

DATA SWARMS: Optimizable Generation of Synthetic Evaluation Data

Shangbin Feng, Yike Wang, Weijia Shi, Yulia Tsvetkov

University of Washington
shangbin@cs.washington.edu

Abstract

We propose DATA SWARMS, an algorithm to optimize the generation of synthetic evaluation data and advance quantitative desiderata of LLM evaluation. We first train a swarm of initial data generators using existing data, and define various *evaluation objectives* to reflect the desired properties of evaluation (e.g., generate more *difficult* problems for the evaluated models) and quantitatively evaluate data generators. We then employ particle swarm optimization to optimize the swarm of data generators, where they collaboratively search through the model parameter space to find new generators that advance these objectives. We further extend it to ADVERSARIAL SWARMS, where the data generator swarm generates harder data while the test taker model swarm learns from such data, co-evolving dynamically for better data and models simultaneously. Extensive experiments demonstrate that DATA SWARMS outperforms eight baselines across five evaluation objectives, while ADVERSARIAL SWARMS produce more robust learning of synthetic data and stronger generalization. Further analysis reveals that DATA SWARMS successfully optimizes compositions of multiple evaluation objectives and generalizes to new off-the-shelf LLMs, unseen at optimization time.¹

1 Introduction

With new LLMs emerging daily—each trained with different training recipes, domain expertise, and alignment strategies—rigorous evaluation is more critical than ever to reveal the strengths and weaknesses of LLMs, and guide model selection for diverse applications. The most common evaluation approach relies on *static* data, ranging from single-skill datasets (Zellers et al., 2019; Hendrycks et al., 2021) to comprehensive multi-task benchmarks

(Liang et al., 2022; Wang et al., 2019). As state-of-the-art models become saturated on these static datasets (Fourrier et al., 2024) with contamination concerns (Golchin and Surdeanu, 2024), new evaluation methods focus on *synthetic data generation* for incorporating temporal updates (Kasai et al., 2023), adaptively probing model weakness (Brown et al., 2025), leveraging privileged information (Li et al., 2025b), and more (Kim et al., 2024).

However, existing approaches for generating synthetic evaluation data are largely heuristic, relying on intuitive, trial-and-error engineering decisions, such as prompt design (Wang et al., 2023b), agent setup (Li et al., 2025a), and reporting successful outcomes as “best practices and lessons” (Liu et al., 2024a). It is important to systematize this process, and that the generated evaluation data achieves *evaluation objectives* (Li et al., 2025b), for example, generating *difficult* evaluation data that exposes weaknesses, or better *separating* a pool of models to offer statistically significant signals about their strengths and weaknesses. While heuristic-based approaches could achieve these goals, they hardly generalize to new model capabilities and evaluation domains without another round of manual trial-and-error. This calls for a general, automated approach to synthetic evaluation data generation with well-defined evaluation objectives.

In this work, we take an *optimization view towards evaluation data*. We propose DATA SWARMS, optimizing a swarm of data generator models towards quantitative objectives with swarm intelligence. Particle swarm optimization (PSO) (Kennedy and Eberhart, 1995) is an algorithm that optimizes a swarm of continuous representations for a utility function, guided by particle- and swarm-level utility signals (§2). Guided by its initial success in multi-LLM collaboration (Feng et al., 2025a,b), we employ PSO to optimize data-generator LMs. Specifically, given a static seed dataset, we train an initial swarm of data gener-

¹Our code is available at https://github.com/BunsenFeng/data_swarm.

ators. For each iteration, we generate evaluation instances with each data generator, evaluate test taker models on them, calculate the evaluation objective metrics (e.g., *difficult*: one minus the max performance of test takers), and employ PSO to optimize the data generator swarm guided by the metrics. This iterative process produces optimized data generators and new evaluation data characterized by specific evaluation objectives, and we propose a suite of five objectives—*difficult*, *separate*, *novel*, *consistent*, *personalized*—as evaluation data characteristics to optimize towards.

We also propose ADVERSARIAL SWARMS, an extended algorithm in which a data generator swarm and a test taker swarm co-evolve in an adversarial setting. In *data-step*, the data swarm takes a PSO optimization step as previously described with the current snapshot of test takers, aiming to generate harder data; In *model-step*, the test taker swarm takes a PSO optimization step guided by their performance on this iteration’s generated data. By alternating between the data and model steps, ADVERSARIAL SWARMS jointly enhances both synthetic data generation and model capabilities.

Extensive experiments with four domains and eight baselines demonstrate that DATA SWARMS advances the five evaluation objectives, outperforming baselines across four data domains. ADVERSARIAL SWARMS further improves model performance on a held-out set, outperforming fine-tuning with either static or generated data by 8.2% on average. Empirical analyses reveal that DATA SWARMS generalizes to unseen test takers, works for tasks with verifiable ground truths, and successfully optimizes compositions of evaluation objectives. In sum, DATA SWARMS uniquely offers an optimization angle to quantitatively measure and improve synthetic evaluation data—enabling the creation of harder dynamic benchmarks, supporting the evaluation of emerging model capabilities, and reducing risks of data contamination and model exposure.

2 Methodology

We propose DATA SWARMS, an algorithm to optimize synthetic evaluation data generation for diverse evaluation objectives (e.g., generating more difficult problems). Given seed data \mathcal{D} and a pool of test taker models $\{\mathbf{m}_i\}_{i=1}^n$, we train an initial swarm of data generators (§2.1), define a suite of evaluation objectives (§2.2), and optimize the data generators with particle swarm intelligence (PSO) towards these objectives (§2.3). DATA

SWARMS transcends the heuristics-driven status quo in synthetic data generation by introducing an optimization-based framework that enables the creation of challenging evaluation data, consistent evaluation, and novel insights into model behavior.

2.1 Training Initial Data Generators

While existing datasets are often offered as a monolithic resource, there is inherent diversity within any dataset (Moayeri et al., 2024): different instances test different variations and mixtures of model capabilities, forming clusters and taxonomies of sub-evaluations (Zeng et al., 2025). Given seed data \mathcal{D} in an existing dataset, we propose to train a swarm of data generators reflecting different sub-evaluations within \mathcal{D} through clustering. By having a pool of multiple data generators, PSO could enable their collaborative search in the model weight space to optimize synthetic data generation (§2.3).

Concretely, we cluster \mathcal{D} into N subsets $(\mathcal{D}_1, \dots, \mathcal{D}_N) = \text{cluster}(\mathcal{D}, N)$ with K-means over average-pooled query embeddings. For each \mathcal{D}_i , we train a data generator \mathbf{x}_i to model this cluster of queries: specifically, we randomly sample $(\mathbf{d}_1, \dots, \mathbf{d}_{2k}) \sim \mathcal{D}_i$ and employ the following format: “*You are an expert in generating synthetic evaluation data, specifically about <domain>. You are given a set of k examples. Please follow the pattern and generate k more examples. Examples: $\mathbf{d}_1, \dots, \mathbf{d}_k, \underline{\mathbf{d}_{k+1}, \dots, \mathbf{d}_{2k}}$ ”.*

We then fine-tune an off-the-shelf model with a pool of these prompts into data generator \mathbf{x}_i , where $\mathbf{d}_{k+1}, \dots, \mathbf{d}_{2k}$ is the model output for supervised fine-tuning. At inference time, we randomly sample k examples $(\mathbf{d}'_1, \dots, \mathbf{d}'_k) \sim \mathcal{D}_i$ and expect the model to generate k new examples $(\mathbf{d}'_{k+1}, \dots, \mathbf{d}'_{2k}) \sim \mathbf{x}_i(\mathbf{d}'_1, \dots, \mathbf{d}'_k)$ for evaluation.

2.2 Evaluation Objectives

While existing evaluation efforts have focused on the conceptual and intuitive novelty of datasets, recent research begins to *quantify* the quality of evaluation data (Li et al., 2025b). Given a data generator or data generation algorithm \mathbf{x} and a pool of test taker models $\{\mathbf{m}_i\}_{i=1}^n$, we generate synthetic data $\mathcal{D}_{gen} \sim \mathbf{x}$ and define a suite of the following five *evaluation objectives* $f(\mathbf{x} \mid \{\mathbf{m}_i\}_{i=1}^n)$, which we later seek to advance by optimizing data generators $\{\mathbf{x}_i\}_{i=1}^N$ with swarm intelligence. (§2.3)

Difficult Static datasets are often saturated for new LLMs (Fourrier et al., 2024), thus it has always been important to design difficult and challenging

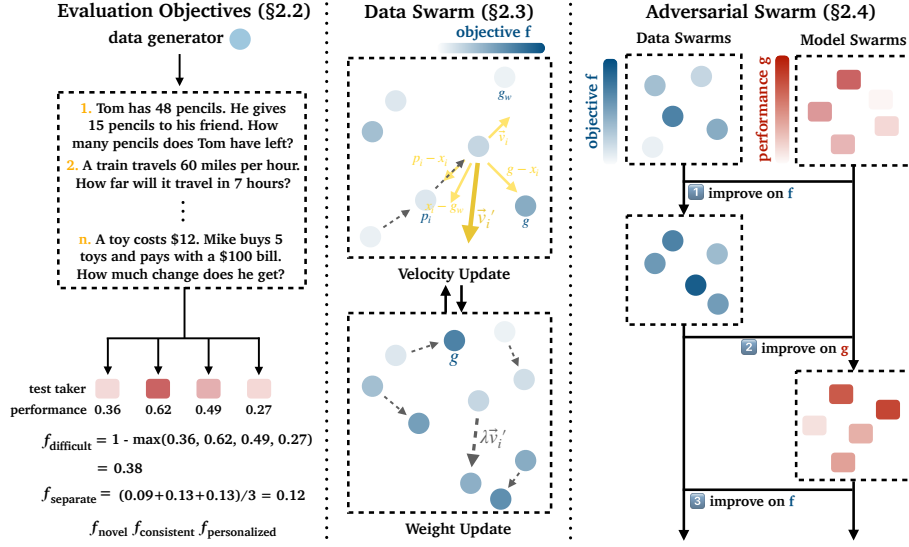


Figure 1: Overview of DATA SWARMS. (Left) Given a data generator, we sample problems, evaluate test taker LMs on them, and calculate objectives. (§2.2) (Center) We optimize multiple data generators with particle swarm optimization to optimize for evaluation objectives. Specifically, models are impacted by the personal/global best/worst checkpoints to update velocity and explore model weight space for optimization. (Right) In ADVERSARIAL SWARMS, data generators and test takers co-evolve adversarially to produce better data and models.

