# PPL-Factory: Task-Aware and Budget-Aware Data Selection from Language Modeling to Reasoning

**Hang Zhang, Warren J. Gross**
Department of Electrical and Computer Engineering, McGill University
Montreal, QC, Canada
hang.zhang3@mail.mcgill.ca, warren.gross@mcgill.ca

## Abstract

Not all training samples contribute equally to large language model fine-tuning. Selecting informative training samples can reduce the computational cost while preserving downstream performance. Many existing data selection methods rely on indirect heuristics, such as data quality, diversity or reasoning trace length. However, the effectiveness of these fixed criteria is task-dependent and difficult to generalize across diverse downstream tasks. Perplexity-based data selection provides a simple and model-aware solution to estimate the sample difficulty, but existing approaches typically score the entire training sequence and ignore the difference in learning objectives of language modeling and reasoning tasks. In this paper, we propose **PPL-Factory**, a simple and interpretable data selection framework that combines task-aware perplexity-based scores and data budget-aware selection criteria. Experiments on GSM8K demonstrate that PPL-Factory outperforms other state-of-the-art data selection methods using only $1\%$ of the training set. With $10\%$ of the data, PPL-Factory exceeds full-data fine-tuning accuracy by 0.9 on GSM8K and 4.8 on MATH using a 3B model, respectively. Overall, our results demonstrate that task-aware and budget-aware perplexity-based selection provides an effective and applicable approach for efficient fine-tuning.

## 1 Introduction

Fine-tuning large language models (LLMs) has become a standard approach for adapting pretrained models on diverse downstream tasks. However, full dataset fine-tuning remains computationally expensive, especially when the parameter size for the model or the available training corpus is large (Albalak et al., 2024). Considering that not all training examples give equal learnable signals to the model improvement, data selection and data pruning approaches (Chen et al., 2024; Li et al., 2024a) are proposed to discard redundant, overly hard, noisy or mislabeled samples for LLMs. Identifying a subset of training data that preserves the informative knowledge is therefore an active research topic.

Perplexity offers a simple and model-aware signal for estimating the difficulty of training examples. Recent studies have shown that both high and medium perplexity improve performance and training efficiency (Ankner et al., 2025). However, prior work has shown that many one-shot coreset selection methods suffer from a dramatic accuracy drop under extremely low data-budget settings and can even perform worse than random sampling (Zheng et al., 2023; Xia et al., 2025), due to insufficient data diversity and informative coverage. Embedding-based data selection approaches (Rubin et al., 2022; Ni et al., 2022) utilize a proxy model to assign a score to each data point in the training set based on its similarity with the validation set. However, semantic similarity extracted from a separate frozen model has a weak connection to the learning behavior of the target model. Gradient similarity-based approaches (Xia et al., 2024; Nikdan et al., 2025) evaluate the utility of each data point and compute its training gradient. Influence functions (Koh and Liang, 2017; Dai et al., 2025) estimate how much a training example affects a model prediction without fully retraining the model. Despite their promising performance, these sophisticated scores often introduce additional computational overhead, as they may require gradient construction, gradient-feature storage, Hessian-vector products, or influence estimation (Hammoudeh and Lowd, 2024; Killamsetty et al., 2021). The utility functions and computational complexities for the previously mentioned approaches are compared in Table 1.

To address the above limitations, we propose **PPL-Factory**, a task-aware and data budget-aware framework for efficient LLM fine-tuning. Unlike prior perplexity-based methods that mainly rely on



*Findings of the Association for Computational Linguistics: IJCNLP-AACL 2026*, pages 4025–4038
November 6-10, 2026 ©2026 Association for Computational Linguistics

| Method | Utility Function | Per-example Computational Cost |
|---|---|---|
| Random | $u(x) \sim \mathcal{U}(0,1)$ | $\mathcal{O}(1)$ |
| Perplexity-Based | $u(x) = \mathrm{PPL}_\theta(x)$ | $C_{\text{forward}}(x)$ |
| Embedding-Based | $u(x) = \mathrm{sim}(h_\theta(x), h_\theta(\mathcal{V}))$ | $C_{\text{forward}}(x) + \mathcal{O}(d\lvert\mathcal{V}\rvert)$ |
| Gradient-Based | $u(x) = \lVert\nabla_\theta \ell_\theta(x)\rVert$ | $C_{\text{forward}}(x) + C_{\text{backward}}(x)$ |
| Influence-Based | $u(x) = -\nabla_\theta \ell_\theta(\mathcal{V})^\top H_\theta^{-1} \nabla_\theta \ell_\theta(x)$ | $C_{\text{forward}}(x) + C_{\text{backward}}(x) + C_{\text{IHVP}}$ |

Table 1: Utility functions and per-example computational cost for data selection methods. Random selection, a common strong baseline (Ivison et al., 2025; Zheng et al., 2023), has the lowest computational complexity. Perplexity scoring computation only requires forward passes without gradient updates. $h_\theta(x) \in \mathbb{R}^d$ denotes the embedding of the training example $x$, $\mathcal{V}$ denotes a small validation set. Embedding-based approaches need a one-time preprocessing cost on validation embeddings. $\ell_\theta(x)$ denotes the token-level negative log-likelihood. Both gradient-based and influence-based approaches involve a backward pass, and influence-based utility further requires the inverse Hessian $H_\theta^{-1}$.

deterministic full-sequence difficulty ranking, our method extends the perplexity-based selection from directly aligned language modeling tasks to reasoning tasks and applies a budget-aware selection criterion. For causal language modeling (CLM) tasks, PPL-Factory scores fixed-length text blocks using token-level negative log-likelihood (NLL). As for the reasoning tasks, where the informative signal is not always uniformly distributed across the full sequence, PPL-Factory adapts a weighted combination of reasoning NLL score and answer NLL score. In addition, we study how the best selection criterion changes across diverse data budgets, and propose a budget-aware selection criterion to preserve the model performance, especially under extremely low data budgets. Our main contributions can be summarized as follows:

- We propose **PPL-Factory**, a joint task-aware and budget-aware framework for efficient LLM fine-tuning. PPL-Factory selects informative training subsets based on likelihood within the target model. Experimental results show that our approach surpasses the state-of-the-art methods on GSM8K using only $1\%$ of the training set. It provides a lightweight alternative to existing selection methods that require external evaluators or expensive quality annotations.

- We develop task-aware NLL-based selection scores for language modeling and reasoning tasks within the PPL-Factory framework. Instead of applying conventional full-sequence perplexity ranking, the proposed scores compute NLL over task-characterized regions. Across four widely used benchmarks, our method consistently outperforms prior perplexity-based selection when using $30\%$ or less of the training set, highlighting the robustness and generalizability under low data budgets.

- We systematically evaluate the fine-tuning performance of PPL-Factory across different training data budgets. The results show that the proposed budget-aware selection strategy outperforms full-data fine-tuning on reasoning tasks and achieves comparable performance on CLM tasks while using substantially fewer training examples.

