BEYOND THE LAST LAYER: RETHINKING GRADIENT PROXIES FOR CORESET SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Gradient-based coreset selection offers a theoretically rigorous framework for data-efficient learning; however, its computational cost necessitates practical approximations. To scale these methods, existing heuristics, such as in CRAIG, approximate full gradient similarity by restricting gradient computation to the final layer, implicitly assuming that the penultimate representation suffices for data selection; a phenomenon we refer to as the *Last-Layer Bias*. In this work, we critically examine this assumption and investigate how the informative gradient signal is distributed across network depth. We propose a family of alternative gradient proxies that exploit information beyond the final layer, including input-level gradients (CRAIG-First), data-driven intermediate layer selection via FlexTuning (CRAIG-FT), and a distributed proxy based on Global Node Importance (CRAIG-NI), which selects sparse, non-contiguous subsets of high-activity neurons. We conduct one of the largest empirical studies in the coreset literature, evaluating 663 tabular classification and regression datasets, and extend our analysis to image classification and instruction tuning in large language models. Our results show that last-layer proxies often exhibit diminishing returns and barely outperform random selection (CRAIG-RL). In contrast, input-level proxies consistently accelerate convergence on tabular datasets, while distributed proxies such as CRAIG-NI generalize robustly across vision and language tasks, with particularly strong gains on visual benchmarks, effectively mitigating the Last-Layer Bias.

1 INTRODUCTION

Neural scaling laws link model performance to dataset size and parameter count Kaplan et al. (2020); Hoffmann et al. (2022), but larger training runs incur substantial computational costs Thompson et al. (2020). *Coreset selection* addresses this challenge by identifying a small, informative subset that approximates the full dataset for training Killamsetty et al. (2021b); Shinde & Madabhushi; San Joaquin et al. (2024); Pooladzandi et al. (2022). Selection methods range from inexpensive heuristics, including natural-language cues Cao et al. (2023), to gradient-based objectives that seek to preserve the aggregate training signal Mirzasoleiman et al. (2020); Killamsetty et al. (2021b). These approaches expose a tension between computational feasibility and signal fidelity: inexpensive scores need not identify examples that preserve the optimization behavior of the full dataset.

Using full-model gradients is costly: beyond extracting per-example gradients, many methods must construct and operate on pairwise similarity matrices. This difficulty applies to conventional neural networks as well as large vision and language models. It motivates approximations such as low-rank gradient representations Xia et al. (2024), restricting the signal to the first-k layers San Joaquin et al. (2024), or using a single layer Killamsetty et al. (2021b;a); Xia et al. (2022); Mirzasoleiman et al. (2020). The last-layer proxy is a common choice, motivated by the semantic information in the penultimate representation. However, semantic richness does not necessarily imply fidelity to the gradient structure needed for coreset selection. We call this reliance on the final layer without establishing its suitability for the task *Last-Layer Bias*.

Evidence from data influence and transfer learning suggests alternatives. The *First is Better* hypothesis points to gradient cancellation in final-layer signals, where conflicting update signals partially neutralize one another Yeh et al. (2022), while early-layer approaches seek to preserve data-dependent geometry San Joaquin et al. (2024). On the other hand, some studies contend that the most informative

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

training signal may reside in intermediate layers Vitel & Chhabra (2025). This perspective is further supported by evidence from transfer learning, where FlexTuning Royer & Lampert (2020) demonstrates that selectively fine-tuning task-relevant intermediate layers can outperform tuning edge layers alone.

This dissonance in the ML literature for Data Influence and transfer learning precipitates a fundamental question: ***Which layer constitutes the optimal proxy for coreset selection?*** Furthermore, we extend this inquiry to a broader hypothesis: ***Should the search for the optimal gradient proxy be strictly confined to contiguous layers, or is the relevant signal for coreset selection sparsely distributed across the model's entire parameter space?***

To rigorously test these hypotheses and address the last-layer bias, we adopt CRAIG (Coresets for Accelerating Incremental Gradient descent) Mirzasoleiman et al. (2020), a prominent gradient-based coreset selection algorithm as our base optimization framework and introduce a novel suite of targeted approximations to probe the locality of the coreset signal. Keeping its selection mechanism fixed allows us to study the source of the proxy signal. We establish **CRAIG-First** and **CRAIG-Last** as boundary baselines. **CRAIG-RL** samples a proxy layer at each epoch to test whether a fixed layer is necessary. **CRAIG-FT** uses validation-guided FlexTuning to select a fixed layer, testing whether the informative signal can be localized. Finally, **CRAIG-NI** constructs a distributed proxy from active neurons across layers, testing whether layer boundaries are too restrictive. Random sample selection provides a separate baseline. Our contributions are threefold:

- We systematically examine last-layer bias through a controlled comparison of boundary, stochastic, validation-guided, and distributed node-level proxies. CRAIG-NI extends the comparison beyond contiguous layers by constructing a sparse proxy from neurons associated with large parameter updates.
- We evaluate 663 tabular datasets (517 classification and 146 regression tasks) using dense MLPs. Exploratory experiments extend the analysis to ResNet18 He et al. (2016) on CIFAR-10 Krizhevsky et al. (2009) and MNIST LeCun et al. (2002), and Gemma3-1B Team et al. (2025) instruction tuning on Dolly-15k Conover et al. (2023).
- We find a modality-dependent quality–cost trade-off. CRAIG-First achieves the highest tabular efficiency with a modest quality reduction. CRAIG-NI offers a stronger balance on vision tasks and is competitive in instruction tuning, where convergence trajectories are needed to interpret efficiency scores. These results challenge a fixed last-layer default without identifying a universally best proxy.

2 Related Work

To contextualize our investigation into the locality of geometric signal, we situate our work within the broader landscape of data efficiency and influence estimation. The prevailing literature can be taxonomized by the choice of gradients, ranging from the entire model to layer based heuristics, crucially, by the spatial assumptions regarding where the most informative signal resides. While early coreset methods focused on handcrafted features, the modern deep learning era has coalesced around gradient-based proxies[1]. However, as illustrated in Table 1, the question of which part of the model yields the most faithful approximation for coresets remains largely assumed rather than empirically validated. This overview highlights the dominance of static layer assumptions and identifies the specific question that our paper seeks to address.

While the methodologies detailed above in Table 1 employ diverse scoring metrics, we leverage the CRAIG method as a controlled optimization testbed. By keeping the selection logic constant, we can isolate and rigorously interrogate the variable of interest: the *source* of the gradient signal. In the following section, we formalize this experimental framework and detail the suite of probing variants ranging from stochastic sampling to sparse node aggregation, specifically designed to challenge the Last-Layer Bias.

[1]We do have papers like Cao et al. (2023) which propose non-gradient approaches using natural language cues, but that is largely confined towards LLMs

Table 1: Comparison of Gradient Proxies in Data Selection Literature. We categorize methods by their *Signal Source* (what proxy was proposed to approximate the gradient) and their *Spatial Assumption* (where this signal is located).

Method	Selection Metric	Signal Source	Spatial / Layer Assumption
CRAIG Mirzasoleiman et al. (2020)	Gradient Clustering	Last Linear Layer	*Static (Last)*
GRAD-MATCH Killamsetty et al. (2021a)	Gradient Matching	Last Linear Layer	*Static (Last)*
GLISTER Killamsetty et al. (2021b)	Bilevel Optimization	Last Linear Layer	*Static (Last)*
GraNd Paul et al. (2021)	Exp. Gradient Norm	Gradient Norm Scalar	*Static (Last)*
SVP Coleman et al. (2019)	Proxy Error	Small Proxy Architecture	*Global (Small Arch)*
First is Better Yeh et al. (2022)	Kernel Representers	First Layer	*Static (First)*
In2Core San Joaquin et al. (2024)	Influence Functions	First-k or Last Layers	*Static (First-k)*
LESS Xia et al. (2024)	Influence (LoRA)	Low-Rank Projection	*Global (LoRA Subspace)*
CRAIG-RL (Ours)	Gradient Clustering	Random Layer Activations	*Dynamic (Random Layer)*
CRAIG-FT (Ours)	Gradient Clustering	FlexTuning Selection Royer & Lampert (2020)	*Static (Best Intermediate)*
CRAIG-NI (Ours)	Gradient Clustering	Global Node Importance	*Dynamic & Sparse Nodes*

3 CRAIG

We use CRAIG Mirzasoleiman et al. (2020) as a controlled testbed for studying proxy location. Given a training set $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{N}$ and model parameters $\theta \in \mathbb{R}^d$, let $\mathcal{L}(\theta) = N^{-1} \sum_i \ell(x_i, y_i)$ denote the empirical loss. Gradient-based coreset selection seeks a weighted subset $\mathcal{S}$, with $|\mathcal{S}| = m \ll N$, whose aggregate gradient approximates that of the full training set:

$$\min_{\mathcal{S},w} \left\| \sum_{i=1}^{N} \nabla_\theta \ell(x_i, y_i) - \sum_{j \in \mathcal{S}} w_j \nabla_\theta \ell(x_j, y_j) \right\|. \tag{1}$$

Directly matching full-model gradients is expensive. Our implementation uses an activation vector $\phi(x)$ as the proxy for each example and applies greedy facility-location selection to cover the training set in the chosen proxy space. We keep this selection mechanism fixed and vary the source of the proxies. Representations entering the final linear layer provide the last-layer baseline; the alternatives are defined in the following section.

Writing the final affine layer as $z_i = W\phi_i + b$, with $\phi_i = \phi(x_i)$, $\ell_i = \ell(x_i, y_i)$, and $\delta_i = \nabla_{z_i} \ell_i$, the last-layer weight-gradient difference satisfies

$$\|\nabla_W \ell_i - \nabla_W \ell_j\|_{\text{F}} \leq \|\phi_i - \phi_j\|_2 \, \|\delta_i\|_2 + \|\phi_j\|_2 \, \|\delta_i - \delta_j\|_2. \tag{2}$$

Activation similarity alone does not guarantee gradient similarity: error signals also matter. We use activation coverage as a practical surrogate for gradient matching. Appendix A provides the derivation and the common greedy selection procedure.

4 Probing Methods

In this section, we systematically operationalize our hypotheses on gradient locality by designing a set of probing methods within the CRAIG framework.

