Better Datasets Start from RefineLab: Automatic Optimization for High-Quality Dataset Refinement

Xiaonan Luo[1*], **Yue Huang**[1*], **Ping He**[2], **Xiangliang Zhang**[1†]

[1]Department of Computer Science and Engineering, University of Notre Dame
[2]Department of Computer Science, Vanderbilt University

Abstract

High-quality Question–Answer (QA) datasets are foundational for reliable Large Language Model (LLM) evaluation, yet even expert-crafted datasets exhibit persistent gaps in domain coverage, misaligned difficulty distributions, and factual inconsistencies. The recent surge in generative model-powered datasets has compounded these quality challenges. In this work, we introduce **RefineLab**, the first LLM-driven framework that automatically refines raw QA textual data into high-quality datasets under a controllable token-budget constraint. RefineLab takes a set of target quality attributes as refinement objectives and performs selective edits within a predefined token budget to ensure practicality and efficiency. In essence, RefineLab addresses a constrained optimization problem: improving the quality of QA samples as much as possible while respecting resource limitations. With a set of available refinement operations, RefineLab takes as input the original dataset, a specified set of target quality dimensions, and a token budget, and determines which refinement operations should be applied to each QA sample. This process is guided by an assignment module that selects optimal refinement strategies to maximize overall dataset quality while adhering to the budget constraint. Experiments demonstrate that RefineLab consistently narrows divergence from expert datasets across coverage, difficulty alignment, factual fidelity, and distractor quality. RefineLab pioneers a scalable, customizable path to reproducible dataset design, with broad implications for LLM evaluation.

Extended version — https://arxiv.org/abs/2511.06530

Introduction

Large language models (LLMs) have achieved remarkable success on a variety of natural language understanding and generation tasks, driving rapid progress in various areas (Zhao et al. 2023; Guo et al. 2023; Qian et al. 2023). To better evaluate and improve the capabilities of these models, central to this progress is the availability of high-quality datasets (Wang et al. 2024; Lin, Hilton, and Evans 2022; Zellers et al. 2019). A high-quality evaluation dataset for LLMs is widely regarded as one that reliably reflects model capabilities, minimizes bias, and facilitates fair comparisons across models. Although this notion is broadly accepted within the community, there is no formal standard that explicitly defines what constitutes a high-quality dataset, or quantifies how the quality of a dataset influences evaluation reliability. In this work, we seek to reduce this ambiguity by identifying and analyzing key dimensions of dataset quality that can be systematically enhanced through careful sample refinement, while accounting for practical constraints such as token-level editing budgets.

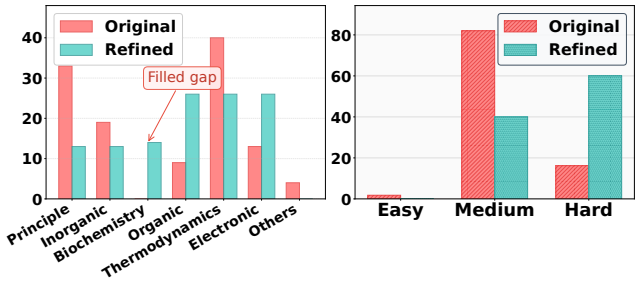

Figure 1: An example of topic coverage (left) and difficulty level distribution (right) in the College Chemistry subset of MMLU before (Original) and after refinement (Refined). Targets follow MIT's undergraduate chemistry major curriculum and feature a stress-testing difficulty profile.

To ground our discussion, we take the widely adopted MMLU dataset (Hendrycks et al. 2021) as an illustrative example. MMLU is often praised for its broad subject coverage and standardized multiple-choice format, which makes it suitable for evaluating knowledge recall and reasoning across diverse domains. However, despite its popularity, MMLU exhibits several key limitations: **1) Topic coverage:** its empirical topic mix can diverge sharply from real-world evaluation standards, for example, in the College Chemistry subset shown in Figure 1 (left), *biochemistry* is completely absent, yet real-world college curricula, including MIT, Stanford, and other world-class universities, always demand a certain portion representation on it, resulting in datasets that fail to adequately reflect the full range of domain-specific skills and knowledge. **2) Imbalanced Difficulty Distribution:** its fixed difficulty profile cannot accommodate varying evaluation goals, whether stress-testing

*These authors contributed equally.
†Corresponding author.

advanced reasoning for math and science or supporting progressive skill diagnostics for language acquisition. As Figure 1 (right) shows, the original College Chemistry subset is dominated by medium-level difficulty questions that lack deep reasoning. Under an example of *stress-testing* target profiles, naive resampling can only remove excess medium items, drastically shrinking the dataset. Further refinement is needed to introduce new examples and maintain the effective dataset size while meeting specific difficulty requirements. **3) Factual and Logical Errors:** its factual or logical mistakes that can persist unchecked, introducing misleading artifacts into model evaluation. Such errors can compromise the validity of benchmarking results. These shortcomings highlight the necessity of refining existing datasets to ensure reliable model evaluation and to drive progress in LLM development and downstream applications.

While dataset refinement is essential, manual approaches are often tedious and inefficient. Recent work has introduced LLM-based methods for automatic dataset refinement, leveraging LLMs to synthesize new data with minimal human involvement (Khan et al. 2025; Li et al. 2025b; Huang et al. 2025; DeSalvo et al. 2025; Li et al. 2025a). Although these approaches are promising, several challenges remain: **1) Lack of explicit control:** LLM-driven pipelines may unintentionally amplify coverage gaps or difficulty imbalances, as synthesized examples are often generated without fine-grained constraints. **2) Quality assurance limitations:** Without rigorous validation, LLM-generated data can introduce new factual or logical errors, complicating overall quality control. **3) Resource allocation dilemma:** Given that invoking high-capability models (e.g., GPT-4) for refinement incurs significant computational cost, efficiently allocating a limited budget of LLM calls across multiple refinement strategies to maximize dataset quality remains a challenging and largely unsolved problem.