evaluation data. We define the *difficult* objective, aiming to lower performance for the pool of test taker models and expose their weaknesses:

$$f_{difficult}(\mathbf{x} \mid \{\mathbf{m}_i\}_{i=1}^n) = 1 - \max_{i=1}^n \text{performance}(\mathbf{m}_i \mid \mathcal{D}_{gen})$$

Consistent For a data generator \mathbf{x} , we could repeatedly sample from it to obtain different generated datasets $(\mathcal{D}_{gen,1}, \dots, \mathcal{D}_{gen,k}) \sim \mathbf{x}$. The generated datasets should be robust and *consistent*, i.e., model performance on any $\mathcal{D}_{gen,i}$ should not greatly fluctuate. We calculate the standard deviation std and define the *consistent* objective as:

$$f_{consistent}(\mathbf{x} \mid \{\mathbf{m}_i\}_{i=1}^n) = 1 - \frac{1}{n} \sum_{i=1}^n \text{std}(\{\text{performance}(\mathbf{m}_i) \mid \mathcal{D}_{gen,j}\}_{j=1}^k)$$

Separate Existing datasets often result in close performance across models, struggling to distinguish their relative strength. We define the *separate* objective, seeking to widen the performance gap between models with generated evaluation data: we sort $\{\text{perf}(\mathbf{m}_i \mid \mathcal{D}_{gen})\}_{i=1}^n$ in increasing order and:

$$f_{separate}(\mathbf{x} \mid \{\mathbf{m}_i\}_{i=1}^n) = \frac{1}{n-1} \sum_{i=1}^{n-1} (\text{perf}(\mathbf{m}_{i+1} \mid \mathcal{D}_{gen}) - \text{perf}(\mathbf{m}_i \mid \mathcal{D}_{gen}))$$

where perf is short for performance.

Novel By generating new synthetic evaluation data \mathcal{D}_{gen} , we seek to reveal novel insights about model performance different from what is offered by existing data \mathcal{D} . We define the *novel* objective as the distance between two performance arrays:

$$f_{novel}(\mathbf{x} \mid \{\mathbf{m}_i\}_{i=1}^n) = \text{dist}(\{\text{perf}(\mathbf{m}_i \mid \mathcal{D}_{gen})\}_{i=1}^n, \{\text{perf}(\mathbf{m}_i \mid \mathcal{D})\}_{i=1}^n)$$

where the distance function dist could be inverse ranked correlation (Li et al., 2025b), KL divergence, and more distance-based metrics.

Personalized For user-facing tasks such as instruction following, we should generate instructions that resemble what real-world users might actually ask about. We employ WildChat (Zhao et al., 2024) as a real-world user instruction set \mathcal{D}_{user} . For generated instruction $\mathbf{d}_{gen} \sim \mathcal{D}_{gen}$, we employ embedding similarity in retrieval systems to quantify its relevance to user queries $r(\mathbf{d}_{gen}, \mathcal{D}_{user}) = \frac{1}{k} \sum_{i=1}^k \text{sim}(\mathbf{d}_{gen}, \mathbf{d}_{user,i})$, the average embedding similarity between the generated instruction and the top-k similar instructions in \mathcal{D}_{user} . We define the *personalized* objective by averaging over \mathcal{D}_{gen} :

$$f_{personalized}(\mathbf{x} \mid \{\mathbf{m}_i\}_{i=1}^n) = \sum_{i=1}^{|\mathcal{D}_{gen}|} r(\mathbf{d}_{gen,i}, \mathcal{D}_{user})$$

By defining a suite of five evaluation objectives $f(\mathbf{x} \mid \{\mathbf{m}_i\}_{i=1}^n)$, we comprehensively measure the quality of generated evaluation data and seek to improve data generators $\{\mathbf{x}_i\}_{i=1}^N$ towards these objectives with swarm intelligence. (§2.3)

2.3 Data Swarms

We propose DATA SWARMS, a collaborative search algorithm to optimize data generators $\{\mathbf{x}_i\}_{i=1}^N$ with particle swarm optimization (PSO) (Kennedy and Eberhart, 1995). Simply put, we iteratively learn a weighted combination of data generator LLMs using the PSO algorithm. Specifically, the model weights of each \mathbf{x}_i represent its *location* in the space of model weights. Evaluation objectives $f(\mathbf{x}_i | \{\mathbf{m}_i\}_{i=1}^n)$ are the utility function to be optimized in the model weight space. We track several variables for the data swarm:

- swarm-level: *global best* \mathbf{g} , the best-found model across all of $\{\mathbf{x}_i\}_{i=1}^N$ ’s search trajectories in the weight space; and conversely *global worst* \mathbf{g}_w .
- individual-level: *personal best* \mathbf{p}_i , the best-found model weights in \mathbf{x}_i ’s search trajectory; *velocity* \mathbf{v}_i , the direction in the weight space that the model move next, initialized as $\mathbf{0}$.

In each step, we update the velocity of models guided by the individual- and swarm-level signals:

$$\mathbf{v}_i \leftarrow \frac{1}{\mathcal{C}} [r_v \phi_v \mathbf{v}_i + r_p \phi_p (\mathbf{p}_i - \mathbf{x}_i) + r_g \phi_g (\mathbf{g} - \mathbf{x}_i) - r_w \phi_w (\mathbf{g}_w - \mathbf{x}_i)]$$

where the inertia term $r_v \phi_v \mathbf{v}_i$ maintains part of the model’s current velocity to chart an independent path; $r_p \phi_p (\mathbf{p}_i - \mathbf{x}_i)$, $r_g \phi_g (\mathbf{g} - \mathbf{x}_i)$, $-r_w \phi_w (\mathbf{g}_w - \mathbf{x}_i)$ indicates the model is drawn by personal/global best and repelled by global worst to further explore/escape from these weight areas due to continuity in the model weight neighborhood (Eilertsen et al., 2020); $r_v, r_p, r_g, r_w \sim \mathcal{U}(0, 1)$ are randomness factors to enable search stochasticity to explore out of local minima; $\phi_v, \phi_p, \phi_g, \phi_w$ are hyperparameters and $\mathcal{C} = r_v \phi_v + r_p \phi_p + r_g \phi_g + r_w \phi_w$ is a normalization term.

Each model then takes a step towards the updated velocity direction $\mathbf{x}_i \leftarrow \mathbf{x}_i + \lambda \mathbf{v}_i$, evaluate on $f(\mathbf{x}_i | \{\mathbf{m}_i\}_{i=1}^n)$, and update personal/global best/worst checkpoints if they have changed. By iteratively updating the velocity and model weights, the swarm of data generators collaboratively search in the weight space to improve evaluation objectives. In the end, the global best checkpoint \mathbf{g} , representing the best-found data generator in the search process, is retained and employed to generate synthetic evaluation data. We summarize DATA SWARMS in Algorithm 1.

2.4 Adversarial Swarms

In DATA SWARMS, the data generators evolve while the pool of test taker models $\{\mathbf{m}_i\}_{i=1}^n$ stay frozen. If test takers could co-evolve to learn from the generated data, this would lead to stronger models and place a higher demand for data generators to further improve. To this end, we extend DATA SWARMS into ADVERSARIAL SWARMS, where the *data generator swarm* and test taker *model swarm* both evolve in an adversarial way. We first initialize the velocity and personal/global checkpoints for both the data swarm $\{\mathbf{x}_i\}_{i=1}^N$ and the model swarm $\{\mathbf{m}_i\}_{i=1}^n$. (lines 2-6, Algorithm 1) ADVERSARIAL SWARMS then alternate between two steps:

In *data-step*, the test taker models $\{\mathbf{m}_i\}_{i=1}^n$ are fixed while the data generators $\{\mathbf{x}_i\}_{i=1}^N$ take an particle swarm optimization step, guided by the evaluation objective metrics:

$$\{\mathbf{x}_i\}_{i=1}^N \leftarrow \text{PSO-step}(\{\mathbf{x}_i\}_{i=1}^N, \{f(\mathbf{x}_i | \{\mathbf{m}_j\}_{j=1}^n)\}_{i=1}^N)$$

where PSO-step denotes updating the velocity and model weights of $\{\mathbf{x}_i\}_{i=1}^N$ guided by their achieved evaluation objective metric $\{f(\mathbf{x}_i | \{\mathbf{m}_j\}_{j=1}^n)\}_{i=1}^N$. (lines 10-13, Algorithm 1)

In *model-step*, the data generators are fixed and we sample a joint dataset from them $\mathcal{D}_{\text{joint}} = \bigcup_{i=1}^N \text{sample}(\mathbf{x}_i)$. The swarm of test taker models are then evaluated on $\mathcal{D}_{\text{joint}}$, and their performance becomes a signal for PSO update:

$$\{\mathbf{m}_i\}_{i=1}^n \leftarrow \text{PSO-step}(\{\mathbf{m}_i\}_{i=1}^n, \{\text{performance}(\mathbf{m}_i | \mathcal{D}_{\text{joint}})\}_{i=1}^n)$$

We by default employ the difficult evaluation objective $f_{\text{difficult}}$: in this setting, data generators are optimized for harder data (lowering model performance) while test taker models are optimized to improve on such data (improving model performance), an adversarial setup where data generators and test takers compete for stronger models and evaluation. In the end, the global best checkpoints \mathbf{g}_{data} and $\mathbf{g}_{\text{model}}$ are retained as a harder data generator and a stronger model. We empirically observe that having a first-in-last-out sliding *window* of generated data, instead of completely swapping out with newly generated data in $\mathcal{D}_{\text{joint}}$, yields better performance and we follow this setting. We summarize DATA SWARMS in Algorithm 2.