4.1 Boundary Baselines

The prevailing approximation in the literature is to restrict the gradient calculation to the weights of the final linear projection layer, denoted as θ_L. This approach, which we term *Last-Layer Bias*, assumes that the embedding generated by the penultimate layer captures the complete geometric complexity of the input data. Under this assumption, the gradient of the last layer serves as a sufficient statistic for the difficulty or informativeness of a sample. We formalize this baseline as **CRAIG-Last**. As described in Appendix B, CRAIG-Last extracts the embeddings $\mathbf{P}_{last}$ from the final layer and performs greedy facility location selection on these low-dimensional vectors. Conversely, to implement CRAIG-First we leverage the embeddings $\mathbf{P}_{first}$ and have one simple modification in the original CRAIG-Last algorithm, $\mathbf{P}_{sel} \leftarrow \mathbf{P}_{first}$.

We also leverage the Random Selection baseline, as shown in the original CRAIG paper Mirzasoleiman et al. (2020) as the main baseline, which serves as the efficiency floor for our experimental validation. More details are shown in Appendix C.

4.2 ALGORITHMIC LAYER SAMPLING

As mentioned in Section 1, we probe the locality of targeted approximation by introducing a novel suite of CRAIG variants. Here, CRAIG-RL and CRAIG-FT are different attempts to algorithmically sample the optimal layer instead of being confined to the first or the last layer.

CRAIG-RL (Random Layer Sampling) challenges the assumption that the last layer is uniquely informative. As described in Appendix D, this method stochastically samples a layer index l_{rand} at each selection epoch. If the geometric signal is indeed uniformly distributed, CRAIG-RL should yield comparable performance to CRAIG-Last. This serves as a control to test if the *optimality* of the last layer is an artifact of convention rather than geometric necessity.

CRAIG-FT (FlexTuning) investigates the hypothesis of *Static Localization*. This method draws direct inspiration from FlexTuning (specifically the Even Faster Flex Tuning paradigm) from the paper Royer & Lampert (2020), which demonstrates that for many transfer learning tasks, fine-tuning a specific intermediate layer yields higher accuracy than fine-tuning the last layer or the full model. This finding suggests that the gradients of a specific intermediate layer may be more representative of the dataset geometry than the final output layer. Guided by this insight, CRAIG-FT posits that while the Last-Layer may be suboptimal, there is a specific, fixed intermediate layer l_{best} that contains the optimal signal. As detailed in Appendix E, we employ the probing strategy from Flex Tuning using a validation set to identify the layer that maximizes generalization performance. We then fix this layer as the static proxy throughout the selection process.

4.3 THE DISTRIBUTED HYPOTHESIS AND GLOBAL NODE IMPORTANCE

Our paper extends the investigation of gradient approximation to a fundamental question: *What if the optimal proxy for dataset geometry is not found in any single static layer, but is spatially distributed across the model's most active neurons?* To operationalize this hypothesis, we introduce **CRAIG-NI**.

Unlike previous methods that restrict selection to a physical layer, CRAIG-NI constructs a *Virtual Gradient* vector derived dynamically from the model's active subspace. To justify this approach, we return to the primary objective of gradient-based coreset selection defined in Eq. 1. While Eq. 1 typically implies optimization over the full parameter space $\theta \in \mathbb{R}^d$, we propose that this high-dimensional optimization can be projected onto a lower-dimensional active subspace without significant loss of precision.

We define *Parameter Velocity*, $\boldsymbol{\nu}$, as the magnitude of change in parameters over a training interval. In standard optimization schemes, velocity is directly proportional to the gradient magnitude:

$$\boldsymbol{\nu}^{(k)} = |\theta_{t+1}^{(k)} - \theta_t^{(k)}| \approx \eta |\nabla_{\theta^{(k)}} \mathcal{L}| \tag{3}$$

Using this metric, we can decompose the gradient matching objective. Let $\mathcal{W}_{active}$ be the set of weights with high velocity (top $M\%$) and $\mathcal{W}_{inactive}$ be the complement. The upper bound from Eq. 2 effectively splits:

$$\begin{aligned} \text{Error}_{\text{total}} \leq & \sum_{w \in \mathcal{W}_{\text{active}}} \|\phi_w(x_i) - \phi_w(x_j)\|_2 \cdot C_1 \\ & + \underbrace{\sum_{w \in \mathcal{W}_{\text{inactive}}} \|\dots\|_2 \cdot C_2}_{\approx 0 \text{ (due to low gradient)}} \end{aligned} \tag{4}$$

To minimize the effective gradient error, we need only minimize the geometric distance $\|\phi(x_i) - \phi(x_j)\|$ for the activations ϕ_w associated with the active weights $\mathcal{W}_{active}$.

CRAIG-NI identifies the subset of neurons corresponding to the top $M\%$ of parameters with the highest velocity. To fairly compare against fixed-layer-based CRAIG methods, we formalize M as $\frac{1}{N_{layers}}$, where N_{layers} is the number of hidden layers in the model[2]. By aggregating the activations

[2]As we essentially leverage the same number of neurons as one layer, the time taken by $CRAIG_Greedy$ in CRAIG-NI will be similar to that in CRAIG-Last.

a_i associated with these high-velocity weights, we form a *Virtual Gradient* vector $\mathbf{V}_x$. This vector represents the input features driving the most significant updates in the network. Algorithm 1 details this procedure, where facility location is performed on this dynamic, high-leverage subspace.

5 EXPERIMENTS

To rigorously validate our hypotheses regarding the locality of geometric information, we conducted a large-scale empirical study spanning diverse modalities and task complexities. In contrast to much of the prior coreset literature, which evaluates on a limited number of image datasets, our design prioritizes statistical significance by evaluating performance across 663 distinct datasets for tabular classification and regression with selection criteria available in Appendix F. We further include exploratory experiments in vision models and instruction tuning for LLMs as pilot studies, aimed at assessing whether the observed trends extend beyond the tabular domain.

Algorithm 1 CRAIG-NI (Global Node Importance)

Require: $\mathcal{D}_{train}$, epochs E, warmup K, subset size m, θ_0.
Require: Parameter: TopPercentage $M\%$.
Ensure: Fine-tuned model parameters θ_E.
Phase 1: Warmup
$\theta_{prev} \leftarrow \theta_0$
for $e = 1 \rightarrow K$ **do**
  $\theta_{temp} \leftarrow \theta_{e-1}$ ▷ Cache state
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{train})$
  $\theta_{prev} \leftarrow \theta_{temp}$
end for
Phase 2: Incremental Selection Loop
for $e = K + 1 \rightarrow E$ **do**
  At start of epoch e**:**
    Step A: Determine Active Nodes (Velocity)
    ▷ Neurons evolving most rapidly
    $\boldsymbol{\delta} = |\theta_{e-1} - \theta_{prev}|$ ▷ Velocity
    Map $\boldsymbol{\delta}$ to input neurons $\mathcal{N}$.
    $\mathcal{I}_{active} \leftarrow$ Top $M\%$ neurons in $\mathcal{N}$.
    Step B: Construct Virtual Vectors
    For each $x \in \mathcal{D}_{train}$:
      Forward pass to get hidden states $\mathbf{H}_x$.
      $\mathbf{V}_x \leftarrow \mathbf{H}_x[\mathcal{I}_{active}]$ ▷ Concat
    Step C: Selection
    Form $\mathbf{P}_{sel}$ from $\{\mathbf{V}_x\}$.
    $\tilde{\mathbf{P}}_{sel} \leftarrow \text{Normalize}(\mathbf{P}_{sel})$.
    $S_e \leftarrow \text{CRAIG_Greedy}(\tilde{\mathbf{P}}_{sel}, m)$.
  Update Model:
  $\theta_{prev} \leftarrow \theta_{e-1}$ ▷ Drift reference
  $\mathcal{D}_{subset} \leftarrow \mathcal{D}_{train}[S_e]$
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{subset})$
end for
return θ_E

5.1 ABLATION FRAMEWORK AND TRAINING PROTOCOL

Experiments were conducted using two dense 5-layered MLP architectures: $[512, 512, 256, 256, 128]$ and $[256, 256, 256, 256, 256]$. To evaluate robustness, for each dataset, we randomly sampled one architecture and one coreset size from the set $m \in \{3\%, 5\%, 7\%, 10\%, 15\%\}$ of the full training set $|\mathcal{D}_{train}|$. We utilized the AdamW optimizer with a learning rate $\eta = 10^{-3}$.

Training Phases. To stabilize selection dynamics, we employ a two-phase protocol: **Phase 1: Warmup.** The model trains on the full dataset $\mathcal{D}_{train}$ (batch size 2) until a convergence threshold is met ($R^2 \geq 0.1$ or set accuracy 0.65), capped at $K = 3$ epochs. **Phase 2: Active Selection.** Following warmup, the model trains for a maximum of 30 epochs. At each epoch e, a subset S_e is selected via the designated proxy method, and the model updates using only S_e (batch size 32).

5.2 EVALUATION METRICS

We report quality Q, sample cost C, and efficiency E relative to the Random sample-selection baseline. For method m and dataset d, let S_{peak} be the cumulative unique samples consumed to reach peak performance:

$$Q_{m,d} = \begin{cases} \text{F1}_{\max}(m,d)/\text{F1}_{\max}(\text{Random},d), & \text{classification}, \\ \text{RMSE}_{\min}(\text{Random},d)/\text{RMSE}_{\min}(m,d), & \text{regression}, \end{cases} \tag{5}$$

$$C_{m,d} = \frac{S_{\text{peak}}(m,d)}{S_{\text{peak}}(\text{Random},d)}, \qquad E_{m,d} = \frac{Q_{m,d}}{C_{m,d}}. \tag{6}$$

Higher Q and E, and lower C, are better. We aggregate per-dataset ratios using medians; detailed tables also report the standard deviation of C. These costs measure sample consumption, not wall-clock time. Median learning trajectories against cumulative data fraction distinguish early learning speed from final performance. The step-based LLM evaluation is defined in Appendix H.

5.3 Extended Experiments

To assess the robustness of our findings beyond the tabular domain, we further extend our evaluation to representative high-dimensional learning settings in Computer Vision (CV) and Natural Language Processing (NLP). For vision tasks, we consider the standard *CIFAR-10* Krizhevsky et al. (2009) and *MNIST* LeCun et al. (2002) benchmarks with a *ResNet-18* He et al. (2016) architecture[3]. For instruction tuning in large language models, we evaluate our methods using the *Gemma3-1B* Team et al. (2025) model on the *Databricks-Dolly-15k* Conover et al. (2023) dataset. We note that the probing strategies introduced in this work do not directly transfer to standard LLM training regimes, which typically involve only a few fine-tuning epochs and parameter-efficient adaptations such as LoRA Hu et al. (2021). Accordingly, we adopt suitable modifications for the LLM setting, with full details provided in Appendix H.