To end this, in this paper, we introduce **RefineLab**, a framework for textual dataset refinement that automatically selects and applies targeted editing operations to improve dataset quality under token-level budget constraints. In RefineLab, dataset *high quality* is operationalized via explicitly specified targets (e.g., desired topic distribution for coverage, difficulty levels, or factual accuracy). These targets can be defined either by users or inferred from real-world evaluation needs. Given such explicit requirements and a set of available refinement strategies, RefineLab employs an assignment module that selects the optimal refinement operations for each sample to maximize overall dataset quality, while respecting a global token-level budget constraint. Based on the estimated cost and quality gain of each refinement option, the assignment problem is formulated as an Integer Linear Programming (ILP) optimization, which determines, for each QA sample (q_i, a_i), which refinement operations (r_k) to apply: x_{ik} =1 if refinement operation r_k is assigned to sample (q_i, a_i), and x_{ik} = 0 otherwise. Overall, the objective is formalized as

$$\max_{x_{ik}\in\{0,1\}} \sum_{i,k} x_{ik} \cdot \Delta_{ik} \quad \text{s.t.} \quad \sum_{i,k} x_{ik} \cdot c_{ik} \leq B \qquad (1)$$

where Δ_{ik} denotes the estimated quality gain from applying r_k to sample (q_i, a_i), c_{ik} represents the token cost of that refinement, and B is the total available token budget. This formulation enables RefineLab to reason about tradeoffs between different refinement operations and their associated costs, providing a principled way to enhance dataset quality under resource constraints.

In this work, we make the following key contributions:

- We propose **RefineLab**, the first LLM-driven framework for dataset refinement that formulates the refinement selection process as an *integer linear program* to optimize quality improvements under a token-level budget.
- We introduce a set of modular refinement operations designed to improve QA datasets along a set of explicit quality targets, including topic coverage, difficulty calibration, and factual consistency.
- We conduct comprehensive experiments on widely-used benchmarks such as MMLU, demonstrating that RefineLab produces higher-quality QA datasets with improved topic coverage, difficulty balance, and overall utility for evaluating LLM performance.

Related Work

Synthetic Dataset Generation. LLMs have enabled scalable, automated construction of synthetic datasets via prompting. Self-Instruct (Wang et al. 2023) bootstraps instruction–response pairs through iterative self-prompting and employs minimal manual filtering to ensure basic correctness. DataGen (Huang et al. 2025) introduces a modular framework using attribute-guided prompts and automated consistency checks to generate diverse text, supplemented by lightweight heuristic filters for domain validity. AutoBencher (Li et al. 2025a) advances a declarative dataset construction paradigm, allowing users to specify high-level dataset schemas from which LLMs generate evaluation items under explicit quality constraints. However, these methods focus on data synthesis that is typically guided by high-level prompts with limited control over sample-level quality dimensions. In contrast, RefineLab targets systematic dataset improvement by selecting editing operations through constrained optimization, enabling fine-grained, budget-aware refinement of real datasets.

Evaluation on Dataset Quality. High-quality datasets are fundamental to effective LLMs training and reliable evaluation. Recent works emphasize that current dataset flaws, including distributional bias, uncalibrated factual errors, lead to misleading performance estimates or inefficient model training (Reuel et al. 2024; Yu et al. 2023; Xie et al. 2023; Miranda et al. 2025; Iskander et al. 2024). As a result, ensuring high-quality datasets becomes essential. Recent efforts have focused on designing metrics that capture multiple, complementary aspects of data validity. For example, factual integrity is typically assessed through error-detection and correction pipelines that retrieve external evidence to validate item correctness, achieving high correction rates in benchmark evaluations (Fatahi Bayat et al. 2023). In multiple-choice settings, distractor quality is measured along dimensions of plausibility and variety:

automatic metrics for incorrectness and semantic similarity provide quantitative assessments (Raina, Liusie, and Gales 2023), student-choice-prediction models estimate diagnostic power by modeling real-user error patterns (Lee, Kim, and Jo 2025), and synthesize best practices for distractor generation and evaluation (Alhazmi et al. 2024). RefineLab is a flexible and extensible framework that can incorporate a wide range of quality metrics along with their corresponding refinement operations, enabling targeted improvements in a controllable and resource-aware manner.

The RefineLab Framework

Notations and Formulation

Given a raw textual dataset with questions q and corresponding answers a, denoted as $D = \{(q_i, a_i)\}_{i=1}^{N}$, and a set of specified quality targets $\mathcal{T} = \{t_1, t_2, \ldots, t_k\}$, the goal of RefineLab is to transform D into a refined dataset D' by applying a set of selected refinement operations. Formally, we define a refinement process

$$\mathcal{R} : (D, \mathcal{T}) \;\mapsto\; D',$$

where $\mathcal{R} = \{r_1, r_2, \ldots, r_k\}$ is the set of available refinement operators applied selectively to each sample (q_i, a_i). The assignment of refinement operations is determined by solving the ILP problem described in Eq. (1), such that the resulting dataset D' better satisfies the specified quality targets $\mathcal{T}$ while adhering to the token-level budget constraint.

Each quality target $t_k \in \mathcal{T}$ represents a desired property of the dataset such as a target distribution over *topics*, *difficulty levels*, or *factual consistency*. These targets can be either explicitly provided by users (e.g., for custom evaluation settings) or automatically derived from real-world benchmarks or downstream task requirements. For simplicity, we assume that each refinement operator $r_k \in \mathcal{R}$ is designed to address a specific quality target $t_k \in \mathcal{T}$. That is, there exists a one-to-one mapping between refinement operations and quality targets, where each r_k is specialized to improve the dataset with respect to the corresponding t_k. Therefore, the quality gain Δ_{ik} in Eq. (1), which represents the improvement from applying refinement operation r_k to sample (q_i, a_i), is defined with respect to the corresponding target t_k, reflecting how well the refined sample aligns with the desired quality dimension addressed by r_k.