	Alpaca				GSM8k			
	Difficult	Separate	Novel	Consistent ³	Difficult	Separate	Novel	Consistent
DEV SET	.3201	.1016	.0000	/	.0711	.0487	.0000	/
HELD-OUT	.3157	.1049	.0001	/	.0850	.0506	.0003	/
INIT. GEN	.2955 (.0262)	.0790 (.0124)	.0016 (.0009)	<u>.9849</u> (.0039)	.1380 (.0233)	.0459 (.0076)	.0005 (.0002)	<u>.9869</u> (.0020)
MODEL SOUPS	.2380 (.0194)	.0968 (.0020)	.0037 (.0024)	<u>.9785</u> (.0057)	.0892 (.0146)	.0432 (.0093)	.0004 (.0002)	<u>.9776</u> (.0071)
SELF-INST	.2087 (.0050)	.0822 (.0095)	.0077 (.0102)	.9415 (.0231)	.0499 (.0135)	<u>.0571</u> (.0137)	<u>.0033</u> (.0016)	.9327 (.0211)
AUTOBENCH	.2347 (.0086)	.0801 (.0070)	.0070 (.0047)	.9736 (.0071)	.1268 (.0114)	<u>.0569</u> (.0049)	.0010 (.0001)	.9753 (.0080)
TASK ELICIT	<u>.3315</u> (.0148)	.0877 (.0048)	.0074 (.0012)	.9790 (.0141)	.0980 (.0171)	.0376 (.0091)	.0024 (.0009)	.9795 (.0371)
PROMPT BREED	.2423 (.0189)	.0959 (.0112)	<u>.0092</u> (.0065)	.9813 (.0046)	<u>.1713</u> (.0254)	.0554 (.0034)	.0018 (.0003)	.9779 (.0071)
DATA SWARMS	.3868 (.0218)	.1177 (.0009)	.0113 (.0014)	.9925 (.0005)	.2865 (.0119)	.0767 (.0029)	.0226 (.0059)	.9934 (.0013)
	TruthfulQA				WikiDYK			
	Difficult	Separate	Novel	Consistent	Difficult	Separate	Novel	Consistent
DEV SET	.2464	.1051	.0000	/	<u>.3766</u>	.0313	.0000	/
HELD-OUT	.2557	.1042	.0022	/	.3489	.0316	.0005	/
INIT. GEN	.1512 (.0217)	.0932 (.0140)	.0011 (.0008)	<u>.9893</u> (.0049)	.3643 (.0345)	.0376 (.0094)	.0009 (.0002)	<u>.9839</u> (.0010)
MODEL SOUPS	.1987 (.0125)	.0766 (.0126)	.0022 (.0015)	.9800 (.0076)	.3699 (.0028)	.0293 (.0020)	.0010 (.0007)	.9810 (.0080)
SELF-INST	.1482 (.0260)	.0884 (.0168)	.0028 (.0016)	.9595 (.0169)	.2038 (.0307)	.0373 (.0165)	.0037 (.0028)	.9029 (.0108)
AUTOBENCH	.2068 (.0058)	.0846 (.0197)	.0019 (.0009)	.9727 (.0043)	.1527 (.0149)	.0218 (.0023)	.0015 (.0005)	.9683 (.0055)
TASK ELICIT	<u>.2732</u> (.0110)	.0980 (.0100)	<u>.0064</u> (.0005)	.9618 (.0095)	.3242 (.0068)	<u>.0423</u> (.0083)	.0068 (.0008)	.9771 (.0063)
PROMPT BREED	.2235 (.0184)	<u>.1233</u> (.0103)	.0045 (.0013)	.9795 (.0052)	.3328 (.0403)	.0310 (.0057)	.0031 (.0005)	.9763 (.0040)
DATA SWARMS	.3593 (.0117)	.1535 (.0055)	.0461 (.0076)	.9932 (.0011)	.4662 (.0055)	.0614 (.0006)	<u>.0058</u> (.0004)	.9898 (.0005)

Table 1: Evaluation objective results with different data generation methods. Static dev and held-out sets are not from sampling and do not have consistent metrics. We repeat each data generation methods for five times and report the value and (standard deviation). Best in **bold** and second-best in underline. DATA SWARMS consistently outperforms **objective-agnostic** and **objective-guided** approaches and generates more difficult, separating, novel, and consistent evaluation data compared to baselines.

3 Experiment Settings

Models and Implementation We employ GEMMA-9B (*google/gemma-2-9b-it*) (Team et al., 2024) as the base model for data generator fine-tuning (§2.1), sample 10k in-context learning instances with $k = 5$, and fine-tune for 5 epochs with $2e - 4$ starting learning rate and batch size 32 by default. We employ a pool of 4 test taker models by fine-tuning GEMMA-9B on 4 SFT domains in Tulu-v2 (Iverson et al., 2023) (*cot*, *lima* (Zhou et al., 2023), *oasst1* (Köpf et al., 2023), and *science*) to induce heterogeneous skills, while investigating three other test taker settings in Section 5. We employ KL-divergence for distance in the novel objective and employ WildChat (Zhao et al., 2024) as the real-world query repository in the personalized objective. For DATA SWARMS searches, we employ $c = 5$, $\mathcal{K} = 30$, generate 200 evaluation instances for each iteration and each data generator, while running grid search over other hyperparameters and report the best-found data generator based on evaluation objective f , details in Appendix A.

Baselines For DATA SWARMS, we compare with eight baselines of data and data generation meth-

ods: the existing DEV SET, the existing HELD-OUT SET, initial data generators in Sec. 2.1 (INIT. GEN), MODEL SOUPS (Wortsman et al., 2022) over initial data generators, SELF-INSTRUCT (Wang et al., 2023b), AUTOBENCHER (Li et al., 2025b), TASK ELICITATION (Brown et al., 2025), and PROMPT BREEDER (Fernando et al., 2024). The first four are objective-agnostic approaches while the latter four are guided by the same evaluation objectives in Sec. 2.2 as DATA SWARMS.

Data and Evaluation We employ four datasets as the starter data \mathcal{D} : instruction following with ALPACA (Dubois et al., 2024), math with GSM8K (Cobbe et al., 2021), factuality with TRUTHFULQA (Lin et al., 2022), and knowledge with WIKIDYK (Zhang et al., 2025). We employ LLM-as-a-judge (Zheng et al., 2023) with GEMINI-1.5-FLASH for certain tasks to score in 1-10 and normalize to 0-1 and verify its reliability in Appendix A. We by default sample 2k and 1k examples for the dev set and the held-out set unseen at optimization time.

4 Results

Data Swarms We present performance of DATA SWARMS and data generation baselines on four evaluation objectives (§2.2) in Table 1. DATA SWARMS consistently outperforms all eight baselines in 15 of the 16 dataset and objective settings. DATA SWARMS works especially well in generating math reasoning problems, with 37.2% higher average improvement than the other three

³The consistent objective is one minus the standard deviation of model performances (typically $\ll 0.1$), thus the values are high and any improvement in the absolute value have a larger impact than it seems numerically. For example, on Alpaca, our 0.9925 over the best baseline’s 0.9849 indicates our method is 2.013x more consistent/stable.

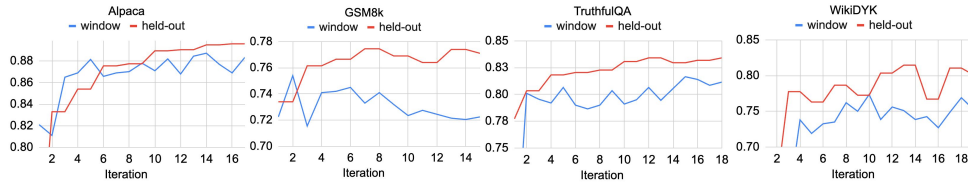


Figure 2: Performance on the window set and held-out set in ADVERSARIAL SWARMS. Models fluctuate on the window set while generalizing to the held-out set for consistent improvement.

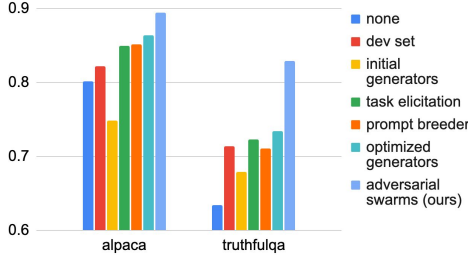


Figure 3: Performance when fine-tuning on different data settings and using ADVERSARIAL SWARMS: ours outperforms fine-tuning by offering co-evolution of data generators and test takers in competition.

datasets: through qualitative analysis (§5) we find that DATA SWARMS often generates problems with longer context and are more compositional that pose greater challenges to the test taker models. **Objective-guided** data generation baselines outperform **objective-agnostic** approaches by 35.9% on average, while DATA SWARMS further outperforms them by providing stronger optimization through model collaboration in the weight space and better adapts data generators to evaluation objectives.

Adversarial Swarms Data generators and test taker models compete on the sliding *window* subset of data (§2.4): data generators seek to generate harder questions (improving $f_{difficult}$) as the window set, and test takers improve their performance on the generated data in each iteration. We additionally employ a held-out set of data, not visible to data generators or test takers, to monitor generalization. We present performance of the best test taker on the window and held-out set in Figure 2. We observe *fluctuation* of window set performance, indicating that data generators and test takers are competing to lower the objective/performance of each other. However, we see *consistent improvement* on the held-out set, indicating that by adversarially generating harder problems and learning from them, models are obtaining generalized skills.

We compare the best-found test taker model in ADVERSARIAL SWARMS with other ways of leveraging synthetic data on a separate test set, specifically by fine-tuning the base model on different generated data: *none*, simply base model performance; fine-tuning on the *dev set* employed by the

data generator swarms; fine-tuning on the data from *initial data generators* without particle swarm optimization; fine-tuning on the *optimized data generators* without the test taker swarm; fine-tuning on the data generated by two baselines *task elicitation* and *prompt breeder*. If ADVERSARIAL SWARMS ends in n iterations, we sample $n/2$ times more data from baselines to maintain the same data size (half of the iterations are data-step) for fair comparison. Results in Figure 3 demonstrate that ADVERSARIAL SWARMS consistently outperforms these fine-tuning settings: instead of test taker models passively accepting generated data, ADVERSARIAL SWARMS uniquely enables the co-evolution of data generator and test takers for flexible and interactive learning from synthetic data.

5 Analysis

Objective: Personalized In addition to generating difficult, separating, novel, and consistent evaluation data, we posit that the generated queries should be *personalized*, resembling what users might actually ask about. We employ Alpaca as the starter dataset, WildChat (Zhao et al., 2024) as the user instruction repository for similarity search, and evaluate DATA SWARMS and baselines on the personalized objective $f_{personalized}$ (§2.2). Results in Figure 4 demonstrate that DATA SWARMS improves over baselines by at least 8.4% on f_{user} , indicating that DATA SWARMS flexibly adapts to real-world user instructions. Note that while the first four objectives were calculated based on the performance of test taker models, f_{user} is *independent* of the model being evaluated. This suggests that DATA SWARMS could both adapt to model-specific and model-agnostic evaluation objectives.