6 Results

In this section, we present a comprehensive evaluation of the layer-aware selection strategies. We analyze performance across four distinct modalities with emphasis on tabular classification and regression, and exploratory experiments on visual perception, and instruction tuning in LLMs to verify the hypothesis that the optimal selection signal is not the static last-layer[4].

Method	All (n=517)				Coreset 3% (n=102)				Coreset 5% (n=105)				Coreset 7% (n=96)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
CRAIG-Last	1.006	0.853	2.265	1.179	1.007	0.845	3.165	1.194	1.008	0.817	3.118	1.232	1.007	0.821	1.492	1.220
CRAIG-First	0.989	0.373	0.993	**2.630**	0.993	0.334	1.240	**2.944**	0.987	0.325	1.116	**3.005**	0.988	0.326	0.862	**2.964**
CRAIG-RL	1.003	0.770	1.937	1.305	1.004	0.747	3.135	1.350	1.003	0.719	2.163	1.389	1.005	0.763	0.891	1.326
CRAIG-NI	1.000	0.550	1.410	1.815	1.000	0.469	2.104	2.114	1.000	0.495	1.699	2.006	1.000	0.516	0.808	1.923
CRAIG-FT	0.996	0.450	1.175	2.193	0.997	0.388	1.454	2.564	0.996	0.430	1.693	2.377	0.996	0.425	0.712	2.277

Method	Coreset 10% (n=106)				Coreset 15% (n=108)				Arch [256x5] (n=269)				Arch [512..128] (n=248)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
CRAIG-Last	1.002	0.870	1.478	1.158	1.007	0.902	0.978	1.126	1.007	0.853	2.179	1.179	1.006	0.849	2.354	1.183
CRAIG-First	0.986	0.415	1.094	**2.389**	0.991	0.408	0.432	**2.382**	0.990	0.378	0.937	**2.628**	0.987	0.368	1.050	**2.638**
CRAIG-RL	1.000	0.771	1.568	1.297	1.004	0.820	0.775	1.225	1.003	0.767	1.801	1.320	1.003	0.771	2.075	1.297
CRAIG-NI	0.999	0.563	1.166	1.763	1.000	0.647	0.657	1.569	1.000	0.554	1.419	1.802	1.000	0.541	1.400	1.824
CRAIG-FT	0.994	0.484	1.061	2.057	0.994	0.482	0.446	2.079	0.996	0.441	0.958	2.202	0.996	0.455	1.370	2.177

Table 2: Classification experiment. Arrows indicate direction of better performance: ↑ = higher is better, ↓ = lower is better

6.1 Tabular Benchmarks

Our extensive evaluation across 663 datasets reveals a consistent efficiency hierarchy that challenges standard selection norms in the tabular domain. As summarized in Table 2 and Appendix J, *CRAIG-First* emerges as the most distinct outlier, achieving the highest efficiency scores ($E \approx 2.63$ for classification and $E \approx 3.63$ for regression). This gain is driven by a drastic reduction in computational cost (C), as the method identifies informative subsets using roughly one-third of the samples required by the baseline. However, the aggressive filtering incurs a minor trade-off in predictive quality ($Q < 1.0$), suggesting that while input-level gradients capture coarse-grained geometry rapidly, they may miss subtle decision boundaries required for perfect fidelity. In contrast, *CRAIG-NI* and *CRAIG-FT* strike a robust balance, maintaining near-optimal quality ($Q \approx 1.0$) while halving the convergence cost relative to the baseline ($C \approx 0.45 - 0.55$). Notably, *CRAIG-Last*, despite being

[3]Specific adaptations and experimental details for vision are provided in Appendix G

[4]We also include additional results as Appendices with Appendix I, and J addressing Tabular classification and regression respectively, and Appendix K and L addressing the vision and LLM experiments respectively.

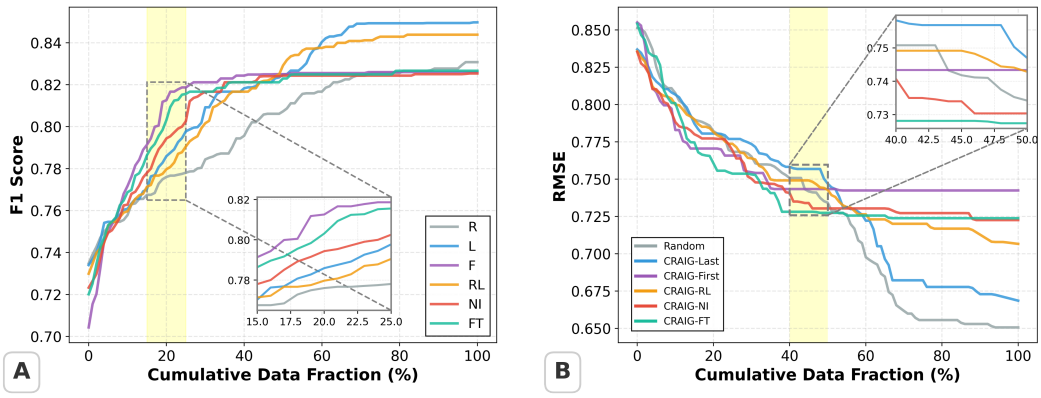

Figure 1: Here (A) represents the median trends across all the classification experiments for F1 Score vs the cumulative data used as a percentage; and (B) is the analogous graph for the regression datasets. For (A), R represents the random selection baseline, and the L, F, RL, NI, and FT represent the associated CRAIG- algorithm.

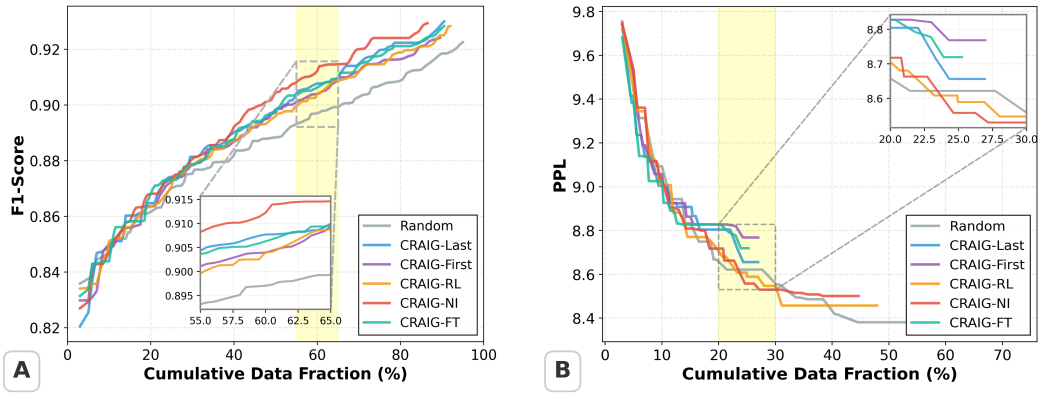

Figure 2: Here (A) represents the trajectory analysis on ResNet-18 for CIFAR-10 and MNIST. (B) represents the trajectory analysis with the perplexity metric on Gemma-3 1B and the Dolly-15k dataset.

the standard heuristic in literature, yields the lowest return on investment. Its efficiency ($E \approx 1.18$ for classification; $E \approx 1.00$ for regression) barely surpasses random selection, indicating that the computational overhead of computing final-layer gradients does not translate into a proportional reduction in sample complexity for these tasks.

To understand the dynamics behind these aggregate scores, we analyzed the training trajectories Figure 1 (A) and (B) which map the evolution of model performance against data consumption. As Figure 1 shows, *CRAIG-First* exhibits a *rapid-rise* trajectory, where performance spikes early in training, often surpassing all other methods in the first few epochs before plateauing. This confirms that the geometric signal is most abundant at the input layer during the initial phase of learning. Conversely, the trajectory for *CRAIG-Last* is sluggish, especially for the regression datasets, it closely mirrors the random baseline for a significant portion of training. This lag suggests that gradients at the final layer remain noisy or uninformative until deep features mature, forcing the model to consume more data to stabilize. *CRAIG-FT* and *CRAIG-NI* occupy the ideal middle ground: their trajectories rise sharply like CRAIG-First but maintain the asymptotic stability of the deeper layers, effectively converging to peak performance much earlier than the baseline.

6.2 Visual Benchmark

We next evaluate the methods on high-dimensional visual data using ResNet18 He et al. (2016) and the CIFAR-10 Krizhevsky et al. (2009) and MNIST datasets LeCun et al. (2002). While Table 6 highlights *CRAIG-First* for its raw efficiency ($E \approx 1.21$), the scatter plot in Figure 7 (B) reveals a critical limitation: it is the sole method clustering below the unity line ($Q < 1.0$), indicating that input-level gradients alone are insufficient for full convergence. In contrast, *CRAIG-NI* achieves the strongest overall trade-off by combining the aggressive cost reduction of the input layer ($C \approx 0.84$) with the high fidelity of deeper layers ($Q \approx 1.01$). The training trajectories in Figure 2 (A) confirm this hybrid advantage; *CRAIG-NI* matches the rapid early-stage learning of *CRAIG-First* but avoids its subsequent plateau, sustaining an F1 progression that rivals the stability of the baseline. This experiment further reinforces the shortcomings of CRAIG-LAST and highlights the need for a more effective heuristic.

6.3 Instruction Tuning in Large Language Models

The consolidated statistics in Table 7 present a nuanced landscape of performance. Based on pure efficiency, *CRAIG-Last* and *CRAIG-FT* appear dominant, achieving scores of $E > 3.0$. This is primarily driven by their exceptionally low computational cost ($C \approx 0.29$), allowing them to reach their respective convergence points with minimal data consumption. We further refer to the training trajectories in Figure 2 (B) to understand the relative rankings of the approaches at the moment of convergence. From Figure 2 (B), we can clearly see that when CRAIG-Last converges, it lags behind the random baseline, CRAIG-RL, and CRAIG-NI. Hence, while CRAIG-NI, CRAIG-RL, and random selection exhibit comparable performance at convergence, CRAIG-NI demonstrates a modest but consistent advantage in convergence speed without premature stopping. This makes CRAIG-NI a favorable choice among CRAIG-based approaches for instruction tuning.

7 Discussion

We observe a dichotomy: tabular efficiency peaks at early layers, while perceptual tasks require deeper layers. This section interprets these findings, focusing on the geometry-semantics trade-off and the flaws in existing selection biases.