The RefineLab framework is shown in the top part of Figure 2. We next introduce the core components of the refinement process, including the design of refinement operations, the mechanism by which the ILP is solved by the assignment engine to determine optimal refinement decisions, and the data sample validation process.

Data Refinement Operators

Coverage Refiner (r_1, r_2). Datasets collected without reference to authorized guidelines often exhibit uneven representation of topics or skills. For example, in the *College Chemistry* subset of MMLU, fewer than 1% data cover organic chemistry, despite organic chemistry typically comprising over 10% of a standard college curriculum. Such under-representation can inflate model ability on over-represented topics while obscuring weaknesses in critical domains. To mitigate this issue, the *Coverage Refiner* realigns the dataset's empirical domain–skill distribution to a specified target vector t. This distribution vector could be provided by the user or drawn from a set of empirically validated and authorized human-examination guidelines, e.g., AP curriculum standards, grade-level learning objectives, to ensure that each domain category is represented according to specification.

The distribution alignment can be achieved by *increasing* the number of samples in underrepresented categories and *reducing* those in overrepresented categories, each incurring different costs. While the *assignment engine* determines which action should be applied to each QA sample, the *Coverage Refiner* operates on the assigned sample (q, a) with two possible actions: (1) **Removal (r_1)**, if fewer samples from the category of (q, a) are needed, it drops (q, a) from the dataset. The probability of removing a sample (q, a) is determined by the degree of *overrepresentation* of its category relative to the target distribution t; (2) **Expansion (r_2)**, if more samples from the category of (q, a) are needed, it generates additional instances by rephrasing (q, a). Specifically, the operator takes in (q, a) and a target topic (t^{topic}), prompts an LLM to produce lexically varied QA pairs that preserve the original question's topic focus and correctness. The generation hint includes the original QA pair, a brief instruction to maintain fidelity to the source topic and reasoning logic, and optionally a target lexical style or variation constraint. Multiple paraphrased versions are sampled, followed by lightweight filtering to eliminate degenerate or duplicate outputs.

Difficulty Calibrator (r_3, r_4, r_5). Real-world datasets frequently exhibit skewed difficulty distributions. For example, prior work has noted that most questions in MMLU emphasize knowledge recall over reasoning, highlighting the need to recalibrate difficulty levels and tailor the question mix to specific research tasks and user requirements. To address such concerns, we design the *Difficulty Calibrator* module to adjust a dataset's difficulty distribution via efficient pairwise comparisons against a curated seed set. This allows us to infer the relative difficulty of each question and subsequently align the overall difficulty distribution with a specified target, e.g., the desired percentage of samples in *easy*, *medium*, and *difficult* categories.

Specifically, calibration can be achieved in three ways: (1) **Removal (r_3)**, dropping samples from difficulty levels that are overrepresented relative to the target distribution; (2) **Generation (r_4)**, synthesizing new questions to increase the proportion of underrepresented difficulty levels; and (3) **Distractor Rewriting (r_5)**, modifying incorrect answer choices to adjust the cognitive load and thereby shift a question's difficulty level without changing its topic or factual correctness. Each of these approaches incurs a different refinement cost, but all contribute to enriching the refined dataset by improving the balance of difficulty levels.

Generation (r_4) creates new QA samples at a desired difficulty level, involves prompt conditioning the LLM with explicit difficulty specifications (t^{diff}), a domain anchor (e.g.,

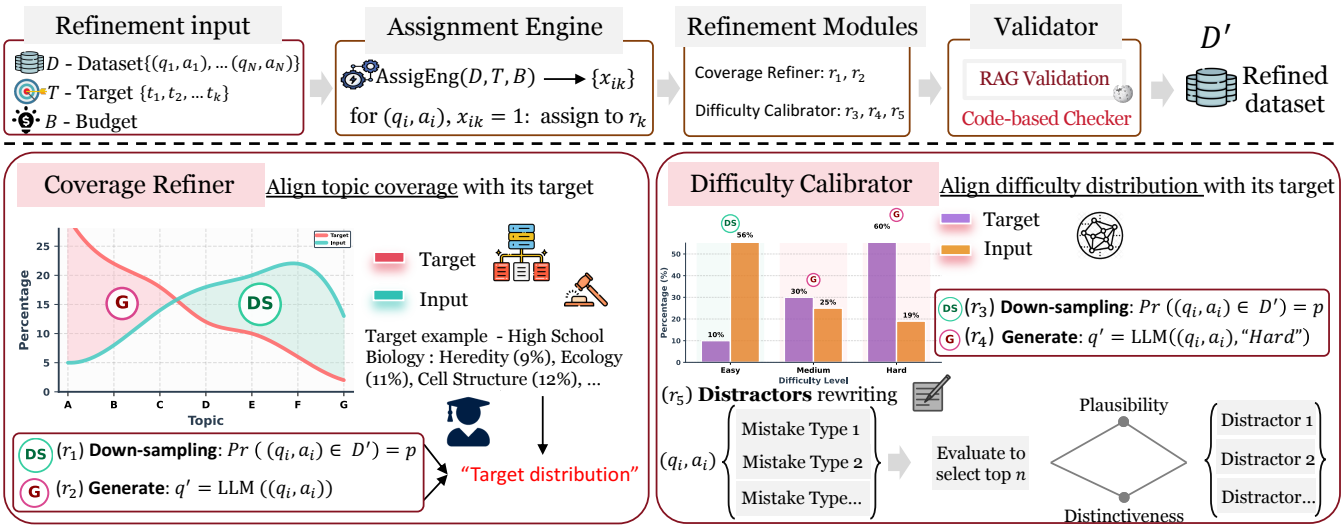

Figure 2: Top: the framework of RefineLab. Bottom: the Refinement Operations.

"college chemistry - thermodynamics"), and a reasoning indicator calibrated to the target level. For example, a *hard* question may require multi-step reasoning or distractor designs that encode subtle misconceptions.

