Bandit Guided Submodular Curriculum for Adaptive Subset Selection

Prateek Chanda[*] **Prayas Agrawal**[*] **Saral Sureka**
Lokesh Reddy Polu **Atharv Kshirsagar** **Ganesh Ramakrishnan**

Department of Computer Science and Engineering,
Indian Institute of Technology Bombay
{prateekch, prayas, ssaral
{lokeshreddypolu, atharvksagar, ganesh}@cse.iitb.ac.in

Abstract

Traditional curriculum learning proceeds from easy to hard samples, yet defining a reliable notion of difficulty remains elusive. Prior work has used submodular functions to induce difficulty scores in curriculum learning. We reinterpret adaptive subset selection and formulate it as a multi-armed bandit problem, where each arm corresponds to a submodular function guiding sample selection. We introduce ONLINESUBMOD, a novel online greedy policy that optimizes a utility-driven reward and provably achieves no-regret performance under various sampling regimes. Empirically, ONLINESUBMOD outperforms both traditional curriculum learning and bi-level optimization approaches across **vision** and **language datasets**, showing superior accuracy-efficiency tradeoffs. More broadly, we show that validation-driven reward metrics offer a principled way to guide the curriculum schedule. Our code is publicly available at GitHub [2].

1 Introduction

Curriculum Learning (CL), inspired by cognitive development, posits that training machine learning models by gradually exposing them to data of increasing complexity can significantly enhance both learning efficiency and generalization performance [4, 59]. The underlying principle is that mastering simpler concepts first provides a robust foundation for acquiring more complex ones, leading to improved convergence and a more effective exploration of the hypothesis space [19]. Empirical evidence shows CL improves model training, particularly in areas like code understanding [33], enhances graph embeddings through complexity-based ordering [57], mitigates catastrophic forgetting [2, 26, 50, 43], and boosts learning efficiency in reinforcement learning [34]. We first provide a formal definition of Curriculum Learning.

Definition 1. *(**Curriculum Learning**) Given a dataset $\mathcal{D} = \bigcup_{i=1}^{k} \mathcal{B}_i$ partitioned into disjoint batches $\mathcal{B}_i$, and a batch difficulty score function $\boldsymbol{d} : \{\mathcal{B}_i\}_{i=1}^{k} \to \mathbb{R}_{\geq 0}$ assigning non-negative difficulty scores, a **batch-wise curriculum** can be represented as a permutation $\pi : [k] \mapsto [k]$ over the ordered indices such that the ordered sequence*

$$\mathcal{C} = (\mathcal{B}_{\pi(1)}, \mathcal{B}_{\pi(2)}, \ldots, \mathcal{B}_{\pi(k)}),$$

*satisfies the **monotonic difficulty score condition**: $\boldsymbol{d}(\mathcal{B}_{\pi(t)}) \leq \boldsymbol{d}(\mathcal{B}_{\pi(t+1)}) \quad \forall t \in \{1, \ldots, k-1\}$.*

Determining Difficulty is challenging A critical challenge in realizing the full potential of Curriculum Learning (CL) is determining the optimal sequence of batches. This is complicated by the fact

[*] These authors contributed equally.

[2] https://github.com/efficiency-learning/banditsubmod/

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

that the difficulty score, denoted as $\boldsymbol{d}$, is typically unknown. Traditional approaches often rely on domain expertise or practitioner's knowledge to assess the hardness or difficulty of samples.

Recent works, such as [19], have proposed using submodular function maximization over data batches as an intrinsic measure of sample difficulty. In particular, **representative submodular functions** representative submodular functions are used to identify easy samples, while **diversity focused submodular functions** are used to capture difficult ones. As a result, the CL objective is typically constructed by prioritizing diversity functions later and representative functions earlier in the training phase. However, this definition of hardness is still restrictive, as it relies on a fixed pretraining phase and does not account for evolving training dynamics.

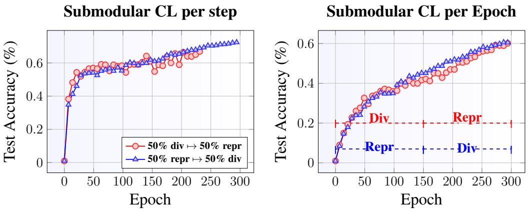

Figure 1: **Sequential Ordering of Submodular Functions**: *Observations on CIFAR100*: Initial training with subsets sampled using representation-based submodular functions followed by diversity results in better performance gains than the opposite order. (1a): First 50% of **steps in an epoch**. (1b): First 50% of **epochs**.

Adaptive Subset Selection Induces CL Many adaptive subset selection methods although can be viewed as forms of curriculum learning, incur substantial computational overhead. For instance, Glister [20] solves costly bilevel optimization involving joint subset selection and model training with validation feedback. GradMatch [18] minimizes gradient matching error by solving complex optimization problems to approximate full-dataset gradients. Importance sampling approaches [7, 46] similarly require expensive importance score estimations. Such costs limit the scalability of advanced curriculum strategies, especially under resource constraints or large datasets.

1.1 Our Contributions

Submodular curriculum learning via online bandits We formulate the curriculum learning problem in conjunction with the adaptive subset selection as a multi-arm bandit problem, where each arm corresponds to a submodular function that captures its unique characteristics, thereby providing a good surrogate difficulty score required for curriculum learning design.

A no-regret greedy policy for adaptive subset selection We introduce ONLINESUBMOD, a novel greedy utility-based policy that leverages feedback from validation performance-driven reward signal to adaptively guide the subset selection process. We prove that ONLINESUBMOD achieves no-regret performance under general sampling regimes, providing theoretical grounding for its learning efficiency.

Validation performance-aware reward design Unlike prior work which uses static heuristics or model-dependent metrics, we define a utility function based on validation performance-driven reward improvements, thereby aligning curriculum progression with actual generalization objectives. **Empirical improvements across modalities** Through extensive experiments on large-scale language and vision benchmarks, we demonstrate that ONLINESUBMOD outperforms traditional curriculum strategies and state-of-the-art adaptive selection methods in terms of accuracy-efficiency trade-offs across diverse subset budgets and training stages.

1.2 Brief Discussion on Related Work & Limitations

Here we detail some of the recent prior work in the space of adaptive subset selection and corresponding limitations.

Leveraging Training Gradient information: Efficiently training robust machine learning models often involves selecting informative data subsets. GLISTER [20] directly addresses this through a mixed discrete-continuous bi-level optimization framework, leveraging validation likelihood for robustness. The concept of *adaptive data subset selection*, where the subset evolves during training, is explored by methodologies like coreset selection [32]. [18] tackles the problem by focusing on minimizing *gradient matching error*, as the quality of this matching significantly impacts convergence. By modeling this error as weakly submodular and using OMP [9], GradMatch achieves tighter convergence guarantees for various convex loss functions. Despite their advancements, many contemporary subset selection techniques, such as coreset selection and related methods [8], pose a considerable computational burden due to their complex optimization processes.

Reweighting Techniques: In this context, [53] offered significant insights into strategies for selecting data subsets that focus on identifying high-quality subsets during the training of models. As we shift towards meta-learning and weighted loss techniques, traditional methods like importance sampling, first introduced by [6], and more contemporary approaches such as focal loss proposed by [25], provide essential perspectives on weighting samples to highlight more challenging examples during training. However, all these strategies entail additional costs. We further share a more detailed Related work section in Appendix G.

2 Notation and Problem Setup

Notation: We consider a supervised learning setup where we have a training dataset $\boldsymbol{\mathcal{D}}_{\text{tr}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, with each instance independently and identically distributed (i.i.d.) according to a distribution $\mathbb{P}_{\mathcal{X}\times\mathcal{Y}}$ over the feature space $\mathcal{X}$ and label space $\mathcal{Y}$. Similarly, we have a validation dataset $\boldsymbol{\mathcal{D}}_{\texttt{val}} = \{(\mathbf{x}_j^{\texttt{val}}, y_j^{\texttt{val}})\}_{j=1}^m$, also drawn i.i.d. from $\mathbb{P}_{\mathcal{X}\times\mathcal{Y}}$. Here, $\mathbf{x} \in \mathcal{X}$ represents the features and $y \in \mathcal{Y}$ represents the labels. Let $\mathcal{M}_{\boldsymbol{\theta}}$ be a model parameterized by $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subseteq \mathbb{R}^{\boldsymbol{d}}$, with $\boldsymbol{\Theta}$ being a compact and convex parameter space. The learning objective is to minimize the empirical risk $L(\mathcal{M}_{\boldsymbol{\theta}}; \boldsymbol{\mathcal{D}}_{\text{tr}})$ [51]. The training process unfolds over a discrete time horizon $T \in \mathbb{Z}^+$. Let $\boldsymbol{\mathcal{F}}$ be the space of set functions, with $\boldsymbol{\mathcal{F}}_{\text{sub}} \subset \boldsymbol{\mathcal{F}}$ denoting the subspace of submodular functions.

Note: Throughout this paper, we use $\mathbf{z}$ to denote a training instance from $\mathcal{B}_t$, unless explicitly labeled as $\mathbf{z}_{\text{val}}$, which refers to a validation instance. In Appendix Section B we provide an extensive notation summary. We provide here some important definitions which would be utilised in the later sections.

Definition 2 (Submodularity). *Given a ground set $\mathcal{V}$, a set function $\boldsymbol{f} : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is submodular if for all $\mathcal{S} \subseteq \mathcal{V}$ and $\mathcal{B} \subseteq \mathcal{A} \subseteq \mathcal{V}$, it holds that $\boldsymbol{f}(\mathcal{S} \cup \mathcal{A}) - \boldsymbol{f}(\mathcal{A}) \leq \boldsymbol{f}(\mathcal{S} \cup \mathcal{B}) - \boldsymbol{f}(\mathcal{B})$.*

Definition 3 (Monotonicity). *A set function $\boldsymbol{f} : 2^{\mathcal{V}} \mapsto \mathbb{R}_{\geq 0}$ is monotone if for all $\mathcal{B} \subseteq \mathcal{A} \subseteq \mathcal{V}$, it holds that $\boldsymbol{f}(\mathcal{B}) \leq \boldsymbol{f}(\mathcal{A})$.*

Definition 4 (Maximum High Value Subset). *Corresponding to a monotone submodular function $\boldsymbol{f}$, the maximum high value subset of cardinality at most β, denoted by $\boldsymbol{f}_{\text{arg}}(\beta) = \mathcal{B}^{opt} \subseteq \mathcal{V}$, is defined as: $\mathcal{B}^{opt} = \underset{\mathcal{B}\subseteq\mathcal{V}; |\mathcal{B}|\leq\beta}{\operatorname{argmax}} \boldsymbol{f}(\mathcal{B})$.*

2.1 Problem Formulation : Adaptive Subset Selection posed as Curriculum Learning

At each discrete time step $t \in [T]$, we consider a mini-batch $\mathcal{B}_t \subseteq \boldsymbol{\mathcal{D}}_{\text{tr}}$ upon which the model $\mathcal{M}_{\boldsymbol{\theta}}$ is trained. Let $\ell : \mathcal{Z} \times \boldsymbol{\Theta} \mapsto \mathbb{R}$ denote the instance-wise loss function, where $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ is the instance space, and the model parameter at time t is denoted by $\boldsymbol{\theta}_t \in \boldsymbol{\Theta}$. The total loss over the mini-batch $\mathcal{B}_t$ is given by $\boldsymbol{\mathfrak{L}}_t(\boldsymbol{\theta}_t) = \sum_{\mathbf{z}\in\mathcal{B}_t} \ell(\mathbf{z}, \boldsymbol{\theta}_t)$. Concurrently, we have access to a validation mini-batch $\mathcal{B}_t^{\texttt{val}} \subseteq \boldsymbol{\mathcal{D}}_{\texttt{val}}$ at each time step t.

Function	$\boldsymbol{f}(X)$
Representative	
Facility Location	$\sum_{i\in\mathcal{V}} \max_{j\in X} s_{ij}$
Graph Cut	$\sum_{i\in\mathcal{V},j\in X} s_{ij} - \rho \sum_{i,j\in X} s_{ij}$
Diversity	
Log Determinant	$\log\det(\mathcal{S}_X)$
Disparity-Min	$\min_{i\neq j\in X}(1 - s_{ij})$
Disparity-Sum	$\sum_{i\neq j\in X}(1 - s_{ij})$

Table 1: *Submodular functions used in arm definitions. $\mathcal{V}$ is the ground set, $X \subseteq \mathcal{V}$, s_{ij} denotes pairwise similarity, and $\mathcal{S}_X$ is the similarity submatrix. ρ indicates the balancing factor between representative and diversity nature. We also utilise mutual information variants (Details in Appendix)*

Gradient Matrix and Mean Gradient: Let $\mathbf{G}_{\boldsymbol{\theta}_t} = \left[\boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_1), \quad \ldots, \quad \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_{|\mathcal{B}_t|})\right] \in \mathbb{R}^{\boldsymbol{d}\times|\mathcal{B}_t|}$ be the batch gradient matrix at time step t, where each column $\boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_i) = \nabla_{\boldsymbol{\theta}} \ell(\mathbf{z}_i, \boldsymbol{\theta}_t) \in \mathbb{R}^{\boldsymbol{d}}$ is the **sample-wise gradient** of the loss function ℓ with respect to the model parameter $\boldsymbol{\theta}_t$, for all $\mathbf{z}_i \in \mathcal{B}_t$. Let $\mathbf{1}_{|\mathcal{B}_t|} \in \mathbb{R}^{|\mathcal{B}_t|\times 1}$ denote the column vector of ones. We define the **per-batch gradient** as $\bar{\mathbf{g}}_{\boldsymbol{\theta}_t}^{(b)} = \frac{1}{|\mathcal{B}_t|} \sum_{\mathbf{z}_i\in\mathcal{B}_t} \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_i) = \frac{1}{|\mathcal{B}_t|} \mathbf{G}_{\boldsymbol{\theta}_t} \mathbf{1}_{|\mathcal{B}_t|}$.

Action Space and Submodular Selection Policy. At each time step $t \in [T]$, the learner observes a mini-batch $\mathcal{B}_t \subseteq \mathcal{D}_{\text{tr}}$ and must select a subset of size β to compute a gradient update. The learner chooses an action $a_t \in \mathscr{A}$ from a discrete action space: $\mathscr{A} := \left\{\boldsymbol{f}^{(1)}, \boldsymbol{f}^{(2)}, \ldots, \boldsymbol{f}^{(\mathcal{K})}\right\}, \quad \boldsymbol{f}^{(a)} \in \mathcal{F}_{\text{sub}}$, where each $\boldsymbol{f}^{(a)} : 2^{\mathcal{B}_t} \to \mathbb{R}$ is a monotone submodular function used to score subsets of $\mathcal{B}_t$. These functions encode different sample selection criteria such as diversity, coverage, and representativeness (see Table 1 for examples). The selected function $\boldsymbol{f}^{(a_t)}$ is then approximately maximized over $\mathcal{B}_t$ under a fixed cardinality constraint to produce a training subset: $\mathcal{S}_t := \arg\max_{S \subseteq \mathcal{B}_t,\, |S| \leq \beta} \boldsymbol{f}^{(a_t)}(S)$, which is typically computed via a greedy algorithm. The model is updated using $\mathcal{S}_t$, and the quality of the update is evaluated using a utility-based reward defined on a held-out validation mini-batch $\mathcal{B}_t^{\mathsf{val}} \subseteq \mathcal{D}_{\mathsf{val}}$.

Specifically, let $\boldsymbol{\vartheta}(a \mid \mathcal{B}_t)$ be the empirical estimate of the expected reward for arm $a \in \mathscr{A}$.

Policy: Greedy Deterministic Selection. We adopt a greedy deterministic policy $\pi : 2^{\mathcal{D}_{\mathsf{tr}}} \to \mathscr{A}$ that selects the arm with the highest estimated reward at each time step i.e. $a_t := \pi(\mathcal{B}_t) := \arg\max_{a \in \mathscr{A}} \boldsymbol{\vartheta}(a_t \mid \mathcal{B}_t)$.. where $\mathcal{U}_t$ is the utility function defined in Section 2.1.

Regret as a Performance Measure We denote by $(*)$ the index of an optimal action, so that $\mu_{(*)}(\mathcal{B}_t)$ represents the expected utility (e.g., value of the selected subset) of an optimal submodular function $\boldsymbol{f}^{(a_t^*)}$ when applied to mini-batch $\mathcal{B}_t$. For each action $a_t \in \mathscr{A}$, we define the *optimality gap* at time t as $\boldsymbol{\Delta}_{(a_t)}(\mathcal{B}_t) := \max\{0,\ \boldsymbol{\vartheta}(a_t^* \mid \mathcal{B}_t) - \boldsymbol{\vartheta}(a_t \mid \mathcal{B}_t)\}$. The *cumulative regret* after T rounds is then defined as $\mathsf{Regret}_T := \sum_{t=1}^{T} \boldsymbol{\Delta}_{a_t}(\mathcal{B}_t)$,. Minimizing Regret_T ensures that the learner approaches the performance of the best submodular selector in hindsight. We define $\boldsymbol{\vartheta}(\bullet \mid \mathcal{B}_t)$ in Sec 2.2

Reward Utility Metric for Performance Evaluation Drawing upon the concept of training data influence [38], we define a utility function $\mathcal{U}_t(\mathcal{B}_t, \mathbf{z}_{\mathsf{val}}) : 2^{\mathcal{D}_{\mathsf{tr}}} \times \mathcal{D}_{\mathsf{val}} \mapsto \mathbb{R}$ to quantify the impact of a training mini-batch $\mathcal{B}_t \subseteq \mathcal{D}_{\mathsf{tr}}$ at time step t on a validation instance $\mathbf{z}_{\mathsf{val}} \in \mathcal{B}_t^{\mathsf{val}}$. Specifically, the utility is the reduction in the loss on the validation instance after one step of stochastic gradient descent:

$$\mathcal{U}_t(\mathcal{B}_t, \mathbf{z}_{\mathsf{val}}) = \ell(\mathbf{z}_{\mathsf{val}}, \boldsymbol{\theta}_t) - \ell(\mathbf{z}_{\mathsf{val}}, \tilde{\boldsymbol{\theta}}_{t+1}(\mathcal{B}_t)), \tag{1}$$

where the updated parameter vector $\tilde{\boldsymbol{\theta}}_{t+1}(\mathcal{B}_t) = \boldsymbol{\theta}_t - \eta_t \boldsymbol{\nabla}_{\boldsymbol{\theta}} \left(\frac{1}{|\mathcal{B}_t|} \sum_{\mathbf{z} \in \mathcal{B}_t} \ell(\mathbf{z}, \boldsymbol{\theta}_t) \right)$.