The Geometry-Semantics Trade-off: Our analysis reveals a distinct modality-dependent behavior: for tabular data, the raw input space utilized by *CRAIG-First* consistently yields the highest efficiency for both regression and classification, suggesting that the initial geometric structure of semantic features is sufficient for selection without the noise of deep transformations. Conversely, for high-dimensional perceptual modalities such as Vision and LLMs, this raw geometry is insufficient; the superior performance of *CRAIG-NI* in these domains indicates that effective coreset selection requires the semantic abstraction provided by deep, learned representations. This validates a trade-off where simple data benefits from early geometric fidelity, while complex data necessitates late semantic processing.

Critique of the Last-Layer Bias: Crucially, our experiments systematically dismantle the prevalence of the Last-Layer Bias, as *CRAIG-Last* failed to achieve top-tier performance in either the massive tabular benchmark or the perceptual tasks. By consistently underperforming compared to *CRAIG-First* in low-dimensional settings and lagging behind *CRAIG-NI* in deep architectures, the results demonstrate that the final linear projection is an unreliable proxy for the full dataset geometry. This suggests that the final layer, while optimized for decision boundaries, likely compresses the granular intra-class information necessary for constructing a robust and diverse coreset.

Generalizability of Global Node Importance: While layer-specific probing (e.g. CRAIG-First) offers a computationally cheap solution for tabular data, *CRAIG-NI* emerges as the robust generalist for deep learning architectures, demonstrating superior scaling on ResNet18 and Gemma-3-1B. By transcending rigid layer boundaries and identifying a sparse subspace of high-velocity neurons globally, *CRAIG-NI* effectively captures distributed signals that single-layer methods miss. This establishes Global Node Importance not merely an alternative, but a necessary evolution for coreset selection in over-parameterized models where the *informative* gradient signal is delocalized across the network depth.

LIMITATIONS

While this study provides one of the largest empirical analyses of gradient-proxy locality for coreset selection, several limitations remain.

Single base framework. Our primary findings are derived within the CRAIG framework. We intentionally use CRAIG as a controlled testbed to isolate the effect of the gradient-proxy location under a fixed selection objective. To assess whether the observed last-layer bias extends beyond CRAIG, we conducted preliminary cross-framework experiments with GLISTER Killamsetty et al. (2021b) on 50 tabular classification datasets. As summarized in Appendix M, the results are qualitatively consistent with our CRAIG findings: *GLISTER-First* and *GLISTER-FT* attain substantially higher efficiency than *GLISTER-Last*, while *GLISTER-NI* remains competitive. Nevertheless, these preliminary results are restricted to tabular tasks and a single additional framework; comprehensive cross-framework and cross-modality validation remains an important direction for future work.

No universally dominant proxy. Our results do not identify a single gradient proxy that is optimal across all modalities. We instead observe a modality-dependent geometry-semantics trade-off: early-layer proxies are highly efficient for tabular data, while distributed or deeper proxies are more effective for vision and language tasks. Accordingly, *CRAIG-NI* is proposed as a robust heuristic rather than a universally superior solution, and its gains vary across datasets and architectures.

Limited vision and language evidence. The vision and LLM experiments are exploratory pilot studies with limited trials: 10 vision trials and 5 LLM trials, each using a single architecture, ResNet-18 and Gemma-3-1B, respectively. Extending these experiments to multiple architectures, model scales, datasets, and larger trial counts is an important future step.

8 CONCLUSION

This study presents the first rigorous empirical challenge to the *Last-Layer Bias*, the prevailing heuristic in gradient-based coreset selection that assumes the final linear projection is the optimal proxy for dataset geometry. By systematically benchmarking diverse gradient proxies across a massive repository of 663 datasets, we demonstrated that this standard assumption is often computationally inefficient and suboptimal. Our results reveal a distinct hierarchy of efficiency: while the standard CRAIG-Last frequently yielded a return on investment barely exceeding random selection, our novel probing variants offered superior alternatives. Specifically, *CRAIG-First* emerged as the most efficient proxy for tabular benchmarks. However, when extending our analysis to vision and instruction tuning in LLMs, we found that *CRAIG-NI* achieved superior performance. These findings demonstrate that the geometric signal required for effective data selection is not statically confined to the model's output, but is instead distributed across the network's depth.

For future research, we aim to address the aforementioned limitations and expand our investigation beyond the specific optimization framework used in this study. We plan to decouple our probes from the CRAIG algorithm and apply them to other gradient-based selection mechanisms. Additionally, we intend to significantly scale our evaluation suite to include a wider array of architectures and massive-scale datasets in the Vision and Large Language Model (LLM) domains.

AI USE STATEMENT

We used AI to edit author-written text for readability, to assist with LaTeX formatting and layout, and to generate supporting code for plotting and result aggregation. We also used these tools to improve productivity during code development. An initial version of the codebase implementing CRAIG and its layer-proxy variants was written manually by the authors. AI coding assistants were then used to help extend it to the vision and instruction-tuning experiments (Appendices G and H) and to the cross-framework evaluation with GLISTER (Appendix M). All AI-edited text was reviewed by the authors, and all AI-assisted code was manually reviewed and validated before being used to produce any reported result. The authors take full responsibility for the content of this paper.

ETHICS STATEMENT

This work studies coreset selection using existing benchmark datasets. Selecting a subset of training data may underrepresent rare examples or amplify existing biases. Our aggregate performance and efficiency results do not establish fairness or safety; applications beyond these benchmarks should assess subgroup performance and potential downstream impacts before deployment.

REPRODUCIBILITY STATEMENT

Section 3 formalizes the CRAIG objective and the activation-based gradient proxy, and Section 4 defines each probing method (CRAIG-First, CRAIG-Last, CRAIG-RL, CRAIG-FT, and CRAIG-NI), including the CRAIG-NI neuron budget $M = 1/N_{layers}$. Pseudocode is provided for every method: facility-location selection (Algorithm 2), CRAIG-Last (Algorithm 3), the Random baseline (Algorithm 4), CRAIG-RL (Algorithm 5), CRAIG-FT (Algorithm 6), and CRAIG-NI (Algorithm 1). All 663 tabular datasets are public OpenML datasets, selected using the filtering criteria and convergence check in Appendix F. Section 5.1 gives the MLP architectures, coreset sizes, optimizer, learning rate, batch sizes, and warmup and selection schedule, and Section 5.2 defines the quality, cost, and efficiency metrics and their median aggregation. Appendices G and H describe the adaptations to ResNet-18 and Gemma-3-1B, including proxy extraction, the kernel-level and merged-LoRA velocity computations, the Dolly-15k length filter, and the QLoRA training configuration. Appendix M specifies the GLISTER variant used for cross-framework validation (Algorithm 7).

REFERENCES

Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Pieter Gijsbers, Frank Hutter, Michel Lang, Rafael G Mantovani, Jan N van Rijn, and Joaquin Vanschoren. Openml benchmarking suites. *arXiv preprint arXiv:1708.03731*, 2017.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. Instruction mining: Instruction data selection for tuning large language models. *arXiv preprint arXiv:2307.06290*, 2023.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*, 2019.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

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*, pp. 770–778, 2016.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL `https://arxiv.org/abs/2106.09685`.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

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*, pp. 5464–5474. PMLR, 2021a.

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, pp. 8110–8118, 2021b.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34: 20596–20607, 2021.

Omead Pooladzandi, David Davini, and Baharan Mirzasoleiman. Adaptive second order coresets for data-efficient machine learning. In *International Conference on Machine Learning*, pp. 17848–17869. PMLR, 2022.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Amélie Royer and Christoph Lampert. A flexible selection scheme for minimum-effort transfer learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2191–2200, 2020.

Ayrton San Joaquin, Bin Wang, Zhengyuan Liu, Nicholas Asher, Brian Lim, Philippe Muller, and Nancy Chen. In2core: Leveraging influence functions for coreset selection in instruction finetuning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10324–10335, 2024.

Tushar Shinde and Manasa Madabhushi. Data-efficient and robust coreset selection via sparse adversarial perturbations. In *NeurIPS 2025 Workshop: Reliable ML from Unreliable Data*.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493*, 2023.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Neil C Thompson, Kristjan Greenewald, Keeheon Lee, Gabriel F Manso, et al. The computational limits of deep learning. *arXiv preprint arXiv:2007.05558*, 10:2, 2020.

Dmytro Vitel and Anshuman Chhabra. First is not really better than last: Evaluating layer choice and aggregation strategies in language model data influence estimation. *arXiv preprint arXiv:2511.04715*, 2025.

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

Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *The Eleventh International Conference on Learning Representations*, 2022.

Chih-Kuan Yeh, Ankur Taly, Mukund Sundararajan, Frederick Liu, and Pradeep Ravikumar. First is better than last for language data influence. *Advances in Neural Information Processing Systems*, 35:32285–32298, 2022.

Dylan Zhang, Qirun Dai, and Hao Peng. The best instruction-tuning data are those that fit. *arXiv preprint arXiv:2502.04194*, 2025.

A CRAIG BACKGROUND AND GREEDY SELECTION

A.1 HIDDEN ACTIVATIONS AND LAST-LAYER GRADIENTS

Following the gradient-proxy motivation of CRAIG Mirzasoleiman et al. (2020), consider the final affine layer $z_i = W\phi_i + b$, where $\phi_i = \phi(x_i)$ is its input representation and $\delta_i = \nabla_{z_i}\ell(x_i, y_i)$ is the corresponding error signal. Using column-vector representations, the weight gradient is

$$\nabla_W \ell(x_i, y_i) = \delta_i \phi_i^\top. \tag{7}$$

To derive Equation 2 in the main text, we add and subtract $\delta_i\phi_j^\top$, obtaining

$$\begin{aligned} &\|\nabla_W \ell(x_i, y_i) - \nabla_W \ell(x_j, y_j)\|_{\mathrm{F}} \\ &= \left\|\delta_i(\phi_i - \phi_j)^\top + (\delta_i - \delta_j)\phi_j^\top\right\|_{\mathrm{F}} \\ &\le \|\phi_i - \phi_j\|_2 \, \|\delta_i\|_2 + \|\phi_j\|_2 \, \|\delta_i - \delta_j\|_2. \end{aligned} \tag{8}$$

The inequality follows from the triangle inequality and $\|uv^\top\|_{\mathrm{F}} = \|u\|_2\|v\|_2$. The first term links activation distance to last-layer gradient distance, while the second captures differences in the error signals. Small activation distances alone therefore do not ensure small gradient distances. They provide a useful surrogate when the error signals are suitably bounded and similar; extending this argument to full-model gradients requires additional assumptions.