## 2 Preliminaries

**Negative Log-Likelihood and Perplexity** NLL measures how unlikely a sequence is under a language model. Given a token sequence $\mathbf{x} = (x_1, x_2 \ldots, x_T)$, the token-normalized NLL is defined as

$$s_{\text{NLL}}(x;\theta) = -\frac{1}{T}\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}). \quad (1)$$

Perplexity is the exponential form of the token-normalized NLL

$$\mathrm{PPL}(x;\theta) = \exp\left(s_{\text{NLL}}(x;\theta)\right). \quad (2)$$

A lower NLL means that the language model predicts the next token relatively well and confidently, so it reflects an easy sample in the set. Middle-band NLL samples can provide informative learning signals while remaining learnable. High NLL values mean the samples are more surprising to the model and therefore more difficult to learn given the pre-trained parameters. Since perplexity is directly derived from the training objective of autoregressive language models, it provides a model-aware measure of sample difficulty.

**One-shot Data Selection** Let $\mathcal{D} = \{z_i\}_{i=1}^{N}$ denote a training dataset with $N$ examples, where each $z_i$ represents a training unit. One-shot data selection aims to select a training subset $\mathcal{D}_\rho \subseteq \mathcal{D}$ before training, given a selection ratio $\rho \in (0, 1]$, such that the evaluation performance on the target dataset can be preserved. This optimization objective can be formulated as

$$\begin{aligned} \arg\min_{\mathcal{D}_\rho \subseteq \mathcal{D}} \ & \mathcal{L}\left(\theta^*(\mathcal{D}_\rho); \mathcal{D}_{\text{eval}}\right), \\ \text{s.t.} \quad & |\mathcal{D}_\rho| = \lfloor \rho N \rfloor, \end{aligned} \tag{3}$$

where $\mathcal{L}(\cdot)$ denotes the loss function, and $\theta^*(\mathcal{D}_\rho)$ denotes the model parameters trained on the selected subset $\mathcal{D}_\rho$.

**Supervised Fine-tuning.** Given a certain dataset containing target sequences, the training strategy is to minimize the NLL of the target sequence $y$ given an input $x$, which is,

$$\mathcal{L}_{\text{SFT}}(\theta) = -\sum_{t=1}^{T} \log p_\theta(y_t \mid x, y_{<t}), \tag{4}$$

where $y_{<t} = (y_1, \cdots, y_{t-1})$ denotes the prefix before the $t$-th token in the target response.

## 3 Methodology

Figure 1 provides an overview of our proposed PPL-Factory. Given a training corpus, PPL-Factory computes task-aware NLL scores using the frozen pre-trained LLM and applies a budget-aware selection rule to obtain a subset for efficient fine-tuning. Then the subset is used to fine-tune the pre-trained LLM to improve the downstream performance.

In the following subsections, we first introduce the theoretical motivation in Section 3.1. We then introduce the task-aware adaptation to the NLL scores for diverse downstream tasks in Section 3.2, from CLM to mathematical reasoning tasks. Finally, Section 3.3 presents our budget-aware selection criteria.

### 3.1 Theoretical Motivation

We consider an offline, one-shot data selection approach for efficient fine-tuning. Let the original training corpus be denoted as $\mathcal{D}$ and let $\rho \in (0, 1]$ denote the data selection ratio. Data selection constructs a subset $\mathcal{D}_\rho$ which is selected from the full-size dataset $\mathcal{D}$ according to a certain criterion, where $\mathcal{D}_\rho$ contains a fraction $\rho$ of the available training samples. Thus, the selection criterion is to identify informative and learnable signals from the original training set and maximize the training utility of the selected subset.

Let $s$ denote the NLL of a training example, which serves as a model-aware difficulty signal for data selection. For a given selection ratio $\rho$, we conceptually define the utility of retaining an example with score $s$ as

$$u_\rho(s) = r_\rho(s) - \gamma_\rho c_\rho(s), \tag{5}$$

where $r_\rho(s)$ is the positive training contribution to fine-tuning and $c_\rho(s)$ denotes the potential negative effects caused by noise, incorrect label, or outlier difficulty. The coefficient $\gamma_\rho$ controls the penalty assigned to the negative effect.

While Equation 5 characterizes the utility of individual examples, an effectively selected subset should keep sufficient coverage of the original distribution. Let $P(s)$ denote the NLL distribution of the full train set, and let $Q_\rho(s)$ denote the selection score distribution of the selected subset under selection ratio $\rho$. We formulate data selection as an informative-coverage maximization problem, where the selected subset is encouraged to retain useful training signals while maintaining distributional coverage with respect to the full corpus

$$\begin{aligned} J_\rho(Q_\rho) = & \int u_\rho(s) Q_\rho(s)\, ds \\ & - \lambda_\rho D_{\text{KL}}(Q_\rho \parallel P), \quad \lambda_\rho > 0. \end{aligned} \tag{6}$$

The first term measures the expected utility of the selected subset, while the KL-divergence term penalizes excessive deviation from the original score distribution. The parameter $\lambda_\rho$ controls the trade-off between concentrating on a high-utility score region and increasing coverage of the full training set. Overall, the optimization problem of the data selection for efficient fine-tuning is now transferred to maximizing the informative-coverage score under a selection ratio $\rho$ and $\lambda_\rho > 0$. For a fixed ratio $\rho$, the corresponding maximizer of Equation 6 has the standard Gibbs form

$$Q_\rho^*(s) = \frac{P(s) \exp(u_\rho(s)/\lambda_\rho)}{\int P(t) \exp(u_\rho(t)/\lambda_\rho)\, dt}. \tag{7}$$

This formulation illustrates that, when the relative utility of examples varies with the data budget, the preferred region may also vary with $\rho$. This observation motivates us to empirically investigate budget-dependent selection strategies in Section 3.3.

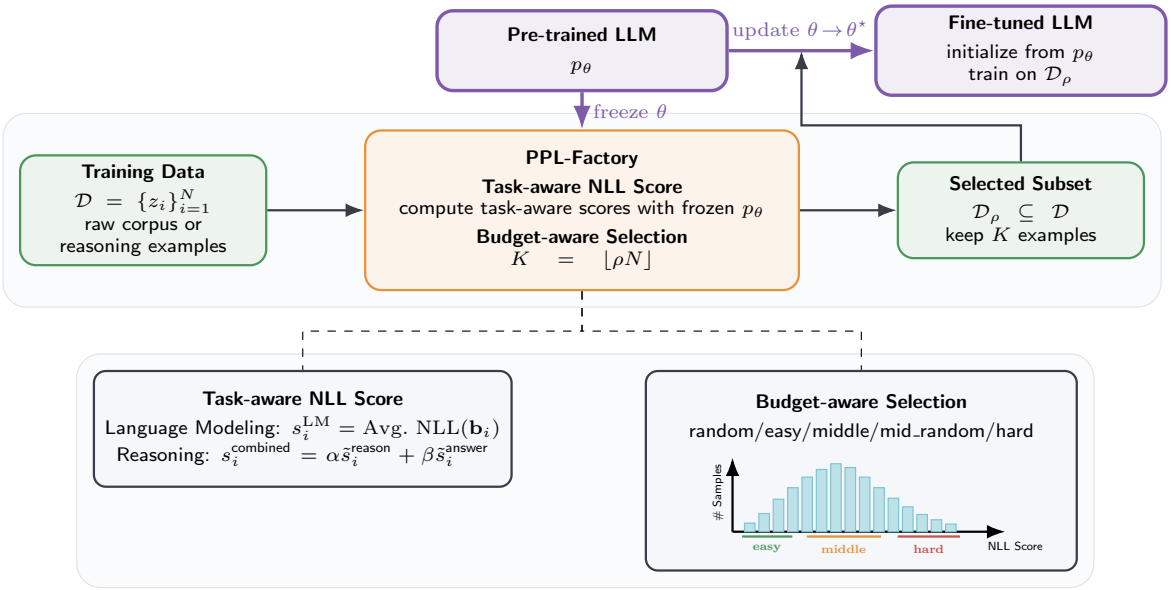



Figure 1: Overview of the proposed PPL-Factory framework. The orange-shaded block represents PPL-Factory. Purple-shaded blocks represent LLMs, green-shaded blocks represent datasets. The top, middle, and bottom rows illustrate the model fine-tuning, data selection flow, and internal components of PPL-Factory, respectively.