First-Order Approximation of Marginal Utility Gain: We define the instance-wise conditional marginal utility gain of including the i-th training instance $\mathbf{z}_i$ into a partially constructed mini-batch $\mathcal{B}_t^{(<i)} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{i-1}\}$ at time step t, with respect to a validation instance $\mathbf{z}_{\mathsf{val}}$, as the change in utility $\mathcal{U}_t$:

$$\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)}, \mathbf{z}_{\mathsf{val}}) = \mathcal{U}_t(\mathcal{B}_t^{(<i)} \cup \{\mathbf{z}_i\}; \mathbf{z}_{\mathsf{val}}) - \mathcal{U}_t(\mathcal{B}_t^{(<i)}; \mathbf{z}_{\mathsf{val}}) \tag{2}$$

$$\approx \eta_t \boldsymbol{\nabla}_{\boldsymbol{\theta}} \ell(\mathbf{z}_i, \boldsymbol{\theta}_t) \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta}} \ell(\mathbf{z}_{\mathsf{val}}, \boldsymbol{\theta}_{t+1}(\mathcal{B}_t^{(<i)})) \tag{3}$$

The approximation in the last step utilizes a first-order Taylor expansion, which is reasonable under the common assumption of a small learning rate η_t. We defer the derivation to Appendix

Second-Order Approximation and Gradient Influence: Further approximating the second term in Equation (3) using another first-order Taylor expansion around $\boldsymbol{\theta}_t$, we obtain:

$$\eta_t \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_i) \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta}} \ell(\mathbf{z}_{\mathsf{val}}, \boldsymbol{\theta}_{t+1}(\mathcal{B}_t^{(<i)})) \approx \eta_t \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_i) \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta}} \ell(\mathbf{z}_{\mathsf{val}}, \boldsymbol{\theta}_t - \eta_t \frac{1}{|\mathcal{B}_t^{(<i)}|} \sum_{\mathbf{z} \in \mathcal{B}_t^{(<i)}} \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}))$$

$$\approx \eta_t \underbrace{\boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_i) \cdot \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_{\mathsf{val}})}_{\text{Gradient Influence Function}(\textbf{Term I})} \underbrace{- \eta_t^2 \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_i)^\top \boldsymbol{\mathcal{H}}_{\mathbf{z}_{\mathsf{val}}}(\boldsymbol{\theta}_t)(\frac{1}{|\mathcal{B}_t^{(<i)}|} \sum_{\mathbf{z} \in \mathcal{B}_t^{(<i)}} \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}))}_{\text{Hessian Weighted Relative Similarity}(\textbf{Term II})} \tag{4}$$

where $\boldsymbol{\mathcal{H}}_{\mathbf{z}_{\mathsf{val}}}(\boldsymbol{\theta}_t) = \nabla^2_{\boldsymbol{\theta}} \ell(\mathbf{z}_{\mathsf{val}}, \boldsymbol{\theta}_t)$ denotes the Hessian of the loss function with respect to the model parameters $\boldsymbol{\theta}$ evaluated at $\boldsymbol{\theta}_t$ for the validation data point $\mathbf{z}_{\mathsf{val}}$.

Gradient Influence Function: The first term indicates the importance score of $\mathbf{z}_i$ *w.r.t* validation data point $\mathbf{z}_{\mathsf{val}}$ which, in essence, captures the effectiveness of the gradient of the training instance $\mathbf{z}_i$ towards the reduction in the validation loss. This term closely resembles the influence function proposed in [38].

Relative Similarity Term The second term indicates the Hessian weighted relative similarity of the current training instance with all other training instances in the batch $\mathcal{B}_t^{(<i)}$.

Hessian Approximation Strategies The Hessian term $\boldsymbol{\mathcal{H}}_{\mathbf{z}_{\mathsf{val}}}(\boldsymbol{\theta}_t)$ in Equation (4) presents a major computational bottleneck due to its high cost. To alleviate this, several approximation strategies are commonly employed: *Kronecker-Factored Approximation* methods [54] exploit layer-wise structure and approximate the Hessian using Kronecker products; *Gauss-Newton Decomposition* [42] replaces the Hessian with the covariance of output gradients, assuming a negligible residual; and the *Identity Approximation* [29, 37] simplifies the Hessian to $\mathbf{I}_d$, yielding a low-cost diagonal preconditioner. In our current list of experiments, we consider Hessian to $\mathbf{I}_d$ as it is has been shown to be usefull with low approximation error in large scale trainings e.g. LLM settings [52]. In Appendix Section D.6, we include other Hessian Approximation strategies which we tried out along with corresponding ablation studies.

2.2 Sample-wise Expected Marginal Gain

We define the sample-wise expected marginal gain as the expectation of the conditional marginal utility gain over a validation instance $\mathbf{z}_t^{\mathsf{val}}$ and a training instance $\mathbf{z}_i$ from the partially constructed mini-batch $\mathcal{B}_t^{(<i)}$ as $\mathbb{E}_{\mathbf{z}_t^{\mathsf{val}} \in \mathcal{B}_t^{\mathsf{val}},\, \mathbf{z}_i \in \mathcal{B}_t^{(<i)}} \left[\boldsymbol{\Delta}\mathcal{U}_t \left(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)},\, \mathbf{z}_t^{\mathsf{val}} \right) \right]$ Here, due to the property of permutation invariance over the samples in $\mathcal{B}_t^{(<i)}$ as shown in Lemma 2, the inner expectation can be written as:

$$\mathbb{E}_{\mathbf{z}_i \in \mathcal{B}_t^{(<i)}} \left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)}, \mathbf{z}_t^{\mathsf{val}}) \right] \triangleq \eta_t \bar{\mathbf{g}}_{\boldsymbol{\theta}_t}^{(b)} \cdot \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_t^{\mathsf{val}}) - \eta_t^2 \bar{\mathbf{g}}_{\boldsymbol{\theta}_t}^{(b)\top} \left(\mathbf{I}_d - \frac{1}{|\mathcal{B}_t|} \mathbf{1}_{d \times |\mathcal{B}_t|} \mathbf{G}_{\boldsymbol{\theta}_t}^\top \right) \boldsymbol{\mathcal{H}}_{\mathbf{z}_t^{\mathsf{val}}}(\boldsymbol{\theta}_t) \bar{\mathbf{g}}_{\boldsymbol{\theta}_t}^{(b)}. \tag{5}$$

A direct greedy approach to maximize the conditional marginal gain at each step t by iteratively selecting the training instance $\mathbf{z}_i^* \notin \mathcal{B}_t^{(<i)}$ that yields the maximal local reduction in validation loss, i.e., $\mathbf{z}_i^* = \operatorname{argmax}_{\mathbf{z}_i \notin \mathcal{B}_t^{(<i)}} \boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)}, \mathbf{z}_t^{\mathsf{val}})$, is computationally prohibitive. Constructing the new subset batch $\mathcal{S}_t$ of size β from the current mini-batch $\mathcal{B}_t$ via this exhaustive greedy maximization starting from an empty set ($\mathcal{B}_t^{(<0)} = \emptyset$) incurs a computational complexity of $\mathcal{O}\binom{|\mathcal{B}_t|}{\beta}$.

Submodular Relaxation for Efficient Selection: To overcome the computational intractability of exact optimization, we introduce a relaxation that exploits the structure of submodular functions to enable efficient selection of high-value subsets. Specifically, for each submodular function arm $a_t \in \mathscr{A}$, we compute an approximately optimal subset $\mathcal{S}_{a_t}^{\mathtt{opt}} \subseteq \mathcal{B}_t$ of size at most β, chosen to maximize the submodular objective $\boldsymbol{f}^{(a_t)}(\mathcal{S})$. Since exact maximization of submodular functions is NP-hard, we adopt a standard greedy algorithm that offers a provable $(1 - 1/e)$-approximation guarantee under cardinality constraints.

Reward Formulation using Submodular Function Arms: We define the overall expected marginal gain $\boldsymbol{\vartheta} : \mathscr{A} \times T \mapsto \mathbb{R}$ for each submodular function arm $a_t \in \mathscr{A}$ at time step t as the expectation of the instance-wise conditional marginal gain $\boldsymbol{\Delta}\mathcal{U}_t$, conditioned on a validation instance and a training instance from the approximately optimal subset $\mathcal{S}_{a_t}^{\mathtt{opt}}$:

$$\boldsymbol{\vartheta}(a_t \mid \mathcal{B}_t) = \mathbb{E}_{\mathbf{z}_t^{\mathsf{val}} \in \mathcal{B}_t^{\mathsf{val}}, \mathbf{z}_i \in \mathcal{S}_{a_t}^{\mathtt{opt}}} \left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{S}_{a_t}^{\mathtt{opt}\,(<i)}, \mathbf{z}_t^{\mathsf{val}}) \right] \tag{6}$$

The best arm is then selected via $\hat{a}_t = \arg \max_{a_t \in \mathscr{A}} (\boldsymbol{\vartheta}(a_t \mid \mathcal{B}_t))$.

2.3 Speedup for OnlineSubmod

Gradient Computation Full-model gradients in deep networks are expensive to compute due to high-dimensionality. For vision tasks, we adopt last-layer gradients following [3], and for LLMs, we compute gradients over LoRA adapters (rank 128) as in [52]. Both reduce overhead while preserving informative signals for subset selection.

OnlineSubmod-Batch To align with batch-level baselines [18], we extend our samplewise formulation to the batch setting, treating each batch as a unit. Let $\mathbf{M}_t^{(b)} = \begin{bmatrix} \tilde{\boldsymbol{g}}_1^{(b)} & \cdots & \tilde{\boldsymbol{g}}_{|\mathbb{S}_t|}^{(b)} \end{bmatrix}$ be the matrix of average gradients $\tilde{\boldsymbol{g}}_i$ for batches $\mathcal{B}_i \in \mathbb{S}_t$, where $\mathbb{S}_t$ denotes the set of sampled batches at time t. The expected conditional marginal gain becomes:

$$\mathbb{E}_{\mathcal{B}_i \in \mathbb{S}_{t_{[:\prec i]}}} \boldsymbol{\Delta}\mathcal{U}_t(\bullet) \triangleq \left[\eta_t \mathbf{M}_t^{(b)} \mathbf{1}_{|\mathbb{S}_t|}\, \boldsymbol{g}(\mathbf{z}_t^{\mathtt{val}}) - \eta_t^2 \mathbf{M}_t^{(b)} \left(\mathbf{1}_{|\mathbb{S}_t|} \mathbf{1}_{|\mathbb{S}_t|}^T - \mathbf{I}_{|\mathbb{S}_t|} \right) \boldsymbol{\mathcal{H}}_{\mathbf{z}_t^{\mathtt{val}}} (\mathbf{M}_t^{(b)})^T \mathbf{1}_{|\mathbb{S}_t|} \right] \tag{7}$$

Other methods can be analogously adapted by substituting samples $\boldsymbol{x}_i$ with batches $\mathcal{B}_i$.

3 Algorithm

ONLINESUBMOD instantiates a contextual multi-armed bandit framework to adaptively select curriculum policies throughout training.

```
Algorithm 1: ONLINESUBMOD
Input: $T \in \mathbb{N}$: Total training steps
       $\{\boldsymbol{f}^{(a)}\}_{a=1}^{\mathcal{K}}$: Candidate submodular arms
       $\lambda(\cdot),\ \pi(\cdot)$: Time-varying exploration parameters
Output: $\boldsymbol{\theta}_{T+1}$: Final model parameter
1 for $t = 1$ to $T$ do
2     Receive batch $\mathcal{B}_t$
3     Sample $\zeta \sim \mathcal{U}(0,1)$
4     Threshold: $\Xi_t \leftarrow \frac{t}{(t+\lambda(t))^{\pi(t)}}$
5     $\hat{a}_t \leftarrow \begin{cases} \arg\max_{a_t \in \mathscr{A}} \boldsymbol{\vartheta}(a_t \mid \mathcal{B}_t) & \text{if } \zeta > \boldsymbol{\Xi}_t \\ \text{Uniform}(\mathscr{A}) & \text{otherwise} \end{cases}$
6     $\mathcal{S}_{(\hat{a}_t)} \leftarrow \arg\max_{|\mathcal{S}| \le \beta, \mathcal{S} \subseteq \mathcal{B}_t} \boldsymbol{f}^{(\hat{a}_t)}(\mathcal{S})$
7     $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \frac{\eta_t}{|\mathcal{S}_{(\hat{a}_t)}|} \sum_{\mathbf{z} \in \mathcal{S}_{(\hat{a}_t)}} \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z})$
8 return $\boldsymbol{\theta}_{T+1}$
```

- ***Step: 1-2*** The model receives a batch $\mathcal{B}_t$ and chooses an arm $\hat{a}_t \in \mathscr{A}$, each corresponding to a distinct submodular utility function $\boldsymbol{f}^{(\hat{a}_t)} : 2^{\mathcal{B}_t} \to \mathbb{R}_{\ge 0}$.
- ***Step: 5*** The arm selection is governed by a exploration threshold $\Xi_t := \frac{t}{(t+\lambda(t))^{\pi(t)}}$, parameterized by time-dependent schedules $\lambda(t)$ and $\pi(t)$ that modulate the annealing from exploration to exploitation. Here, $\lambda(t)$ **(Exploration Dampening)** and $\pi(t)$ **(Exploration Sharpness)** act as curriculum schedulers. If a uniform sample satisfies $\zeta > \boldsymbol{\Xi}_t$, the algorithm enters ***Exploitation Phase*** and selects the arm maximizing $\boldsymbol{\vartheta}(a \mid \mathcal{B}_t)$; otherwise, an arm is sampled uniformly at random (***Exploration Phase***).
- ***Step: 6*** Once an arm $\hat{a}_t$ is selected, the algorithm performs approximate maximization over $\mathcal{B}_t$ with respect to $\boldsymbol{f}^{(\hat{a}_t)}$, selecting a subset $\mathcal{S}_{(\hat{a}_t)}$.
- ***Step: 7*** The model parameters $\boldsymbol{\theta}_t$ are then updated using a stochastic gradient step computed only on the selected subset.

4 Theoretical Results

In this section, we present the main theoretical results of our work, focusing on regret guarantees for our best-arm selection policy. Specifically, we analyze the regret incurred by our method relative to the performance of the optimal arm in hindsight. This requires a set of structural assumptions (pertaining to describe properties of the exploration dynamics, utility approximation quality, and the existence of a reward gap between optimal and suboptimal arms).

Assumption (a) (**Constant Fractional Exploration Dampening**): The exploration dampening parameter $\lambda(t)$ is time-invariant $\lambda(t) = \epsilon$ where $\quad \epsilon \in (0, 1)$.

Assumption (b) (**Optimality Gap**): There exists an optimality gap $\boldsymbol{\varrho}$ such that for every suboptimal arm $a_t \in \mathscr{A} \setminus \{a^*\} : 0 \le \boldsymbol{\varrho} \le \boldsymbol{\Delta}_{(a_t)}(\mathcal{B}_t)$.

Assumption (c) (**Fractional Exploration Sharpness**): The exploration sharpness parameter $\pi(t)$ is a bounded quantity $\pi(t) \in (0, 1)$.

Assumption (d) (**Utility Metric Approximation**): The utility metric $\mathcal{U}_t(\cdot, \cdot)$ satisfies the approximation bound as per Theorem 2 (Appendix) with constants $\mathfrak{C}_{(a)}$ for each arm $a \in \mathscr{A}$ and let n_a be a specific constant associated with arm a such that Theorem 2 (Appendix) holds true.

Theorem 1 (**Regret Guarantees**). *Under Assumptions **a - d**, for all* $t > t_0$*, with probability at least*

$$1 - \mathcal{K} \exp\left(-\frac{3(t-2)(1+(1-\pi)\epsilon)}{28\mathcal{K}(2-\pi)}\right),$$

the expected instantaneous regret incurred by the arm selection policy satisfies

$$\begin{aligned} \mathbb{E}[\text{Regret}_t] &:= \mathbb{E}_{\mathcal{B}_t} \mathbb{E}_{\hat{a}_t \in \mathscr{A}} \mathbb{E}_{\boldsymbol{\vartheta}} \left[\boldsymbol{\vartheta}(a_t^* \mid \mathcal{B}_t) - \boldsymbol{\vartheta}(\hat{a}_t \mid \mathcal{B}_t)\right] \\ &= O\left(\frac{1}{t}\right) + O\left(\frac{\mathcal{K}^{3/2}(\max_a \mathfrak{C}_{(a)} + \mathfrak{C}_*)}{\boldsymbol{\varrho}} \sqrt{\frac{\log t}{t}}\right), \end{aligned} \tag{8}$$

where $\mathfrak{C}_*$ *is the approximation constant corresponding to the optimal arm* a^*.