A.2 THE FACILITY-LOCATION SELECTION MECHANISM

We employ greedy facility-location selection to choose representative examples that cover the dataset in the chosen proxy space. This procedure is shared by the probing variants, allowing the comparison to focus on the source of the proxy vectors. The greedy approximation property concerns this proxy-space objective and does not itself establish a guarantee for full-gradient matching.

Algorithm 2 CRAIG-Greedy (Facility Location Selection)

Require: Candidate Vectors $\mathbf{P} = \{v_1, \ldots, v_N\}$ (Hidden States), subset size m.
Ensure: Selected indices S.
Initialize: $S \leftarrow \emptyset, \mathcal{U} \leftarrow \{1, \ldots, N\}$.
Distance Cache: $D_{min}[i] \leftarrow \infty, \forall i \in \mathcal{U}$.
for $k = 1 \rightarrow m$ **do**
  Find best candidate j^***:**
  $j^* \leftarrow \arg\max_{j \in \mathcal{U}} \sum_{i \in \mathcal{U}} \max(0, D_{min}[i] - \text{dist}(v_i, v_j))$
  ▷ Select point maximizing marginal gain
  Update Set and Cache:
  $S \leftarrow S \cup \{j^*\}$
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \{j^*\}$
  for $i \in \mathcal{U}$ **do**
    $D_{min}[i] \leftarrow \min(D_{min}[i], \text{dist}(v_i, v_{j^*}))$
  end for
end for
return S

B CRAIG-LAST

Algorithm 3 CRAIG-Last (Dynamic Last-Layer Proxies)

Require: $\mathcal{D}_{train}$, epochs E, warmup K, subset size m, θ_0.
Ensure: Fine-tuned model parameters θ_E.
Phase 1: Warmup
for $e = 1 \to K$ **do**
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{train})$
end for
Phase 2: Incremental Selection Loop
for $e = K+1 \to E$ **do**
  At start of epoch e**:**
    1. Forward pass θ_{e-1} on $\mathcal{D}_{train}$.
    2. Extract last layer embeddings: $\mathbf{P}_{sel} \leftarrow \mathbf{P}_{last}$.
    3. Normalize: $\tilde{\mathbf{P}}_{sel} \leftarrow \text{Normalize}(\mathbf{P}_{sel})$.
    4. $S_e \leftarrow \text{CRAIG-Greedy}(\tilde{\mathbf{P}}_{sel}, m)$.
  $\mathcal{D}_{subset} \leftarrow \mathcal{D}_{train}[S_e]$
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{subset})$
end for
return θ_E

C RANDOM SELECTION ALGORITHM

Algorithm 4 Baseline: Random Selection

Require: Full training dataset $\mathcal{D}_{train}$, subset size m, epochs E.
Ensure: Trained model parameters θ_E.
Initialize: θ_0.
for $e = 1 \to E$ **do**
  At start of epoch e**:**
  Randomly sample indices S_e of size m from $\{1, \ldots, |\mathcal{D}_{train}|\}$.
  $\mathcal{D}_{subset} \leftarrow \mathcal{D}_{train}[S_e]$
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{subset})$
end for
return θ_E

D CRAIG-RL

Algorithm 5 CRAIG-RL (Dynamic Random-Layer Proxies)

Require: $\mathcal{D}_{train}$, epochs E, warmup K, subset size m, θ_0.
Ensure: Fine-tuned model parameters θ_E.
Phase 1: Warmup (As above)
Phase 2: Incremental Selection Loop
for $e = K+1 \to E$ **do**
  At start of epoch e**:**
    1. Forward pass θ_{e-1} on $\mathcal{D}_{train}$.
    2. Sample random layer index $l_{rand} \sim \text{Uniform}(1, L)$.
    3. Extract random layer embeddings: $\mathbf{P}_{sel} \leftarrow \mathbf{P}_{l_{rand}}$.
    4. Normalize: $\tilde{\mathbf{P}}_{sel} \leftarrow \text{Normalize}(\mathbf{P}_{sel})$.
    5. $S_e \leftarrow \text{CRAIG_Greedy}(\tilde{\mathbf{P}}_{sel}, m)$.
  $\mathcal{D}_{subset} \leftarrow \mathcal{D}_{train}[S_e]$
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{subset})$
end for
return θ_E

E CRAIG-FT

Algorithm 6 CRAIG-FT (Fast FlexTuning Implementation)

Require: $\mathcal{D}_{train}, \mathcal{D}_{val}$, epochs E, warmup K, subset size m.
Ensure: Fine-tuned model parameters θ_E.
Phase 1: Warmup
for $e = 1 \to K$ **do**
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{train})$
end for
Identify Best Layer (Fast Probe):
▷ Train all layers once, then test individual contributions
$\theta_{temp} \leftarrow \text{TrainOneEpoch}(\theta_K, \mathcal{D}_{train})$ ▷ Full fine-tuning step
for $l = 1 \to L$ **do**
▷ Construct proxy model θ'_l: Layer l from θ_{temp}, others from θ_K
  $\theta'_l \leftarrow \text{Combine}(\theta_{temp}[l], \theta_K[\setminus l])$
  $Acc_l \leftarrow \text{Evaluate}(\theta'_l, \mathcal{D}_{val})$
end for
$l_{best} \leftarrow \arg\max_l Acc_l$
Phase 2: Incremental Selection Loop
for $e = K + 1 \to E$ **do**
  At start of epoch e**:**
    1. Forward pass θ_{e-1} on $\mathcal{D}_{train}$.
    2. Extract embeddings from l_{best}: $\mathbf{P}_{sel} \leftarrow \mathbf{P}_{l_{best}}$.
    3. Normalize: $\tilde{\mathbf{P}}_{sel} \leftarrow \text{Normalize}(\mathbf{P}_{sel})$.
    4. $S_e \leftarrow \text{CRAIG-Greedy}(\tilde{\mathbf{P}}_{sel}, m)$.
  $\mathcal{D}_{subset} \leftarrow \mathcal{D}_{train}[S_e]$
  $\theta_e \leftarrow \text{TrainOneEpoch}(\theta_{e-1}, \mathcal{D}_{subset})$
end for
return θ_E

F Dataset Curation and Filtering

We curated a massive repository of datasets from OpenML Bischl et al. (2017), employing a multi-stage filtering process to ensure data quality and model learnability. The selection pipeline yielded a final suite of **517 classification** and **146 regression** tasks.

Filtering Criteria. We queried the OpenML database for active, verified datasets with sample sizes $N \in [1000, 3000]$ and feature dimensionality $d \leq 1000$. We strictly excluded datasets with missing values to prevent artifacts from imputation from confounding the geometric analysis.

Convergence Verification. To ensure that variance in results is attributable to the selection proxy rather than data noise, we trained a probe FeedForward Network (Architecture: $[512, 512, 256, 256, 128]$) for 30 epochs on each candidate dataset. We retained only those where the probe achieved non-trivial convergence (Accuracy/F1 ≥ 0.6 for classification; $R^2 \geq 0.0$ for regression). The search criteria are summarized in Table 3.

Table 3: Dataset Search and Filtering Criteria.

Criterion	Classification	Regression
Source	OpenML	OpenML
Samples (N)	$500 - 3k$	$500 - 3k$
Features (d)	≤ 1000	≤ 1000
Metric Thresh.	F1 ≥ 0.6	$R^2 \geq 0.0$
Count	**517**	**146**

G Extended Experiments: Adapting to Vision

While our primary large-scale analysis focuses on tabular data to ensure statistical breadth, validating our hypotheses on high-dimensional perceptual data is crucial. We adapted the CRAIG framework from simple Feed-Forward Networks (FFNs) to Convolutional Neural Networks (CNNs) using the CIFAR-10 and MNIST datasets. This transition necessitates specific architectural and algorithmic adaptations to handle spatial hierarchies and pre-trained weights.

G.1 Architecture and Proxy Extraction

We replace the configurable MLP used in tabular experiments with a **ResNet-18** architecture pre-trained on ImageNet. This introduces significant complexity compared to the sequential linear layers of the FFN. To map our layer-selection hypotheses (CRAIG-First/Last/RL/FT) to this deeper architecture, we employ a *Block-Level Aggregation* strategy.

Unlike the direct hidden state access in simple MLPs, extracting intermediate representations from a ResNet requires registering forward hooks. We capture activations from 18 individual components, including the initial convolution (`conv1`), the four internal residual blocks (`layer1` through `layer4`), and the final fully connected layer (`fc`). To convert 4D convolutional feature maps (Batch $\times$ Channels $\times$ Height $\times$ Width) into compatible 1D proxy vectors for facility location, we apply Global Average Pooling:

$$\mathbf{P}_{conv} = \text{AdaptiveAvgPool2D}(\mathbf{A}_{hook}, (1, 1)) \tag{9}$$

where $\mathbf{A}_{hook}$ is the activation tensor.

To maintain methodological consistency with the 5-6 layer depth of our tabular models, we aggregate the 18 atomic layers into 6 logical blocks: `conv1`, `layer1`, `layer2`, `layer3`, `layer4`, and `fc`. For methods like CRAIG-FT and CRAIG-RL, the selection algorithm operates over these aggregated blocks, concatenating internal activations where necessary to represent the block's geometry.

G.2 Algorithmic Adaptations and Geometric Consistency

The transition from tabular MLP architectures to Convolutional Neural Networks requires re-validating the mathematical assumptions underlying the CRAIG framework, specifically the gradient upper bound derived in Eq. 2 and the subspace decomposition in Equation 4.

Tensor-to-Vector Projection ($\phi(x)$). In Section 3, the gradient proxy relies on the Euclidean distance between activation vectors $\|\phi(x_i) - \phi(x_j)\|_2$. However, in a ResNet-18, the raw hidden state $\mathbf{H}_l$ at layer l is a 4D tensor in $\mathbb{R}^{B\times C\times H\times W}$. To satisfy the vector space requirement of the Facility Location algorithm without violating the geometric bounds, we employ Global Average Pooling (GAP) as a projection operator $\mathcal{P}$:

$$\phi_l(x) = \mathcal{P}(\mathbf{H}_l(x)) = \frac{1}{H \times W} \sum_{h,w} \mathbf{H}_{l,:,h,w}(x) \tag{10}$$

This operation maps the spatial manifold to a flat feature vector $\phi_l(x) \in \mathbb{R}^C$, ensuring that the pairwise distances calculated during selection remain valid proxies for the gradient difference norm $\|\nabla_W \ell_i - \nabla_W \ell_j\|_F$.