### 3.2 Task-Aware NLL Scores

Since perplexity directly reflects how likely a language model considers a sequence under its learned distribution, it is directly aligned with the next-token prediction objective. Motivated by this alignment, we use NLL as a practical scoring form of perplexity. We then adapt the NLL computation to reasoning task structures, so that the score captures the most relevant learning signal for different downstream tasks.

**Causal Language Modeling Tasks** In CLM fine-tuning, the training corpus is typically tokenized, concatenated, and divided into fixed-length blocks. Therefore, we pack the corpus into token blocks $\mathbf{b}_i = (b_{i,1}, \ldots, b_{i,B})$ where $B$ is the block size. Then we calculate NLL using the average next-token prediction loss over all tokens in a packed block. Since all packed blocks have the same length, the resulting NLL-based score provides a comparable block-level measure of difficulty. For each block $\mathbf{b}_i$, the language modeling score is computed as

$$s_i^{\text{LM}} = -\frac{1}{B-1} \sum_{t=1}^{B-1} \log p_\theta(b_{i,t+1} \mid b_{i,1:t}). \quad (8)$$

Based on the score, we rank the blocks by difficulty. A subset $\mathcal{D}_\rho$ can be selected given the selection criterion and the selection ratio. A detailed description of the proposed automatic selection approach is provided in Algorithm 1.

**Mathematical Reasoning Tasks** Unlike CLM corpora, the learning signal in the reasoning benchmarks is not uniformly distributed over the full sequence. Reasoning datasets are typically organized as instruction-response pairs, where the response contains the main task-specific supervision for fine-tuning (Huerta-Enochian and Ko, 2024). Both the chain-of-thought reasoning and the final answer are closely related to reasoning performance (Wei et al., 2022; Fang et al., 2025).

Instead of computing NLL over fixed-length packed blocks, we compute it over the reasoning steps and the final answer. For each structured example $(q_i, y_i)$, $q_i$ denotes the question and $y_i$ denotes the target response. The response is further decomposed as $y_i = [y_i^{\text{reason}}, y_i^{\text{answer}}]$. The reasoning and answer NLL scores are computed as

$$s_i^{\text{reason}} = -\frac{1}{|y_i^{\text{reason}}|} \sum_{t \in y_i^{\text{reason}}} \log p_\theta(y_{i,t} \mid q_i, y_{i,<t}),$$
$$s_i^{\text{answer}} = -\frac{1}{|y_i^{\text{answer}}|} \sum_{t \in y_i^{\text{answer}}} \log p_\theta(y_{i,t} \mid q_i, y_{i,<t}). \quad (9)$$

We define the overall selection score as a normalized weighted combination of average token-level

**Algorithm 1:** Packed-block Data Selector

**Input:** Dataset $\mathcal{D}$, language model $\mathcal{M}$, block size $B$, selection ratio $\rho$, `selection` $\in$ {`random`, `easy`, `hard`, `middle`, `mid_random`}, lower quantile $q_l$, upper quantile $q_u$

**Output:** Selected packed-block dataset $\mathcal{D}_\rho$

Tokenize $\mathcal{D}$, concatenate tokens, and split into blocks $\mathcal{B} = \{\mathbf{b}_i\}_{i=1}^{N}$ of length $B$
$K \leftarrow \max(1, \lfloor \rho N \rfloor)$
**if** `selection` $=$ `random` **then**
    Randomly select $K$ blocks
**else**
    $s_i \leftarrow -\frac{1}{B-1} \sum_{t=1}^{B-1} \log p_{\mathcal{M}}(b_{i,t+1} \mid b_{i,1:t}), \forall \mathbf{b}_i \in \mathcal{B}$
    **if** `selection` $=$ `easy` **then**
        select $K$ blocks with lowest $s_i$
    **else if** `selection` $=$ `hard` **then**
        select $K$ blocks with highest $s_i$
    **else if** `selection` $=$ `middle` **then**
        select $K$ blocks closest to $\text{median}(\{s_i\})$
    **else if** `selection` $=$ `mid_random` **then**
        $\mathcal{D}' \leftarrow \{\mathbf{b}_i : Q_{q_l}(\{s_i\}) \le s_i \le Q_{q_u}(\{s_i\})\}$
        randomly sample $K$ blocks from $\mathcal{D}'$, with fallback to $\mathcal{B} \setminus \mathcal{D}'$
**return** selected blocks as $\mathcal{D}_\rho$

**Algorithm 2:** Reasoning Data Selector

**Input:** Dataset $\mathcal{D} = \{(q_i, y_i)\}_{i=1}^{N}$, language model $\mathcal{M}$, selection ratio $\rho$, `scoring` $\in$ {`response`, `reason`, `answer`, `combined`}, `selection` $\in$ {`random`, `easy`, `hard`, `middle`, `mid_random`}, multiplier $\kappa$

**Output:** Selected reasoning dataset $\mathcal{D}_\rho$

Decompose each $y_i = [y_i^{\text{reason}}, y_i^{\text{answer}}]$
$K \leftarrow \max(1, \lfloor \rho N \rfloor)$
**if** `selection` $=$ `random` **then**
    Randomly select $K$ examples
**else**
    Compute task-aware NLL score $s_i$ using Eqs. 9 – 10
    **if** `selection` $=$ `easy` **then**
        select $K$ examples with lowest $s_i$
    **else if** `selection` $=$ `hard` **then**
        select $K$ examples with highest $s_i$
    **else if** `selection` $=$ `middle` **then**
        select $K$ examples closest to $\text{median}(\{s_i\})$
    **else if** `selection` $=$ `mid_random` **then**
        $\mathcal{P} \leftarrow$ the $\min(N, \lceil \kappa K \rceil)$ examples closest to $\text{median}(\{s_i\})$
        randomly sample $K$ examples from $\mathcal{P}$
**return** selected examples as $\mathcal{D}_\rho$

NLL scores

$$s_i^{\text{combined}} = \alpha \tilde{s}_i^{\text{reason}} + \beta \tilde{s}_i^{\text{answer}}, \qquad (10)$$

where $\alpha$ and $\beta$ are the weights for the reasoning steps and the final answer. $\tilde{s}_i$ is the normalized score using z-score normalization. This design measures the model's uncertainty in generating the solution process and deriving the final answer from the question, thereby producing a more task-aware selection score for mathematical reasoning fine-tuning.

Instead of applying fixed lower and higher quantile thresholds as in Algorithm 1, we construct a selection pool whose size is $\kappa\times$ the target number of selected samples and then randomly sample from this pool, where $\kappa$ is the middle-pool multiplier. The selection approach is detailed in Algorithm 2.