The theorem guarantees that, under the specified assumptions, the arm selection strategy based on maximizing the expected marginal utility gain converges to the optimal arm almost surely, with the regret decreasing at a rate combining a fast $1/t$ decay and a slower $\sqrt{\frac{\log t}{t}}$ decay modulated by constants related to the utility approximation and the number of arms. The presence of the optimality gap ϱ in the denominator highlights the difficulty of distinguishing between arms when their utility values are close. We also showcase proofs in Appendix when Assumption (**a**) and Assumption (**c**) are relaxed with no constraints on the bounds of $\lambda(\cdot)$ and $\pi(\cdot)$.

4.1 Supporting Lemmas

Here we detail out Supporting Lemmas that are utilised in proofs and derivations above.

Lemma 1 (**Permutation Invariance of Expected Marginal Gain).** *Let* Π *denote the set of all permutations over the elements of* $\mathcal{B}_t^{(<i)}$. *Then the expected marginal gain* $\mathbb{E}_{\mathbf{z}_i \in \mathcal{B}_t^{(<i)}}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)}, \mathbf{z}_t^{\mathsf{val}})\right]$ *is invariant under any permutation* $\pi \in \Pi$, *i.e.,*

$$\mathbb{E}_{\mathbf{z}_i \in \mathcal{B}_t^{(<i)}}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)}, \mathbf{z}_t^{\mathsf{val}})\right] = \mathbb{E}_{\mathbf{z}_i \in \pi(\mathcal{B}_t^{(<i)})}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \pi(\mathcal{B}_t^{(<i)}), \mathbf{z}_t^{\mathsf{val}})\right].$$

We provide the detailed derivations for all proofs in Appendix H.

5 Experimental Setup

We evaluate ONLINESUBMOD across diverse datasets to highlight its advantages in terms of both accuracy and computational efficiency. All vision-related experiments are conducted using NVIDIA 3 × A6000 GPUs, while large language model (LLM) experiments are performed on 8 × H100 GPUs to ensure fair comparisons with all baselines. We share more details in Appendix Section D.

5.1 Finetuning Large Language Models

Model-Training-Evaluation Pairs. We evaluate ONLINESUBMOD using combinations of two LLMs: `LLAMA-2-7B` [49] and `MISTRAL-7B` [16] finetuned on LESS [55], with performance assessed on MMLU and TYDIQA (Table 1). We use batch size of 16 and use 2 random validation points for computing the reward utility. We select 50% of the batch data for gradient updates during each step.

Table 1: Performance comparison across tasks. Bold indicates best performance in each column.

Method	Avg.	Soc.	Pol.	Hist.	Anat.	ML.	Eth.	Gen.	Bio.	Chem.	TydiQA
GradNorm	46.4%	61.0%	62.5%	**52.1%**	40.5%	40.2%	43.0%	46.7%	42.9%	32.3%	54.6%
MaxLoss	45.2%	60.2%	64.4%	48.0%	39.5%	38.1%	44.4%	43.8%	42.6%	31.1%	55.4%
RhoLoss	46.4%	60.6%	66.2%	49.4%	41.5%	40.2%	42.8%	46.1%	41.1%	33.7%	55.2%
SBERT	45.8%	62.3%	63.7%	47.0%	43.1%	36.8%	43.4%	44.2%	42.0%	32.4%	54.2%
GREATS	47.8%	63.2%	66.2%	48.3%	42.6%	41.1%	46.2%	48.9%	43.1%	33.6%	55.7%
ONLINESUBMOD	**49.6%**	**65.3%**	**67.4%**	**52.1%**	**45.2%**	**42.7%**	**48.6%**	**50.9%**	**45.1%**	**35.7%**	**55.9%**

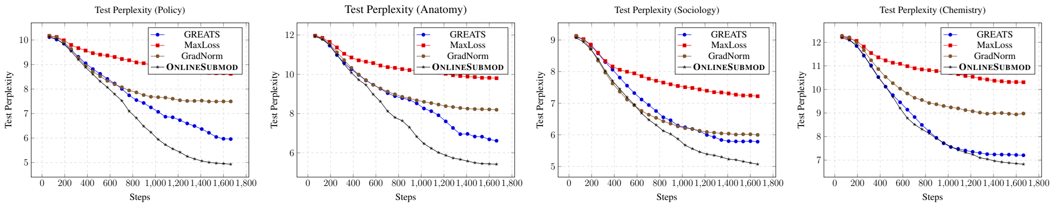

Figure 2: **Test perplexity dynamics** on `LLAMA-2-7B` during training with various **online batch selection strategies** on `MMLU`. We evaluate on `US Foreign Policy`, `Anatomy`, `Sociology`, and `Chemistry`. ONLINESUBMOD significantly outperforms baselines.

Baselines: We compare our algorithm with a variety of online batch selection algorithms: ❶ MAX-LOSS [27], which selects training data points with the highest loss values. ❷ GRADNORM [17], which prioritizes training data points with the highest gradient norms,❸ RHO-LOSS[30], using LLaMA-3.1-8B-Instruct as the reference and LLaMA-2-7B as the target. ❹ SBERT, which selects batches by semantic similarity to validation data using Sentence-BERT embeddings [40]. ❺ GREATS [52] which has a similar utility metric as ours, but the optimization objective instead relies on directly selecting samples greedily that maximizes the utility reward Eqn (4) rather than utilizing any monotone submodular characteristics.

Observations: As can be seen from the perplexity curves (Figure 8) and downstream performance (Table 1), ONLINESUBMOD significantly outperforms other existing baselines, thereby indicating how principled validation performance aware reward signal combined with induced submodular curriculum results in better generalization than static heuristic based approaches.

5.2 Image Classification

We showcase the utility of our method across 5 datasets primarily CIFAR10, CIFAR100 [22],TINYIMAGENET[23], MNIST [24] and SVHN [36]. We compare ONLINESUBMOD with: ❶ GRADMATCH: [18], ❷ CRAIG [32], ❸ GLISTER [20], ❹ RHO-LOSS [30] and ❺ BOSS [1]. All models are trained on a ResNet backbone, 300 epochs, with 20 epochs warm-start (we provide more details regarding cold start vs warm start in Appendix Sec: C.3). We provide a detailed summary of individual baselines, comparision with more datasets and hyperparameters in Appendix D.

Performance Metrics across all baselines We work with the batch-wise variant for ONLINESUBMOD (Section 2.3) keeping in line with other methods. To compare various baselines, we utilize **Speedup** as a relative measure of the training times for each baseline in relation to full batch training. Our goal is to identify a baseline that achieves both high speedup and high test accuracy. We evaluate on budget fractions ($\frac{\beta}{\mathcal{B}_t} \times 100\%$) of 10%, 30% and 50%. From Table 2, we observe that our method significantly outperforms baselines in accuracy, with speedup extremely close to the optimal speedup, obtained by MILO which relies on an expensive offline filtering step based on some assorted selection of submodular functions, contrast to our method which dynamically selects subsets in an online manner. However, this gap is minimal, while our method achieves higher accuracy. For completeness, we also showcase per samplewise selection in Figure 3.

Method	**TinyImageNet**			**CIFAR-100**			**CIFAR-10**		
	10%	**30%**	**50%**	**10%**	**30%**	**50%**	**10%**	**30%**	**50%**
CRAIG [32]	0.524 / 4.82	0.555 / 2.41	0.615 / 1.7	0.672 / 5.1	0.723 / 2.5	0.751 / 1.5	0.900 / 6.7	0.924 / 1.9	0.931 / 1.15
MILO [19]	0.532 / 8.62	0.593 / 3.1	0.623 / 2.6	0.723 / 10.1	0.746 / 3.5	0.756 / 1.95	0.922 / 5.8	0.932 / 2.05	**0.941** / 2.15
GRADMATCH [18]	0.526 / 5.92	0.581 / 2.62	0.619 / 2.1	0.683 / 6.9	0.746 / 3.1	0.753 / 1.3	0.922 / 4.3	0.932 / 1.95	0.941 / 1.48
GLISTER [20]	0.515 / 5.5	0.563 / 2.65	0.621 / 1.7	0.642 / 7.7	0.723 / 2.6	0.746 / 1.2	0.911 / 4.5	0.921 / 1.7	0.926 / 1.3
RHO-LOSS [30]	0.544 / 5	0.597 / 2.57	0.621 / 2	0.713 / 3.9	0.748 / 1.9	0.757 / 1.2	0.901 / 2.5	0.915 / 1.6	0.941 / 1.15
BOSS [1]	0.526 / 5.4	0.601 / 2.9	0.621 / 2.15	0.717 / 7.8	0.737 / 3	0.754 / 1.9	0.916 / 4.9	0.930 / 1.8	0.938 / 1.53
ONLINESUBMOD	**0.553** / 8.43	**0.607** / 3.08	**0.626** / 2.6	**0.736** / 9.2	**0.754** / 3.3	**0.758** / 1.92	**0.924** / 5.4	**0.937** / 2	0.941 / 2.08

Table 2: **Batchwise version performance:** Accuracy vs Speedup. 🟩 ↦ highest accuracy 🟨 ↦ 2nd highest accuracy 🟥 ↦ 3rd highest accuracy.: Performance comparison across different datasets and fractions. Bold indicates the best performance in each column.

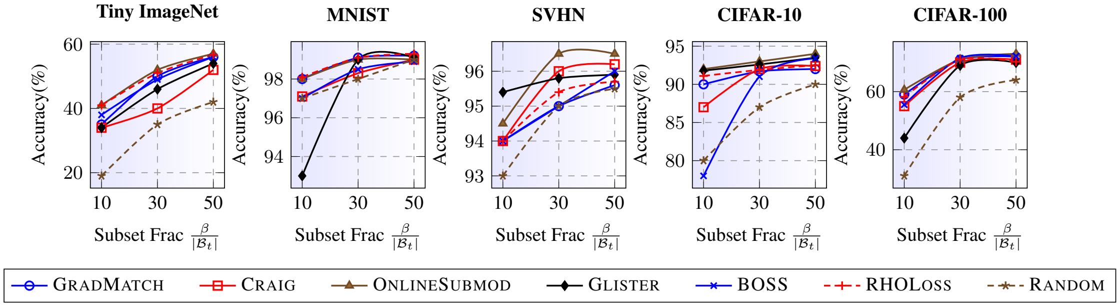

Figure 3: **Samplewise Submodular Curriculum:** ONLINESUBMOD consistently achieves top-1 accuracy across all subset sizes on TINYIMAGENET, SVHN, CIFAR-10, and CIFAR-100, and remains competitive on MNIST. Notably, it matches or outperforms all baselines at early subset fractions (10%, 30%) on all datasets except MNIST.

5.3 Ablation Study results

To understand how the choice of submodular function at each step affects the model performance under ***Exploration***/***Exploitation*** scheme, it is important to understand how the underlying variables affect the overal submodular function selection and thereby model performance at each step.

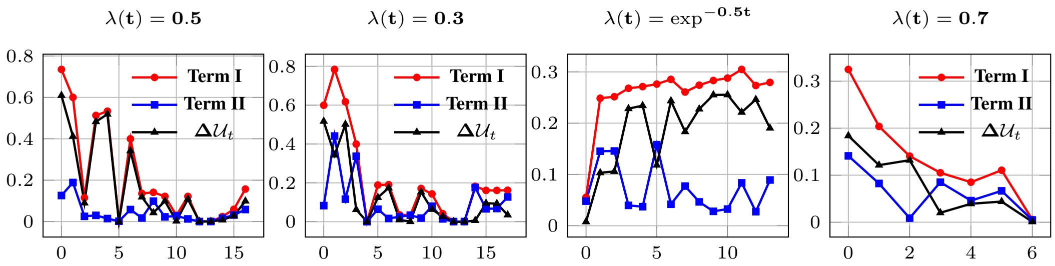

Figure 4: Evolution of **Term I** and **Term II** (Eq 4) across training epochs on CIFAR-100. Here we study the effect of the $\lambda(t)$ and $\pi(t)$. Note for time independent constants we ignore the argument inside $\lambda(\cdot), \pi(\cdot)$.

Formally, $\lambda(t)$: (***Exploration Dampening***) modulates the inertia of exploration. Larger values induce slower increases in Ξ_t, prolonging stochastic exploration across arms, while smaller values accelerate convergence to greedy selection. On the other hand $\pi(t)$ (***Exploration Sharpness***) controls the curvature of the annealing schedule. *High $\pi(t)$ enforces an abrupt shift to exploitation*, *while low $\pi(t)$ yields smoother, prolonged exploration phases*. As shown in Figure 5, for any $\lambda(t)$, increasing π leads to a higher degree of exploitation. For effective learning, the policy must exploit frequently while retaining sufficient exploration to ensure coverage of the state space. $\pi = 1.5$ offers a suitable trade-off—predominantly exploiting with occasional exploration—whereas $\pi = 1.0$ explores too uniformly and $\pi = 0.5$ almost always explores. Hence, $\pi = 1.5$ emerges as the most effective choice.

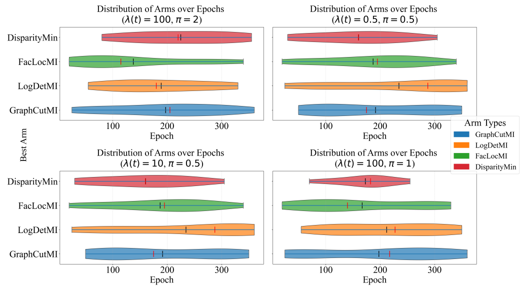

Figure 5: Arm selection distribution over epochs on CIFAR-100. ***Diversity based submodular functions*** become increasingly active during training.

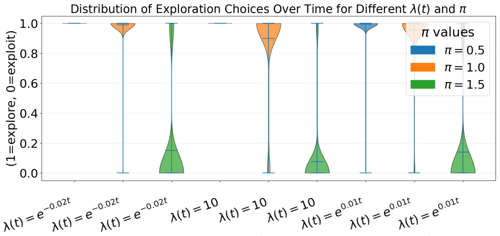

Figure 6: Cumulative ***exploration*** vs. ***exploitation*** choices over time on CIFAR-100.

Computation Breakdown	**Average time**
Gradient Computation	630 ms
Submodular Maximization	0.8 ms
Total time for Subset Selection	640 ms

Table 3: Runtime breakdown showing submodular selection adds negligible overhead.

Computational overhead of Submodular Optimization: Submodular maximization is NP-hard, but most practical solvers use the greedy algorithm, which guarantees a $(1-1/e)$ approximation [35]. Table 3 discusses the tradeoff incurred for the submodular maximization problem w.r.t overall subset selection that involves gradient computation. In our LLM fine-tuning setup on MMLU, LLaMa2-7B using LoRA of rank 128, Table 3 shows that submodular selection takes 0.8 ms on average, while gradient computation takes 630 ms—a 800× gap. Thus, gradient computation remains the primary bottleneck, and submodular selection adds negligible overhead.

6 Conclusion

We introduce OnlineSubmod, a bandit-guided framework for online submodular subset selection that provides a principled alternative to traditional curriculum learning paradigms. By dynamically optimizing a utility-driven reward function, OnlineSubmod effectively balances the trade-off between accuracy and efficiency across diverse training budgets. Our extensive empirical evaluation demonstrates consistent gains over strong state-of-the-art baselines on multiple benchmarks. Future work will focus on extending the proposed greedy utility metric to train neural scoring models, thereby enabling scalable and adaptive subset selection in large-scale pretraining regimes.

7 Acknowledgements

We thank the anonymous reviewers for their constructive feedback and insightful suggestions that helped improve the quality of this work. 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. We also acknowledge the computing resources provided by the Department of Computer Science and Engineering at IIT Bombay. In addition, we are exceptionally grateful to the *BharatGen Initiative* [3] for providing compute resources for conducting large scale language model experiments. Finally, we thank our colleagues and collaborators for valuable discussions and feedback throughout the course of this research process.

References

[1] Abhinab Acharya, Dayou Yu, Qi Yu, and Xumin Liu. Balancing feature similarity and label variability for optimal size-aware one-shot subset selection. In *Forty-first International Conference on Machine Learning*.

[2] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11254–11263, 2019.

[3] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *ICLR*, 2020.

[4] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[5] Andrew A Bian, Joachim M Buhmann, and Andreas Krause. Guarantees for greedy maximization of non-submodular functions with applications. In *International Conference on Machine Learning (ICML)*, pages 498–507, 2017.

[6] Lawrence D Brown. Estimation with incompletely specified loss functions (the case of several location parameters). *Journal of the American Statistical Association*, 70(350):417–427, 1975.

[7] Daniele Calandriello, Michal Derezinski, and Michal Valko. Sampling from a k-dpp without looking at all items. *Advances in Neural Information Processing Systems*, 33:6889–6899, 2020.

[8] Prateek Chanda, Shrey Modi, and Ganesh Ramakrishnan. Bayesian coreset optimization for personalized federated learning. In *The Twelfth International Conference on Learning Representations*.

[9] Ethan R Elenberg, Rajiv Khanna, Alexandros G Dimakis, and Sahand Negahban. Restricted strong convexity implies weak submodularity. *The Annals of Statistics*, 46(6B):3539–3568, 2018.

[10] Kazuya Fujita, Kensuke Okada, and Kentaro Katahira. The fisher information matrix: A tutorial for calculation for decision making models. 2022.

[11] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[12] László Györfi, Michael Kohler, Adam Krzyzak, Harro Walk, et al. *A distribution-free theory of nonparametric regression*, volume 1. Springer, 2002.

[13] Nicholas Harvey, Christopher Liaw, and Tasuku Soma. Improved algorithms for online submodular maximization via first-order regret bounds. *Advances in Neural Information Processing Systems*, 33:123–133, 2020.