Kernel-Level Velocity and Active Subspaces. For CRAIG-NI, the concept of "Active Nodes" must be adapted from scalar neurons to convolutional kernels. Recall that Equation 4 decomposes the error into active and inactive weight subspaces $\mathcal{W}_{active}$. In a convolutional layer, a single "feature" is generated by a 3D kernel tensor $K \in \mathbb{R}^{C_{in}\times k\times k}$.

To identify the active subspace $\mathcal{I}_{active}$, we aggregate the Parameter Velocity $\boldsymbol{\nu}$ over the input and spatial dimensions. For the j-th output channel (node), the velocity is defined as:

$$\boldsymbol{\nu}^{(j)} = \sum_{c=1}^{C_{in}} \sum_{u=1}^{k} \sum_{v=1}^{k} \left| \theta_{t+1}^{(j,c,u,v)} - \theta_t^{(j,c,u,v)} \right| \tag{11}$$

We then select the top $M\%$ of channels with the highest aggregated $\boldsymbol{\nu}^{(j)}$. The *Virtual Gradient* $\mathbf{V}_x$ is subsequently constructed by concatenating the pooled representations $\phi(x)$ corresponding exclusively

to these active channels. This ensures that our selection mechanism minimizes the geometric error specifically within the high-velocity subspace that dominates the training trajectory.

H Extended Experiments: Adapting to LLMs

To test the universality of our findings beyond tabular regression, we extend our evaluation to the domain of Generative AI, specifically Instruction Tuning for Large Language Models (LLMs). This transition introduces significant challenges in dimensionality and architecture. We address these by adapting the CRAIG framework to operate within the constraints of Parameter-Efficient Fine-Tuning (PEFT).

We utilize **Gemma-3-1b** as our base model. The model is fine-tuned on the Databricks-Dolly-15k dataset, a standard corpus for instruction following containing 15k high-quality human-generated records. To ensure training stability and manage computational constraints during the data selection process, we filtered the dataset based on token length. We analyzed the distribution of instruction-response lengths and applied a cutoff to retain approximately 90% of the data, removing extreme outliers that could skew the proxy embeddings or unnecessarily exceed context windows.

Percentile	Length Filter (Words)	Data Retained
50%	≤ 45	50.8% (7,624 examples)
75%	≤ 85	75.2% (11,288 examples)
85%	≤ 118	85.1% (12,779 examples)
90% (Selected)	$\mathbf{\leq 150}$	**90.1% (13,519 examples)**
95%	≤ 218	95.0% (14,266 examples)

Table 4: Distribution of sequence lengths in the Dolly-15k dataset. We selected the 90th percentile filter (≤ 150 words) for our experiments.

After having a suitable dataset, we also adopt a *step-based* training paradigm to construct incremental coresets instead of performing coreset selection every epoch. As for instruction tuning in LLMs, it is typical to only train for 1-3 epochs Ouyang et al. (2022); Zhang et al. (2025), and hence we rely on training steps. The training consists of a fixed warmup of 30 steps, followed by active selection intervals of 30 steps. We employ 4-bit quantization (QLoRA) Dettmers et al. (2023) to manage memory footprints, using a batch size of 16 with gradient accumulation steps of 4 (Effective Batch Size = 64).

H.1 Low-Rank Gradient Proxies

A critical theoretical question is whether the hidden state proxy $\phi(x)$ defined in Eq. 2 remains valid when the model is not fully updated. In our setup, we freeze the pre-trained weights W_0 and optimize rank-r decomposition matrices A and B such that the weight update is given by $W = W_0 + \Delta W = W_0 + BA$. Recall that for a standard linear layer, $\nabla_W \mathcal{L} = \delta\phi(x)^\top$. By the chain rule, the gradients with respect to the adapter matrices A and B are projections of the full weight gradient:

$$\nabla_B \mathcal{L} = \nabla_W \mathcal{L} \cdot A^\top = (\delta\phi(x)^\top)A^\top = \delta(\phi(x)^\top A^\top) \tag{12}$$

Eq. 12 demonstrates that the gradients of the trainable parameters B are explicitly linearly dependent on the input activation $\phi(x)$. Consequently, the geometric distance bound derived in Eq. 2 still holds: ensuring coverage in the activation space $\phi(x)$ guarantees coverage of the gradient space for the adapters. For the specific case of Causal Language Modeling, we define the representative embedding $\phi(x)$ as the hidden state of the *last token* at a given layer, under the standard assumption that causal attention aggregates the sequence information into this final position.

H.2 Adapting CRAIG Variants for Transformers

The transition to Transformers requires specific modifications to the proxy extraction and velocity computation mechanisms:

Hook-Based Proxy Extraction: Unlike the simple feed-forward architecture, where activations are directly accessible, we utilize PyTorch forward hooks to intercept $2\phi(x)$ during the forward pass.3 For a sequence of length T, we extract $\phi(x) \in \mathbb{R}^d$ corresponding to the token at position T (the last token) to represent the instruction-response pair.

CRAIG-NI with Merged Velocity: Calculating Parameter Velocity $\boldsymbol{\nu}$ for CRAIG-NI becomes non-trivial due to quantization. Since the base weights are 4-bit quantized and frozen, we compute the effective velocity by virtually merging the LoRA adapters. Let $\theta^{(t)} = W_0 + B^{(t)}A^{(t)}$. The velocity for a specific layer is computed as the element-wise L1 norm of the delta between steps:

$$\boldsymbol{\nu}^{(t)} = \left|(B^{(t)}A^{(t)}) - (B^{(t-1)}A^{(t-1)})\right| 1 = \sum i,j \left|(B^{(t)}A^{(t)})ij - (B^{(t-1)}A^{(t-1)})ij\right| \tag{13}$$

This metric captures the aggregate magnitude of updates flowing through each Transformer block, allowing us to identify the *active* subspace even when the base model parameters W_0 remain static.

H.2.1 LLM Evaluation Metrics

For language modeling, the primary metric of quality is Perplexity (PPL) Radford et al. (2019); Shumailov et al. (2023), defined as the exponentiation of the cross-entropy loss: $\text{PPL} = \exp(\mathcal{L})$. To maintain consistency with the regression metrics, where higher values of Q indicate superiority, we invert the ratio for the Quality metric. Let S_{conv} be the number of steps required to reach a convergence threshold (loss $\leq \lambda$). **Quality (Q_m):** The ratio of the baseline's best perplexity to the method's best perplexity.

$$Q_m = \frac{\min(\text{PPLRandom})}{\min(\text{PPL}m)} \tag{14}$$

Cost (C_m): The ratio of steps taken to converge.

$$C_m = \frac{S_{\text{conv}}(m)}{S_{\text{conv}}(\text{Random})} \tag{15}$$

Efficiency (E_m): The aggregate score balancing generation quality and training speed.

$$E_m = \frac{Q_m}{C_m} \tag{16}$$

Consistent with the regression tasks, an efficiency score $E_m \gg 1.0$ indicates that the method achieves lower perplexity (better understanding) in fewer training steps than random selection.

I TABULAR BENCHMARK: CLASSIFICATION

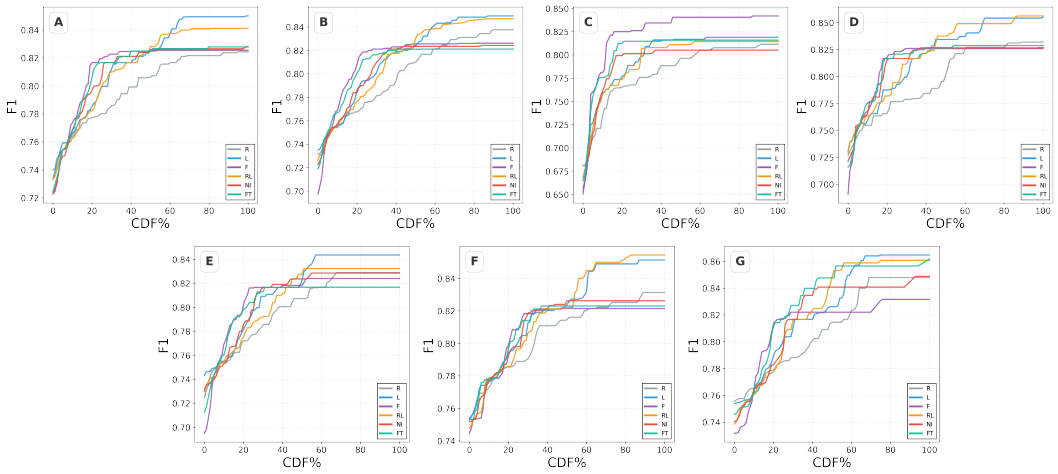

Figure 3: Trajectory analysis for the classification Experiments. Combined visualization of F1 score progression as a function of cumulative data fraction (CDF%). (A, B) Results grouped by neural network architecture. (C–G) Results grouped by coreset size: 3% (C), 5% (D), 7% (E), 10% (F), and 15% (G). Each curve represents the median best-yet F1 score across all datasets for each method. Methods: R (Random), L (CRAIG-Last), F (CRAIG-First), RL (CRAIG-RL), NI (CRAIG-NI), FT (CRAIG-FT).

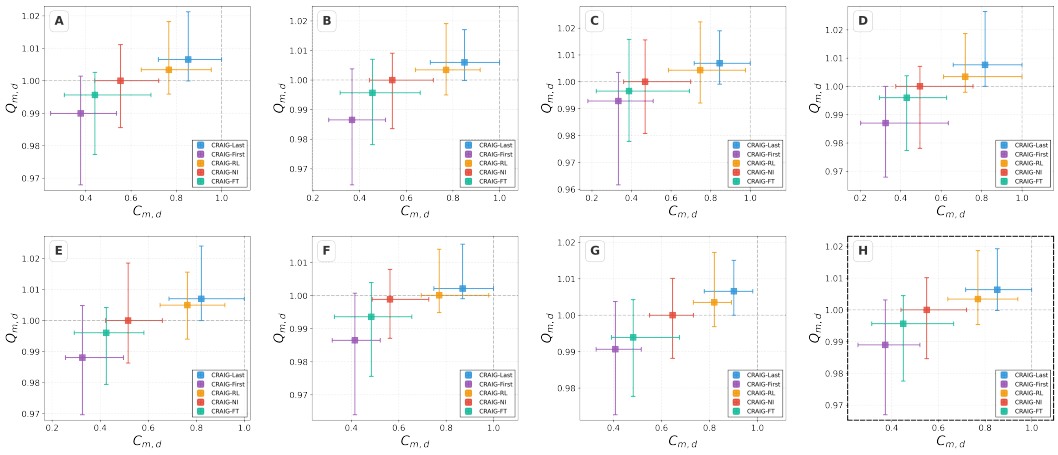

Figure 4: Quality vs Cost Trade-off Analysis for Classification. Scatter plots showing the trade-off between normalized quality ($Q_{m,d}$) and normalized cost ($C_{m,d}$) for each coreset selection method. (A, B) Results grouped by architecture. (C–G) Results grouped by coreset size: 3% (C), 5% (D), 7% (E), 10% (F), and 15% (G). (H) Cumulative results across all configurations (highlighted with dashed border). Points represent median values with error bars showing interquartile range. Reference lines at $Q = 1$ and $C = 1$ indicate Random baseline performance.