Distractor Rewriting (r_5) is a cost-effective approach for adjusting the difficulty of multiple-choice questions. It modifies one or a few distractors while keeping the correct option intact. This method is generally less expensive than generating completely new questions, as it focuses on improving the quality of existing distractors rather than creating entirely new samples. It takes in a QA pair (q, a) and a target difficulty level (t^{diff}), then undergoes the following process to adjust the difficulty. The process begins by prompting an LLM to identify the set of relevant mistake types for the QA sample, and to generate candidate distractors that exemplify each specific mistake type. These candidates are scored based on both their plausibility (i.e., how likely they are to be a reasonable distractor), measured by a confidence score given by the LLM generator as a plausible answer, and their typological distinctiveness (i.e., how different they are from the correct answer), computed via embedding-based cosine distance from the correct answer. The top n distractors are selected, ensuring that each mistake category is represented at least once in the final set of distractors, enriching the question's difficulty without introducing significant overhead. Conversely, reducing difficulty can be achieved by removing or simplifying distractors with high semantic proximity to the correct answer or low plausibility.

To ensure that generated questions align with the intended difficulty levels, each new sample is compared against a curated seed set using pairwise Elo scoring (Elo 1978). First, exemplars with fixed difficulty ratings are sampled from the nearest semantic cluster of the generated question. An LLM is prompted to judge which sample is more difficult in each pairwise comparison. Based on the binary outcomes, the sample's provisional Elo score e_i is updated iteratively by

$$e_i \leftarrow e_i + K_{\text{Elo}}\big(r_{ij} - p_{ij}\big), \quad p_{ij} = \big(1 + 10^{(e_{\text{ex}} - e_i)/\eta}\big)^{-1},$$

where e_{ex} is the exemplar's Elo rating, and K_{Elo}, η are hyper-parameters. After convergence, the final score e_i is mapped to a discrete difficulty band using pre-defined thresholds, and only those samples whose estimated difficulty matches the target are retained.

To determine whether a given sample should undergo *Distractor Rewriting* or full *Generation*, the system considers both the difficulty gap and token-level cost. If the difficulty change required (e.g., moving from easy to hard) exceeds what is achievable through distractor modification alone, generation is preferred despite higher cost. Otherwise, rewriting is selected for its cost efficiency. This decision is encoded in the *assignment engine* based on estimated difficulty quality gain and refinement cost.

Assignment Engine

As shown in Algorithm 1, the main objective of the *Assignment Engine* is to solve the ILP task and determine, for each QA pair (q_i, a_i), the optimal refinement operation $x_{ik} \in \{0, 1\}$, along with the corresponding refinement targets: t_i^{topic} for coverage expansion and (t_i^{diff}) for difficulty calibration. These targets are sampled from underrepresented categories proportional to their mass of gaps between the empirical and target distribution. As shown in Eq. (1), the ILP involves estimating Δ_{ik}, the quality gain from applying r_k to (q_i, a_i), and the token cost c_{ik} associated with that refinement.

The *Assignment Engine* estimates Δ_{ik} and c_{ik} in two steps using representative subsets. First, for each refinement operation r_k and candidate target z (e.g., a specific topic or difficulty level), we apply r_k to a pilot batch $\mathcal{B}_{k,z}$ of QA samples conditioned on target z. The average improvement in the relevant quality metric is recorded as:

$$\Delta_{k,z} \approx \frac{1}{|\mathcal{B}_{k,z}|} \sum_{(q_j, a_j) \in \mathcal{B}_{k,z}} \Delta_{\text{metric}}(q_j, a_j, r_k, z),$$

where $\Delta_{\text{metric}}(\cdot)$ denotes the target-specific quality gain.

Algorithm 1: Assignment Engine

Input: dataset $D = \{(q_i, a_i)\}_{i=1}^{N}$; operations $\{r_k\}_{k=1}^{K}$; target topic and difficulty distributions $\mathcal{T}$; budget B

Output: $\{x_{ik}\}$, $\{t_i^{\text{topic}}\}$, $\{t_i^{\text{diff}}\}$

Identify underrepresented bins in $\mathcal{T}$, e.g., topics $\mathcal{T}^u_{\text{topic}}$ and difficulties $\mathcal{T}^u_{\text{diff}}$
for each $(q_i, a_i) \in D$ **do**
    Sample $t_i^{\text{topic}} \sim \mathcal{T}^u_{\text{topic}}, \quad t_i^{\text{diff}} \sim \mathcal{T}^u_{\text{diff}}$
end for
for each r_k and target $z \in \mathcal{T} \cup \mathcal{D}$ **do**
    Apply r_k on pilot batch $\mathcal{B}_{k,z}$ to estimate $\Delta_{k,z}$, $c_{k,z}$
end for
for each (q_i, a_i) and operation r_k **do**
    Set $\Delta_{ik} \leftarrow \Delta_{k,z_i}, \quad c_{ik} \leftarrow c_{k,z_i}$
end for
Solve ILP:

$$\max_{x_{ik} \in \{0,1\}} \sum_{i,k} \Delta_{ik} x_{ik} \quad \text{s.t.} \quad \sum_{i,k} c_{ik} x_{ik} \leq B$$

Similarly, the average token cost for invoking r_k under target z is estimated by:

$$c_{k,z} \approx \frac{1}{|\mathcal{B}_{k,z}|} \sum_{(q_j, a_j) \in \mathcal{B}_{k,z}} \text{Token}(q_j, a_j, r_k, z).$$

Then, for each QA pair (q_i, a_i), the assignment engine sets $\Delta_{ik} \leftarrow \Delta_{k,z_i}$ and $c_{ik} \leftarrow c_{k,z_i}$, where z_i is the sampled t_i^{topic} or t_i^{diff}, depending on the type of refinement. This batched estimation scheme enables the ILP to reason about cost–benefit trade-offs efficiently, without the overhead of probing every sample individually. To mitigate the computational cost of solving the ILP at scale, we instead solve its LP relaxation to obtain fractional assignments $\tilde{x}_{ik} \in [0, 1]$, followed by a rounding step that converts them to binary x_{ik} while preserving the budget constraint.