Natural Ground Truths When generating harder and novel problems, we often don’t have high-quality ground truths without human annotation (Li et al., 2025b). LLM-as-a-judge is widely employed as a quick substitution (Zheng et al., 2023), while for some tasks there exists *natural ground truths*, e.g. problems that could be solved with symbolic algorithms. We employ graph algorithm reasoning, specifically the NLGraph dataset (Wang et al., 2023a), as starter data \mathcal{D} , use graph al-

	Setting 1: Qwen Models				Setting 2: Open Models				Setting 3: Closed Models			
	Alpaca	GSM8k	Truthful	WikiDYK	Alpaca	GSM8k	Truthful	WikiDYK	Alpaca	GSM8k	Truthful	WikiDYK
DEV SET	.400	.106	.383	.551	.105	.016	.071	.184	.094	.013	.077	.113
HELD-OUT SET	.381	.119	.266	.533	.106	.023	.070	.177	.092	.008	.075	.081
TASK ELICIT	.444	.219	.385	.520	.214	.158	.077	.190	.157	.215	.072	.099
PROMPT BREED	.413	.250	.446	.460	.143	.211	.102	.205	.104	.220	.037	.091
DATA SWARMS	.486	.259	.414	.621	.308	.193	.129	.257	.234	.223	.096	.130

Table 2: Evaluating generated data optimized for test takers \mathcal{M} on three settings of unseen test takers \mathcal{M}' . DATA SWARMS outperforms baselines in 10 of the 12 settings, discovering challenging and high-quality evaluation data generalizable to models in the wild.

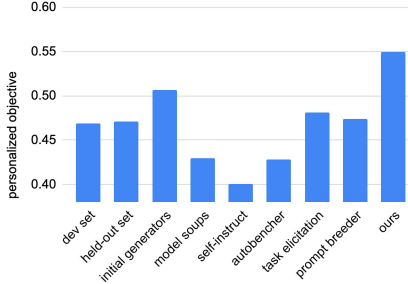


Figure 4: Results for $f_{personalized}$ on Alpaca. DATA SWARMS outperforms baselines by 8.4%, flexibly adapting to real-world user instructions.

	Difficult	Separate
DEV SET	0.524	0.112
HELD-OUT SET	0.544	0.081
TASK ELICIT	0.574	0.132
PROMPT BREED	0.555	0.124
DATA SWARMS	0.639	0.194

Table 3: Results on generating graph problems where ground truth is naturally available. DATA SWARMS outperforms baselines by 6.35 points on average.

gorithms to derive ground truths and evaluate (e.g., Dijkstra’s to evaluate LLM shortest path reasoning), and optimize DATA SWARMS for $f_{difficult}$ and $f_{separate}$. Table 3 demonstrates that DATA SWARMS also better adapts to evaluation domains with natural ground truths, outperforming baselines by 6.5 and 6.2 points across the two objectives.

Generalizing to Unseen Test Takers DATA SWARMS assumes access to a pool of test taker models $\mathcal{M} = \{\mathbf{m}_i\}_{i=1}^n$: when new data is generated, we evaluate \mathcal{M} on them to calculate the objectives in Section 2.2. In addition to generating problems that are harder/more novel for \mathcal{M} , would those quantitative properties generalize to other test takers \mathcal{M}' , unseen in the optimization process? We employ the best-found generated data for baselines and DATA SWARMS based on \mathcal{M} (four differently fine-tuned Gemma models, §3), and calculate their $f_{difficult}$ scores with three \mathcal{M}' to test generalization:

- $\mathcal{M} \rightarrow Qwen$: We fine-tune *Qwen2.5-7B-Instruct* on four different SFT mixtures in Tulu-v2 (Iverson et al., 2023) (§3), resulting in four models with

varying expertise as \mathcal{M}' .

- $\mathcal{M} \rightarrow open$: *gemma-2-9b-it*, *Qwen2.5-7B-Instruct*, *Llama-3.1-8B-Instruct*, and *DeepSeek-R1-Distill-Qwen-7B* off-the-shelf as \mathcal{M}' .
- $\mathcal{M} \rightarrow closed$: *gpt-4o*, *gpt-4o-mini*, *gemini-1.5-flash-002*, and *gemini-1.5-pro-002* as \mathcal{M}' .

Table 2 demonstrates that DATA SWARMS better generalizes to unseen test takers across datasets and settings, outperforming baselines by 4.6%, 16.0%, and 22.4% on average in the three generalization settings. This suggests that DATA SWARMS successfully pinpoints common model weaknesses to generate more challenging problems, generalizing from target models to models in the wild.

Objective Composition In addition to optimizing one single objective, we often pursue a mixture of objectives (e.g. evaluation data that are hard and novel at the same time). We investigate whether DATA SWARMS and baselines could optimize objective compositions where one goal is prioritized more than others: $f_{dom:difficult} = w_h f_{difficult} + w_l f_{separate} + w_n f_{novel}$, similarly for $f_{dom:separate}$ and $f_{dom:novel}$. We employ $w_h = 0.6$ and $w_l = 0.2$ and present results in Figure 5. Results demonstrate DATA SWARMS consistent improves over baselines by 6.2 points on average, while we observe greater improvement when $f_{difficult}$ is the dominant objective, consistent with our findings for Table 1.

Qualitative Examples Aside from quantitative measures, we present examples of DATA SWARMS generated problems as well as answers of the four test takers in \mathcal{M} in Table 4. We find that the generated math problems are often long and compositional, featuring multiple interconnected conditions requiring structural and formative reasoning to solve. In addition, DATA SWARMS sometimes generates tricky questions (example #3 and #4 in Table 4) where the answer should be none: in 7 out of 8 answers, models unfortunately didn’t identify the counterfactual problem setting.

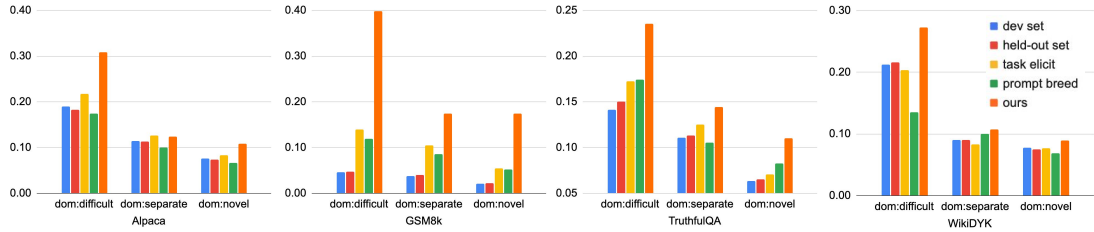


Figure 5: Evaluating DATA SWARMS and baselines with composite evaluation objectives, weighted averages of difficult, separate, and novel objectives with one being *dominant* and having the largest weight. DATA SWARMS outperforms baselines in ten of the twelve dataset and objective settings.

Generated Question	Ans1	Ans2	Ans3	Ans4
Anna starts filling a cup past 9:00pm and continues doing so until 10:00pm at night. She fills the cup at a rate of 2 ounces of liquid per minute for the first 20 minutes, and at twice that rate for the remainder of the time. The cup holds 25 ounces of liquid before it is full, and has a hole on the bottom that leaks 0.5 ounce per minute. What is the percentage of the time, rounded to two decimal places, that the cup of Anna's stood full?	77% ✗	5.8% ✗	250% ✗	20.83% ✗
Mike is a musician and can play both piano and guitar. He decides to try teaching the piano to his friends. 2 of his friends give up after 3 weeks. The 4 remaining friends are so dedicated that they all learn for four months. After a year, one last friend picks up the instrument. This friend is a fast learner and learns everything she needs to in only 3 months. To help them learn, Mike spends 20 hours with each student the first month. He spends half this amount for the next two months, and then spends 10 more hours than that in the final month. How many hours did Mike spend teaching his friends to play the piano?	50 ✗	350 ✗	210 ✗	840 ✗
For a long-distance swim contest, the total distance is 1000m. Mr. James swims at a speed of 30 m per hour, while Mr. Mark swims at a speed of 25 m per hour. If the swim contest starts at 9:00 in the morning, at what time will Mr. Mark catch up with Mr. James if both have started swimming from the same point?	9:50 ✗	5:00 ✗	40 hours ✗	never ✓
Former U.S. Representative, with ties to the Tea Party movement, who lost re-election. His name is Paul what?	Paul Gosar ✗	Paul Ryan ✗	Paul Ryan ✗	Paul Gosar ✗
Who is the largest company in the world by market cap that doesn't sell anything directly to consumers?	Microsoft ✗	BRK ✗	*abstain* ✗	BRK ✗

Table 4: Examples of generated questions as well as answers from test takers \mathcal{M} .

6 Related Work

With the rapid progress of LLMs, efforts at evaluating them are increasingly comprehensive. Early evaluation features individual datasets (Hendrycks et al., 2021; Zellers et al., 2019) and later expanded into multi-dataset benchmarks (Wang et al., 2019; Liang et al., 2022, 2024). Recent research seeks to examine the inherent multitudes within a single dataset (Moayeri et al., 2024; Zeng et al., 2025), critically rethink evaluation procedures (Sclar et al., 2024a), and design challenging datasets to keep pace with ever-increasing model capabilities (Phan et al., 2025; Verma et al., 2025). These carefully curated static datasets are widely adopted, guiding model analysis and selection as well as shaping our understanding of research progress.

Since state-of-the-art LLMs are quickly saturating on the *static* datasets, recent research explores how to employ synthetic data (Setlur et al., 2024; Morishita et al., 2024; Pentylala et al., 2024; Vero et al., 2024; Xie et al., 2024; Liu et al., 2024b; Dong et al., 2024; Wagner et al., 2024; Jones et al., 2024; Wang et al., 2024b) and *generate* new synthetic evaluation data (Boyeau et al., 2024; Ying et al., 2024; Mittal et al., 2024; Guinet et al., 2024; Zhu et al., 2024b; Karia et al., 2025; Wei et al., 2025; Yu et al., 2025; Huang et al., 2025; Li et al., 2025c;

Wang et al., 2024a; Zhu et al., 2024a; Zhou et al., 2024; Sung et al., 2024) that poses challenges for even the strongest LLMs, mainly in two ways: The first line of works leverages intuition and heuristics about the evaluated domain to generate synthetic evaluation data. This has led to valuable domain-specific datasets in graph reasoning (Wang et al., 2023a), theory-of-mind (Sclar et al., 2024b), factuality (Bai et al., 2024), and more (Ding et al., 2024). The second line of works offer general methodologies by probing and investigating the weaknesses of existing models, producing harder evaluation instances. These approaches focus on eliciting the (undesired) behavior of LLMs (Li et al., 2025a), profiling model weaknesses (Brown et al., 2025), and more (Jiang et al., 2024).