J TABULAR BENCHMARK: REGRESSION

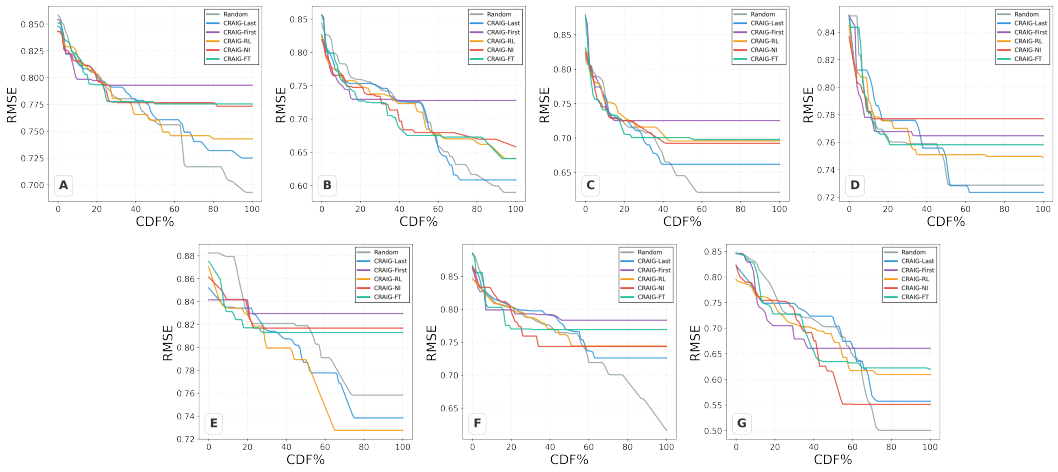

Figure 5: Trajectory analysis for the regression Experiments. Combined visualization of RMSE progression as a function of cumulative data fraction (CDF%). (A, B) Results grouped by neural network architecture. (C–G) Results grouped by coreset size: 3% (C), 5% (D), 7% (E), 10% (F), and 15% (G). Each curve represents the median best-yet RMSE across all datasets for each method. Lower RMSE indicates better performance.

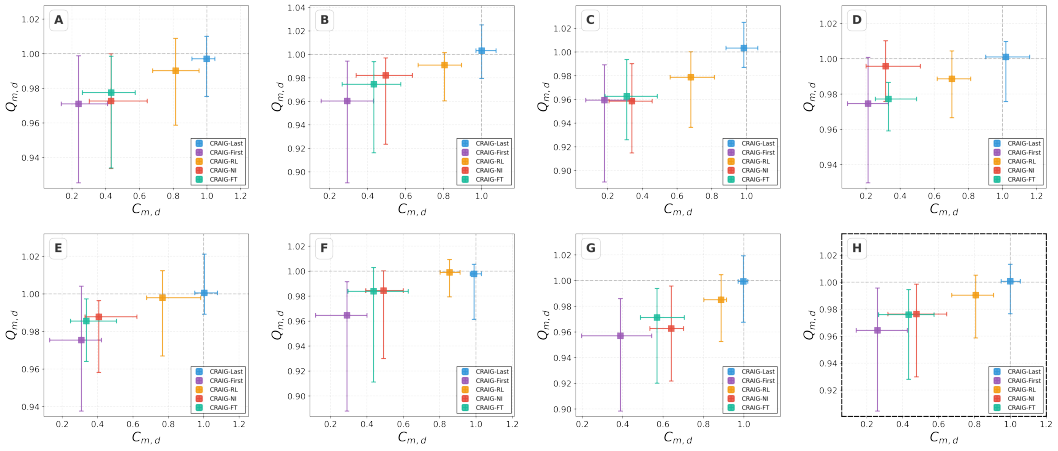

Figure 6: Quality vs Cost Trade-off Analysis for Regression. Scatter plots showing the trade-off between normalized quality ($Q_{m,d}$) and normalized cost ($C_{m,d}$) for each coreset selection method on regression tasks. (A, B) Results grouped by architecture. (C–G) Results grouped by coreset size: 3% (C), 5% (D), 7% (E), 10% (F), and 15% (G). (H) Cumulative results across all configurations (highlighted with dashed border). Quality is defined as Random RMSE / Method RMSE (higher is better).

Method	All (n=146)				Coreset 3% (n=32)				Coreset 5% (n=27)				Coreset 7% (n=21)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
CRAIG-Last	1.001	1.000	1.751	1.000	1.003	0.985	2.912	1.028	1.001	1.020	0.327	0.972	1.000	1.004	2.677	0.985
CRAIG-First	0.964	0.260	0.287	**3.634**	0.959	0.181	0.311	**5.298**	0.975	0.212	0.171	**4.464**	0.975	0.309	0.256	**3.364**
CRAIG-RL	0.990	0.808	0.473	1.202	0.979	0.679	0.604	1.425	0.989	0.704	0.284	1.395	0.998	0.768	0.325	1.240
CRAIG-NI	0.976	0.477	0.348	1.989	0.959	0.340	0.321	2.796	0.996	0.315	0.206	3.134	0.988	0.408	0.402	2.387
CRAIG-FT	0.976	0.433	0.323	2.328	0.963	0.310	0.251	3.014	0.977	0.332	0.167	2.739	0.985	0.337	0.271	2.998

Method	Coreset 10% (n=28)				Coreset 15% (n=38)				Arch [256x5] (n=68)				Arch [512..128] (n=78)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
CRAIG-Last	0.998	0.989	0.513	0.997	0.999	0.998	0.328	1.001	1.003	1.001	0.763	1.003	0.997	0.998	2.283	0.992
CRAIG-First	0.965	0.291	0.281	**3.454**	0.957	0.390	0.317	**2.453**	0.960	0.293	0.258	**3.369**	0.971	0.239	0.311	**3.823**
CRAIG-RL	0.999	0.855	0.417	1.147	0.985	0.887	0.525	1.119	0.991	0.805	0.403	1.209	0.990	0.815	0.524	1.188
CRAIG-NI	0.984	0.492	0.251	1.888	0.963	0.641	0.388	1.481	0.982	0.495	0.317	1.942	0.973	0.432	0.372	2.228
CRAIG-FT	0.984	0.437	0.471	2.289	0.971	0.570	0.289	1.641	0.975	0.433	0.361	2.289	0.977	0.433	0.287	2.373

Table 5: Regression experiment

Method	All (n=10)				Coreset 3% (n=2)				Coreset 5% (n=2)				Coreset 7% (n=2)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
CRAIG-Last	1.008	0.902	0.022	1.117	1.003	0.889	0.012	1.129	1.009	0.895	0.011	1.127	1.008	0.897	0.007	1.124
CRAIG-First	1.000	0.845	0.063	1.184	0.995	0.739	0.016	**1.348**	1.001	0.804	0.026	**1.246**	0.999	0.845	0.022	1.184
CRAIG-RL	1.003	0.933	0.021	1.077	0.999	0.924	0.006	1.081	1.005	0.920	0.013	1.092	1.003	0.914	0.035	1.099
CRAIG-NI	1.008	0.845	0.032	**1.196**	1.005	0.886	0.023	1.134	1.008	0.839	0.005	1.202	1.008	0.826	0.001	**1.220**
CRAIG-FT	1.005	0.909	0.023	1.106	1.003	0.922	0.006	1.087	1.008	0.896	0.009	1.124	1.005	0.889	0.020	1.131

Method	Coreset 10% (n=2)				Coreset 15% (n=2)				MNIST (n=5)				CIFAR-10 (n=5)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
CRAIG-Last	1.009	0.898	0.004	1.123	1.008	0.945	0.003	1.066	1.005	0.901	0.018	1.113	1.011	0.902	0.025	1.121
CRAIG-First	1.002	0.863	0.034	1.162	1.001	0.927	0.016	1.081	1.002	0.866	0.064	1.156	0.998	0.823	0.062	**1.212**
CRAIG-RL	1.006	0.927	0.009	1.085	1.006	0.954	0.005	1.054	1.003	0.930	0.024	1.080	1.004	0.936	0.019	1.074
CRAIG-NI	1.011	0.844	0.001	**1.197**	1.007	0.907	0.004	**1.110**	1.007	0.844	0.036	**1.195**	1.010	0.846	0.027	1.197
CRAIG-FT	1.007	0.907	0.003	1.111	1.006	0.949	0.001	1.060	1.004	0.910	0.027	1.105	1.007	0.909	0.020	1.108

Table 6: Consolidated Quality vs Cost Statistics for Classification (ResNet18)