Validator

Previous studies have uncovered factual errors in widely used datasets. For example, the GSM8K dataset contains a proportion of incorrect solution labels, and some sociology sections of MMLU also include historically inaccurate statements. To mitigate these risks, the *Validator* employs two complementary pipelines tailored to question type.

For mathematical or multi-step reasoning items, we use PoT prompting (Chen et al. 2023): for each question–answer pair (q_i, a_i), we prompt the LLM to produce a derivation trace tri_i, then apply a code-based checker $\mathcal{J}$ such that $\mathcal{J}(tri_i) = \text{True}$ indicates procedural correctness. For factual or knowledge-based questions, we extract a keyword set w_i, retrieve relevant passages $Rp(w_i)$ from Wikipedia, and feed $\{q_i, a_i, Rp(w_i)\}$ into a verification prompt that asks the LLM to detect and correct any discrepancies. By combining PoT for numerical rigor with RAG (Lewis et al. 2021) for factual grounding, RefineLab achieves high integrity across both reasoning and factual domains.

Experiments

To evaluate RefineLab, we conduct extensive experiments to answer the following evaluation questions (EQ):

- **EQ1:** How RefineLab improves dataset quality across various dimensions;
- **EQ2:** How RefineLab balances cost and quality;
- **EQ3:** How refined datasets affect LLM evaluation;
- **EQ4:** How refined datasets help LLM fine-tuning.

Experiments Setup

Dataset. We evaluate RefineLab on the refinement of multiple widely-used datasets: **MMLU**: A multitask benchmark for evaluating knowledge recall and reasoning across 57 academic subjects. **RACE** (Lai et al. 2017): A reading comprehension dataset sourced from English exams for Chinese middle and high school students. **GSM8K** (Cobbe et al. 2021): A grade school math word problem dataset requiring multi-step arithmetic and algebraic reasoning. **OpenbookQA** (Mihaylov et al. 2018): A commonsense and science QA dataset based on an open-book of elementary-level science facts. **HumanEval** (Chen et al. 2021): A code generation and completion benchmark featuring programming tasks with functional correctness tests.

Hyperparameters. The hyperparameter setting includes default difficulty level targets $[0.0, 0.4, 0.6]$ (stress-testing profile with 40% medium, 60% hard) and coverage targets combining college curriculum, high-school AP guidelines, and professional certification test guides. Budgets $B \in \{0.25C, 0.5C, 1C\}$ are set relative to the cost C on the full refinement operation. Elo updates $K = 64$, $\eta = 400$ with three difficulty bands. The top 3 (n = 3) distractors are selected for distractor rewriting. GPT-4o is used as the default backbone LLM for refinement operations if needed.

Experimental Results

EQ1: Dataset Quality Improvement. We evaluate the quality of RefineLab-refined dataset across several dimensions, which are introduced below, along with their respective evaluation metrics. 1) **Alignment of Coverage and Difficulty.** We compute the Jensen–Shannon Divergence (JSD) between the empirical distribution p and the target distribution t, defined as $\text{JSD}(p\|t) = \frac{1}{2}\text{KL}(p\|M) + \frac{1}{2}\text{KL}(t\|M)$, where $M = \frac{1}{2}(p + t)$ is the mixture distribution and $\text{KL}(p\|M) = \sum_i p(i) \log \frac{p(i)}{M(i)}$, where i indicates the i-th category (topic or difficulty level) of the distribution. JSD is symmetric and bounded between 0 and 1, with lower values indicating better alignment with the target. 2) **Effectiveness of Distractor Rewriting.** For each multiple-choice question, we evaluate the effectiveness of distractor rewriting, by computing the Shannon entropy of distractor-type proportions $p_1, \ldots, p_K$ as $H = -\sum_{k=1}^{K} p_k \log p_k$, and normalize it by the maximum entropy $\log K$, where K indicates total number of mistake types, yielding the normalized entropy $H_{\text{norm}} = H/\log K$. We report the average of H_{norm} across the dataset; higher values indicate greater distractor diversity

Dataset	Coverage (JSD)↓		Difficulty (JSD)↓		Distractor Diversity↑		Correction Ratio (%)↑	Error Rate (%)↓
	Orig.	Refined	Orig.	Refined	Orig.	Refined	Refined	Refined
MMLU	0.046	0.001	0.142	0.005	0.362	0.585	90.0	2.3
RACE	0.124	0.010	0.143	0.042	0.416	0.597	95.0	1.2
GSM8K	0.112	0.005	0.133	0.005	–	–	95.0	3.0
OpenbookQA	0.300	0.012	0.320	0.090	0.239	0.456	94.4	1.0
HumanEval	0.280	0.002	0.131	0.021	–	–	–	1.8

Table 1: Dataset quality evaluation before (Orig.) and after refinement (Refined). Correction Ratio and Error Rate are evaluated only on refined samples. (–) indicates not applicable, e.g., no errors in HumanEval or non-MCQ datasets.