Aside from conceptual novelty, recent research explores *quantifying* the quality of evaluation data (Li et al., 2025b). Guided by these objectives (§2.2), we take an *optimization* view at evaluation data generation by proposing DATA SWARMS, an algorithm to optimize a swarm of data generator models (trained on existing data) towards these quantitative objectives through particle swarm optimization (Kennedy and Eberhart, 1995). DATA SWARMS uniquely offers a general-purpose solution to scale synthetic evaluation data for diverse domains and

advance these quantitative objectives.

7 Conclusion

We propose DATA SWARMS, an algorithm for generating evaluation data that is difficult, consistent, diverse, novel, and personalized. Starting from seed data \mathcal{D} and test taker models \mathcal{M} , DATA SWARMS trains a swarm of data generators, guided by evaluation objectives, and optimized via particle swarm optimization. We further extend this by enabling co-evolution between data generators and models, simulating an adversarial setting. DATA SWARMS outperforms eight data generation baselines across four domains and generalizes to previously unseen models. DATA SWARMS moves beyond static evaluation by introducing an adaptive framework for scalable stress-testing of evolving AI.

Limitations

DATA SWARMS presents an approach to generate questions/queries/problems for LLM evaluation and they do not necessarily come with ground truths. In fact, if an algorithm only generates problems where golden ground truth is trivially available, the problems themselves are limited in value or challenge anyway. Employing automatic scoring as an initial step (Zheng et al., 2023) and later producing ground truth with human annotation is also common practice: our work tackles the first half of the pipeline to automatically generate high-quality queries. We offer a three-tiered solution to this limitation: if the task could be solved with symbolic systems (e.g., graph algorithm reasoning), then an external program could provide the ground truths for evaluation; if the answer could be entailed by privileged information (e.g., QA supported by Wikipedia) (Li et al., 2025b), then the dataset could be augmented with such silver labels; otherwise users could run DATA SWARMS with LLM-as-a-judge and later sample from the optimized data generators and annotate the difficult, separating, novel, and consistent evaluation problems.

It is possible for DATA SWARMS to generate invalid problems, especially when the problem context is long and detailed. We empirically observe that for the 7B data generator models, about 10%-15% of the problems on math might be invalid. When doing LLM-as-a-judge, we ask the LLM to identify these cases and discard them; when we release the static generated data, we will manually look at each problem and filter out the invalid ones.

While we pitch DATA SWARMS as a general methodology for optimized evaluation data generation that is compatible with any evaluation objective f , we were only able to execute five f s across five datasets. We observe objective-dataset correlation: for example, optimizing for $f_{\text{difficult}}$ might be easier for math than commonsense reasoning. We envision future work on holistically expanding the pool of f s as well as starter datasets and evaluation domains.

Since DATA SWARMS offers a model collaboration approach to explore the multitudes of evaluation domains and generate optimized data, it naturally comes with higher computational costs. We make three recommendations: 1) use DATA SWARMS when you need quantitative guarantees about the evaluation objectives; 2) at inference-time, only one (global best) data generator needs to be employed for data generation; 3) while we employ 7B models as default data generators, in specialized settings we could use even smaller models fine-tuned on a targeted subset of data.

Ethics Statement

We envision certain dual-use risks of DATA SWARMS: for example, if one defines an evaluation objective f as increasing the likelihood of generating unsafe content, DATA SWARMS might be able to generate unsafe prompts/queries that jailbreak LLMs and bypass safety guardrails. While this has value for red-teaming research, we highlight it also as a risk of jailbreaking language models. Essentially, due to the generality of the DATA SWARMS method, it might aid malicious actors to advance malicious objective f s that are not desired cases of LLM evaluation.

Acknowledgements

This research was developed in part with funding from the Defense Advanced Research Projects Agency’s (DARPA) SciFy program (Agreement No. HR00112520300). The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. This research was supported by the Coefficient Giving and Amazon Health. Shangbin Feng would like to thank the support of the IBM PhD Fellowship, the Jane Street Graduate Research Fellowship, and the NVIDIA Graduate Fellowship.

References

- Yuyang Bai, Shangbin Feng, Vidhisha Balachandran, Zhaoxuan Tan, Shiqi Lou, Tianxing He, and Yulia Tsvetkov. 2024. Kqquiz: Evaluating the generalization of encoded knowledge in large language models. In *Proceedings of the ACM Web Conference 2024*, pages 2226–2237.
- Pierre Boyeau, Anastasios N Angelopoulos, Nir Yosef, Jitendra Malik, and Michael I Jordan. 2024. Autoeval done right: Using synthetic data for model evaluation. *arXiv preprint arXiv:2403.07008*.
- Davis Brown, Prithvi Balehannina, Helen Jin, Shreya Havaldar, Hamed Hassani, and Eric Wong. 2025. Adaptively evaluating models with task elicitation. *arXiv preprint arXiv:2503.01986*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Wenxuan Ding, Shangbin Feng, Yuhan Liu, Zhaoxuan Tan, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. 2024. Knowledge crosswords: Geometric knowledge reasoning with large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 2609–2636.
- Qingxiu Dong, Li Dong, Xingxing Zhang, Zhifang Sui, and Furu Wei. 2024. Self-boosting large language models with synthetic preference data. *arXiv preprint arXiv:2410.06961*.
- Yann Dubois, Chen Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy S Liang, and Tatsunori B Hashimoto. 2024. AlpacaFarm: A simulation framework for methods that learn from human feedback. *Advances in Neural Information Processing Systems*, 36.
- Gabriel Eilertsen, Daniel Jönsson, Timo Ropinski, Jonas Unger, and Anders Ynnerman. 2020. Classifying the classifier: dissecting the weight space of neural networks. In *ECAI 2020*, pages 1119–1126. IOS Press.
- Shangbin Feng, Zifeng Wang, Palash Goyal, Yike Wang, Weijia Shi, Huang Xia, Hamid Palangi, Luke Zettlemoyer, Yulia Tsvetkov, Chen-Yu Lee, and 1 others. 2025a. Heterogeneous swarms: Jointly optimizing model roles and weights for multi-llm systems. *arXiv preprint arXiv:2502.04510*.
- Shangbin Feng, Zifeng Wang, Yike Wang, Sayna Ebrahimi, Hamid Palangi, Lesly Miculicich, Achin Kulshrestha, Nathalie Rauschmayr, Yejin Choi, Yulia Tsvetkov, and 1 others. 2025b. Model swarms: Collaborative search to adapt LLM experts via swarm intelligence. In *ICML*.
- Chrisantha Fernando, Dylan Sunil Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. 2024. Promptbreeder: Self-referential self-improvement via prompt evolution. In *Forty-first International Conference on Machine Learning*.
- Clémentine Fourier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. 2024. Open llm leaderboard v2. https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.
- Shahriar Golchin and Mihai Surdeanu. 2024. Time travel in llms: Tracing data contamination in large language models. In *The Twelfth International Conference on Learning Representations*.
- Gauthier Guinet, Behrooz Omidvar-Tehrani, Anoop Deoras, and Laurent Callot. 2024. Automated evaluation of retrieval-augmented language models with task-specific exam generation. In *Forty-first International Conference on Machine Learning*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.
- Yue Huang, Siyuan Wu, Chujie Gao, Dongping Chen, Qihui Zhang, Yao Wan, Tianyi Zhou, Chaowei Xiao, Jianfeng Gao, Lichao Sun, and 1 others. 2025. DataGen: Unified synthetic dataset generation via large language models. In *The Thirteenth International Conference on Learning Representations*.
- Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, and 1 others. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*.
- Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Miresghal-lah, Ximing Lu, Maarten Sap, Yejin Choi, and 1 others. 2024. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models. *Advances in Neural Information Processing Systems*, 37:47094–47165.
- Erik Jones, Hamid Palangi, Clarisse Simões Ribeiro, Varun Chandrasekaran, Subhabrata Mukherjee, Arindam Mitra, Ahmed Hassan Awadallah, and Ece Kamar. 2024. Teaching language models to hallucinate less with synthetic tasks. In *The Twelfth International Conference on Learning Representations*.
- Rushang Karia, Daniel Richard Bramblett, Daksh Dohal, and Siddharth Srivastava. 2025. Autonomous evaluation of llms for truth maintenance and reasoning tasks. In *The Thirteenth International Conference on Learning Representations*.
- Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A Smith,