### 3.3 Budget-Aware Selections

Motivated by the budget dependence illustrated by Equation 7, we empirically construct a budget-aware data selection strategy using the NLL scores described in Section 3.2. When the selection ratio is relatively high, we use easier samples to avoid extremely difficult or noisy outliers. As the budget becomes more limited, we shift the selection toward mid-range samples, which provide more informative learning signals while avoiding both overly simple and overly difficult examples. For very small budgets, we further introduce randomness into the middle-band selection to preserve coverage of the training distribution and reduce the risk of selecting a narrow, biased subset. This randomization mitigates score noise and improves diversity without introducing additional scoring or evaluator costs. More details and analysis are discussed in Appendix B.

## 4 Experiments and Results

In this section, we present a comprehensive validation of our task-aware and budget-aware PPL-Factory. We first detail our evaluation benchmarks and model configuration in Section 4.1. Then, we list the training details in Section 4.2. In Section 4.3 and Section 4.4, we compare our results on multiple datasets against baselines and analyze key components through ablation studies. Finally, we analyze the computational complexity and fine-tuning cost in Section 4.5.

### 4.1 Benchmarks and Models

We evaluate PPL-Factory on four datasets, including CLM tasks and mathematical reasoning tasks.

In the experiments on CLM tasks, we fine-tune LLaMA3.2-1B, LLaMA3.2-3B, and LLaMA3.1-8B (Grattafiori et al., 2024) on our selected subset from WikiText-2 and WikiText-103 (Merity et al., 2016) to evaluate the model performance. The reported metric is perplexity.

For reasoning tasks, we adopt LLaMA3.2-1B-Instruct, LLaMA3.2-3B-Instruct, and LLaMA3.1-8B-Instruct as backbone models. We validated our PPL-Factory on two mathematical benchmarks, GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021). Performance is measured by zero-shot exact match accuracy on GSM8K and greedy Pass@1 accuracy on MATH. Each problem is evaluated using a single deterministic generation.

### 4.2 Training Details

All experiments use full parameter fine-tuning and are conducted on a single NVIDIA H100 with 80G memory. We follow standard practice in LLM fine-tuning (Wolf et al., 2020). For the data selectors, we keep `block_size` $= 2048$ in the packed-block data selector. Lower quantile and upper quantile are $q_l = 0.1$ and $q_u = 0.8$, respectively. These values are selected to keep more learnable, easier samples and remove more outliers. In reasoning task settings, we have `max_length` $= 2048$ for GSM8K, and `max_length` $= 4096$ for MATH since its problems and solutions are much longer and more complex. The weights in Equation 10 are $\alpha = 1$, $\beta = 0.5$. This choice assigns primary importance to the reasoning NLL score because it captures the model's intermediate multi-step problem-solving ability. The answer-level NLL is used as an auxiliary term to capture the difficulty of generating the final answer, but with a smaller weight to avoid overemphasizing short answer spans. We set the middle-pool multiplier $\kappa = 2.0$ when $\rho \leq 0.2$ and $\kappa = 1.5$ otherwise to keep the selection algorithm focusing on the middle band instead of introducing much randomness.

For experiments with smaller data budgets (1%–10%), we conduct five independent runs with different random seeds. For experiments with larger data budgets (20%–100%), we perform three independent runs. To ensure fair comparison, we use consistent optimization seeds across all experiments. We report the mean value of the evaluation metric in our analysis. The fine-tuning settings are discussed in Appendix D.2.

### 4.3 Results

Table 2 reports the evaluation performance on the four benchmarks under low data budget settings, with selection ratios ranging from 10% to 30%. We focus on this regime because preserving model performance with limited training data is particularly challenging and is critical to evaluating the effectiveness of data selection methods. Table 2a on the left illustrates the evaluation perplexity of different data selection methods on WikiText-2 and WikiText-103. Table 2b on the right compares accuracy on GSM8K and MATH.

**Causal Language Modeling Performance** Overall, PPL-Factory provides the most stable performance across WikiText-2 and WikiText-103. In nearly all settings, it achieves the lowest evaluation perplexity among the data selection baselines, demonstrating its effectiveness in identifying informative subsets for language-modeling fine-tuning. Although full-data fine-tuning remains the upper reference in most cases, PPL-Factory substantially narrows the gap using only 10% of the training data. It shows the effectiveness of applying random selection to the middle band under low data budget settings.

**Reasoning Tasks Accuracy** Compared with existing perplexity-based methods, PPL-Factory achieves higher accuracy on both GSM8K and MATH under all the experimental settings. In addition, when fine-tuning LLaMA3.2-3B-Instruct model on GSM8K with 30% and 10% of data, PPL-Factory yields better performance than the model fine-tuned on the entire training dataset. Experimental results on MATH also demonstrate that

(a) Language Modeling: Perplexity

| Selection Method | WikiText-2 | | | WikiText-103 | | |
|---|---|---|---|---|---|---|
| | [30%] | [20%] | [10%] | [30%] | [20%] | [10%] |
| **LLaMA3.2-1B Base Model** | | | | | | |
| Full Fine-tuning | 8.79 [100%] | | | 8.00 [100%] | | |
| Random | 9.21 | 9.37 | 9.69 | 8.18 | 8.28 | 8.41 |
| Medium PPL | 9.22 | 9.36 | 10.52 | 8.22 | 8.30 | 8.45 |
| High PPL | 9.25 | 9.37 | 9.63 | 8.30 | 8.38 | 8.51 |
| **PPL-Factory** | **9.20** | **9.22** | **9.55** | **8.15** | **8.26** | **8.38** |
| **LLaMA3.2-3B Base Model** | | | | | | |
| Full Fine-tuning | 7.43 [100%] | | | 6.59 [100%] | | |
| Random | 8.11 | 8.77 | 9.61 | 6.94 | 7.07 | 7.17 |
| Medium PPL | 8.08 | 8.70 | 9.52 | **6.90** | 7.05 | 7.15 |
| High PPL | 8.09 | 8.67 | 9.69 | 6.98 | 7.09 | 7.19 |
| **PPL-Factory** | **8.04** | **8.64** | **9.43** | 6.92 | **7.02** | **7.13** |
| **LLaMA3.1-8B Base Model** | | | | | | |
| Full Fine-tuning | 6.14 [100%] | | | 5.90 [100%] | | |
| Random | 6.32 | 6.40 | 6.66 | 6.14 | 6.20 | 6.37 |
| Medium PPL | 6.32 | 6.42 | 6.64 | 6.16 | 6.19 | 6.34 |
| High PPL | 6.32 | 6.40 | 6.61 | 6.29 | 6.33 | 6.40 |
| **PPL-Factory** | **6.30** | **6.36** | **6.59** | **6.10** | **6.16** | **6.31** |