Budget	Strategy	Cov. (JSD)↓	Diff. (JSD)↓	Dist. (Div.) ↑
0.25C	**RefineLab**	**0.183**	0.388	**0.243**
	Greedy	0.252	**0.328**	0.243
	Uniform	0.250	0.418	0.213
0.5C	**RefineLab**	**0.063**	**0.317**	**0.286**
	Greedy	0.190	0.342	0.286
	Uniform	0.183	0.397	0.259
1.0C	**RefineLab**	**0.019**	**0.295**	**0.331**
	Greedy	0.063	0.322	0.300
	Uniform	0.065	0.412	0.250

Table 2: Cost and quality balance of different strategies

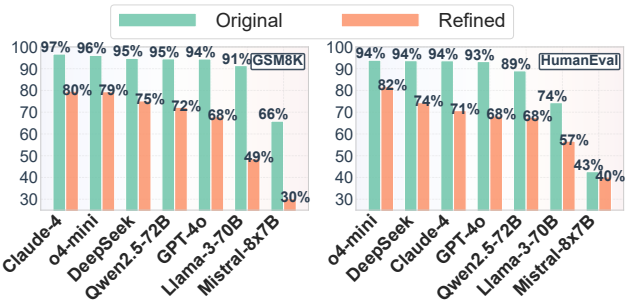

Figure 3: Comparison of LLM performance on original and refined datasets. Left: GSM8K (original vs. refined). Right: HumanEval (original vs. refined).

and improved question discriminability. 3) **Successful Correction Ratio.** We measure the correction rate on samples originally flagged as *error* by the Validator, i.e., reporting the proportion that are successfully corrected after refinement based on human annotation. 4) **Overall Error Rate.** we draw a stratified random sample of 400 items from each refined dataset, and report the overall error rate based on human evaluation. The human evaluation process involves independent review of each sample by three human annotators with domain expertise. Each annotator assesses whether the QA pair contains any factual, logical, or structural flaws. Annotators are blind to the refinement strategy and base their judgments solely on correctness. The majority vote is used to determine the ground-truth correctness of each sample.

Table 1 summarizes the evaluation results. As it shows, RefineLab substantially reduces divergence from the target distributions: coverage JSD decreases by over 90% and difficulty JSD by over 70%. Additionally, the effectiveness of distractor rewriting is evident, with distractor diversity increasing from 40% to 90%, demonstrating that the refined distractors capture a broader and more informative set of misconception patterns. The high correction rates (over 94.4%) further indicate that the Validator successfully corrects the vast majority of detected errors. Finally, across all datasets, RefineLab yields remarkably low error rates (typically below 4%), indicating that it reliably produces high-quality, valid QA pairs even under constrained generation.

EQ2: Cost and Quality Balance. To evaluate the effectiveness of our *Assignment Engine* on balancing cost and quality, we conduct experiments on a pilot subset, comprising 10% of the MMLU dataset, under three budget settings: $B \in \{0.25\,C,\ 0.5\,C,\ 1.0\,C\}$, where C denotes the cost of running all refinement on the pilot subset. We compare three allocation strategies: **a)** our **RefineLab**; **b)** a **Greedy** approach that iteratively selects the (q_i, a_i)–r_k pair with the highest utility–cost ratio, allocating the budget accordingly; **c)** a **Uniform** way that evenly splits the budget B across all operations, with random data selection.

For each strategy and budget, we obtain a refined subset and measure the alignment of coverage and difficulty, as well as the distractor diversity. The total cost per QA sample remains highly affordable, typically up to $0.03 per interaction—assuming all refinement operations are applied with GPT-4o as the backbone LLM. As shown in Table 2, RefineLab consistently achieves the best or near-best performance across all metrics, particularly under tight budgets where intelligent allocation is critical. For instance, at $B = 0.25\,C$, RefineLab achieves significantly lower topic coverage JSD than Greedy, while maintaining strong difficulty alignment. Greedy occasionally performs well on isolated metrics (e.g., difficulty at low budget), but lacks global coordination. As the budget increases, all strategies improve, but RefineLab maintains a clear lead, demonstrating its ability to coordinate multi-dimensional quality improvements more effectively.

EQ3: Impact of Refined Datasets on LLM Evaluation. We evaluate how RefineLab-refined datasets can affect LLM evaluation by investigating whether **a) model performances can be better distinguished**; and **b) the refined dataset yields consistent model rankings** under different prompt-

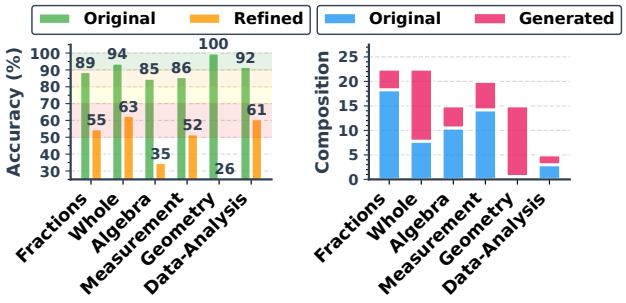

Figure 4: Performance breakdown on original and refined GSM8K (left), and refined dataset composition (right).

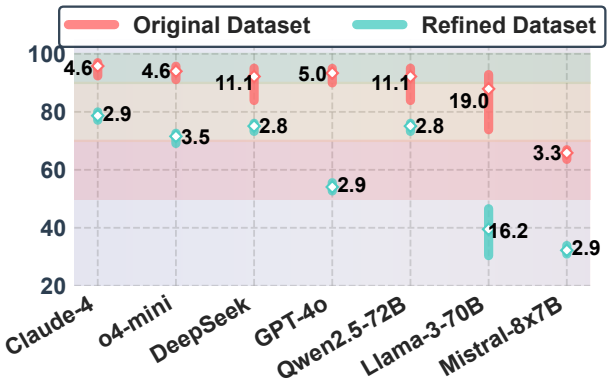

Figure 5: Performance variation on GSM8K across prompting strategies, comparing original vs. refined datasets.