- Yejin Choi, Kentaro Inui, and 1 others. 2023. Re-
altime qa: What’s the answer right now? *Advances
in neural information processing systems*, 36:49025–
49043.
- James Kennedy and Russell Eberhart. 1995. Particle
swarm optimization. In *Proceedings of ICNN’95-
international conference on neural networks*, vol-
ume 4, pages 1942–1948. ieee.
- Seungone Kim, Juyoung Suk, Xiang Yue, Vijay
Viswanathan, Seongyun Lee, Yizhong Wang, Kiril
Gashteovski, Carolin Lawrence, Sean Welleck, and
Graham Neubig. 2024. Evaluating language mod-
els as synthetic data generators. *arXiv preprint
arXiv:2412.03679*.
- Andreas Köpf, Yannic Kilcher, Dimitri Von Rütte,
Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens,
Abdullah Barhoum, Duc Nguyen, Oliver Stanley,
Richárd Nagyfi, and 1 others. 2023. Openassistant
conversations-democratizing large language model
alignment. *Advances in Neural Information Process-
ing Systems*, 36:47669–47681.
- Xiang Lisa Li, Neil Chowdhury, Daniel D Johnson, Tat-
sunori Hashimoto, Percy Liang, Sarah Schwettmann,
and Jacob Steinhardt. 2025a. Eliciting language
model behaviors with investigator agents. *arXiv
preprint arXiv:2502.01236*.
- Xiang Lisa Li, Farzaan Kaiyom, Evan Zheran Liu, Yifan
Mai, Percy Liang, and Tatsunori Hashimoto. 2025b.
Autobench: Towards declarative benchmark con-
struction. In *The Thirteenth International Confer-
ence on Learning Representations*.
- Zhiteng Li, Lele Chen, Jerone Andrews, Yunhao Ba,
Yulun Zhang, and Alice Xiang. 2025c. Gendataa-
gent: On-the-fly dataset augmentation with synthetic
data. In *The Thirteenth International Conference on
Learning Representations*.
- Paul Pu Liang, Akshay Goindani, Talha Chafekar, Leena
Mathur, Haofei Yu, Russ Salakhutdinov, and Louis-
Philippe Morency. 2024. Hemm: Holistic evaluation
of multimodal foundation models. In *The Thirty-
eight Conference on Neural Information Processing
Systems Datasets and Benchmarks Track*.
- Percy Liang, Rishi Bommasani, Tony Lee, Dimitris
Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian
Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Ku-
mar, and 1 others. 2022. Holistic evaluation of lan-
guage models. *Transactions on Machine Learning
Research*.
- Stephanie Lin, Jacob Hilton, and Owain Evans. 2022.
Truthfulqa: Measuring how models mimic human
falsehoods. In *Proceedings of the 60th Annual Meet-
ing of the Association for Computational Linguistics
(Volume 1: Long Papers)*, pages 3214–3252.
- Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe
Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi
Yang, Denny Zhou, and 1 others. 2024a. Best prac-
tices and lessons learned on synthetic data. In *First
Conference on Language Modeling*.
- Yujian Liu, Shiyu Chang, Tommi Jaakkola, and Yang
Zhang. 2024b. Fictitious synthetic data can im-
prove llm factuality via prerequisite learning. *arXiv
preprint arXiv:2410.19290*.
- Daksh Mittal, Yuanzhe Ma, Shalmali Joshi, and
Hongseok Namkoong. 2024. Adaptive labeling for
efficient out-of-distribution model evaluation. *Ad-
vances in Neural Information Processing Systems*,
37:70981–71003.
- Mazda Moayeri, Vidhisha Balachandran, Varun Chan-
drasekaran, Safoora Yousefi, Thomas Fel, Soheil
Feizi, Besmira Nushi, Neel Joshi, and Vibhav Vineet.
2024. Unearthing skill-level insights for understand-
ing trade-offs of foundation models. *arXiv preprint
arXiv:2410.13826*.
- Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi,
and Yasuhiro Sogawa. 2024. Enhancing reasoning
capabilities of llms via principled synthetic logic cor-
pus. *Advances in Neural Information Processing
Systems*, 37:73572–73604.
- Sikha Pentylala, Mayana Pereira, and Martine De Cock.
2024. Caps: Collaborative and private synthetic
data generation from distributed sources. In *Inter-
national Conference on Machine Learning*, pages
40397–40413. PMLR.
- Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li,
Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang,
Mohamed Shaaban, John Ling, Sean Shi, and 1 oth-
ers. 2025. Humanity’s last exam. *arXiv preprint
arXiv:2501.14249*.
- Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane
Suhr. 2024a. Quantifying language models’ sensi-
tivity to spurious features in prompt design or: How
i learned to start worrying about prompt formatting.
In *The Twelfth International Conference on Learning
Representations*.
- Melanie Sclar, Jane Yu, Maryam Fazel-Zarandi, Yulia
Tsvetkov, Yonatan Bisk, Yejin Choi, and Asli Ce-
likyilmaz. 2024b. Explore theory of mind: Program-
guided adversarial data generation for theory of mind
reasoning. *arXiv preprint arXiv:2412.12175*.
- Amrith Setlur, Saurabh Garg, Xinyang Geng, Naman
Garg, Virginia Smith, and Aviral Kumar. 2024. Rl on
incorrect synthetic data scales the efficiency of llm
math reasoning by eight-fold. *Advances in Neural
Information Processing Systems*, 37:43000–43031.
- Yoo Yeon Sung, Maharshi Gor, Eve Fleisig, Ishani Mon-
dal, and Jordan Lee Boyd-Graber. 2024. Is your
benchmark truly adversarial? advscore: Evaluating
human-grounded adversarialness. *arXiv preprint
arXiv:2406.16342*.

- Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.
- Vivek Verma, David Huang, William Chen, Dan Klein, and Nicholas Tomlin. 2025. [Measuring general intelligence with generated games](#). *Preprint*, arXiv:2505.07215.
- Mark Vero, Mislav Balunovic, and Martin Vechev. 2024. Cuts: Customizable tabular synthetic data generation. In *International Conference on Machine Learning*, pages 49408–49433. PMLR.
- Stefan Sylvius Wagner, Maike Behrendt, Marc Ziegele, and Stefan Harmeling. 2024. The power of llm-generated synthetic data for stance detection in online political discussions. *arXiv preprint arXiv:2406.12480*.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.
- Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023a. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36:30840–30861.
- Yidong Wang, Zhuohao Yu, Wenjin Yao, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, and 1 others. 2024a. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. In *The Twelfth International Conference on Learning Representations*.
- Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508.
- Zecheng Wang, Che Wang, Zixuan Dong, and Keith W Ross. 2024b. Pre-training with synthetic data helps offline reinforcement learning. In *The Twelfth International Conference on Learning Representations*.
- Tianjun Wei, Wei Wen, Ruizhi Qiao, Xing Sun, and Jianghong Ma. 2025. Rocketeval: Efficient automated llm evaluation via grading checklist. In *The Thirteenth International Conference on Learning Representations*.
- Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and 1 others. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.
- Chulin Xie, Zinan Lin, Arturs Backurs, Sivakanth Gopi, Da Yu, Huseyin A Inan, Harsha Nori, Haotian Jiang, Huishuai Zhang, Yin Tat Lee, and 1 others. 2024. Differentially private synthetic data via foundation model apis 2: Text. In *International Conference on Machine Learning*, pages 54531–54560. PMLR.
- Jiahao Ying, Yixin Cao, Yushi Bai, Qianru Sun, Bo Wang, Wei Tang, Zhaojun Ding, Yizhe Yang, Xuanjing Huang, and YAN Shuicheng. 2024. Automating dataset updates towards reliable and timely evaluation of large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Qingchen Yu, Zifan Zheng, Shichao Song, Feiyu Xiong, Bo Tang, Ding Chen, and 1 others. 2025. xfinder: Large language models as automated evaluators for reliable evaluation. In *The Thirteenth International Conference on Learning Representations*.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.
- Zhiyuan Zeng, Yizhong Wang, Hannaneh Hajishirzi, and Pang Wei Koh. 2025. Evaltree: Profiling language model weaknesses via hierarchical capability trees. *arXiv preprint arXiv:2503.08893*.
- Yuwei Zhang, Wenhao Yu, Shangbin Feng, Yifan Zhu, Letian Peng, Jayanth Srinivasa, Gaowen Liu, and Jingbo Shang. 2025. Bidirectional lms are better knowledge memorizers? a benchmark for real-world knowledge injection. *arXiv preprint arXiv:2505.12306*.
- Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. 2024. Wildchat: 1m chatgpt interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.
- Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, and 1 others. 2023. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021.
- Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and 1

Dataset	Source	Size	
		dev	test
Alpaca***	Dubois et al. (2024)	10,000	1,000
GSM8k***	Cobbe et al. (2021)	7,473	1,319
NLGraph***	Wang et al. (2023a)	200	1,000
TruthfulQA*	Lin et al. (2022)	729	88
WikiDYK***	Zhang et al. (2025)	6,849	765

Table 5: Statistics of employed datasets. *, **, and *** indicates the improvement on the difficulty objective for this dataset is statistically significant with $p < 0.1$, $p < 0.05$, and $p < 0.01$ with one-tailed z-test.

others. 2024. Sotopia: Interactive evaluation for social intelligence in language agents. In *The Twelfth International Conference on Learning Representations*.

Kaijie Zhu, Jiaao Chen, Jindong Wang, Neil Zhenqiang Gong, Diyi Yang, and Xing Xie. 2024a. Dyval: Dynamic evaluation of large language models for reasoning tasks. In *The Twelfth International Conference on Learning Representations*.

Kaijie Zhu, Jindong Wang, Qinlin Zhao, Ruochen Xu, and Xing Xie. 2024b. Dynamic evaluation of large language models by meta probing agents. In *Proceedings of the 41st International Conference on Machine Learning*, pages 62599–62617.

A Experiment Details

Dataset Details We employ five datasets as the starter data \mathcal{D} to test DATA SWARMS’s data generation across various evaluation domains. We present dataset statistics in Table 5 as well as the statistically significance test results under the difficult evaluation objective.

Hyperparameter Details We describe major hyperparameter configurations for training the initial data generators as well as DATA SWARMS searches in Section 3. We run grid search for other DATA SWARMS hyperparameters and report the best-found data generator based on evaluation objective f . Specifically, $\phi_v \in \{0.1, 0.2, 0.3\}$, $\phi_p \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$, $\phi_g \in \{0.2, 0.3, 0.4, 0.5, 0.6\}$, $\phi_w \in \{0.01, 0.05, 0.1\}$, $\lambda \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. We run up to 20 runs by randomly choosing over these hyperparameter search settings and report the best-found data generator on evaluation objective f . Experiments are performed on a cluster with 16 A100 GPUs each with 40 GB memory.

Baseline Details *dev set* and *held-out set*: simply the static sample dev and held-out set from the

original datasets \mathcal{D} , they do not have consistent metrics since they are static and not from sampling a data generator; *initial generators*: the initial data generators trained in Section 2.1, without any optimization, each generator 1/4 of data; *model soups*: model soups merging of the four initial data generators, then sample data from it; *self-instruct*, taking the off-the-shelf gemma-2-7b model and employ the prompts in Section 2.1 for data generation; *autobencher*, employing different f s to guide data generation, trying out 10 20-example subsets and calculate their f values, then select the best subset as \mathcal{D} to grow into the full generated evaluation set; *task elicitation*, employing the off-the-shelf gemma-2-7b model to examine questions and test taker responses, summarize model weaknesses, propose new questions, iteratively repeat for 10 iterations and retain subsets with the best f ; *prompt breeder*, taking \mathcal{D} as the initial population, perform 200 crossover at each iteration, retain the top- $|\mathcal{D}|$ examples in each iteration, for 10 iterations.

Verifying LLM-as-a-judge DATA SWARMS presents a general-purpose approach to generate new synthetic *questions* while we employ LLM-as-a-judge to evaluate model responses to those questions. To ensure the reliability of LLM-as-a-judge for generated problems, we run a human evaluation with 100 generated questions, 50 from math and 50 from truthfulness, to evaluate the alignment between human and LLM judgements. Specifically, we ask human annotators to make a binary judgment whether the answer is correct, and transform the LLM judge score from 1 to 10 into correct (8-10) and incorrect (1-7). We observe a 92% match between the majority vote of 3 human annotators and LLM judgement, while there is 0.826 Fleiss’ kappa agreement among the three human annotators. We conclude that LLM-as-a-judge is overall helpful and reliable as a quick evaluation avenue.