[3] BharatGen: http://bharatgen.tech/

[14] Hamed Hassani, Mahdi Soltanolkotabi, and Amin Karbasi. Gradient methods for submodular maximization. *Advances in Neural Information Processing Systems*, 30, 2017.

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

[16] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[17] Angelos Katharopoulos and François Fleuret. Not All Samples Are Created Equal: Deep Learning with Importance Sampling. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 2536–2545. PMLR, 10–15 Jul 2018. Also available as arXiv:1803.00942.

[18] Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pages 5464–5474. PMLR, 2021.

[19] Krishnateja Killamsetty, Alexandre V Evfimievski, Tejaswini Pedapati, Kiran Kate, Lucian Popa, and Rishabh Iyer. Milo: Model-agnostic subset selection framework for efficient model training and tuning. *arXiv preprint arXiv:2301.13287*, 2023.

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

[21] Suraj Kothawade, Vishal Kaushal, Ganesh Ramakrishnan, Jeff Bilmes, and Rishabh Iyer. Prism: A rich class of parameterized submodular information measures for guided data subset selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10238–10246, 2022.

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

[23] Yann Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[24] Yann LeCun, Corinna Cortes, Chris Burges, et al. Mnist handwritten digit database, 2010.

[25] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[26] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

[27] Ilya Loshchilov and Frank Hutter. Online Batch Selection for Faster Training of Neural Networks, 2015.

[28] Adyasha Maharana, Prateek Yadav, and Mohit Bansal. D2 pruning: Message passing for balancing diversity and difficulty in data pruning. *arXiv preprint arXiv:2310.07931*, 2024.

[29] James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR, 2015.

[30] Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pages 15630–15649. PMLR, 2022.

[31] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrak, and Andreas Krause. Fast constrained submodular maximization: Personalized data summarization. In *International Conference on Machine Learning (ICML)*, pages 1358–1367, 2016.

[32] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020.

[33] Marwa Naïr, Kamel Yamani, Lynda Said Lhadj, and Riyadh Baghdadi. Curriculum learning for small code language models. *arXiv preprint arXiv:2407.10194*, 2024.

[34] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020.

[35] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14:265–294, 1978.

[36] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[37] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[38] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930, 2020.

[39] Michael Rawson and Radu Balan. Convergence guarantees for deep epsilon greedy policy learning. *arXiv preprint arXiv:2112.03376*, 2021.

[40] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[41] Tim Roughgarden and Joshua R Wang. An optimal algorithm for online unconstrained submodular maximization. *arXiv preprint arXiv:1806.03349*, 2018.

[42] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.

[43] Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Sayna Ebrahimi, and Hao Wang. Continual learning of large language models: A comprehensive survey. *arXiv preprint arXiv:2404.16789*, 2024.

[44] Matthew Staib, Bryan Wilder, and Stefanie Jegelka. Distributionally robust submodular maximization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 506–516. PMLR, 2019.

[45] Matthew Streeter, Daniel Golovin, and Andreas Krause. Online learning of assignments. *Advances in neural information processing systems*, 22, 2009.

[46] Shivakanth Sujit, Somjit Nath, Pedro Braga, and Samira Ebrahimi Kahou. Prioritizing samples in reinforcement learning with reducible loss. *Advances in Neural Information Processing Systems*, 36:23237–23258, 2023.

[47] Artin Tajdini, Lalit Jain, and Kevin Jamieson. Nearly minimax optimal submodular maximization with bandit feedback. *Advances in Neural Information Processing Systems*, 37:96254–96281, 2024.

[48] Haoru Tan, Sitong Wu, Wei Huang, Shizhen Zhao, and Xiaojuan Qi. Data pruning by information maximization. *International Conference on Learning Representations*, 2025.

[49] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models, 2023.

[50] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

[51] Vladimir Vapnik. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4, 1991.

[52] Jiachen Tianhao Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. Greats: Online selection of high-quality data for llm training in every iteration. *Advances in Neural Information Processing Systems*, 37:131197–131223, 2024.

[53] Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. Submodular subset selection for large-scale speech training data. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3311–3315. IEEE, 2014.

[54] Yikai Wu, Xingyu Zhu, Chenwei Wu, Annie Wang, and Rong Ge. Dissecting hessian: Understanding common structure of hessian in neural networks. *arXiv preprint arXiv:2010.04261*, 2020.

[55] Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *International Conference on Machine Learning (ICML)*, 2024.

[56] Sifan Yang, Yuanyu Wan, and Lijun Zhang. Online nonsubmodular optimization with delayed feedback in the bandit setting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 21992–22000, 2025.

[57] Zheng Zhang, Junxiang Wang, and Liang Zhao. Curriculum learning for graph neural networks: Which edges should we learn first. *Advances in Neural Information Processing Systems*, 36, 2024.

[58] Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. Coverage-centric coreset selection for high pruning rates. *arXiv preprint arXiv:2210.15809*, 2022.

[59] Yuwei Zhou, Zirui Pan, Xin Wang, Hong Chen, Haoyang Li, Yanwen Huang, Zhixiao Xiong, Fangzhou Xiong, Peiyang Xu, Wenwu Zhu, et al. Curbench: Curriculum learning benchmark. In *Forty-first International Conference on Machine Learning*.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims in the abstract and introduction are consistent with the technical contributions and empirical results presented in the paper. We clearly state our proposed method, theoretical foundations, and experimental validation, and these are substantiated in the body of the work without overstatement or omission.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes we have discussed the limitations of our work at specific portions of the paper, and have also added in Appendix

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The theoretical proofs are provided in the Appendix H

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The full codebase is provided along with the supplementary material

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The full codebase is provided along with the supplementary material along with running instructions commands in the Readme. Further, we test our algorithm on open source datasets only which we have cited sufficiently and have provided links in the codebase Readme file.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All specific training details are specified in the Appendix D

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report std error of most our results over 3 runs per baseline on average.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All specific training and compute details are specified in the Appendix D

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Yes we have reviewed the Code of Ethics Guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: We have not discussed any potential societal impacts (neither positive nor negative). We do hope that since we are able to show significant efficiency improvement both across different modalities (especially in LLM settings) this may be of siginificant potential impact for Large scale LLM training.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: In this work, we are not releasing any generative models or new datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Our paper's methodology is not involved in LLM usage and nor is the experimental pipeline.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Supplementary Material: Bandit Guided Submodular Curriculum for Adaptive Subset Selection

Contents

Supplementary Material: Bandit Guided Submodular Curriculum for Adaptive Subset Selection

A Organization of the Appendix

The appendix is organized as follows. Section I provides a summary of the impact of our work.Section B provides a summary of the notation used throughout the paper. Section H presents our main theoretical results. Section D outlines the experimental setup and implementation details for both vision and language model tasks. Section G discusses additional related work. Section F describes the various submodular functions employed in our experiments.

B Notation Summary

Table 4: Table of Notations

Topic	Notation	Explanation
Data (sub)Sets Indices	$\boldsymbol{\mathcal{D}}_{\texttt{train}}$	Entire Training Set consisting of n instances
	$\boldsymbol{\mathcal{D}}_{\texttt{val}}$	Entire Validation Set consisting of m instances
	$\mathbf{z}_i$	i-th training instance in a batch
	$\mathcal{B}_t$	Denotes the full sized t-th step train minibatch : $\{\boldsymbol{x}_p\}_{p=1}^{\lvert\mathcal{B}_t\rvert}$
	$\mathcal{B}_t^{\texttt{val}}$	Denotes the full sized t-th step validation minibatch
	$\mathcal{B}_t^{(<i)}$	Denotes the t-th step minibatch being constructed uptil $\boldsymbol{x}_{i-1}$ i.e. $\{\boldsymbol{x}_p\}_{p=1}^{i-1}$
	$\mathcal{S}_{(a_t)}^{\texttt{opt}}$	Optimal subset obtained when submodular function $f^{(a_t)}$ is applied
Parameters	$\boldsymbol{\theta}^*$	Optimal model parameter (vector)
	$\boldsymbol{\theta}_t$	Model parameter at t^{th} step
	$\boldsymbol{\theta}_{t+1}$	Model parameter at $(t+1)^{\text{th}}$ step
Loss Function	ℓ	Strongly convex instance-wise loss function
	$\boldsymbol{\mathcal{L}}_t$	Total loss over mini-batch
	$\mathcal{U}_t(\mathcal{B}_t; z_t^{\text{val}})$	Utility metric capturing validation loss drop for a particular validation data point
	$\mathcal{U}_t(\mathcal{B}_t; B_t^{\text{val}})$	Aggregated utility metric over validation set
Hyperparams	$\lambda(t)$	(*Exploration Dampening*) modulates the inertia of exploration
	$\boldsymbol{\vartheta}$	Reward function
	$\boldsymbol{\Xi}_t$	Exploration-exploitation threshold
	$\boldsymbol{\mathcal{F}}_{\texttt{sub}}^{\text{div}}, \boldsymbol{\mathcal{F}}_{\texttt{sub}}^{\text{repr}}$	Diversity/representative function subsets
	$\zeta \sim \text{Uniform}(0,1)$	Random sample for trade-off
	$\pi(t)$	(*Exploration Sharpness*) controls the curvature of the annealing schedule rule

C Implementation Details

C.1 Details about model architectures used

Vision Model Architecture Details:

The ResNet18 model [15] architecture begins with a *basic block*, which is composed of two main sections. The first section consists of a convolution layer followed by a batch normalization layer, and then a ReLU activation function. The second section similarly comprises a convolution layer followed by batch normalization. This entire *basic block* is repeated twice for each of the four layers in the network. These layers progress with input dimensions of $[64, 128, 256, 512]$ to form the complete ResNet18 architecture.

Language Model Architecture Details

The LLaMA-2-7B model is a decoder-only transformer comprising approximately 7 billion parameters. It includes 32 transformer layers, each built with pre-normalization using RMSNorm and employing the SwiGLU activation function. The self-attention mechanism uses multi-head causal attention with 32 heads and a hidden dimensionality of 4096. Rotary positional embeddings (RoPE) are applied to the query and key vectors within each attention head. The model begins with a learned token embedding layer and concludes with a tied output projection layer to predict the next token. Mistral-7B-v0.3 is architecturally similar to LLaMA-2-7B, also featuring 32 transformer layers and a 4096-dimensional hidden state, but introduces several efficiency-focused modifications. It uses grouped-query attention (GQA) with 8 query groups across 32 heads, improving inference throughput. Furthermore, Mistral replaces full causal attention with sliding-window attention to handle long contexts more efficiently. As with LLaMA, it utilizes RoPE for positional encoding and SwiGLU activations. These optimizations maintain strong modeling performance while enabling greater scalability in both training and inference settings.

C.2 Details on submodular functions implementation

We provide detailed formulations of the specific submodular functions employed as arms in our experiments in Section F. From an implementation standpoint, each submodular arm operates on a similarity kernel computed over the set of instances within a given batch. This kernel, typically represented as a symmetric positive semi-definite matrix, encodes pairwise affinities between samples based on their embedding representations (e.g., cosine similarity or RBF kernel). Once the similarity structure is established, any submodular function can be instantiated over this ground set—such as facility location, log-determinant, or graph-cut functions—depending on the desired coverage, diversity, or representativeness property being optimized.

To operationalize this, we leverage the `Submodlib` library[4] , an open-source framework maintained by the Decile organization[5] , which provides efficient and modular implementations of a wide family of submodular functions. The library supports both dense and sparse similarity representations and includes greedy as well as lazy-greedy optimization routines, enabling scalable computation even for large batch sizes.

C.3 Gradient Computation

Computing full-model gradients in modern deep networks is computationally prohibitive due to the extremely high dimensionality of parameter spaces—often exceeding billions of parameters for large vision or language models. Moreover, for the purpose of subset selection, what is typically required is not the full parameter gradient but an informative proxy that captures the *relative contribution* of individual samples to the model's training dynamics.

Following this motivation, we adopt *partial-gradient approximations* that preserve discriminative signal while substantially reducing computational cost. Specifically, for vision models, we compute gradients only with respect to the *last linear classification layer*, as in [3]. This choice leverages the empirical observation that gradients in earlier layers are highly correlated and redundant, and that last-layer gradients retain sufficient information to distinguish hard, redundant, or noisy samples based on their contribution to the decision boundary.

For large language models (LLMs), computing full backpropagation across all transformer layers is infeasible. We therefore restrict gradient computation to *Low-Rank Adaptation (LoRA)* adapter parameters (rank 128), following the setup of [52]. This approach not only reduces memory and compute overhead by several orders of magnitude but also captures localized curvature information relevant to the fine-tuning or instruction-following objective. Since LoRA adapters are trained in the low-dimensional subspace most sensitive to task adaptation, their gradients provide a faithful and low-noise estimate of per-sample learning signals.

Importantly, both approximations maintain *gradient informativeness* under the assumption that the selected subspace (last layer or adapter) spans the most discriminative directions of parameter updates. Prior empirical evidence (see [3, 52]) shows that subset selection, influence estimation, and sample

[4] https://submodlib.readthedocs.io/en/latest/

[5] https://decile.org/

reweighting methods computed in these reduced spaces closely match those computed with full gradients. In our experiments, we verify that this approximation incurs negligible performance degradation while providing up to $30\times$ faster per-batch computation. Thus, the proposed gradient computation scheme achieves a favorable balance between computational efficiency and fidelity of learning signal for submodular subset selection.

Warm-starting Data Selection. A common challenge in data subset selection methods lies in the instability of early-stage gradients. During the initial epochs of training, model parameters are far from any local minimum, and per-sample gradients tend to be highly noisy and uninformative. Consequently, performing subset selection too early can result in biased or suboptimal subsets that fail to represent the underlying data distribution or learning dynamics. To mitigate this issue, for the image experiments, we conduct a *warm-start* strategy, wherein the model is first trained for a small number of epochs on the full dataset before invoking any subset selection procedure.

Concretely, for each algorithm considered in this paper (i.e., ONLINESUBMOD GRADMATCH, GRADMATCHPB, CRAIG, CRAIGPB, and GLISTER,), we include a warm-start variant. Let T denote the total number of training epochs and k the subset size. We define two quantities: T_f (the number of full-training epochs prior to subset selection) and T_s (the number of epochs during which subset selection is active). We set these in proportion as $T_s = \kappa T, T_f = \frac{T_s k}{n}$, where n is the total number of training samples and $\kappa \in (0, 1]$ is the fraction of total training epochs used for subset selection. This parametrization ensures that the effective compute budget remains comparable across methods, while allowing early-stage training to stabilize the model representation before adaptive data selection begins.

Empirically, we observe that performing a few epochs of full-data warm-up (T_f) consistently improves convergence stability and downstream accuracy across all subset selection algorithms. The warm-start phase enables the gradient space to form a meaningful geometry, allowing the submodular or gradient-based selection objectives to more accurately identify informative and diverse samples. In contrast, starting selection from random initialization often leads to premature overfitting or unstable subset composition due to noisy or poorly conditioned gradients.

Setting T_f too large, however, diminishes the benefit of subset selection, as the model effectively performs full training with minimal adaptive sampling. In this limit, the behavior approaches that of the full-batch baseline with early stopping, which we include as a control setting in our experiments. Thus, the warm-start scheme provides a principled balance between computational efficiency and representational stability—retaining the benefits of subset-based training while ensuring robust and smooth convergence.

C.4 Evaluation metrics

Image Classification: For image classification experiments, we report the standard *test accuracy* as the primary performance metric, measured as the proportion of correctly classified samples on the held-out validation or test split. This metric provides a direct and interpretable indicator of the model's generalization performance under different subset selection strategies.

In addition to accuracy, we evaluate the *computational efficiency* of our method by comparing the total training time required to reach convergence across different selection policies. To ensure a fair comparison, all other hyperparameters—including optimizer configuration, learning rate schedule, batch size, and data augmentations—are held fixed across runs. The only varying factor is the subset selection mechanism applied at each training step.

We define the *speedup* metric with respect to the baseline model trained using full-batch selection (i.e., without any submodular or adaptive sampling). Formally, if T_{full} denotes the wall-clock training time for the full-batch model and T_{sub} denotes the time under our submodular selection strategy, the speedup is given by Speedup $= \frac{T_{\text{full}}}{T_{\text{sub}}}$. A higher speedup thus indicates a more efficient training regime, achieved without sacrificing downstream accuracy. In practice, we observe consistent gains in training efficiency—typically in the range of $3\times$–$8\times$—depending on the dataset and the choice of submodular objective, confirming that adaptive selection substantially reduces redundant gradient computations while maintaining comparable predictive performance.

D Experimental Setup Details

D.1 Software and Hardware

Vision Experiments All experiments were conducted using Python 3.10.13 and PyTorch 2.1.2. Our proposed methods, ONLINESUBMOD and ONLINESUBMOD-Batch, along with their corresponding ablations, were trained on NVIDIA RTX A6000 GPUs (48 GB). Baseline methods, including RHO-LOSS and BOSS, were also trained using the same GPU configuration to ensure comparability.

For reference, a typical training run of our ResNet18-based model on an RTX A6000 consists of 300 epochs, with each epoch averaging approximately one minute (excluding certain baselines). Model checkpointing is employed to retain only the best-performing model based on validation accuracy, as well as the final model. Running multiple training jobs concurrently on the same GPU incurs only a slight overhead in training time due to resource contention.

D.2 Language Model Experiments

We experiment for the LLM finetuning setup using a RANDOM subset of 9 datasets from MMLU, and on TydiQA. We choose Sociology, Policy, History, Anatomy, ML, Ehics, Genetics, High School Biology, High School Chemistry. All language model experiments, including both our proposed methods and the baselines, were conducted using 8 NVIDIA H100 GPUs. Additionally, Weights & Biases (WandB) [6]wandb was used to manage and monitor all experiments. For all experiments we take batch size of 16, initial learning rate of 2e-5 using adam optimizer with default state, finetuned on 10% of LESS[55] version of OpenWebText.