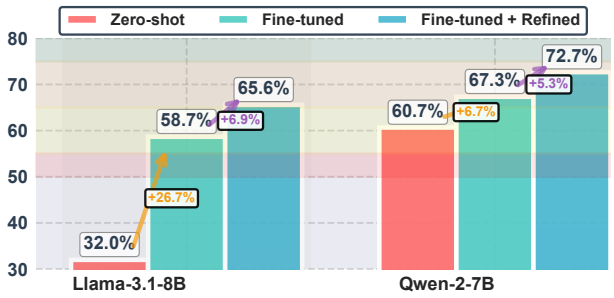

Figure 6: Performance improvements from fine-tuning with RefineLab augmentations compared to the original dataset.

ing strategies. We employ 7 popular LLMs here: o4-mini (OpenAI 2025), GPT-4o (OpenAI 2024), Claude-4-Sonnet (Anthropic 2025), DeepSeek-V3 (DeepSeek-AI et al. 2025), Llama-3-70B-Instruct (AI@Meta 2024), Qwen2.5-72B-Instruct (Yang et al. 2025), and Mistral-8x7B-Instruct-v0.1 (Jiang et al. 2024).

a) Improved Performance Distinction. Figure 3 presents a comparison of LLM performance on both original and refined datasets. On the left, it shows the performance of various LLMs on the original GSM8K dataset versus the refined version. On the right, a similar comparison is made for HumanEval, illustrating how the refined dataset affects model performance. On GSM8K, the original dataset shows that top-tier LLMs (the top 5) are clustered within a narrow 2.2% range, masking the true performance gaps. After refinement, the performance spread expands to 11.8%, with the gap between o4-mini and Mistral-8x7B-Instruct-v0.1 increasing from 30.8% to 49.4%. Similarly, on HumanEval, the original dataset has a narrow 0.6% spread among the top models, which grows to 13.4% after refinement, with weaker models showing more noticeable relative shifts. To better understand this widening gap, we analyze Llama-3-70B-Instruct performance by topic in refined GSM8K (Figure 4). The refined dataset increases representation in categories such as *Geometry*, which are underrepresented in the original, by generating *hard* examples. It corresponds to the largest performance drops, with accuracies decreasing by over 60%. In contrast, accuracy drops less on topics such as *Whole Numbers*. This selective degradation reveals that RefineLab surfaces previously obscured model weaknesses by rebalancing coverage and injecting difficulty into subdomains, thereby enhancing the dataset's ability to distinguish true capability differences among LLMs.

b) Consistent Model Rankings. Figure 5 plots the variation in model performance across different prompting strategies for GSM8K. Specifically, we evaluate on the GSM8K dataset using 6 prompting variants drawn from: *role play prompting*, *chain-of-thought reasoning*, and *self-consistency prompting*. We observe that the average performance range across LLMs shrinks from 8.4% on the original data to 4.9% after refinement. This indicates that the refined dataset enhances consistency in model rankings.

EQ4: Refined Datasets for LLM Fine-Tuning. To assess RefineLab as a data augmentation tool, we fine-tune Llama-3.1-8B and Qwen-2-7B using 1200 GSM8K original and RefineLab-refined data, and evaluate on a 150-item test set, combining both original and refined samples. Figure 6 shows that RefineLab augmentations produce improvements compared with the original dataset. Llama-3.1-8B improves from 58.7% to 65.6%, while Qwen-2-7B rises from 67.3% to 72.7%, confirming consistent benefits across architectures.

Conclusion

We introduce RefineLab, an LLM-driven, budget-aware framework designed to refine QA datasets by optimizing key quality dimensions under token-level cost constraints. RefineLab integrates refinement operations, including coverage alignment and difficulty calibration, and formulates the operation assignment as an ILP problem to enable efficient token-budget allocation. Experiments show that RefineLab markedly reduces distributional gaps, achieves near-zero label errors, enhances LLMs evaluation, and boosts fine-tuning efficiency. Future work will extend RefineLab to handle multi-turn and open-ended QA format, where refinement must account for longer contexts and dialogue coherence.

Acknowledgments

This work is supported by the National Science Foundation (NSF) under award No. 2321054.

References

AI@Meta. 2024. Llama 3 Model Card. https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Alhazmi, E.; Sheng, Q. Z.; Zhang, W. E.; Zaib, M.; and Alhazmi, A. 2024. Distractor Generation in Multiple-Choice Tasks: A Survey of Methods, Datasets, and Evaluation. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 14437–14458. Miami, Florida, USA: Association for Computational Linguistics.

Anthropic. 2025. Claude Sonnet 4 model card. https://docs.anthropic.com/en/docs/about-claude/models/overview. Accessed: 2025-07-25.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; de Oliveira Pinto, H. P.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; Ray, A.; Puri, R.; Krueger, G.; Petrov, M.; Khlaaf, H.; Sastry, G.; Mishkin, P.; Chan, B.; Gray, S.; Ryder, N.; Pavlov, M.; Power, A.; Kaiser, L.; Bavarian, M.; Winter, C.; Tillet, P.; Such, F. P.; Cummings, D.; Plappert, M.; Chantzis, F.; Barnes, E.; Herbert-Voss, A.; Guss, W. H.; Nichol, A.; Paino, A.; Tezak, N.; Tang, J.; Babuschkin, I.; Balaji, S.; Jain, S.; Saunders, W.; Hesse, C.; Carr, A. N.; Leike, J.; Achiam, J.; Misra, V.; Morikawa, E.; Radford, A.; Knight, M.; Brundage, M.; Murati, M.; Mayer, K.; Welinder, P.; McGrew, B.; Amodei, D.; McCandlish, S.; Sutskever, I.; and Zaremba, W. 2021. Evaluating Large Language Models Trained on Code. arXiv:2107.03374.

Chen, W.; Ma, X.; Wang, X.; and Cohen, W. W. 2023. Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks. arXiv:2211.12588.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168.

DeepSeek-AI; Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; Dai, D.; Guo, D.; and Others. 2025. DeepSeek-V3 Technical Report. arXiv:2412.19437.

DeSalvo, G.; Kagy, J.-F.; Karydas, L.; Rostamizadeh, A.; and Kumar, S. 2025. SoftSRV: Learn to Generate Targeted Synthetic Data. arXiv:2410.16534.

Elo, A. E. 1978. *The Rating of Chessplayers, Past and Present*. Arco Publishing.

Fatahi Bayat, F.; Qian, K.; Han, B.; Sang, Y.; Belyy, A.; Khorshidi, S.; Wu, F.; Ilyas, I.; and Li, Y. 2023. FLEEK: Factual Error Detection and Correction with Evidence Retrieved from External Knowledge. In Feng, Y.; and Lefever, E., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 124–130. Singapore: Association for Computational Linguistics.