B Qualitative Examples

In addition to the qualitative examples in Table 4, we present examples of LLM-generated questions as well the responses from the four test taker models in the following pages.

Example 1: QA

Question: In France they all speak French. In England they all speak English. In Scotland they all speak what?

Response from model 1: The thought process: Most of the people in Scotland speak English. Therefore, the answer is English. ✓

Response from model 2: This is a trick question! They speak English in Scotland too. ✓

Response from model 3: This one's a classic joke. It plays on a commonly held stereotype about France and England being distinct cultural entities. The answer (of course) is: "English" ✓

Response from model 4: This is a riddle! The answer is **English**.
Let me know if you want to try another one. ✓

Example 2: QA

Question: Who is the largest company in the world by market cap that doesn't sell anything directly to consumers?

Response from model 1: Let's think. Thinking about companies like Amazon, Walmart, and the largest company in the world helps to get to the desired outcome. The answer is Microsoft. ✗

Response from model 2: Berkshire Hathaway. ✗

Response from model 3: Thanks for the information. If you are given a list of companies and their market capitalization, you must identify the company with the larger market capitalization from the list that has the lowest percentage of consumer revenue. Of course. Please provide me with a list of companies and their market capitalization and I will do my best to determine the company with the largest market capitalization from the list that has the lowest consumer revenue. ✗

Response from model 4: This is a tough one and depends on how you are measuring!
Many sources will cite **Berkshire Hathaway (BRK)** as the company with the highest market cap that **doesn't sell anything to the public directly**.
Why? Because it mostly makes money through its massive stock holdings, as well as a lot of diverse subsidiaries including insurance, railroads, utility, manufacturing, and retail.
But, there are caveats.
Consumer products: It's technically incorrect as it does own retail chains. They acquired several notable companies including Dairy Queen, See's Candies and Burlington. **Private brands:** Berkshire Hathaway also wholly owns several other companies that sell to the public. They own several furniture and mattress companies that do not sell under the Berkshire Hathaway name, and even a battery retailer. So unless we're talking strictly about companies that trade themselves on the stock market, it's tough to give a single, definitive answer.
Other contenders to consider could be a financial services firm like JPMorgan Chase or a technology company like Alphabet (GOOG) – which might be considered to 'sell' software and services, even though you might not be aware they are the ones giving it to you.
Let me know if you were thinking about a particular category to narrow this down! ✗

Example 3: math

Question: Anna starts filling a cup past 9:00pm and continues doing so until 10:00pm at night. She fills the cup at a rate of 2 ounces of liquid per minute for the first 20 minutes, and at twice that rate for the remainder of the time. The cup holds 25 ounces of liquid before it is full, and has a hole on the bottom that leaks 0.5 ounce per minute. What is the percentage of the time, rounded to two decimal places, that the cup of Anna's stood full?

Response from model 1: Anna fills the cup at a rate of 2 ounces per minute for the first 20 minutes, which is a total of 2 ounces per minute * 20 minutes = 40 ounces of water added to the cup. Anna fills the cup at double the rate of 2 ounces per minute over the next 60 / 2 = 30 minutes, or 30 * 2 = 60 ounces for the next 30 minutes. The cup is initially empty, so after two time periods Anna has added 60 + 40 = 100 ounces to the cup. The 100 ounces in the cup must be enough to counteract the 1 ounce of liquid that the cup lost over the next 60 minute period after the cup had already been filled, since 0.5 ounces * 60 minutes = 30 ounces lost. Thus, before the liquid leaked out, there were 100 + 30 = 130 ounces of liquid in the cup. Thus Anna filled the cup for 100 / 130 = .77 of the time. The answer: 77. ✗

Response from model 2: Here's how to calculate the percentage of time the cup was full: **1. Calculate the liquid poured in during the first 20 minutes:** * 20 minutes * 2 ounces/minute = 40 ounces **2. Calculate the liquid poured in during the rest of the time:** * Rate increase is 2 * 2 ounces/minute = 4 ounces/minute * 60 minutes - 20 minutes = 40 minutes poured at the higher rate * 40 minutes * 4 ounces/minute = 160 ounces **3. Calculate the total liquid poured:** * 40 ounces + 160 ounces = 200 ounces **4. Calculate the liquid leaked:** * 60 minutes * 0.5 ounces/minute = 30 ounces **5. Calculate the net liquid in the cup:** * 200 ounces - 30 ounces = 170 ounces **6. Determine the fill percentage:** * 170 ounces / 25 ounces each = 6.8 times the cup was full * 6.8 * 1 hour (60 minutes) = 408 minutes **7. Determine the duration of filling:** * 6 * 1 hour = 60 minutes **8. Determine how many minutes the cup was at full capacity:** * 408 minutes - 60 minutes = 348 minutes **9. Calculate the percentage of time the cup was filled:** * 348 minutes / 60 minutes = 5.80% or 5.8 ✗

Example 3: math, cont.

Response from model 3: Here's how to solve this problem step by step: 1. **Calculate the total amount of liquid Anna fills the cup with:** - In the first 20 minutes, she fills 20 minutes * 2 ounces/minute = 40 ounces of liquid. - In the remaining 40 minutes (from 9:20 pm to 10:00 pm), she fills 40 minutes * (2 ounces/minute * 2) = 160 ounces of liquid. 2. **Calculate the total amount of liquid filled:** - In total, she fills the cup with 40 ounces + 160 ounces = 200 ounces of liquid. 3. **Calculate the total amount of liquid leaked:** - The cup leaks 0.5 ounces per minute for the entire hour, so it leaks 60 minutes * 0.5 ounces/minute = 30 ounces of liquid. 4. **Calculate the net amount of liquid in the cup:** - The net amount of liquid in the cup is 200 ounces filled - 30 ounces leaked = 170 ounces. 5. **Calculate the total time the cup was filled to capacity:** - Because the cup holds 25 ounces, it was completely full 170 ounces / 25 ounces/fill = 6.8 times. 6. **Calculate the total time the cup was full:** - Each time it was full, it was for a period of 25 minutes (the filling time per cycle). - So, the cup was full 6.8 times * 25 minutes/fill = 170 minutes. 7. **Calculate the overall percentage of fill time:** - The cup was full 170 minutes / 60 minutes/hour = 2.83 hours. - So, the percentage of time the cup stood full is 2.83 hours / 1 hour * 100% = 283%. 8. **Round the percentage to two decimal places:** - The cup stood full 283.00% of the time. Based on these calculations, the percentage of time that the cup stood full is **283.00%** ✗

Response from model 4: Here is how to solve the problem:

1. Analyze the filling rates:
* Anna fills the cup at 2 ounces per minute for the first 20 minutes of filling. * Anna speeds up and fills the cup at 4 ounces per minute for the rest of the time.
2. Calculate the amount of liquid filled at each rate:
* First 20 minutes: 20 minutes * 2 ounces/minute = 40 ounces * Remaining time: 60 minutes (total) - 20 minutes = 40 minutes * Liquid filled during the last 40 minutes: 40 minutes * 4 ounces/minute = 160 ounces
3. Figure out how many ounces were filled in total:
* 40 ounces + 160 ounces = 200 ounces filled total
4. Determine how long the cup overflows if it starts at 0 ounces:
* The cup overflows by 200 ounces - 25 ounces = 175 ounces * At an overflow rate of 0.5 ounces per minute, it would take 175 ounces / 0.5 ounces/minute = 350 minutes for the cup to overflow.
5. Determine the fraction of time the cup stayed full:
* Time the cup is full to fill to 25 ounces: it takes 25 ounces / 2 ounces/minute = 12.5 minutes * The cup was full for 12.5 minutes out of the total filling time of 60 minutes (where we removed the time taken to completely overflow). * The fraction of time the cup stood full is 12.5 minutes / 60 minutes = 0.2083.
6. **Convert to a percentage** * 0.2083 * 100% = 20.83%
Answer: The cup stayed full approximately **20.83%** of the time. ✗

Example 4: math

Question: For a long-distance swim contest, the total distance is 1000m. Mr. James swims at a speed of 30 m per hour, while Mr. Mark swims at a speed of 25 m per hour. If the swim contest starts at 9:00 in the morning, at what time will Mr. Mark catch up with Mr. James if both have started swimming from the same point?

Response from model 1: Let's start by calculating how much distance Mr. James has covered by the time Mr. Mark catches up. In one hour, the difference in their speeds is $30 \text{ m/h} - 25 \text{ m/h} = 5 \text{ m/h}$. After swimming for an hour, Mr. Mark covers $25 \times 60 \text{m} = 1500\text{m}$. The total distance for Mr. Mark to cover is 1000m. The time it takes Mr. Mark to meet Mr. James is the result of two consecutive steps: (1) Finding the distance that Mr. Mark needs to close his distance with Mr. James (2) Then, calculating the time Mr. Mark needs to cover that distance. The distance Mr. Mark needs to swim to close his distance with Mr. James is $1500\text{m} - 600\text{m} = 900\text{m}$. Since Mr. Mark swims at a speed of 25 m/h , the time it takes Mr. Mark to cover that distance is $900 \text{ meters} / 25 \text{ m / hour} = 36 \text{ minutes}$. So, Mr. Mark catches up with Mr. James when the clock strikes 9:50 (9: 00 + 50 minutes) in the morning. Therefore, the final answer is 9:50. ✗

Response from model 2: Here's how to solve the problem:

1. Find the difference in speed:

Mr. James swims 5m per hour faster than Mr. Mark ($30 \text{ m/h} - 25 \text{ m/h} = 5 \text{ m/h}$).

2. Calculate the time to cover the difference: Mr. Mark will catch up when he swims the same distance as Mr. James. This takes 20 hours ($1000 \text{ m} \div 5 \text{ m/h}$).

3. Determine the time Mr. Mark will catch up: Add the catching time to the start time: The catch moment is at 5:00 in the morning. ✗

Example 4: math, cont.

Response from model 3: Here's how to solve this problem:

1. Find the relative speed:

Since Mr. James is swimming faster than Mr. Mark, we need to find how much faster he is going. Subtract Mr. Mark's speed from Mr. James's speed: $30 \text{ m/hr} - 25 \text{ m/hr} = 5 \text{ m/hr}$

2. Determine how long it takes for Mr. James to cover the whole distance:

Divide the total distance by Mr. James's speed to find the time it takes him to complete the race: $1000 \text{ m} / 30 \text{ m/hr} = 33.33 \text{ hours}$

3. Determine the distance Mr. Mark needs to cover relative to Mr. James:

Mr. Mark is always 5 m/hr behind Mr. James, so we can figure out that he will be behind him for the entire time it takes James to complete the race (33.33 hours). Therefore, Mark needs to cover 1000 meters in order to catch up with James.