K RESNET18

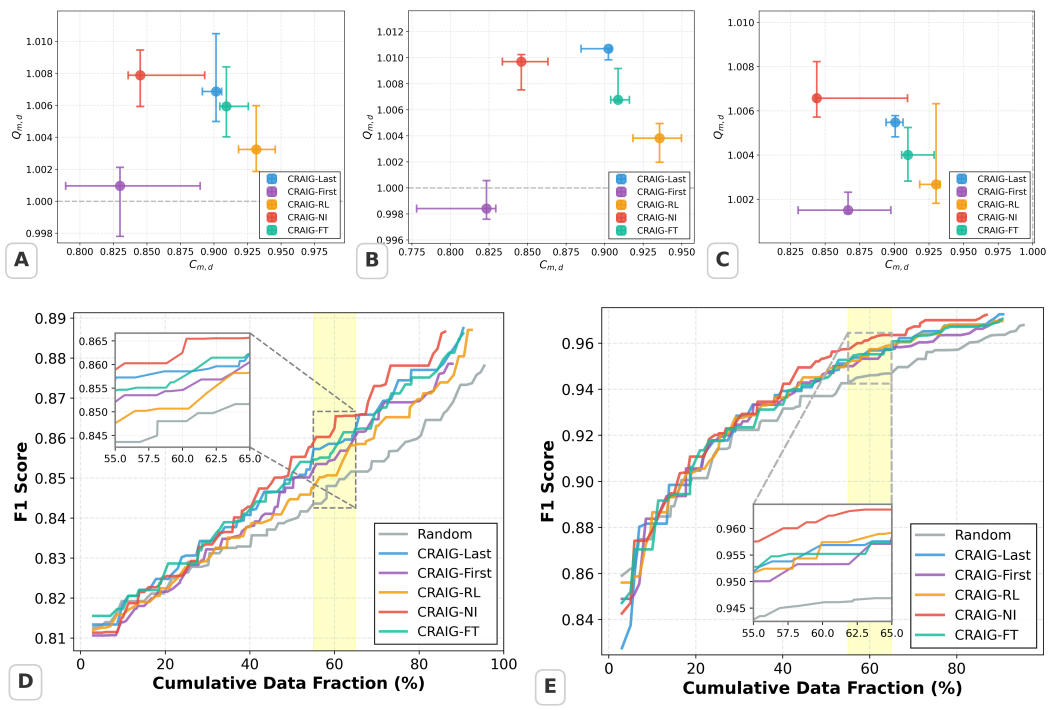

Figure 7: Vision Experiments with ResNet-18. Comprehensive analysis of coreset selection methods on vision tasks using ResNet-18. Top row: Quality vs cost trade-off analysis for (A) combined results, (B) CIFAR-10, and (C) MNIST. Bottom row: trajectory analysis for (D) CIFAR-10 and (E) MNIST, showing the effectiveness of informed coreset selection across different image classification benchmarks.

L GEMMA3-1B

Table 7: Consolidated Quality vs Cost Statistics for LLM Gemma3, here Q represents the PPL with a lesser PPL being the better score.

Method	All (n=5)				Coreset 3% (n=1)			Coreset 5% (n=1)			Coreset 7% (n=1)		
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	Q	C	E	Q	C	E	Q	C	E
CRAIG-Last	0.943	0.293	0.024	**3.238**	0.941	0.334	**2.822**	0.943	0.281	**3.362**	0.941	0.285	**3.305**
CRAIG-First	0.952	0.337	0.056	2.827	0.958	0.477	2.009	0.952	0.337	2.827	0.949	0.350	2.713
CRAIG-RL	0.989	0.677	0.041	1.453	0.983	0.677	1.453	0.989	0.753	1.313	0.991	0.700	1.416
CRAIG-NI	0.983	0.655	0.035	1.488	0.979	0.667	1.467	0.974	0.655	1.488	0.983	0.611	1.608
CRAIG-FT	0.952	0.307	0.024	3.098	0.955	0.359	2.658	0.952	0.307	3.102	0.949	0.306	3.098

Method	Coreset 10% (n=1)			Coreset 15% (n=1)		
	Q	C	E	Q	C	E
CRAIG-Last	0.950	0.293	3.238	0.969	0.335	2.892
CRAIG-First	0.947	0.334	2.840	0.956	0.335	2.859
CRAIG-RL	0.986	0.633	1.556	0.991	0.660	1.501
CRAIG-NI	0.984	0.675	1.457	0.983	0.583	1.687
CRAIG-FT	0.946	0.291	**3.247**	0.962	0.329	**2.922**

M CROSS-FRAMEWORK VALIDATION WITH GLISTER

To test whether the Last-Layer Bias is an artifact of CRAIG's facility-location objective specifically, we extend our layer-proxy suite to GLISTER Killamsetty et al. (2021b), a validation-anchored bi-level selection method whose selection mechanism, greedy alignment with a validation-set direction, is structurally distinct from CRAIG's coverage-based facility location. We build a ***GLISTER-inspired*** scoring rule: candidates are ranked by their cosine alignment, in the chosen proxy layer's activation space, with the mean validation-set representation, and selected via GLISTER's stochastic greedy procedure (Algorithm 7). Consistent with the activation-as-gradient-proxy assumption used throughout our CRAIG experiments (Section 3), this substitutes GLISTER-Online's original last-layer *gradient* approximation, $\delta(x)\phi(x)^\top$, combining prediction error and penultimate activation, with the activation-only proxy $\phi(x)$. This isolates the layer/proxy axis under a controlled comparison rather than reproducing GLISTER-Online's validation-loss meta-approximation in full. Table 8 reports median normalized quality (Q), normalized sample cost (C), standard deviation of cost (σ_C), and efficiency ($E = Q/C$) across 50 random OpenML classification datasets.

Method	All (n=50)				Coreset 3% (n=12)				Coreset 5% (n=6)				Coreset 7% (n=11)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
GLISTER-Last	0.972	0.300	0.312	3.127	0.958	0.153	0.267	5.270	0.973	0.266	0.316	3.521	0.972	0.217	0.325	4.057
GLISTER-First	0.958	0.120	0.324	7.439	0.931	0.070	0.257	12.352	0.980	0.076	0.365	12.611	0.962	0.117	0.253	7.416
GLISTER-RL	0.977	0.514	0.305	1.935	0.967	0.358	0.325	2.581	0.983	0.446	0.293	2.255	0.987	0.480	0.284	2.042
GLISTER-NI	0.960	0.317	0.657	3.016	0.964	0.153	0.255	5.858	0.968	0.196	0.319	5.292	0.969	0.410	0.286	2.485
GLISTER-FT	0.964	0.153	0.315	5.548	0.932	0.074	0.256	11.367	0.989	0.141	0.352	7.374	0.987	0.131	0.250	4.953

Method	Coreset 10% (n=11)				Coreset 15% (n=10)				Arch [512..128] (n=28)				Arch [256x5] (n=22)			
	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$	$Q\uparrow$	$C\downarrow$	$\sigma_C\downarrow$	$E\uparrow$
GLISTER-Last	0.972	0.414	0.316	2.352	0.978	0.368	0.298	2.461	0.977	0.336	0.291	2.756	0.934	0.195	0.334	4.524
GLISTER-First	0.981	0.127	0.382	7.355	0.962	0.183	0.331	5.039	0.952	0.130	0.305	7.511	0.967	0.115	0.345	7.439
GLISTER-RL	0.972	0.580	0.299	1.724	0.978	0.575	0.278	1.645	0.975	0.521	0.299	1.833	0.983	0.490	0.308	1.987
GLISTER-NI	0.948	0.386	1.243	2.475	0.966	0.432	0.288	2.080	0.967	0.294	0.826	3.384	0.941	0.319	0.323	2.778
GLISTER-FT	0.964	0.237	0.363	3.912	0.962	0.271	0.313	3.493	0.964	0.159	0.298	5.972	0.962	0.153	0.334	5.048

Table 8: Consolidated Quality vs Cost Statistics for GLISTER on Tabular Classification.

Notably, the relative standing of LAST and RL is not consistent across frameworks: CRAIG-RL modestly outperforms CRAIG-Last (E=1.305 vs. 1.179, Table 2), whereas GLISTER-Last outperforms GLISTER-RL by a wide margin (E=3.127 vs. 1.935, Table 8). This suggests the specific failure mode of last-layer selection is mechanism-dependent: under CRAIG's coverage objective, the last layer is redundant with a random layer, while under GLISTER's validation-alignment objective, it retains some signal. The finding that generalizes across both frameworks is narrower but more robust: the last layer is never the *best* available proxy, and input-level (FIRST) or data-driven (FT) proxies consistently dominate it in both tabular settings (E_{First}=2.630, E_{FT}=2.193 for CRAIG; E_{First}=7.439, E_{FT}=5.548 for GLISTER).

Algorithm 7 GLISTER with Different Layer Selection Policies

Require: Warmup model θ; train set $\mathcal{U}$; validation set $\mathcal{V}$; budget m; layer policy $\pi \in \{\text{LAST}, \text{FIRST}, \text{RL}, \text{NI}, \text{FT}\}$; candidate pool size C

$\mathcal{L} \leftarrow$ hidden layers of θ plus output layer
Determine active layer ℓ using π:
  LAST: $\ell \leftarrow \arg\max$ layer index;
  FIRST: $\ell \leftarrow \arg\min$ layer index;
  RL: $\ell \sim \text{Unif}(\mathcal{L})$;
  NI: build NI virtual vectors from top-velocity neurons (cross-layer);
  FT: $\ell \leftarrow \arg\max_{\ell' \in \mathcal{L}}$ validation score from FlexTuning proxy networks
if $\pi = \text{NI}$ **then**
  for each $x \in \mathcal{U}$ **do**
    $\mathbf{p}^{\mathcal{U}}(x) \leftarrow$ concatenated activations of velocity-selected neurons
  end for
  for each $x \in \mathcal{V}$ **do**
    $\mathbf{p}^{\mathcal{V}}(x) \leftarrow$ same NI construction
  end for
else
  for each $x \in \mathcal{U}$ **do**
    $\mathbf{p}^{\mathcal{U}}(x) \leftarrow$ activation of layer ℓ for x
  end for
  for each $x \in \mathcal{V}$ **do**
    $\mathbf{p}^{\mathcal{V}}(x) \leftarrow$ activation of layer ℓ for x
  end for
end if
Normalize rows: $\tilde{\mathbf{p}}(x) \leftarrow \mathbf{p}(x)/\|\mathbf{p}(x)\|_2$
Validation direction: $\mathbf{g}_{\mathcal{V}} \leftarrow \frac{1}{|\mathcal{V}|}\sum_{x \in \mathcal{V}} \tilde{\mathbf{p}}^{\mathcal{V}}(x), \quad \mathbf{g}_{\mathcal{V}} \leftarrow \mathbf{g}_{\mathcal{V}}/\|\mathbf{g}_{\mathcal{V}}\|_2$
Score each training point: $s(x) \leftarrow \langle \tilde{\mathbf{p}}^{\mathcal{U}}(x), \mathbf{g}_{\mathcal{V}} \rangle$
$S \leftarrow \emptyset, \mathcal{R} \leftarrow \mathcal{U}$
for $t = 1$ to m **do**
  Sample candidate set $\mathcal{C} \subseteq \mathcal{R}$, $|\mathcal{C}| \leq C$
  $x^\star \leftarrow \arg\max_{x \in \mathcal{C}} s(x)$
  $S \leftarrow S \cup \{x^\star\}, \mathcal{R} \leftarrow \mathcal{R} \setminus \{x^\star\}$
end for
Train one epoch on S (same incremental loop as CRAIG ablation)
return updated θ