(b) Mathematical Reasoning: Accuracy

| Selection Method | GSM8K | | | MATH | | |
|---|---|---|---|---|---|---|
| | [30%] | [20%] | [10%] | [30%] | [20%] | [10%] |
| **LLaMA3.2-1B-Instruct Model** | | | | | | |
| Full Fine-tuning | 39.88 [100%] | | | 19.4 [100%] | | |
| Random | 36.62 | 36.16 | 35.94 | **21.0** | 19.8 | 19.8 |
| Medium PPL | 35.41 | 34.19 | 34.95 | 19.0 | 17.2 | 20.4 |
| High PPL | 35.71 | 34.34 | 31.92 | 20.2 | 21.2 | 21.0 |
| **PPL-Factory** | **39.42** | **36.92** | **36.32** | 20.6 | **22.4** | **21.8** |
| **LLaMA3.2-3B-Instruct Model** | | | | | | |
| Full Fine-tuning | 61.87 [100%] | | | 27.2 [100%] | | |
| Random | 62.40 | 59.89 | 62.55 | 26.0 | 27.6 | 28.8 |
| Medium PPL | 62.77 | 61.20 | 59.89 | 26.2 | 25.4 | 24.2 |
| High PPL | 59.89 | 60.50 | 59.89 | 27.4 | 29.4 | 29.0 |
| **PPL-Factory** | **63.00** | **61.56** | **62.77** | **28.0** | **29.8** | **32.0** |
| **LLaMA3.1-8B-Instruct Model** | | | | | | |
| Full Fine-tuning | 71.42 [100%] | | | 36.2 [100%] | | |
| Random | 69.06 | 66.79 | 64.67 | 35.6 | 34.4 | 35.2 |
| Medium PPL | 65.50 | 65.20 | 65.13 | 31.6 | 33.4 | 33.2 |
| High PPL | 63.76 | 59.44 | 58.76 | 33.8 | 32.0 | 34.6 |
| **PPL-Factory** | **70.46** | **68.34** | **67.25** | **35.8** | **34.6** | **35.6** |

Table 2: Results of PPL-Factory under different data budgets. Table(a) on the left reports perplexity on WikiText-2 and WikiText-103, where lower is better. Table(b) on the right reports accuracy on GSM8K and MATH, where higher is better. Medium PPL and High PPL are reproduced from (Ankner et al., 2025), using deterministic categories. Best results within each column are in **bold**. Data selection ratios are in [brackets].

| **Method** | [1%] | [5%] | [10%] |
|---|---|---|---|
| LLaMA3.1-8B-Instruct | 57.49 [0%] | | |
| + Full Fine-tuning | 71.39 [100%] | | |
| + Random | 58.09 | 64.17 | 65.67 |
| + Medium PPL (Ankner et al., 2025) | 58.07 | 67.70 | 66.43 |
| + High PPL (Ankner et al., 2025) | 56.33 | 60.88 | 59.71 |
| + GraNd (Paul et al., 2021) | 57.23 | 67.65 | 70.05 |
| + EL2N (Paul et al., 2021) | 60.16 | 61.50 | 71.66 |
| + CCS (Zheng et al., 2023) | 60.70 | 63.64 | 68.72 |
| + Data Whisperer (Wang et al., 2025) | 62.57 | **69.68** | **72.46** |
| + **PPL-Factory (Ours)** | **62.94** | 69.61 | 70.24 |

Table 3: Exact-match accuracy on GSM8K dataset, using LLaMA3.1-8B-Instruct Model. [0%] means the model is not fine-tuned on the downstream dataset. [100%] refers to the model being fine-tuned with the full-size training set, without data selection.

PPL-Factory provides strong subset fine-tuning performance. More specifically, for 1B and 3B models, fine-tuning on the selected subsets achieves higher accuracy than full fine-tuning across all the 30%, 20%, and 10% selection ratios, indicating that task-aware selection can remove less useful samples while preserving more effective reasoning signals.

To enable a direct comparison with recent data selection methods, we follow the experimental setting of Data Whisperer (Wang et al., 2025) for the results reported in Table 3. Compared to other data selection methods, PPL-Factory achieves higher accuracy with only 1% data. Although using 10% data does not achieve higher accuracy than EL2N and Data Whisperer, our PPL-Factory is computationally resource-friendly. EL2N is memory-intensive because its token-level prediction involves vocabulary-sized logits, whereas our NLL scoring can be computed with inference-only memory and scalar scores. The attention-based Data Whisperer was run on eight NVIDIA A100 GPUs (Wang et al., 2025), reflecting substantially greater computational resource requirements. Experimental results show that our task-aware NLL score works effectively on reasoning tasks, and especially fits the low data-budget settings.

**Budget-Aware Selection Criterion** The best data selection criterion shifts with the fine-tuning data budget. Among the five categories, `random`, `easy`, `middle`, `mid_random`, and `hard`, easy samples contribute the most learning signals when a relatively large portion of the training set is preserved. These low-perplexity examples can provide stable and reliable weight updates, making selection effective. In contrast, when only a very limited subset is kept, i.e., 30% or less of the full-size dataset, deterministic selection from a narrow, biased perplexity region may reduce data diversity and over-concentrate on similar training examples. Under this condition, `mid_random`, which focuses on a wider middle band and randomly selects samples from the candidate pool, achieves better empirical evaluation performance. In other

| Method | [30%] | [20%] | [10%] |
|---|---|---|---|
| Random | 36.62 | 36.16 | 35.94 |
| Response NLL | 36.47 | 36.39 | 35.25 |
| Reasoning NLL | 38.97 | 35.48 | 36.01 |
| Answer NLL | 38.67 | 34.34 | 34.72 |
| **PPL-Factory (Ours)** | **39.42** | **36.92** | **36.32** |

Table 4: Ablation of our PPL-Factory compared with NLL variants fine-tuned on GSM8K using LLaMA3.2-1B-Instruct Model. Response NLL uses the full response to calculate the selection score. Reasoning NLL is equivalent to setting the weights in Equation 10 to $\alpha = 1,\ \beta = 0$, and answer NLL is equivalent to set $\alpha = 0,\ \beta = 1$.

words, introducing randomness enhances the diversity of the selected data, and contributes positively to fine-tuning. It balances training effectiveness and efficiency without involving extra computational cost overhead. Thus, we conclude that the empirically best selection criterion is not only determined by the model size or the task, but also by the percentage of the training samples.

### 4.4 Ablation

Table 4 presents the GSM8K accuracy of our combined NLL score in comparison to different NLL scoring variants. The results show that PPL-Factory consistently achieves the highest exact-match accuracy across all selection ratios. Compared with random selection, PPL-Factory improves the accuracy by 2.80, 0.76, and 0.38 points using 30%, 20%, and 10% of training samples, respectively. Among the single-component variants, reasoning NLL and answer NLL can provide competitive performance with 30% data, but their performance becomes less stable when the data budget decreases. Response NLL also shows limited improvement, suggesting that using the full response as a single scoring target may dilute the contribution of the reasoning and final-answer components. In contrast, PPL-Factory combines both reasoning steps and answer information and achieves the best performance in all three settings, indicating that both components in Equation 10 are useful for selecting informative reasoning examples.

### 4.5 Computational Complexity and Training Cost

In the PPL-Factory framework, for a dataset with $N$ examples, each NLL-based score is computed by forward inference using the target model, where the subset is directly determined by the target model without using proxy evaluation. The main efficiency gain comes from reducing the size of the training set. With a selection ratio $\rho \in (0, 1]$, the number of training examples is reduced from $N$ to $\rho N$, and the fine-tuning cost decreases approximately linearly with the number of processed tokens. Although the scoring step adds a one-time inference overhead, this cost is much smaller than the cost of full fine-tuning. Experimental results show that our method achieves a trade-off between performance and training cost.