Additional Experiment Results: For MMLU we also showcase additional experiments on LLaMa2-7b and Mistral-7b for TydiQA, later in the appendix.

Mathematical definitions of the submodular objectives used as arms are provided in Appendix F. For this experiment, each arm is a mutual information variant of a classical submodular function, designed to maximize $I_f(X;Q) = f(X) + f(Q) - f(X \cup Q)$, where X is the candidate training set, Q is the validation set, and f is a base submodular function (Facility-Location, Graph-Cut, or Log-Determinant).

We use mutual information forms to ensure the selected subset is explicitly conditioned on the current validation set, making the acquisition process adaptive to the downstream task. Features for X and Q are derived either from Sentence-BERT embeddings or from gradient vectors, with the latter shown to yield better alignment with task-specific error signals and improved selection performance.

D.3 Vision Model Experiments

The experimental setup was configured to evaluate the proposed method on several datasets, including CIFAR-10, CIFAR-100, Tiny-ImageNet-200, and SVHN. The data module used a batch size of 128, with four workers for data loading. The model architecture employed was ResNet18 [15], and the training followed a curriculum-based mode, progressively utilizing 10%, 30%, and 50% of the training data. The optimizer used was SGD with a learning rate of 0.05, momentum of 0.9, weight decay of 0.0005, and Nesterov momentum enabled.

For all our settings (across different baselines and dataset), we consider ResNet18 [15] as our primary model with the following architecture and training details:

In our training setup, we employed batch-wise Nesterov accelerated gradient descent with a batch size of 128. The optimization configuration included a learning rate of 0.05 and a momentum of 0.9, alongside a cosine-annealing scheduler.

Across all dataset comparisons, we set the submodular function budget β to 10%, 30%, and 50% of the entire batch size.

Dataset Specifics. We conduct experiments across a range of standard vision benchmarks. For the **MNIST** dataset, we use 60,000 training instances, 10,000 test instances, and 10,000 validation instances, with training proceeding until full convergence, typically around **200 epochs**. On CIFAR-

[6] https://wandb.ai/site/

10, we use 50,000 training instances, 10,000 test instances, and 10,000 validation instances, with models trained for up to **300 epochs**. For **CIFAR-100**, we similarly use 50,000 training examples spread across 100 classes (500 per class), and a validation set of 10,000 examples (100 per class). The **SVHN** dataset comprises 73,257 training images across 10 classes with variable class frequencies, and a validation set of 26,032 images distributed proportionally. Finally, for **TinyImageNet**, we use 100,000 training images across 200 classes (500 per class), and a validation set of 10,000 images (50 per class), covering the same label space as the training data.

D.4 Baseline Training Details

We compare our method OnlineSubmod with several state of the art baselines for our experiments:

Max-Loss [27]: Within each training batch, the loss is computed for every example. A fixed fraction (e.g., top-K%) of samples with the highest per-example loss is selected for gradient computation and model update. This assumes that high-loss samples are currently mis-predicted and could contribute the most to updating the decision boundary.

GradNorm [17]: The l_2 norm of each example's per-sample gradient i.e. $\|\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{z}; \boldsymbol{\theta})\|_2$ is computed, and a subset with the highest norms is selected for each batch. This prioritizes examples inducing the largest parameter updates under the current model, helping direct learning toward sensitive or uncertain regions.

Rho-Loss [30]: Each example's *reducible loss* is estimated as the difference between the current model's loss and its *irreducible loss*, the latter approximated by a small auxiliary model trained on held-out clean data. Examples with high reducible loss are selected, as they are considered learnable but not yet learned, making them useful for continued training. For LLM experiments, we use `LLaMa3-7b-instruct` as our auxiliary model. For image experiments, we begin by training an irreducible model on the specific task for 100 epochs. Subsequently, we precompute the irreducible losses for the training set, which are required for the target model training. During the target model training phase, we train the model for 300 epochs across the CIFAR-10, CIFAR-100, SVHN, and TinyImageNet datasets, using subset ratios of 0.1, 0.3, and 0.5. We employ the ResNet-18 architecture for both the irreducible model and the target model. For training, we use the SGD optimizer with Nesterov accelerated gradient descent, a batch size of 128, and the following configuration: a learning rate of 0.05, momentum of 0.9, and a cosine-annealing scheduler. One observation we made is that RHO Loss converges to reasonably good accuracy within a few epochs, but further training does not significantly improve performance, and it fails to reach the accuracy levels of other state-of-the-art (SOTA) baselines.

SBert [40]: In this case, each training and validation example is encoded into a sentence embedding using a pre-trained SBERT model. Cosine similarity is computed between training examples and the validation set, and those with the highest average similarity are selected. This favors examples that align semantically with the validation distribution.

For **GREATS** [52], each example's impact on the loss is approximated using a first-order Taylor expansion of the objective. For model parameters θ and a batch $\{x_1, \ldots, x_n\}$, gradients $\nabla \mathcal{L}(x_i; \theta)$ are used to estimate loss reduction. A greedy selection strategy then chooses the subset expected to most decrease validation loss under this approximation.

For fair comparison against our model we considered the configuration where subset selection happens at every epoch for all the 3 baselines with a *lazy* optimizer. Due to our multi-class image classification setup we utilise *CrossEntropy* loss for our model training.

BOSS [1]: For BOSS, to select the subset, we first initialized a model by training it using the full dataset. With the help of the training dynamics obtained from the initialized model, we calculated the difficulty score for each sample that is used to select the subset. We evaluated the selected subset keeping the subset fixed and using it to train a new RANDOM initialized model. For the difficulty score, we experimented using the EL2N score because it can be efficiently computed early on during training.We trained the model for 300 epochs across the CIFAR-10, CIFAR-100, SVHN and TinyImageNet datasets, using subset ratios of 0.1, 0.3, and 0.5. We employed ResNet18 model using SGD with a learning rate of 0.1, and momentum of 0.9 with a batch size of 128.

D.5 Comparison between DINO and Gradient-Based Features for Submodular Selection

To evaluate how closely our feature representations must align with the downstream objective, we compared two ways of representing each training item when optimising submodular acquisition functions (and their mutual–information variants):

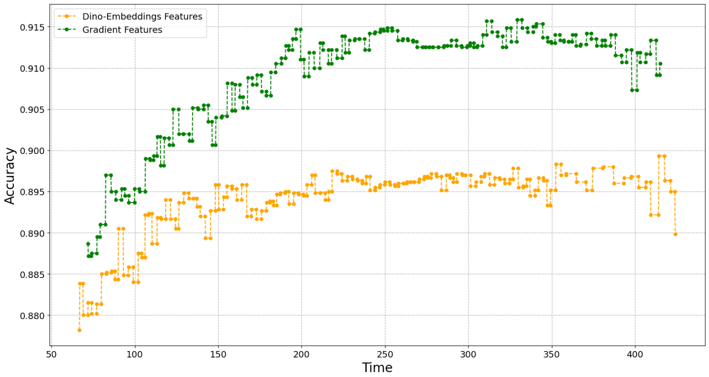

Figure 7: Comparison of Fashion-MNIST with DINO-embeddings, and with Gradient Features for submodular optimization

1. **DINO embeddings**. We obtain a fixed d-dimensional feature vector for every image by running it through a frozen DINO vision transformer, exactly as one would use a CLIP encoder. These representations are task–agnostic and remain static throughout training.
2. **Gradient-based features**. At every training step we compute the gradient of the scalar loss with respect to the parameters of the final layer. We average these per-example gradients within the mini-batch to form a single vector[7]. For mutual–information objectives, validation gradients serve as the query features.

Figure 7 shows that gradient features yield substantially higher test accuracy on FASHION-MNIST across all subset sizes: they encode task-specific error signals that guide the submodular optimiser toward examples most useful for loss reduction, whereas DINO embeddings capture only generic visual similarity. Hence, directly leveraging gradients as features is the more effective choice for data subset selection in this setting.

D.6 Fisher Information Matrix

Fisher Information Matrix Approximation An alternative and potentially more informative approach to approximating the Hessian is through the use of the Fisher Information Matrix (FIM) [10]. The FIM provides insights into the curvature of the loss landscape and can serve as a useful surrogate for the Hessian. While the exact computation of the FIM requires calculating an expectation, which can be computationally intensive, it can be efficiently approximated using an exponential moving average of the outer product of the gradients from the validation data points.

Let $\boldsymbol{\Omega}_i := \boldsymbol{g}(\mathbf{z}_i, \boldsymbol{\theta}_t)\boldsymbol{g}(\mathbf{z}_i, \boldsymbol{\theta}_t)^\top$ denote the outer product of the gradient for the ith data point in the current batch $\mathcal{B}_t$. The approximate FIM $\widehat{\boldsymbol{\mathcal{H}}}^{(t)}_{\mathcal{B}_t}$ at time step t for the current mini-batch $\mathcal{B}_t$ can be computed recursively as

$$\widehat{\boldsymbol{\mathcal{H}}}^{(t)}_{\mathcal{B}_t} = \begin{cases} \frac{1}{|\mathcal{B}_0|}\sum_{\mathbf{z}_i \in \mathcal{B}_0} \boldsymbol{\Omega}_i, & \text{if } t = 0 \\ (1-\alpha)\widehat{\boldsymbol{\mathcal{H}}}^{(t-1)}_{\mathcal{B}_{t-1}} + \\ \alpha \frac{1}{|\mathcal{B}_t|}\sum_{\mathbf{z}_i \in \mathcal{B}_t} \boldsymbol{\Omega}_i, & \text{else} \end{cases} \tag{9}$$

where $\widehat{\boldsymbol{\mathcal{H}}}^{(t-1)}_{\mathcal{B}_{t-1}}$ is the approximate FIM at the previous time step $t-1$ for the mini-batch $\mathcal{B}_{t-1}$, and $\alpha \in (0, 1]$ is the smoothing parameter for the exponential moving average. This recursive formulation provides a computationally efficient approach to approximating the Hessian, particularly in high-dimensional settings where direct Hessian computation is prohibitively expensive.

[7] Using the batch-wise average was consistently superior to concatenating per-example gradients, and last-layer gradients are sufficient while keeping the computation inexpensive.

E Additional Experiments

E.1 Sensitivity of Validation Dataset Configuration

To better understand the influence of validation data composition on model performance, we investigate how sensitive the algorithm is to different validation set configurations. In particular, we examine what occurs during the early stages of training when the validation dataset includes harder samples that the model has not yet adequately learnt. This analysis provides a deeper perspective on how the distribution and difficulty of validation examples can affect optimization dynamics and generalization, thereby strengthening the empirical validity of our findings.

Specifically, we aim to understand the following questions:

- **Q1:** To what extent is the algorithm's performance sensitive to the configuration and composition of the validation dataset?
- **Q2:** How does the presence of harder, yet-unlearned samples in the validation set during early training stages affect convergence and generalization?

To better understand this issue, we conducted a controlled experiment on CIFAR-100 (300 epochs) where we varied the hardness of the validation dataset. Hardness was measured via gradient norm where the gradient is calculated w.r.t model parameter at that time step. In accordance with other literature, a crude way to approximate difficulty of a sample is to check if the gradient norm is high. (higher gradient norm $\sim$ harder example). We compared four validation subset configurations:

Validation Subset	10%	20%	30%
Easiest	72.3	74.4	76.03
EasyHard	73.1	74.5	76.4
HardEasy	72.29	74.6	76.2
Hardest	71.31	74.3	75.9

Table 5: Final test accuracies under different validation subset configurations.

- **Easiest:** Lowest gradient norms
- **EasyHard:** Easy samples early, hard samples later
- **HardEasy:** Hard samples early, easy samples later
- **Hardest:** Highest gradient norms

Each configuration was evaluated at validation subset sizes of 10%, 20%, and 30%.

Observations:
Validation sets composed of the most difficult examples tend to yield lower performance, particularly when smaller subsets are used. This decline likely stems from noisy or overly pessimistic reward signals during the early stages of training. In contrast, mixed validation configurations such as **EasyHard** and **HardEasy** generally perform best, indicating that a balanced distribution of sample difficulty across training can enhance robustness. These findings suggest that further exploring how validation sample difficulty and ordering interact especially through the lens of curriculum learning could be a promising direction for future work. Importantly, even validation sets containing difficult samples early in training do not lead to instability or model collapse.

E.2 Effect of Submodular Functions Individually and RANDOM Selection over Arms

To assess the contribution of the multi-armed bandit formulation in our framework, we perform ablation experiments on CIFAR-100 (10% subset, 300 epochs) under two simplified settings: (a) using a single, fixed submodular arm throughout training (i.e., no bandit-driven adaptation), and (b)RANDOMly selecting an arm at each round (i.e., no explore–exploit balancing). These ablations isolate the effect of static versus dynamic subset selection policies on training efficiency and generalization.

We compare various submodular selection strategies that define the reward structure of the curriculum. *Representative* functions (e.g., **GraphCut**, **FacilityLocation**) promote coverage and ensure the selected subset reflects the global data distribution, while *Diversity*-oriented functions (e.g., **DisparitySum**, **LogDeterminant**) encourage maximal dissimilarity among chosen samples. These functions capture different inductive biases: representation versus decorrelation.

Selection Strategy	**Accuracy (%)**
DisparitySum (Div., Static)	68.6
FacilityLocation (Rep., Static)	72.0
LogDeterminant (Div., Static)	71.1
GraphCut (Rep., Static)	72.6
Random arm per round	72.0
ONLINESUBMOD (ours)	**73.6**

Table 6: Performance comparison of individual and RANDOM arm selection strategies on CIFAR-100. ONLINESUBMOD adaptively balances diversity and representativeness over training epochs.

Our proposed **ONLINESUBMOD** method dynamically alternates between these functions through an adaptive explore–exploit policy governed by the bandit controller. This dynamic weighting enables the model—whether a **ResNet-18** backbone or a small **LLM fine-tuning setup**—to exploit high-yield submodular arms while continually exploring others that may improve validation loss or perplexity.

Observations:
Different submodular functions show complementary but limited strengths. Coverage-based methods such as *GraphCut* and *FacilityLocation* converge quickly early in training, while diversity-based ones like *DisparitySum* and *LogDet* encourage better generalization but can become unstable when applied uniformly. Random arm selection gives reasonable results, suggesting that diversity matters, but it lacks feedback to adapt to validation performance. In contrast, the proposed **ONLINESUBMOD** approach adjusts arm selection based on past rewards, maintaining a stable balance between exploration and exploitation. This supports our main hypothesis that adaptive, reward-driven selection leads to more robust and generalizable outcomes than static or random strategies.

E.3 Additional Experiments on Vision datasets

Table 7 summarizes batchwise data selection results across multiple vision datasets. Across all budgets and datasets, ONLINESUBMOD consistently achieves the highest test accuracy while maintaining competitive or lower training time compared to prior methods. Notably, it surpasses strong baselines such as GRADMATCH, MILO, and RHO-LOSS, particularly at low data budgets (10%–30%), indicating superior sample efficiency and adaptivity under constrained training regimes. The improvement is most pronounced on CIFAR100 and TINYIMAGENET, where the model benefits from dynamic online selection over diverse feature manifolds. In contrast, static coreset-based methods (e.g., CRAIG, GLISTER) exhibit slower convergence and lower performance as the budget increases.

Here, red highlights the best result and blue denotes the second-best result for each setting. Overall, these results confirm that ONLINESUBMOD provides a strong trade-off between accuracy and computational efficiency across datasets of varying complexity.

Table 7: (Batchwise) Data Selection Results on Vision Datasets

Dataset	**Selection Strategy**	**Test accuracy (%)**			**Training time (hrs)**		
	Budget(%)	10%	30%	50%	10%	30%	50%
CIFAR10	FULL (skyline for test accuracy)	95.09	95.09	95.09	1.73	1.73	1.73
	RANDOM (skyline for training time)	77.49	89.62	91.85	0.29	0.75	0.85
	CRAIG	90.07	92.4	93.12	0.26	0.62	1.54
	GLISTER	91.15	92.18	92.65	0.38	1.05	1.34
	GRADMATCH	92.27	93.28	93.15	0.42	0.95	1.21
	MILO	92.25	93.21	94.16	0.34	0.85	0.89
	RHO-LOSS	90.16	91.54	94.03	0.76	1.13	1.54
	BOSS	91.64	93.04	93.8	0.36	0.94	1.18
	ONLINESUBMOD (ours)	92.44	93.75	94.18	0.32	0.87	0.83
CIFAR100	FULL (skyline for test accuracy)	76.8	76.8	76.8	1.52	1.52	1.52
	RANDOM (skyline for training time)	35.03	61.93	64.67	0.15	0.42	0.78
	CRAIG	67.25	72.38	73.12	0.31	0.62	1.12
	GLISTER	64.27	72.36	74.62	0.26	0.57	1.3
	GRADMATCH	68.34	74.63	72.36	0.22	0.48	1.22
	MILO	72.36	74.66	75.60	0.15	0.44	0.82
	RHO-LOSS	71.37	74.82	75.74	0.53	0.86	1.46
	BOSS	71.73	73.77	75.41	0.27	0.53	0.85
	ONLINESUBMOD (ours)	73.67	75.46	75.78	0.165	0.47	0.82
SVHN	FULL (skyline for test accuracy)	96.49	96.49	96.49	6.436	6.436	6.436
	RANDOM (skyline for training time)	93.47	95.31	95.84	0.6383	1.90	3.19
	CRAIG	95.27	96.15	96.40	0.934	2.332	4.17
	GLISTER	95.52	95.69	96.42	0.83	2.42	4.26
	GRADMATCH	95.64	96.4	96.42	0.789	2.398	4.19
	MILO	95.62	96.36	96.41	0.69	2.09	3.25
	RHO-LOSS	94.64	94.27	94.85	1.08	2.56	3.94
	BOSS	94.31	95.75	96.01	0.76	2.39	3.56
	ONLINESUBMOD (ours)	95.68	96.38	96.46	0.68	2.12	3.28
TINYIMAGENET	FULL (skyline for test accuracy)	64.36	64.36	64.36	15.4	15.4	15.4
	RANDOM (skyline for training time)	19.61	35.68	43.84	1.82	4.92	6.12
	CRAIG	52.42	55.56	61.48	3.27	6.46	9.23
	GLISTER	51.54	56.37	62.15	2.84	5.93	9.47
	GRADMATCH	52.63	58.19	61.93	2.63	5.94	7.24
	MILO	53.24	59.36	62.28	1.81	4.97	6.16
	RHO-LOSS	54.46	59.78	62.15	3.16	6.38	7.94
	BOSS	52.63	60.17	62.13	2.85	5.47	7.26
	ONLINESUBMOD (ours)	55.3	60.74	62.58	1.84	5.16	6.14

E.4 Additional Experiments on Large Language Models

We evaluate the evolution of test perplexity during pretraining on the MMLU benchmark using the LLAMA-2-7B model under different **online batch selection strategies**. Each method is trained under identical hyperparameter and compute budgets to ensure fair comparison.