Guo, T.; Guo, K.; Nan, B.; Liang, Z.; Guo, Z.; Chawla, N. V.; Wiest, O.; and Zhang, X. 2023. What can Large Language Models do in chemistry? A comprehensive benchmark on eight tasks. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.

Huang, Y.; Wu, S.; Gao, C.; Chen, D.; Zhang, Q.; Wan, Y.; Zhou, T.; Xiao, C.; Gao, J.; Sun, L.; and Zhang, X. 2025. DataGen: Unified Synthetic Dataset Generation via Large Language Models. In *The Thirteenth International Conference on Learning Representations*.

Iskander, S.; Tolmach, S.; Shapira, O.; Cohen, N.; and Karnin, Z. 2024. Quality Matters: Evaluating Synthetic Data for Tool-Using LLMs. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 4958–4976. Miami, Florida, USA: Association for Computational Linguistics.

Jiang, A. Q.; Sablayrolles, A.; Roux, A.; Mensch, A.; Savary, B.; Bamford, C.; Chaplot, D. S.; de las Casas, D.; Hanna, E. B.; Bressand, F.; Lengyel, G.; Bour, G.; Lample, G.; Lavaud, L. R.; Saulnier, L.; Lachaux, M.-A.; Stock, P.; Subramanian, S.; Yang, S.; Antoniak, S.; Scao, T. L.; Gervet, T.; Lavril, T.; Wang, T.; Lacroix, T.; and Sayed, W. E. 2024. Mixtral of Experts. arXiv:2401.04088.

Khan, Z.; Stengel-Eskin, E.; Cho, J.; and Bansal, M. 2025. DataEnvGym: Data Generation Agents in Teacher Environments with Student Feedback. In *The Thirteenth International Conference on Learning Representations*.

Lai, G.; Xie, Q.; Liu, H.; Yang, Y.; and Hovy, E. 2017. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In Palmer, M.; Hwa, R.; and Riedel, S., eds., *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 785–794. Copenhagen, Denmark: Association for Computational Linguistics.

Lee, Y.; Kim, S.; and Jo, Y. 2025. Generating Plausible Distractors for Multiple-Choice Questions via Student Choice Prediction. In Che, W.; Nabende, J.; Shutova, E.; and Pilehvar, M. T., eds., *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 23669–23692. Vienna, Austria: Association for Computational Linguistics. ISBN 979-8-89176-251-0.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; tau Yih, W.; Rocktäschel, T.; Riedel, S.; and Kiela, D. 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv:2005.11401.

Li, X. L.; Kaiyom, F.; Liu, E. Z.; Mai, Y.; Liang, P.; and Hashimoto, T. 2025a. AutoBencher: Towards Declarative Benchmark Construction. In *The Thirteenth International Conference on Learning Representations*.

Li, Z.; Chen, L.; Andrews, J.; Ba, Y.; Zhang, Y.; and Xiang, A. 2025b. GenDataAgent: On-the-fly Dataset Augmenta-

tion with Synthetic Data. In *The Thirteenth International Conference on Learning Representations*.

Lin, S.; Hilton, J.; and Evans, O. 2022. TruthfulQA: Measuring How Models Mimic Human Falsehoods. arXiv:2109.07958.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *EMNLP*.

Miranda, B.; Lee, A.; Sundar, S.; Casasola, A.; Schaeffer, R.; Obbad, E.; and Koyejo, S. 2025. Beyond Scale: The Diversity Coefficient as a Data Quality Metric for Variability in Natural Language Data. arXiv:2306.13840.

OpenAI. 2024. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/. Accessed: 2025-07-25.

OpenAI. 2025. Introducing OpenAI o3 and o4-mini. https://openai.com/index/introducing-o3-and-o4-mini/. Accessed: 2025-07-25.

Qian, C.; Liu, W.; Liu, H.; Chen, N.; Dang, Y.; Li, J.; Yang, C.; Chen, W.; Su, Y.; Cong, X.; et al. 2023. Chatdev: Communicative agents for software development. *arXiv preprint arXiv:2307.07924*.

Raina, V.; Liusie, A.; and Gales, M. 2023. Assessing Distractors in Multiple-Choice Tests. In Deutsch, D.; Dror, R.; Eger, S.; Gao, Y.; Leiter, C.; Opitz, J.; and Rücklé, A., eds., *Proceedings of the 4th Workshop on Evaluation and Comparison of NLP Systems*, 12–22. Bali, Indonesia: Association for Computational Linguistics.

Reuel, A.; Hardy, A.; Smith, C.; Lamparth, M.; Hardy, M.; and Kochenderfer, M. 2024. BetterBench: Assessing AI Benchmarks, Uncovering Issues, and Establishing Best Practices. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N. A.; Khashabi, D.; and Hajishirzi, H. 2023. Self-Instruct: Aligning Language Models with Self-Generated Instructions. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 13484–13508. Toronto, Canada: Association for Computational Linguistics.

Wang, Y.; Ma, X.; Zhang, G.; Ni, Y.; Chandra, A.; Guo, S.; Ren, W.; Arulraj, A.; He, X.; Jiang, Z.; Li, T.; Ku, M.; Wang, K.; Zhuang, A.; Fan, R.; Yue, X.; and Chen, W. 2024. MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark. arXiv:2406.01574.

Xie, S. M.; Santurkar, S.; Ma, T.; and Liang, P. 2023. Data Selection for Language Models via Importance Resampling. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yang, Q. A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Tang, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; and Qiu, Z. 2025. Qwen2.5 Technical Report. arXiv:2412.15115.

Yu, Y.; Zhuang, Y.; Zhang, J.; Meng, Y.; Ratner, A.; Krishna, R.; Shen, J.; and Zhang, C. 2023. Large Language Model as Attributed Training Data Generator: A Tale of Diversity and Bias. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).