## 5 Related Work

Data selection aims to identify a smaller but more useful subset that preserves downstream performance. Existing methods can be broadly categorized into difficulty or perplexity-based selection, coverage-based selection, and influence-based selection.

**Difficulty- or Perplexity-based Selection** Some data pruning methods, such as GraNd and EL2N (Paul et al., 2021), estimate the importance of examples from early training dynamics by the L2 norm of error vectors. RHO-loss (Mindermann et al., 2022) prioritizes examples that are learnable, useful, but not yet learned. $\mathcal{H}$-score (Hosseini et al., 2025) ranks each training example by the success rate in classifying it correctly across multiple fine-tuning runs.

Perplexity-based data pruning has been shown to be effective for pre-training (Marion et al., 2023). Recent work (Ankner et al., 2025) demonstrates that a small reference model can provide a useful perplexity-based rank for selecting examples, which can be used to train larger models. These methods suggest that model-estimated difficulty can serve as a proxy estimation for data utility.

**Coverage-based Selection** Coverage-centric Coreset Selection (Zheng et al., 2023) argues that maintaining coverage is important under high pruning rates. The existing selection algorithms may over-concentrate a narrow region of the data distribution and suffer from a dramatic accuracy drop under extremely low data-budget settings.

**Influence-based Selection** LESS (Xia et al., 2024) selects influential examples for target instruction tuning by comparing low-dimensional gradient features between candidate examples and

target-task demonstrations. Influence-based pruning (Humane et al., 2025; Li et al., 2024b) improves reasoning-data selection over heuristic criteria, but it requires costly influence estimation and shows limited capability in cross-model transferability.

**Our PPL-Factory Framework** Different from methods that rely on external LLM references and expensive gradient similarity, our work studies a simple and scalable one-shot data selection framework based on task-aware NLL scores and budget-aware selection criteria. Our study connects perplexity-based data selection approaches with supervised fine-tuning for both CLM tasks and mathematical reasoning tasks. Empirical evidence shows that our PPL-Factory outperforms deterministic perplexity-based selection and achieves the highest accuracy among the compared approaches at the $1\%$ data budget, while remaining competitive at larger budgets, including gradient-based GraNd and EL2N, coverage-based CCS, and attention-based Data Whisperer. Meanwhile, PPL-Factory preserves a simple design with low resource requirements.

## 6 Conclusion

In this paper, we presented PPL-Factory, a task-aware and budget-aware data selection framework for efficient fine-tuning of large language models. Experiments on CLM benchmarks show that PPL-Factory preserves most of the full-data fine-tuning performance using only $10\%$ of the training corpus. On GSM8K, PPL-Factory achieves higher accuracy than state-of-the-art selection methods using $1\%$ of the training data. It also surpasses full-data fine-tuning in several low-budget settings on both reasoning benchmarks, GSM8K and MATH, by removing examples with weak or noisy learning signals. Overall, PPL-Factory reduces training cost across both language modeling and reasoning tasks by integrating the simplicity of the perplexity-based scores adapted to different task structures with budget-aware selection strategies.

## Limitations

While our work demonstrates the promise of offline one-shot data selection for both causal language modeling and mathematical reasoning, several limitations remain. Our study mainly focuses on supervised fine-tuning, where reducing the number of training examples directly lowers computational cost, especially when the dataset size dominates the overall training expense. We leave the extension of PPL-Factory to knowledge distillation as future work. In particular, applying perplexity-based selection to teacher–student learning may further reduce training cost by enabling smaller student models to learn from more informative subsets of the distillation data. With an appropriate student architecture, we believe PPL-Factory can provide an efficient solution for improving knowledge transfer while using fewer training samples.

## References


Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. 2024. A survey on data selection for language models. *Transactions on Machine Learning Research*. Survey Certification, Featured Certification.

Zachary Ankner, Cody Blakeney, Kartik Sreenivasan, Max Marion, Matthew L Leavitt, and Mansheej Paul. 2025. Perplexed by perplexity: Perplexity-based data pruning with small reference models. In *The Thirteenth International Conference on Learning Representations*.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024. Alpagasus: Training a better alpaca with fewer data. In *The Twelfth International Conference on Learning Representations*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Qirun Dai, Dylan Zhang, Jiaqi W. Ma, and Hao Peng. 2025. Improving influence-based instruction tuning data selection for balanced learning of diverse capabilities. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 7079–7102, Suzhou, China. Association for Computational Linguistics.

Lizhe Fang, Yifei Wang, Zhaoyang Liu, Chenheng Zhang, Stefanie Jegelka, Jinyang Gao, Bolin Ding, and Yisen Wang. 2025. What is wrong with perplexity for long-context language modeling? In *The Thirteenth International Conference on Learning Representations*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh

Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Zayd Hammoudeh and Daniel Lowd. 2024. Training data influence analysis and estimation: a survey. *Machine Learning*, 113(5):2351–2403.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Sayed Mohammadreza Tayaranian Hosseini, Seyyed Hasan Mozafari, Brett H. Meyer, James J. Clark, and Warren J. Gross. 2025. Automatic pruning of fine-tuning datasets for transformer-based language models. In *Proceedings of The 3rd Conference on Lifelong Learning Agents*, volume 274 of *Proceedings of Machine Learning Research*, pages 1–28. PMLR.

Mathew Huerta-Enochian and Seung Yong Ko. 2024. Instruction fine-tuning: Does prompt loss matter? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22771–22795, Miami, Florida, USA. Association for Computational Linguistics.

Prateek Humane, Paolo Cudrano, Daniel Z Kaplan, Matteo Matteucci, Supriyo Chakraborty, and Irina Rish. 2025. Influence functions for efficient data selection in reasoning. In *NeurIPS 2025 Workshop on Efficient Reasoning*.

Hamish Ivison, Muru Zhang, Faeze Brahman, Pang Wei Koh, and Pradeep Dasigi. 2025. Large-scale data selection for instruction tuning. *Preprint*, arXiv:2503.01807.

Krishnateja Killamsetty, Durga S, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. 2021. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5464–5474. PMLR.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1885–1894. JMLR.org.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024a. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14255–14273, Bangkok, Thailand. Association for Computational Linguistics.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024b. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7602–7635, Mexico City, Mexico. Association for Computational Linguistics.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. 2023. When less is more: Investigating data pruning for pretraining llms at scale. *Preprint*, arXiv:2309.04564.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. 2022. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15630–15649. PMLR.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mahdi Nikdan, Vincent Cohen-Addad, Dan Alistarh, and Vahab Mirrokni. 2025. Efficient data selection at scale via influence distillation. In *Advances in Neural Information Processing Systems*, volume 38, pages 143229–143265. Curran Associates, Inc.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: finding important examples early in training. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA. Curran Associates Inc.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Shaobo Wang, Xiangqi Jin, Ziming Wang, Jize Wang, Jiajun Zhang, Kaixin Li, Zichen Wen, Zhong Li, Conghui He, Xuming Hu, and Linfeng Zhang. 2025. Data whisperer: Efficient data selection for task-specific LLM fine-tuning via few-shot in-context learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23287–23305, Vienna, Austria. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Tingyu Xia, Bowen Yu, Kai Dang, An Yang, Yuan Wu, Yuan Tian, Yi Chang, and Junyang Lin. 2025. Rethinking data selection at scale: Random selection is almost all you need. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 2698–2711, Suzhou, China. Association for Computational Linguistics.

Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. 2023. Coverage-centric coreset selection for high pruning rates. In *The Eleventh International Conference on Learning Representations*.

# A Overview

This appendix provides details of our theoretical analysis, experimental settings and additional evaluation results.

# B Theoretical Analysis

## B.1 Proof of the Optimizer in Equation 7

We consider the informative-coverage score defined in Section 3.1:

$$J_\rho(Q_\rho) = \int u_\rho(s) Q_\rho(s)\, ds - \lambda_\rho D_{\mathrm{KL}}(Q_\rho \,\|\, P), \quad \lambda_\rho > 0 \tag{11}$$

where $u_\rho(s)$ denotes the conceptual utility of retaining an example with score $s$ under selection ratio $\rho$, and $\lambda_\rho > 0$ controls the preference for preserving coverage relative to concentrating on high-utility regions.

The function $u_\rho(s)$ is deliberately budget-dependent. Intuitively, when $\rho \in (0, 1]$ is relatively large, the selected subset already preserves most of the original data distribution, so the main benefit comes from removing a small, low-quality tail. When $\rho$ is relatively small, however, each retained example must carry more learning value, and a useful selector may need to place greater emphasis on informative but still learnable examples.

**Proposition** For a fixed ratio $\rho$, the unique maximizer of Equation 11 is

$$Q^*_\rho(s) = \frac{P(s)\exp(u_\rho(s)/\lambda_\rho)}{\int P(t)\exp(u_\rho(t)/\lambda_\rho)\, dt} \tag{12}$$

*Proof.* Consider the Lagrangian

$$\begin{aligned}\mathcal{L}(Q_\rho, \eta) = &\int u_\rho(s) Q_\rho(s)\, ds \\ &- \lambda_\rho \int Q_\rho(s) \log \frac{Q_\rho(s)}{P(s)}\, ds \\ &+ \eta \left( \int Q_\rho(s)\, ds - 1 \right)\end{aligned} \tag{13}$$

Taking the derivative with respect to $Q_\rho(s)$ and setting it to zero gives

$$u_\rho(s) - \lambda_\rho \left( \log \frac{Q_\rho(s)}{P(s)} + 1 \right) + \eta = 0 \tag{14}$$

It yields

$$Q_\rho(s) = P(s)\exp(u_\rho(s)/\lambda_\rho)\, C, \tag{15}$$

for the normalization constant $C$.

Enforcing $\int Q_\rho(s)\, ds = 1$ yields the maximizer in Equation 12. Since the objective in Equation 11 is strictly concave in $Q_\rho$, the solution is unique. □

This result characterizes the optimizer of the KL-regularized objective.

## B.2 Discussion on Pruning Rates

The shape of the optimal selector is controlled by the budget-dependent utility $u_\rho(s)$. Thus, the preferred NLL region may shift with the size of available training samples. We compare the evaluation perplexity on WikiText-2 and WikiText-103 using different selection ratios on training samples in Figures 2 – 4.

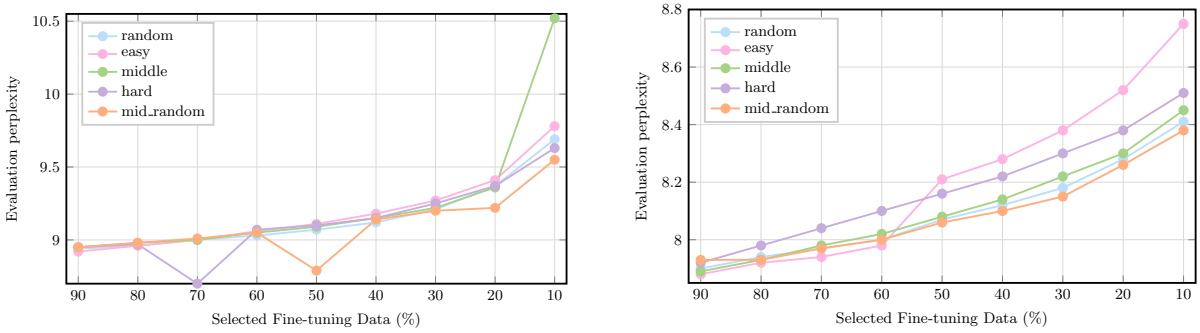



Figure 2: Comparison of the evaluation perplexity using different subsets from WikiText-2 (left) and WikiText-103 (right) datasets, using LLaMA3.2-1B model.

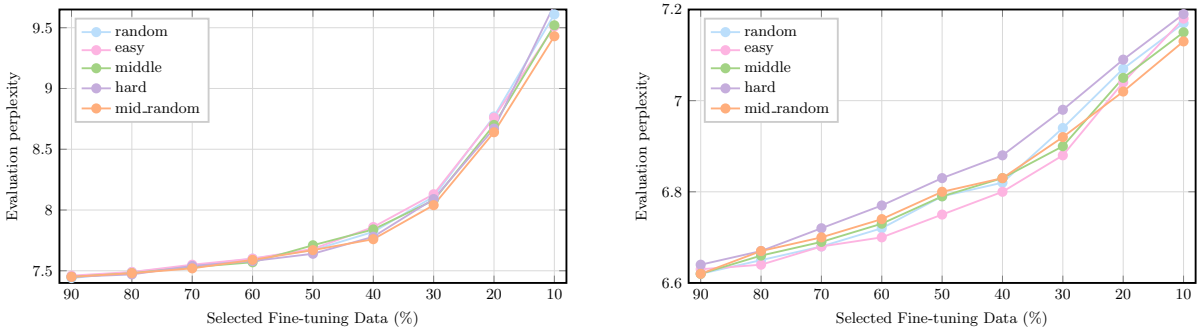



Figure 3: Comparison of the evaluation perplexity using different subsets from WikiText-2 (left) and WikiText-103 (right) datasets, using LLaMA3.2-3B model.

**High Selection Ratio** When using a relatively higher selection ratio, for example, retaining 60%–90% of the training data, selecting `easy` examples with a lower NLL score can outperform the other four categories, `random`, `middle`, `mid_random`, and `hard`. However, the kept subset remains close in distribution to the full dataset. The main advantage of selection under this circumstance is often the removal of the hardest tail in a low-density region, which may contain noisy, atypical, or low-return examples.

**Low Selection Ratio** Under a very limited computational budget, i.e., to force the selection ratio to be lower, middle-band selection with randomization can become more effective. Under this condition, training samples with a low NLL score contain limited learning signal that would contribute to the training and weight updates. Thus, these examples may become redundant, while examples with very high NLL scores may remain too noisy or difficult. The best trade-off can then shift to the middle region, but it still needs the diversity from randomness.

**Conclusion** The empirically best selection criterion depends not only on the model size and task, but also on the retained-data ratio. The framework above suggests that there is no universally optimal fixed NLL region for data selection. Instead, the preferred region is budget-dependent:

$$Q_\rho^\star(s) \propto P(s)\exp(u_\rho(s)/\lambda_\rho). \qquad (16)$$

For a high selection ratio, $u_\rho(s)$ may favor easy examples, which is consistent with favoring low-NLL examples in our empirical results. For a low selection ratio, $u_\rho(s)$ may peak in the middle, making middle-biased randomized selection preferable. An implementation table for our proposed PPL-Factory, specifying dataset, data budget, the scoring function and the selection criterion, is provided in Table 5.