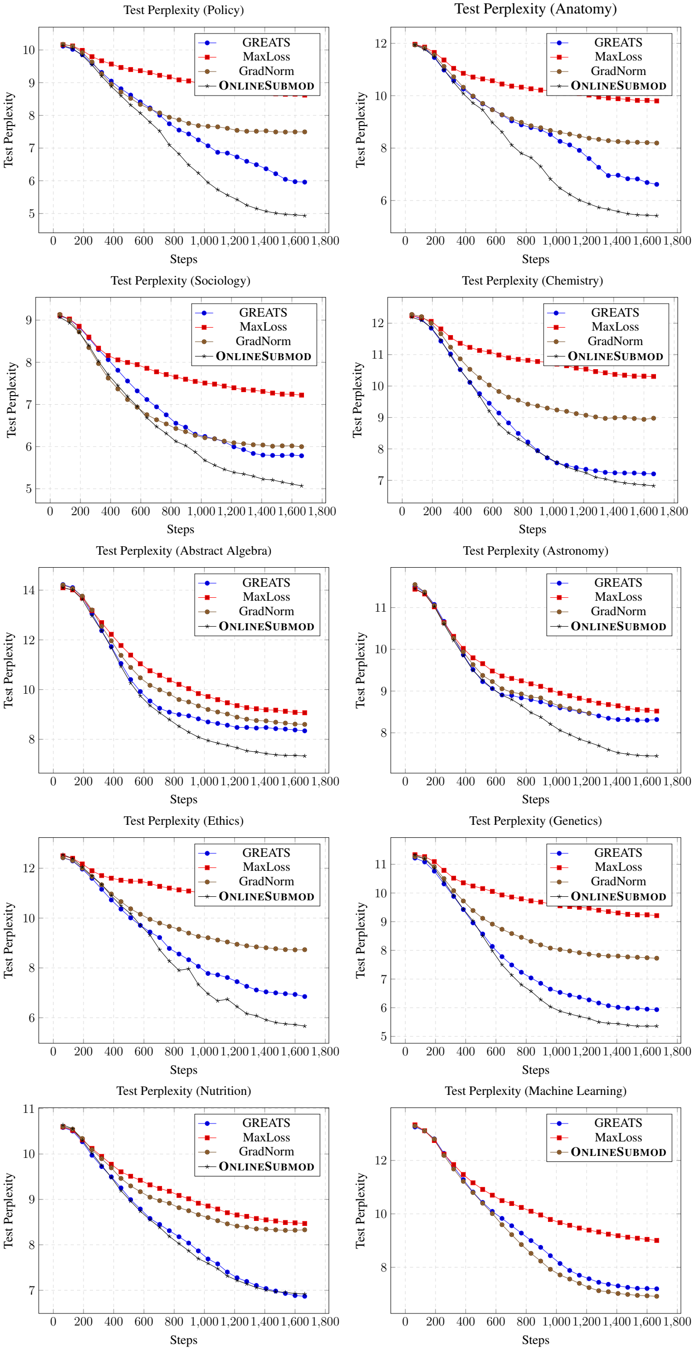

Figure 8: **Test perplexity dynamics** on LLAMA-2-7B during training with various **online batch selection strategies** on MMLU. ONLINESUBMOD significantly outperforms baselines.

F Details of Submodular Function used in all our training settings

We describe here the submodular functions we broadly utilised for all our experiments.

F.1 Diversity based Submodular Function

Here we share the details on the diversity based submodular functions we used for our training purposes.

Definition 1. ***Log-determinant Function*** *is a diversity-based submodular function. It is non-monotone in nature. Let* $\mathbf{L}$ *denote a positive semidefinite kernel matrix and* $\mathbf{L_S}$ *denote the subset of rows and columns indexed by set* $\mathbf{S}$*. Log-determinant function* f *is specified as:*

$$f(\mathbf{S}) = \textit{logdet}(\mathbf{L_S}) \tag{10}$$

The log-det function models diversity and is closely related to a determinantal point process.

Definition 2. ***Disparity Sum Function*** *characterizes diversity by considering the sum of distances between every pair of points in a subset* $\mathbf{S}$*. For any two points* $i, j \in \mathbf{S}$*, let* d_{ij} *denote the distance between them.*

$$f(\mathbf{S}) = \sum_{i,j \in \mathbf{S}} d_{ij} \tag{11}$$

The aim is to select a subset $\mathbf{S}$ *such that* $f(\mathbf{S})$ *is maximized.*

Definition 3. ***Disparity Min Function*** *characterizes diversity by considering the minimum distance between any two non-similar points in a subset* $\mathbf{S}$*.*

$$f(\mathbf{S}) = \min_{i,j \in \mathbf{S}, i \neq j} d_{ij} \tag{12}$$

The aim is to select a subset $\mathbf{S}$ *such that* $f(\mathbf{S})$ *is maximized.*

F.2 Representative based Submodular Function

Here we share the details on the representative based submodular functions we used for our training purposes.

Definition 4. ***Facility Location Function*** *characterizes the representativeness in the dataset by considering the minimum distance between any two non-similar points in a subset* $\mathbf{S}$*.*

$$f(\mathbf{S}) = \sum_{i \in \mathcal{V}} \max_{j \in \mathbf{S}} d_{ij} \tag{13}$$

The aim is to select a subset $\mathbf{S}$ *such that* $f(\mathbf{S})$ *is maximized.*

Definition 5. ***Graph Cut Function*** *characterizes representativeness by using the parameter* λ *which governs the tradeoff between representation and diversity. When* λ *becomes large, graph cut function also tries to model diversity in the subset.* $\mathbf{S}$.

$$f(\mathbf{S}) = \sum_{i \in \mathcal{V}, j \in \mathbf{S}} d_{ij} - \lambda \sum_{i,j \in \mathbf{S}} d_{ij} \tag{14}$$

The aim is to select a subset $\mathbf{S}$ *such that* $f(\mathbf{S})$ *is maximized.*

Submodular Mutual Information We first provide a definition of Submodular Mutual Information:

$$\mathcal{I}_f(\mathcal{A}; \mathcal{B}) = f(\mathcal{A}) + f(\mathcal{B}) - f(\mathcal{A} \cup \mathcal{B})$$

Definition 6. ***Log-Determinant Mutual Information Function*** *is an instantiation of a submodular mutual information function using a* `LogDeterminantFunction`*. Let* $S_{A,B}$ *be the cross-similarity matrix between the items in sets* A *and* B*. Also, denote* $S_{AB} = S_{A \cup B}$*. We construct a similarity matrix* S^η *(on a base matrix* S*) such that the cross-similarity between* A *and* Q *is multiplied by* η *(i.e.,* $S^\eta_{A,Q} = \eta S_{A,Q}$*) to control the trade-off between query relevance and diversity. Higher values of* η *ensure greater query-relevance while lower values favor diversity. Using a similarity matrix defined above and with* $f(A) = \log\det(S^\eta_A)$*, we have:*

$$I_f(A; Q) = \log\det(S_A) - \log\det(S_A - \eta^2 S_{A,Q} S_Q^{-1} S_{A,Q}^T) \tag{15}$$

Definition 7 (**Generalized Submodular Mutual Information**)**.** *Let Ω be a ground set and $\mathcal{V} \subseteq \Omega$ be a domain of interest. Let $f : 2^{\Omega} \to \mathbb{R}_{\geq 0}$ be a* restricted submodular function*, i.e., submodular when restricted to subsets of $\mathcal{V}$. A* Submodular Mutual Information (SMI) *function defined via such a function f is called a* Generalized Submodular Mutual Information (GMI) *function.*

A notable instance of GMI is the Concave Over Modular (COM) *function [21], defined for subsets $\mathcal{A} \subseteq \mathcal{V}$ and $\mathcal{Q} \subseteq \mathcal{V}'$ as:*

$$I_{f_\eta}(\mathcal{A}; \mathcal{Q}) = \eta \sum_{i \in \mathcal{A}} \psi \left(\sum_{j \in \mathcal{Q}} s_{ij} \right) + \sum_{j \in \mathcal{Q}} \psi \left(\sum_{i \in \mathcal{A}} s_{ij} \right),$$

where $\eta \in \mathbb{R}_{\geq 0}$ controls the trade-off between query-relevance and diversity, $\psi : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a concave function,$S = [s_{ij}]$ is a kernel similarity matrix such that $s_{ij} = \mathbb{1}(i = j)$ for $i, j \in \mathcal{V}$ or $i, j \in \mathcal{V}'$.

Definition 8 (Facility Location Mutual Information (FL1MI))**.** *Let $I_f(\mathcal{A}; \mathcal{Q})$ denote a* Submodular Mutual Information (SMI) *function. An instantiation of SMI using the* `FacilityLocationFunction` *is known as the* Facility Location Mutual Information (FL1MI) *function.*

Formally, given subsets $\mathcal{A} \subseteq \mathcal{V}$ and $\mathcal{Q} \subseteq \mathcal{V}'$, FL1MI is defined as:

$$I_f(\mathcal{A}; \mathcal{Q}) = \sum_{i \in \mathcal{V}} \min \left(\max_{j \in \mathcal{A}} s_{ij},\ \eta \max_{j \in \mathcal{Q}} s_{ij} \right),$$

where: $\eta \in \mathbb{R}_{\geq 0}$ is a relevance-diversity trade-off parameter, s_{ij} denotes similarity between elements i and j in the kernel similarity matrix S, $\mathcal{V}$ is the candidate set and $\mathcal{V}'$ is the query set domain.

Definition 9 (Graph Cut Mutual Information (GCMI))**.** *Let $I_f(\mathcal{A}; \mathcal{Q})$ denote a* Submodular Mutual Information (SMI) *function. An instantiation of SMI using the* `GraphCutFunction` *is called the* Graph Cut Mutual Information (GCMI) *function.*

Formally, for subsets $\mathcal{A} \subseteq \mathcal{V}$ and $\mathcal{Q} \subseteq \mathcal{V}'$, GCMI is defined as:

$$I_f(\mathcal{A}; \mathcal{Q}) = 2\lambda \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{Q}} s_{ij},$$

where $\lambda \in \mathbb{R}_{\geq 0}$ controls the scale of mutual information,s_{ij} denotes the similarity between elements i and j in the kernel similarity matrix S,$\mathcal{V}$ is the candidate set and $\mathcal{V}'$ is the query set domain.

G Additional Related Work

G.1 Online Submodular Maximization

A growing body of research has advanced our understanding of online submodular maximization under diverse feedback models and constraint classes. A notable contribution is the recent work on [13], which introduces a principled framework leveraging first-order regret bounds from online linear optimization to derive improved guarantees in submodular settings. At each round t, the algorithm selects a feasible set $S_t \in \mathcal{C} \subseteq 2^V$, where $\mathcal{C}$ encodes combinatorial constraints such as matroids or cardinality bounds, and observes an adversarially chosen submodular function f_t. For monotone submodular functions under matroid constraints, the method achieves a $(1 - c/e - \epsilon)$-approximate regret bound of $\mathcal{O}(kT \log(n/k))$, improving on earlier results by Streeter and Golovin [45], and Golovin and Krause [11], even in the absence of curvature (i.e., $c = 1$). For non-monotone unconstrained submodular functions, a novel algorithm based on Blackwell approachability achieves a $1/2$-regret of $\mathcal{O}(n\sqrt{T})$, extending Roughgarden and Wang [41].

These developments complement recent advances in bandit and semi-bandit feedback models, including those by Hassani et al. [14] and [47], who analyze online submodular optimization in stochastic and adversarial environments, obtaining nearly minimax optimal regret bounds. Related efforts have also explored limited feedback settings with structure-aware exploration strategies (e.g., combinatorial Thompson sampling or optimism-based approaches), enhancing sample efficiency in large-scale decision spaces.

In the full-information setting, the online continuous greedy algorithm of Bian et al. [5] offers near-optimal $(1 - 1/e)$-regret for monotone submodular functions under matroid constraints, while extensions [56] tackle delayed feedback. Meanwhile, online versions of lazy greedy [31] and distributionally robust submodular maximization [44] have enabled scalable implementations in real-world domains such as streaming recommendation and dataset summarization.

Although submodular maximization is NP-hard in general, approximation algorithms often yield near-optimal performance in practice across diverse applications [45, 11]. These include influence maximization, budget-constrained recommendation, and online resource allocation, all of which benefit from the expressive yet structured nature of submodular objectives. As such, the aforementioned theoretical advances not only deepen our algorithmic understanding but also broaden the applicability of online submodular optimization frameworks to practical domains involving limited feedback, combinatorial constraints, and dynamic inputs.

G.2 Pruning Mechanisms

Several recent works have explored data pruning and subset selection for efficient training, including D2Pruning [28], InfoMax [48], and CCS [58]. While these methods offer valuable insights into coreset selection and dataset reduction, they predominantly operate in static, full-dataset settings, in contrast to our dynamic, batch-level framework.

InfoMax formulates an objective that can be interpreted as a reformulation of the Graph Cut function, which is monotone submodular, and leverages similarity kernels such as DINO embeddings (for image tasks) or gradient-based features. This aligns conceptually with our approach, where Graph Cut is explicitly implemented as a bandit arm. However, InfoMax selects samples over the entire dataset in a static manner, whereas our framework is modular and dynamic: an InfoMax-like objective can be treated as a bandit arm and applied in batch-level pruning during training. This flexibility enables more scalable deployment in real-world training pipelines where adaptive, online selection is crucial.

D2Pruning frames data subset selection as a subgraph pruning task over the dataset, representing data points as nodes in a similarity graph. Selection is performed using message passing algorithms, which can be computationally intensive and challenging to scale to large datasets. Like InfoMax, D2Pruning operates at a static, dataset-wide level, making it less suitable for dynamic batch-level pruning.

Similarly, CCS addresses static selection by optimizing for both coverage and diversity. A key contribution of CCS is its theoretical characterization of the pruning budget, identifying thresholds beyond which accuracy degradation becomes catastrophic. While our method does not primarily operate at the full-dataset level, we note that analogous effects could, in principle, occur in batch-level selection; investigating such phenomena is a potential avenue for future work.

In summary, although InfoMax, D2Pruning, and CCS provide important foundations for data pruning and coreset strategies, they are largely static and dataset-wide in nature. Our approach extends these ideas to a dynamic, scalable setting by leveraging a bandit-driven curriculum where batch-level pruning decisions are guided by validation performance. Moreover, the modularity of our framework allows for seamless integration of alternative reward signals, such as forgetting scores or other criteria discussed in prior works, enabling flexible and adaptive training in large-scale environments.

H Main Theoretical Results

H.1 Proof for permutation invariance of Expected Marginal Gain

Lemma 1 (**Permutation Invariance of Expected Marginal Gain**). *Let* Π *denote the set of all permutations over the elements of* $\mathcal{B}_t^{(<i)}$. *Then the expected marginal gain* $\mathbb{E}_{\mathbf{z}_i \in \mathcal{B}_t^{(<i)}}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)}, \mathbf{z}_t^{\mathsf{val}})\right]$ *is invariant under any permutation* $\pi \in \Pi$, *i.e.,*

$$\mathbb{E}_{\mathbf{z}_i \in \mathcal{B}_t^{(<i)}}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \mathcal{B}_t^{(<i)}, \mathbf{z}_t^{\mathsf{val}})\right] = \mathbb{E}_{\mathbf{z}_i \in \pi(\mathcal{B}_t^{(<i)})}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \pi(\mathcal{B}_t^{(<i)}), \mathbf{z}_t^{\mathsf{val}})\right].$$

Proof. Let $S = \mathcal{B}_t^{(<i)}$, with $|S| = n$, and let $\mathbf{z}_{\mathsf{val}} = \mathbf{z}_{\mathsf{val}}^t$. We denote $\boldsymbol{g}_i := \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_i)$, $\boldsymbol{g}_v := \boldsymbol{g}_{\boldsymbol{\theta}_t}(\mathbf{z}_{\mathsf{val}})$, and $\boldsymbol{\mathcal{H}}_v := \boldsymbol{\mathcal{H}}_{\mathbf{z}_{\mathsf{val}}}(\boldsymbol{\theta}_t)$.
Then the expected marginal gain as per Eq 4 is given by:

$$\frac{1}{n}\sum_{\mathbf{z}_i \in S}\left[\eta_t\, \boldsymbol{g}_i \cdot \boldsymbol{g}_v - \eta_t^2\, \boldsymbol{g}_i^\top \boldsymbol{\mathcal{H}}_v\left(\frac{1}{n}\sum_{\mathbf{z} \in S}\boldsymbol{g}_{\mathbf{z}}\right)\right].$$

Let $\bar{\boldsymbol{g}} := \frac{1}{n}\sum_{\mathbf{z}\in S}\boldsymbol{g}_{\mathbf{z}}$. Then:

$$= \eta_t\, \bar{\boldsymbol{g}} \cdot \boldsymbol{g}_v - \eta_t^2\left(\frac{1}{n}\sum_{\mathbf{z}_i \in S}\boldsymbol{g}_i^\top \boldsymbol{\mathcal{H}}_v \bar{\boldsymbol{g}}\right) = \eta_t\, \bar{\boldsymbol{g}} \cdot \boldsymbol{g}_v - \eta_t^2\, \bar{\boldsymbol{g}}^\top \boldsymbol{\mathcal{H}}_v \bar{\boldsymbol{g}}.$$

This expression depends only on the multiset S, not the order of its elements. Therefore, for any permutation $\pi(S)$, the same value holds:

$$\mathbb{E}_{\mathbf{z}_i \in \pi(S)}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \pi(S), \mathbf{z}_{\mathsf{val}})\right] = \eta_t\, \bar{\boldsymbol{g}} \cdot \boldsymbol{g}_v - \eta_t^2\, \bar{\boldsymbol{g}}^\top \boldsymbol{\mathcal{H}}_v \bar{\boldsymbol{g}}.$$

Hence,

$$\mathbb{E}_{\mathbf{z}_i \in S}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid S, \mathbf{z}_{\mathsf{val}})\right] = \mathbb{E}_{\mathbf{z}_i \in \pi(S)}\left[\boldsymbol{\Delta}\mathcal{U}_t(\mathbf{z}_i \mid \pi(S), \mathbf{z}_{\mathsf{val}})\right],$$

which proves the claim. □

H.2 Theorem: Capacity-Controlled Risk Convergence Theorem

Theorem 2 (**Capacity-Controlled Risk Convergence**). *[([12]) **Theorem 16.3**] Let* $\mathcal{M}_{\boldsymbol{\Theta}}$ *be a neural network with* $\mathbf{d}$ *parameters belonging to the parameter space* $\boldsymbol{\Theta}$ *with an objective to minimize the empirical risk over the training data,* $\boldsymbol{\mathcal{D}} = \{(\mathcal{X}_i, \mathcal{Y}_i)\}_{i=1}^n$ *where* $\mathcal{X}_i \in \mathbb{R}^m$ *and* $\mathcal{Y}$ *are almost surely bounded and where* $\mathcal{Y}_i = \boldsymbol{\vartheta}(\mathbf{z}_i) \sim \mathcal{N}(\mu_{\mathbf{z}_i}, \sigma_{\mathbf{z}_i})$ *where* $\boldsymbol{\vartheta} : \mathbb{R}^m \to \mathbb{R}$ *and* $\mathbb{P}$ *denotes the data distribution. Then for* $\mathbf{d}$ *large enough, we have the following, for any* $\mathfrak{c} > 0$.

$$\mathbb{E}_{\boldsymbol{\mathcal{D}}}\int_{\mathbf{z}}\left\|\mathcal{M}_{\boldsymbol{\Theta}}(\mathbf{z}) - \mathbb{E}\left[\boldsymbol{\vartheta}(\mathbf{z})\right]\right\|^2 \partial\mathbb{P}(\mathbf{z}) \le \mathfrak{c}\sqrt{\frac{\ln(\boldsymbol{d})}{\boldsymbol{d}}} \tag{16}$$

The above theorem follows from [39] and [12] and is useful to prove further bounds in our case as below.

H.3 No-Regret Bounds under Constant $\lambda(\cdot)$

We first state here the main theorem under the following assumptions as stated in our main text:

Let $\boldsymbol{\tau}_{(a)}^R(t)$ denote the number of times the a-th submodular function $\boldsymbol{f}^{(a)}$ is chosen in the first $t-1$ steps by the uniform branch of the algorithm.

Assumption (a) (**Constant Fractional Exploration Dampening**): The exploration dampening parameter $\lambda(t)$ is time-invariant $\lambda(t) = \epsilon$ where $\epsilon \in (0, 1)$.

Assumption (b) (**Optimality Gap**): There exists an optimality gap $\boldsymbol{\varrho}$ such that for every suboptimal arm $a_t \in \mathscr{A} \setminus \{a^*\} : 0 \le \boldsymbol{\varrho} \le \boldsymbol{\Delta}_{(a_t)}(\mathcal{B}_t)$.

Assumption (c) (**Fractional Exploration Sharpness**): The exploration sharpness parameter $\pi(t)$ is a bounded quantity $\pi(t) \in (0, 1)$.

Assumption (d) (**Utility Metric Approximation**): The utility metric $\mathcal{U}_t(\cdot, \cdot)$ satisfies the approximation bound as per Theorem 2 (Appendix) with constants $\mathfrak{C}_{(a)}$ for each arm $a \in \mathscr{A}$ and let n_a be a specific constant associated with arm a such that Theorem 2 (Appendix) holds true.

Theorem 1 (**Regret Guarantees**). *Under Assumptions* ***a - d****, for all* $t > t_0$*, with probability at least*

$$1 - \mathcal{K} \exp\left(-\frac{3(t-2)(1+(1-\pi)\epsilon)}{28\mathcal{K}(2-\pi)} \right),$$

the expected instantaneous regret incurred by the arm selection policy satisfies

$$\begin{aligned} \mathbb{E}[\mathrm{Regret}_t] &:= \mathbb{E}_{\mathcal{B}_t} \mathbb{E}_{\hat{a}_t \in \mathscr{A}} \mathbb{E}_{\boldsymbol{\vartheta}} \left[\boldsymbol{\vartheta}(a_t^* \mid \mathcal{B}_t) - \boldsymbol{\vartheta}(\hat{a}_t \mid \mathcal{B}_t) \right] \\ &= O\left(\frac{1}{t}\right) + O\left(\frac{\mathcal{K}^{3/2}(\max_a \mathfrak{C}_{(a)} + \mathfrak{C}_*)}{\boldsymbol{\varrho}} \sqrt{\frac{\log t}{t}} \right), \end{aligned} \tag{8}$$

where $\mathfrak{C}_*$ *is the approximation constant corresponding to the optimal arm* a^*.

Proof.

$$\begin{aligned} &\mathbb{E}_{\mathcal{B}_t} \mathbb{E}_{a_t \in [\mathcal{K}]} \mathbb{E}_{\boldsymbol{\vartheta}} \left[\boldsymbol{\vartheta}(\boldsymbol{f}^{(a_t^*)} | \mathcal{B}_t) - \boldsymbol{\vartheta}(\boldsymbol{f}^{(a_t)} | \mathcal{B}_t) \right] \\ &= \mathbb{E}_{\mathcal{B}_t} \left[\mu_{(*)}(\mathcal{B}_t) - \mathbb{E}_{a \in [\mathcal{K}]} \mathbb{E}_{\boldsymbol{\vartheta}} \left[\boldsymbol{\vartheta}(\boldsymbol{f}^{(a_t)} | \mathcal{B}_t) \right] \right] \quad \text{where } \mu_{(\bullet)} = \mathrm{E}_{\boldsymbol{\vartheta}}[\boldsymbol{\vartheta}(\bullet)] \\ &= \mathbb{E}_{\mathcal{B}_t} \left[\mu_{(*)}(\mathcal{B}_t) - \sum_{j=1}^{\mathcal{K}} \mu_a(\mathcal{B}_t) \mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)} \mid \mathcal{B}_t) \right] \\ &= \mathbb{E}_{\mathcal{B}_t} \sum_a \boldsymbol{\Delta}_a(\mathcal{B}_t) \mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)} \mid \mathcal{B}_t) \\ &= \sum_a \mathbb{E}_{\mathcal{B}_t} \boldsymbol{\Delta}_a(\mathcal{B}_t) \mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)} \mid \mathcal{B}_t) \end{aligned}$$

$$\begin{aligned} &\mathbb{E}_{\mathcal{B}_t} \boldsymbol{\Delta}_a(\mathcal{B}_t) \mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)} \mid \mathcal{B}_t) \\ &\leq \mathbb{E}_{\mathcal{B}_t} \boldsymbol{\Delta}_a(\mathcal{B}_t) \left[\boldsymbol{\Xi}_t / \mathcal{K} + \mathbb{P}(\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t)}} \geq \mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t^*)}}) \right] \end{aligned}$$

Let $\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t)}}$ indicates the trained neural network in accordance to [39] for action $\boldsymbol{f}^{(a_t)}$. By Markov's inequality

$$\begin{aligned} \mathbb{P}(\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t)}} \geq \mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t^*)}}) &\leq \mathbb{P}\Big(\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t)}} \geq \mu_{(a)}(\mathcal{B}_t) + \boldsymbol{\Delta}_a(\mathcal{B}_t)/2 \Big) + \\ &\qquad \mathbb{P}\Big(\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t^*)}} \leq \mu_{(*)}(\mathcal{B}_t) - \boldsymbol{\Delta}_a(\mathcal{B}_t)/2 \Big) \\ &= \int_{\mathbb{1}\{\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t)}} \geq \mu_{(a)}(\mathcal{B}_t) + \boldsymbol{\Delta}_a(\mathcal{B}_t)/2\}} \partial \mathbb{P}_a + \int_{\mathbb{1}\{\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t^*)}} \leq \mu_{(*)}(\mathcal{B}_t) - \boldsymbol{\Delta}_a(\mathcal{B}_t)/2\}} \partial \mathbb{P}_* \\ &\leq \int_{\mathbb{1}\{|\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t)}} - \mu_{(a)}(\mathcal{B}_t)| \geq \boldsymbol{\Delta}_{(a)}(B_t)/2\}} \partial \mathbb{P}_a + \int_{\mathbb{1}\{|\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t^*)}} - \mu_{(*)}(\mathcal{B}_t)| \geq \boldsymbol{\Delta}_{(a)}(B_t)/2\}} \partial \mathbb{P}_* \\ &\leq \int 4 \frac{|\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t)}} - \mu_{(a)}(\mathcal{B}_t)|^2}{\boldsymbol{\Delta}_{(a)}(B_t)^2} \partial \mathbb{P}_a + \int 4 \frac{|\mathcal{M}_{\boldsymbol{\Theta}, \boldsymbol{f}^{(a_t^*)}} - \mu_{(*)}(\mathcal{B}_t)|^2}{\boldsymbol{\Delta}_{(a)}(B_t)^2} \partial \mathbb{P}_* \end{aligned} \tag{17}$$

Based on *Proposition* 1 we have $\boldsymbol{\tau}_{(a)}^R(t) \geq \frac{t-2}{2\mathcal{K}(2-\pi)}(1 + (1-\pi)\epsilon)$ for all $a \in$ A. Let $\mathfrak{C}_{(a)}$ indicate the constant from Theorem 2 and let n_a be the minimal training data size. We choose $t_0 > e^{(2\mathcal{K} \max\{e, \max_a n_a\})}$. Since the $x \to \sqrt{\frac{\ln(x)}{x}}$ is monotone decreasing for $x > e$, the above expression is further bounded by

$$\begin{aligned} &\leq \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{(a)}(\mathcal{B}_t)\frac{\epsilon_t}{\mathcal{K}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_{(a)}\sqrt{\frac{\ln(\boldsymbol{\tau}_{(a)}(t))}{\boldsymbol{\tau}_{(a)}(t)}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_{*}\sqrt{\frac{\ln(\boldsymbol{\tau}_{*}(t))}{\boldsymbol{\tau}_{*}(t)}} \\ &\leq \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{(a)}(\mathcal{B}_t)\frac{\epsilon_t}{\mathcal{K}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_{(a)}\sqrt{\frac{\ln(\boldsymbol{\tau}_{i}^{R}(t))}{\boldsymbol{\tau}_{i}^{R}(t)}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_{*}\sqrt{\frac{\ln(\boldsymbol{\tau}_{*}^{R}(t))}{\boldsymbol{\tau}_{*}^{R}(t)}} \\ &\leq \frac{\mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{(a)}(\mathcal{B}_t)}{t\mathcal{K}} + \frac{4}{\boldsymbol{\varrho}}\left[\mathfrak{C}_{(a)} + \mathfrak{C}_{*}\right]\sqrt{\frac{\ln(\frac{t-2}{2\mathcal{K}(2-\pi)}(1+(1-\pi)\epsilon))}{\frac{t-2}{2\mathcal{K}(2-\pi)}(1+(1-\pi)\epsilon)}} \end{aligned} \tag{18}$$

Thus we have the following:

$$\begin{aligned} &\sum_{\boldsymbol{f}^{(a_t)}} \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{a}(\mathcal{B}_t)\mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)} \mid \mathcal{B}_t) \\ &\leq \frac{\max_{\boldsymbol{f}^{(a_t)}\in\boldsymbol{\mathcal{F}}_{\text{sub}}} \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{a}(\mathcal{B}_t)}{t} + \\ &\mathcal{K}^{3/2}\sqrt{2(2-\pi)}\frac{4}{\boldsymbol{\varrho}}\left[\max_{a}\mathfrak{C}_{(a)} + \mathfrak{C}_{*}\right] \\ &\sqrt{\frac{\ln((t-2)(1+\epsilon-\pi\epsilon)) - \ln(2\mathcal{K}(2-\pi))}{(t-2)(1+\epsilon-\pi\epsilon)}} \end{aligned} \tag{19}$$

To showcase the lower bound, we have for a-th arm not optimal that,

$$\mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{a}(\mathcal{B}_t)\mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)} \mid \mathcal{B}_t) \geq \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{a}(\mathcal{B}_t)\frac{\boldsymbol{\Xi}_t}{\mathcal{K}} \geq \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\varrho}\frac{\boldsymbol{\Xi}_t}{\mathcal{K}} \geq \frac{\boldsymbol{\varrho}}{t\mathcal{K}} \tag{20}$$

□

Lemma 2 (Bound on Uniform Arm Selection Frequency). *Since $\boldsymbol{\tau}_{(a)}^{R}(t)$ denotes the number of times the a-th submodular function $\boldsymbol{f}^{(a)}$ is chosen in the first $t-1$ steps by the uniform branch of the algorithm, we have the following:*

$$\begin{aligned} &\mathbb{P}\left(\bigcap_{a=1}^{\mathcal{K}}\{\boldsymbol{\tau}_{(a)}^{R}(t) \geq \frac{t-2}{2\mathcal{K}(2-\pi)}(1+(1-\pi)\epsilon)\}\right) \\ &\geq 1 - \mathcal{K}\exp\left(-\frac{3(t-2)(1+(1-\pi)\epsilon)}{28\mathcal{K}(2-\pi)}\right) \end{aligned}$$

Proof.

$$\begin{aligned} \mathbb{E}(\boldsymbol{\tau}_{(a)}^{R}(t)) &= \sum_{r=1}^{t-1}\mathbb{P}(\zeta < \boldsymbol{\Xi}_r \cap \boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)}) \\ = \sum_{r=1}^{t-1}\mathbb{P}(\zeta < \boldsymbol{\Xi}_r)\mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)}) &= \sum_{r=1}^{t-1}\frac{\boldsymbol{\Xi}_r}{\mathcal{K}} = \frac{1}{\mathcal{K}}\sum_{r=1}^{t-1}\frac{r}{(r+\lambda(r))^{\pi}} \\ &\geq \frac{1}{\mathcal{K}}\int_{x=1}^{x=t-1}\frac{x}{(x+\lambda(x))^{\pi}}\partial x \geq \frac{t-2}{\mathcal{K}(2-\pi)}(1+(1-\pi)\epsilon) \end{aligned} \tag{21}$$

where, the last inequality is based on *Proposition* 1. We define the variance of $\boldsymbol{\tau}_{(a)}^{R}(t)$ as $\sigma(\boldsymbol{\tau}_{(a)}^{R}(t))$ and the corresponding upperbound as $\mathcal{Z}(\sigma(t))$

$$\sigma(\boldsymbol{\tau}_{(a)}^{R}(t)) = \sum_{r=1}^{t-1}\frac{\boldsymbol{\Xi}_r}{\mathcal{K}}(1-\frac{\boldsymbol{\Xi}_r}{\mathcal{K}}) \leq \frac{1}{\mathcal{K}}\sum_{r=1}^{t-1}\boldsymbol{\Xi}_r = \frac{1}{\mathcal{K}}\sum_{r=1}^{t-1}\frac{r}{(r+\lambda(r))^{\pi}} = \mathcal{Z}(\sigma(t))$$

Using Bernstein's inequality

$$\begin{aligned}
&\mathbb{P}\bigg(\boldsymbol{\tau}_{(a)}^{R}(t) \leq \frac{\mathcal{Z}(\sigma(t))}{2}\bigg) = \mathbb{P}\bigg(\boldsymbol{\tau}_{(a)}^{R}(t) - \mathcal{Z}(\sigma(t)) \leq -\frac{\mathcal{Z}(\sigma(t))}{2})\bigg) \\
&\leq \exp\left(\frac{\frac{-\mathcal{Z}(\sigma(t))^2}{8}}{\sigma(\boldsymbol{\tau}_{(a)}^{R}(t)) + \frac{1}{3}\frac{\mathcal{Z}(\sigma(t))}{2}}\right) \leq \exp\left(\frac{\frac{-\mathcal{Z}(\sigma(t))^2}{8}}{\mathcal{Z}(\sigma(t)) + \frac{1}{3}\frac{\mathcal{Z}(\sigma(t))}{2}}\right) \\
&\leq \exp\left(-\frac{3\mathcal{Z}(\sigma(t))}{28}\right) \leq \exp\left(-\frac{3(t-2)(1+(1-\pi)\epsilon)}{28\mathcal{K}(2-\pi)}\right)
\end{aligned}$$