## C Random Data Selection and Analysis

Many existing works show that random selection may exceed the performance of carefully designed deterministic selection methods, especially under extremely low data budgets. Our proposed PPL-Factory introduces randomness when the selection criterion is `mid_random`. To isolate the effect

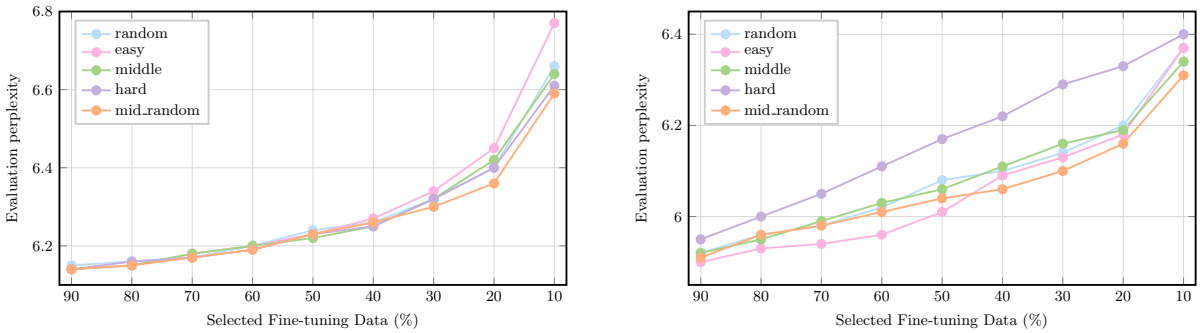



Figure 4: Comparison of the evaluation perplexity using different subsets from WikiText-2 (left) and WikiText-103 (right) datasets, using LLaMA3.1-8B model.

| Dataset | Data Budget (%) | Score | Selection Rule |
|---|---|---|---|
| WikiText-2/103 | 10 – 30 | Block NLL | `mid_random` |
| | 40 – 50 | Block NLL | `middle` |
| | 60 – 90 | Block NLL | `easy` |
| GSM8K/MATH | 10 – 30 | Combined NLL | `mid_random` |
| | 40 – 50 | Combined NLL | `middle` |
| | 60 – 90 | Combined NLL | `easy` |

Table 5: Implementation settings of PPL-Factory across datasets and data budgets.

of score-informed selection from the randomness component, we add a controlled analysis where all methods use the same number of selected samples, the same candidate-pool size, and the same training seeds. In particular, we use a shuffled-score control. It first computes an NLL score for each sample, then randomly shuffles the scores across samples before applying the same randomized middle-band selection rule as PPL-Factory. Therefore, shuffled-score preserves the score distribution and selection mechanism, but removes the true alignment between samples and their NLL scores. The results are shown in Table 6.

These results provide three observations. First, Shuffled-score does not consistently outperform Random, suggesting that the middle-band randomization mechanism alone does not explain the improvement. Second, Response-NLL mid-random improves over Response-NLL middle at 20% and 10%, suggesting that randomized sampling from the middle candidate pool can improve diversity under lower budgets. Third, PPL-Factory achieves the best accuracy across all three budgets, suggesting that combining the proposed combined-NLL score with budget-aware mid-random selection is more effective than using either randomization or plain response-level NLL alone. Thus, the comparison directly measures the contribution of our proposed data selection method.

| Selection Method | [30%] | [20%] | [10%] |
|---|---|---|---|
| Random | 62.40 | 59.89 | 62.55 |
| Shuffled-score | 61.87 | 60.88 | 60.12 |
| Response-NLL middle | 62.77 | 61.18 | 59.89 |
| Response-NLL mid-random | 62.77 | 61.33 | 60.50 |
| Combined-NLL middle | 62.85 | 61.26 | 61.87 |
| **PPL-Factory** | **63.00** | **61.56** | **62.77** |

Table 6: Comparison between selection methods on GSM8K dataset, using LLaMA3.2-3B-Instruct model, across diverse data budgets. Response-NLL uses the plain full response NLL score, while Combined-NLL uses our proposed combined score in PPL-Factory. Middle uses a deterministic middle-band selection, while mid-random randomly samples from the middle-band candidate pool.

## D Reproducibility Details

### D.1 Data Split

In this section, we describe the benchmarks used in our experiments in detail. The training and evaluation splits are summarized in Table 7.

**WikiText-2 and WikiText-103** For language modeling experiments, we use the official

| Dataset | Selection Pool | Evaluation Set | Metric |
|---|---|---|---|
| WikiText-2 | WikiText-2 train (36,718 samples) | WikiText-2 validation (3,760 samples) | Perplexity |
| WikiText-103 | WikiText-103 train (1,801,350 samples) | WikiText-103 validation (3,760 samples) | Perplexity |
| GSM8K | GSM8K train (7,473 samples) | GSM8K test (1,319 samples) | Exact Match |
| MATH | MATH-minus-MATH500 (12,000 samples) | MATH500 (500 samples) | Greedy Pass@1 |

Table 7: Dataset splits used in our experiments.

WikiText-2 and WikiText-103 splits. The WikiText language modeling dataset is a collection of tokens extracted from Wikipedia. The training split is used for data selection and subsequent fine-tuning. Specifically, the text is tokenized, concatenated, and divided into fixed-length token blocks, and selection is performed at the block level. The official validation and test splits are kept separate from the selection process and are used only for perplexity evaluation. WikiText-2 contains 36,718 training lines, 3,760 validation lines, and 4,358 test lines, while WikiText-103 contains 1,801,350 training lines, 3,760 validation lines, and 4,358 test lines.

**GSM8K** GSM8K (Grade School Math 8K) is a dataset of 8.5K high-quality, linguistically diverse grade school math word problems. For GSM8K, we use the official training split for data selection and fine-tuning. The official test split is used only for evaluation. The GSM8K training split contains 7,473 examples, and the test split contains 1,319 examples. We report zero-shot exact-match accuracy on the evaluation set.

**MATH** The MATH benchmark (Hendrycks et al., 2021) has 12,500 competition-level math problems with step-to-step solutions. MATH500 (Lightman et al., 2024) comes from the nonstandard MATH split, where 500 held-out problems are used for evaluation. For MATH data selection and fine-tuning, we use MATH-minus-MATH500, a training pool containing 12,000 examples derived from MATH by removing the 500 MATH500 problems. Following common practice, we evaluate the fine-tuned models on MATH500. No MATH500 examples are included in the selected training subset.

### D.2 Fine-tuning Details

For the results reported in Table 2, the fine-tuning learning rate is set to $2 \times 10^{-5}$ for all 1B, 3B, and 8B models. For WikiText-2 and WikiText-103, we perform evaluation during training and save the best model with the lowest evaluation perplexity. For the $10\%$ and $20\%$ data budget settings on WikiText-2, we train for 2 epochs to avoid overfitting. All remaining settings are trained for 3 epochs. For GSM8K and MATH, evaluation is performed after fine-tuning, and all experiments are conducted for 3 epochs.

In addition, for the comparison with recent data selection methods in Table 3, we follow the experimental protocol of Data Whisperer. Specifically, model fine-tuning is performed with LoRA adapters, with a learning rate of $1 \times 10^{-4}$ and a batch size of 8.