By union bound method

$$\begin{aligned}
&\mathbb{P}\bigg(\bigcup_{a=1}^{\mathcal{K}}\{\boldsymbol{\tau}_{(a)}^{R}(t) \leq \frac{t-2}{2\mathcal{K}(2-\pi)}(1+(1-\pi)\epsilon)\}\bigg) \\
&\leq \mathcal{K}\mathbb{P}\bigg(\boldsymbol{\tau}_{(1)}^{R}(t) \leq \frac{t-2}{2\mathcal{K}(2-\pi)}(1+(1-\pi)\epsilon)\bigg) \leq \mathcal{K}\exp\left(-\frac{3(t-2)(1+(1-\pi)\epsilon)}{28\mathcal{K}(2-\pi)}\right)
\end{aligned}$$

Therefore

$$\mathbb{P}\bigg(\bigcap_{a=1}^{\mathcal{K}}\{\boldsymbol{\tau}_{(a)}^{R}(t) \geq \frac{t-2}{2\mathcal{K}(2-\pi)}(1+(1-\pi)\epsilon)\}\bigg) \geq 1 - \mathcal{K}\exp\left(-\frac{3(t-2)(1+(1-\pi)\epsilon)}{28\mathcal{K}(2-\pi)}\right)$$

□

Proposition 1 (**Integral Lower Bound (Constant λ)**). *Let* $\lambda(t) = \epsilon$ *with* $0 < \epsilon < 1$, *and* $0 < \pi < 1$. *Then, for*

$$I_t = \int_{x=1}^{t-1} \frac{x}{(x+\lambda(x))^{\pi}}\,\mathrm{d}x,$$

we have

$$I_t \;\geq\; \int_{x=1}^{t-1} \frac{x}{(x+\epsilon)^{\pi}}\,\mathrm{d}x \;\geq\; \frac{t-2}{2-\pi}\,(1+(1-\pi)\epsilon).$$

Proof.

$$
\begin{aligned}
\int_{x=1}^{x^{t-1}} \frac{x}{(x+\lambda(x))^{\pi}}\,\mathrm{d}x &= \int_{x=1}^{x=t-1} \frac{x}{(x+\epsilon)^{\pi}}\,\mathrm{d}x && \text{(Substitute } \lambda(x)=\epsilon\text{)}\\
&= \frac{[t-(1-\epsilon)]^{2-\pi}}{2-\pi} - \frac{(1+\epsilon)^{2-\pi}}{2-\pi}\\
&\quad - \epsilon\cdot\frac{[t-(1-\epsilon)]^{1-\pi}}{1-\pi} + \epsilon\cdot\frac{(1+\epsilon)^{1-\pi}}{1-\pi}\\
&= [t-(1-\epsilon)]^{1-\pi}\left(\frac{(1-\pi)(t-1)-\epsilon}{(1-\pi)(2-\pi)}\right)\\
&\quad - [1+\epsilon]^{1-\pi}\left(\frac{1-\pi-\epsilon}{(1-\pi)(2-\pi)}\right)\\
&= [m+\epsilon]^{1-\pi}\left(\frac{(1-\pi)m-\epsilon}{(1-\pi)(2-\pi)}\right)\\
&\quad - [1+\epsilon]^{1-\pi}\left(\frac{(1-\pi)-\epsilon}{(1-\pi)(2-\pi)}\right) && \text{(Let } m=t-1\text{)}\\
&= [m+\epsilon]^{k}\left(\frac{km-\epsilon}{k(k+1)}\right) - [1+\epsilon]^{k}\left(\frac{k-\epsilon}{k(k+1)}\right) && \text{(Let } k=1-\pi\text{)}\\
&= \frac{1}{k(k+1)}\left([m+\epsilon]^{k}(km-\epsilon)+(\epsilon-k)[1+\epsilon]^{k}\right)\\
&\geq \frac{1}{k(k+1)}\left([1+\epsilon]^{k}(km-\epsilon)+(\epsilon-k)[1+\epsilon]^{k}\right)\\
&\quad \text{(Since } m+\epsilon\geq 1+\epsilon\text{)}\\
&= \frac{1}{k(k+1)}[1+\epsilon]^{k}(km-\epsilon+\epsilon-k)\\
&= \frac{1}{k(k+1)}[1+\epsilon]^{k}(km-k)\\
&= \frac{1}{k+1}[1+\epsilon]^{k}(m-1)\\
&\geq \frac{1}{k+1}[1+k\epsilon](m-1) \qquad \text{(Using } (1+\epsilon)^{k}\geq 1+k\epsilon \text{ for small } \epsilon\text{)}\\
&= \frac{1}{2-\pi}[1+(1-\pi)\epsilon](t-2) \qquad \text{(Substitute } k=1-\pi,\ m=t-1\text{)}\\
&= \frac{t-2}{2-\pi}(1+(1-\pi)\epsilon)
\end{aligned}
\tag{22}
$$

□

The above integral computation is used in the main paper for our proofs.

H.4 Regret bounds in the case of growing with time exploration dampening function

Proposition 2 (Integral Lower Bound (Exponential Growing λ)). *Let $\lambda(t)=1-e^{-\mathrm{i}t}$ be a time-growing function with rate $\mathrm{i}>0$, and let $0<\pi<1$. Then, for*

$$I_t = \int_{x=1}^{t-1} \frac{x}{(x+\lambda(x))^{\pi}}\,\mathrm{d}x,$$

the following lower bound holds:

$$I_t \;\geq\; \int_{x=1}^{t-1} \frac{x}{\left(x+1-e^{-\mathrm{i}x}\right)^{\pi}}\,\mathrm{d}x \;\geq\; \left(\frac{1}{2\mathrm{i}}\Big[\ln\!\big(2e^{\mathrm{i}(t-1)}-1\big)-\ln\!\big(2e^{\mathrm{i}}-1\big)\Big]\right)^{\pi}.$$

Proof.

$$\begin{aligned}
\int_{x=1}^{t-1} \frac{x}{(x+\lambda(x))^{\pi}}\,\mathrm{d}x &= \int_{x=1}^{t-1} \frac{x}{(x+1-e^{-\mathfrak{i}x})^{\pi}}\,\mathrm{d}x \\
&\geq \int_{x=1}^{t-1} \frac{x^{\pi}}{(x+1-e^{-\mathfrak{i}x})^{\pi}}\,\mathrm{d}x \\
&\geq \int_{x=1}^{t-1} \left(\frac{xe^{\mathfrak{i}x}}{xe^{\mathfrak{i}x}+e^{\mathfrak{i}x}-1}\right)^{\pi}\,\mathrm{d}x \\
&\geq \int_{x=1}^{t-1} \left(\frac{xe^{\mathfrak{i}x}}{xe^{\mathfrak{i}x}+xe^{\mathfrak{i}x}-x}\right)^{\pi}\,\mathrm{d}x \\
&= \int_{x=1}^{t-1} \left(\frac{e^{\mathfrak{i}x}}{2e^{\mathfrak{i}x}-1}\right)^{\pi}\,\mathrm{d}x && \text{Using Jensen's Inequality} \\
&\geq \left[\int_{x=1}^{t-1} \frac{e^{\mathfrak{i}x}}{2e^{\mathfrak{i}x}-1}\,\mathrm{d}x\right]^{\pi} \\
&= \left(\frac{1}{2\mathfrak{i}}\left[\ln(2e^{\mathfrak{i}(t-1)}-1)-\ln(2e^{\mathfrak{i}}-1)\right]\right)^{\pi}
\end{aligned}$$

□

Lemma 3 (Exploration Dampening: Annealing)**.** *Since $\boldsymbol{\tau}^{R}_{(a)}(t)$ denotes the number of times the a-th submodular action is chosen in the first* $t-1$ *steps by the uniform branch of the algorithm, $a \in [\mathcal{K}]$, then in the case of $\lambda(t) = 1 - e^{-\mathfrak{i}t}$, for $\mathfrak{i} \geq 0$(i.e. growing exploration dampening probability), we have the following:*

$$\begin{aligned}
&\mathbb{P}\Big(\bigcap_{j=1}^{\mathcal{K}}\{\boldsymbol{\tau}^{R}_{(a)}(t) \geq \frac{1}{2\mathcal{K}}\Big(\frac{1}{2\mathfrak{i}}\Big[\ln(2e^{\mathfrak{i}(t-1)}-1)-\ln(2e^{(a)}-1)\Big]\Big)^{\pi}\}\Big) \\
&\geq 1-\mathcal{K}\exp\Big(-\frac{3}{28\mathcal{K}}\Big(\frac{1}{2\mathfrak{i}}\Big[\ln(2e^{\mathfrak{i}(t-1)}-1)-\ln(2e^{(a)}-1)\Big]\Big)^{\pi}\Big)
\end{aligned}$$

Proof.

$$\begin{aligned}
\mathbb{E}(\boldsymbol{\tau}^{R}_{a}(t)) &= \sum_{r=1}^{t-1}\mathbb{P}(\zeta < \boldsymbol{\Xi}_r \cap f_t = f^{j}) \\
= \sum_{r=1}^{t-1}\mathbb{P}(\zeta<\boldsymbol{\Xi}_r)\mathbb{P}(\boldsymbol{f}^{(a_t)} = \boldsymbol{f}^{(a)}) &= \sum_{r=1}^{t-1}\frac{\boldsymbol{\Xi}_r}{\mathcal{K}} = \frac{1}{\mathcal{K}}\sum_{r=1}^{t-1}\frac{r}{(r+\lambda(r))^{\pi}} \\
\geq \frac{1}{\mathcal{K}}\int_{x=1}^{x=t-1}\frac{x}{(x+\lambda(x))^{\pi}}\partial x &= \frac{1}{\mathcal{K}}\int_{x=1}^{x=t-1}\frac{x}{(x+1-e^{-\mathfrak{i}x})^{\pi}}\partial x \\
&\geq \frac{1}{\mathcal{K}}\Big(\frac{1}{2\mathfrak{i}}\Big[\ln(2e^{\mathfrak{i}(t-1)}-1)-\ln(2e^{a}-1)\Big]\Big)^{\pi}
\end{aligned}$$

The last inequality comes from *Proposition* 2
We define the variance of $\boldsymbol{\tau}^{R}_{(a)}(t)$ as $\sigma(\boldsymbol{\tau}^{R}_{(a)}(t))$ and the corresponding upperbound as $\mathcal{Z}(\sigma(t))$

$$\sigma(\boldsymbol{\tau}^{R}_{a}(t)) = \sum_{r=1}^{t-1}\frac{\boldsymbol{\Xi}}{\mathcal{K}}(1-\frac{\boldsymbol{\Xi}}{\mathcal{K}}) \leq \frac{1}{\mathcal{K}}\sum_{r=1}^{t-1}\boldsymbol{\Xi}_r = \frac{1}{\mathcal{K}}\sum_{r=1}^{t-1}\frac{r}{(r+1-e^{-\mathfrak{i}r})^{\pi}} = \mathcal{Z}(\sigma(t))$$

Using Bernstein's inequality

$$
\begin{aligned}
\mathbb{P}\bigg(\boldsymbol{\tau}_a^R(t) \leq \frac{\mathcal{Z}(\sigma(t))}{2}\bigg) &= \mathbb{P}\bigg(\boldsymbol{\tau}_a^R(t) - \mathcal{Z}(\sigma(t)) \leq -\frac{\mathcal{Z}(\sigma(t))}{2})\bigg) \\
&\leq \exp\left(\frac{\frac{-\mathcal{Z}(\sigma(t))^2}{8}}{\sigma(\boldsymbol{\tau}_a^R(t)) + \frac{1}{3}\frac{\mathcal{Z}(\sigma(t))}{2}}\right) \leq \exp\left(\frac{\frac{-\mathcal{Z}(\sigma(t))^2}{8}}{\mathcal{Z}(\sigma(t)) + \frac{1}{3}\frac{\mathcal{Z}(\sigma(t))}{2}}\right) \\
\leq \exp\left(-\frac{3\mathcal{Z}(\sigma(t))}{28}\right) &\leq \exp\left(-\frac{3}{28\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\Big[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\Big]\right)^{\pi}\right)
\end{aligned}
$$

By union bound method

$$
\begin{aligned}
&\mathbb{P}\bigg(\bigcup_{j=1}^{\mathcal{K}}\{\boldsymbol{\tau}_{(a)}^R(t) \leq \frac{1}{2\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\Big[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\Big]\right)^{\pi}\}\bigg) \\
&\leq \mathcal{K}\mathbb{P}\bigg(\boldsymbol{\tau}_1^R(t) \leq \frac{1}{2\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\Big[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\Big]\right)^{\pi}\bigg) \\
&\leq \mathcal{K}\exp\left(-\frac{3}{28\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\Big[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\Big]\right)^{\pi}\right)
\end{aligned}
$$

Therefore

$$
\begin{aligned}
&\mathbb{P}\bigg(\bigcap_{j=1}^{\mathcal{K}}\{\boldsymbol{\tau}_{(a)}^R(t) \geq \frac{1}{2\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\Big[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\Big]\right)^{\pi}\}\bigg) \\
&\geq 1 - \mathcal{K}\exp\left(-\frac{3}{28\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\Big[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\Big]\right)^{\pi}\right)
\end{aligned}
$$

□

Theorem 3 (Regret Guarantees). *Under Assumptions **b - d**, for all $t > t_0$ and with $\lambda(t) = 1 - e^{-\mathrm{i}t}$, with probability at least*

$$1 - \mathcal{K}\exp\left(-\frac{3}{28\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\left[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\right]\right)^{\pi}\right)$$

the expected instantaneous regret incurred by the arm selection policy satisfies

$$\begin{aligned}\mathbb{E}[\mathrm{Regret}_t] &:= \mathbb{E}_{\mathcal{B}_t}\mathbb{E}_{\hat{a}_t\in\mathscr{A}}\mathbb{E}_{\boldsymbol{\vartheta}}\left[\boldsymbol{\vartheta}(a_t^* \mid \mathcal{B}_t) - \boldsymbol{\vartheta}(\hat{a}_t \mid \mathcal{B}_t)\right] \\ &= O\left(\frac{1}{t}\right) + O\left(\frac{4(\mathfrak{C}_{(a)} + \mathfrak{C}_*)}{\varrho}\, t^{-\pi/2}\sqrt{\ln t}\right),\end{aligned} \tag{23}$$

where $\mathfrak{C}_*$ is the approximation constant corresponding to the optimal arm a^*.

Proof. Continuing from the same step in Section H.3 Theorem 1, we have the following: For the case of $\boldsymbol{\tau}^R_{(a)}(t) \geq \frac{1}{2\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\left[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\right]\right)^{\pi}$ for all a via *Proposition* 2. Let $\mathfrak{C}_{(a)}$ indicate the constant from Theorem 2 and let n_a be the minimal training data size. We choose $t_0 > e^{(2\mathcal{K}\max\{e,\max_a n_a\})}$. Since the $x \to \sqrt{\frac{\ln(x)}{x}}$ is monotone decreasing for $x > e$, the above expression is further bounded by

$$\begin{aligned}&\leq \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{(a)}(\mathcal{B}_t)\frac{\epsilon_t}{\mathcal{K}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_{(a)}\sqrt{\frac{\ln(\boldsymbol{\tau}_{(a)}(t))}{\boldsymbol{\tau}_{(a)}(t)}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_*\sqrt{\frac{\ln(\boldsymbol{\tau}_*(t))}{\boldsymbol{\tau}_*(t)}} \\ &\leq \mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{(a)}(\mathcal{B}_t)\frac{\epsilon_t}{\mathcal{K}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_{(a)}\sqrt{\frac{\ln(\boldsymbol{\tau}^R_i(t))}{\boldsymbol{\tau}^R_i(t)}} + \frac{4}{\boldsymbol{\varrho}}\mathfrak{C}_*\sqrt{\frac{\ln(\boldsymbol{\tau}^R_*(t))}{\boldsymbol{\tau}^R_*(t)}} \\ &\leq \frac{\mathbb{E}_{\mathcal{B}_t}\boldsymbol{\Delta}_{(a)}(\mathcal{B}_t)}{t\mathcal{K}} + \frac{4}{\boldsymbol{\varrho}}\left[\mathfrak{C}_{(a)} + \mathfrak{C}_*\right]\sqrt{\frac{\ln(\frac{1}{2\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\left[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\right]\right)^{\pi})}{\frac{1}{2\mathcal{K}}\left(\frac{1}{2\mathrm{i}}\left[\ln(2e^{\mathrm{i}(t-1)}-1) - \ln(2e^{(a)}-1)\right]\right)^{\pi}}}\end{aligned} \tag{24}$$

□

I Broader Impact

The primary aim of our work is to improve the data efficiency of machine learning training pipelines via submodular subset selection. By leveraging principled selection algorithms—such as monotone submodular functions, we can reduce the number of training examples needed without sacrificing model performance. This contributes directly to more sustainable and accessible machine learning, especially in scenarios where training data or compute is limited.

Societal and Environmental Benefits: Reducing the amount of data required for training has multiple practical benefits. For large-scale models, this can translate into lower energy consumption and a reduced carbon footprint. For smaller research labs or applications in low-resource settings, our approach can make training state-of-the-art models more feasible.

Equity and Fairness: By allowing for careful and task-informed selection of training data, our methods could help surface underrepresented or domain-critical samples early in training. However, care must be taken to ensure that the subset selection process does not reinforce existing dataset biases. We encourage practitioners to combine our framework with fairness-aware selection techniques and to audit the resulting models for any performance disparities across groups.

Scientific Impact: More broadly, this work highlights the growing role of data-centric approaches in machine learning research, particularly for compute efficient machine learning research.