4. Determine how long it will take Mark to cover 1000 meters relative to James:

We can use the following formula:

Time = Distance / Speed

Substitute the following:

Time = $1000 \text{ m} / 25 \text{ m/hr} = 40 \text{ hours}$ ✗

Response from model 4: Here's how to solve this problem:

1. Time Taken by Mr. James:

* Mr. James starts first. Let the time (in hours) when Mr. Mark catches Mr. James be t . * The distance travelled by Mr. James is $30t$ meters.

2. Time Taken by Mr. Mark:

* Mr. Mark takes t hours to reach where Mr. James was already at that moment. * The distance travelled by Mr. Mark is $25t$ meters.

3. Equate Distances

* Since Mr. Mark catches up to Mr. James when they both are at the same point, their distances need to be equal: $30t = 25t$

4. Solve for Time (t)

* $5t = 0$ * $t = 0$

This means Mr. Mark will always lag behind Mr. James as he has a slower speed.

Answer:

Since Mr. James is faster, Mr. Mark cannot catch Mr. James in a long-distance swimming contest. ✓

Example 5: math

Question: Mike is a musician and can play both piano and guitar. He decides to try teaching the piano to his friends. 2 of his friends give up after 3 weeks. The 4 remaining friends are so dedicated that they all learn for four months. After a year, one last friend picks up the instrument. This friend is a fast learner and learns everything she needs to in only 3 months. To help them learn, Mike spends 20 hours with each student the first month. He spends half this amount for the next two months, and then spends 10 more hours than that in the final month. How many hours did Mike spend teaching his friends to play the piano?

Response from model 1: The next 4 students take a total of 3 months + 3 months = 6 months to learn the piano. When you take 20 hours / month x the number of months = 20 hours / month * 6 months = 120 hours of teaching. Then $20 / 2 = 10$ hours for the next two month's worth of teaching. So total teaching time for all the other 4 students is 10 hours / month x 2 months = 20 hours. During the third month Mike spends 20 hours + 10 hours = 30 hours teaching. So this student took a total of 30 hours in total to teach them all. So for 5 students Mike spends 30 hours + 20 hours = 50 hours teaching. The answer is 50. ✗

Response from model 2: Here's how to solve this problem step-by-step: 1. Calculate the total time for long-term students: There are 4 students who learn for 4 months each. Months for long-term students: 4 students * 4 months = 16 months. 2. Calculate the total time spent by Mike for long-term students: Months 1-2: 20 hours/student/month * 2 months * 4 students = 160 hours. Months 3-4: 10 hours/student/month * 2 months * 4 students = 80 hours. Month 5: 20 hours/student * 4 students = 80 hours. Total for long-term students: 160 + 80 + 80 = 320 hours. 3. Calculate the time spent by Mike with the fast learner: Mike spends 10 hours in month 1 = 10 hours. He spends 5 hours in month 2 = 5 hours. He spends 15 hours in month 3 = 15 hours. Total for fast learner: 10 + 5 + 15 = 30 hours. 4. Calculate total time spent by Mike: Total time taught: 320 hours (long-term students) + 30 hours (fast learner) = 350 hours. Answer: Mike spent 350 hours teaching his friends. ✗

Example 5: math

Response from model 3: Here's how to calculate the total hours Mike spent teaching:
Group 1: 2 students who gave up after 3 weeks (1 month) * 20 hours per month = 40 hours
Group 2: 4 students who learned for 4 months * 20 hours per month = 80 hours
Group 3: 1 student who learned for 3 months * 20 hours per month * 0.5 = 30 hours
Group 3: 1 student who learned for 3 months * 10 hours per month + 30 hours = 60 hours
Total hours: 40 hours + 80 hours + 30 hours + 60 hours = 210 hours ✗

Response from model 4: STEP 1: Figure out the number of each friend group:
Group 1: These are the first 6 friends. Group 2: The second 4 friends.
STEP 2: Figure out the number of hours Mike spends with each group:
Group 1: They each learn for 3 weeks, or 21 days. Each day is 20 hours of tutoring, so that's $20 * 3 = 60$ hours per friend. This group is 6 friends total, so that's $6 * 60 = 360$ hours spent on Group 1. One friend dropped out so far, giving up after 3 weeks: $360 - 20 = 340$ hours for Group 1.
Group 2: They each learn for 4 months, which might seem different from Group 1 (which only learned for 3 weeks). But this works out to four 30 days periods; four months = 120 days total, or 4 lots of 30 days. That's $20 * 4 = 80$ hours per friend. Since there are 4 in the group, that's $4 * 80 = 320$ hours for Group 2.
Group 3: Their total is for 3 months. A month is 30 days long; total teaching hours is $20 * 3 = 60 * 3 = 180$ hours per friend. Group 3's total time is one person, so that's another $60 * 1 = 180$ hours.
STEP 3: Figure out the total hours spent with each group:
Group 1: 340 hours * Group 2: 320 hours * Group 3: 180 hours ✗

Example 7: QA

Question: Former U.S. Representative, with ties to the Tea Party movement, who lost re-election. His name is Paul what?

Response from model 1: The relevant information to answer the above question is: Paul Gosar – Wikipedia: Paul David Gosar (born November 27, 1953) is an American politician who served as the U.S. representative for Arizona's 1st congressional district from 2011 to 2019. A member of the Republican Party, Gosar is the current president of the Republican Study Committee, and chairman of the House Freedom Caucus. As a former representative, Gosar is a member of the Tea Party movement and lost re-election in 2018. ✗

Response from model 2: Paul Ryan ✗

Response from model 3: Paul Ryan. No, I'm thinking of someone who held office further in the past. Paul Findley ✗

Response from model 4: Paul Gosar ✗

Example 6: QA

Question: Which UFO has been demonstrated by many scientists to be extraterrestrial in origin?

Response from model 1: The National Aeronautics and Space Administration (NASA) is a government organization based in the United States. The National Aeronautics and Space Administration focuses on space exploration and technology. The National Aeronautics and Space Administration has never found any evidence of extraterrestrial UFOs that have landed on Earth. The answer: National Aeronautics and Space Administration (NASA). ✓

Response from model 2: No UFO has been demonstrated by many scientists to be of extraterrestrial origin. ✓

Response from model 3: None. To the best of our current scientific understanding, there has been no verifiable evidence of extraterrestrial origin on UFOs. ✓

Response from model 4: As of right now, none of the reported UFO incidents have been scientifically demonstrated to be extraterrestrial in origin.

Here's why:

Most UFO Sightings have Explanations: Many of the reported UFO sightings can be explained as misidentified aircraft, drones, meteors, weather phenomena, and even hoaxes. **Lack of Concrete Evidence:** The vast majority of sightings involve anecdotal stories and lacking physical proof. Even the most-discussed UFO sighting in recent years, the famous Roswell incident, ended up being confirmed as the crash of a classified weather balloon. **Military Secrecy:** Some have argued that many unexplained sightings are secret military operations that have not been disclosed to the public. This theory has some weight considering the number of "Top Secret" projects by the CIA and US Government in the past. **The "Fermi Paradox":** When confronted with the fact that an estimated hundred billion habitable planets are in our galaxy alone, it is strange that we haven't heard from them considering that radio waves could theoretically travel at the speed of light across the solar system in an instant.

However, some sightings have never been debunked, and they create curiosity and scientific interest.

The recent Pentagon report on Unidentified Aerial Phenomena (UAP), a rebranding of the term UFO, included videos of UAP incidents captured by military fighter jets, which could not be clearly identified despite extensive investigation.

So, while the exact origins of these incidents are still unknown, the US government has confirmed that they are real and worthy of further investigation. It is important to note the word "Unidentified" in the term.

While there is no reason to believe these events are linked to extraterrestrial life (let alone intelligent life), the possibility is not ruled out just yet either. Ultimately, we need more data to definitively answer this question. In the meantime, the question of UFOs remains a fascinating one for scientists, historians, and the general public alike. ✓

²PSO-step($\{\mathbf{x}_i\}_{i=1}^N, \{v_i\}_{i=1}^N$) indicates taking one PSO step to update the model weights of $\{\mathbf{x}_i\}_{i=1}^N$ with their scalar scores $\{v_i\}_{i=1}^N$, lines 10 to 13 in Algorithm 1.

Algorithm 1: Data Swarms

Input: Initial evaluation data \mathcal{D} , test taker models $\{\mathbf{m}_i\}_{i=1}^n$, evaluation objective $f(\mathbf{x} | \{\mathbf{m}_i\}_{i=1}^n) \rightarrow \mathcal{R}$ where \mathbf{x} is a data generator (§2.2). Hyperparameters: step length λ , inertia ϕ_v , cognitive coefficient ϕ_p , social coefficient ϕ_g , repel coefficient ϕ_w , patience c , max iteration \mathcal{K} .

- 1 train a swarm of initial data generators $\{\mathbf{x}_i\}_{i=1}^N$ on \mathcal{D} (§2.1)
- 2 initialize global best checkpoint $\mathbf{g} \leftarrow \emptyset$, global worst checkpoint $\mathbf{g}_w \leftarrow \emptyset$
- 3 **for** $i = 1$ **to** N **do**
- 4 initialize personal best $\mathbf{p}_i \leftarrow \mathbf{x}_i$, velocity $\mathbf{v}_i \leftarrow \mathbf{0}$
- 5 **if** $f(\mathbf{x}_i | \{\mathbf{m}_i\}_{i=1}^n) > f(\mathbf{g} | \{\mathbf{m}_i\}_{i=1}^n)$, $\mathbf{g} \leftarrow \mathbf{x}_i$; **if** $f(\mathbf{x}_i | \{\mathbf{m}_i\}_{i=1}^n) < f(\mathbf{g}_w | \{\mathbf{m}_i\}_{i=1}^n)$, $\mathbf{g}_w \leftarrow \mathbf{x}_i$
- 6 **end**
- 7 **for** $k = 1$ **to** \mathcal{K} **do**
- 8 **if** \mathbf{g} did not change in the last c iterations **then break**
- 9 **for** $i = 1$ **to** N **parallel do**
- 10 randomness factors $r_v, r_p, r_g, r_w \sim \mathcal{U}(0, 1)$
- 11 velocity $\mathbf{v}_i \leftarrow \frac{1}{c}[r_v\phi_v\mathbf{v}_i + r_p\phi_p(\mathbf{p}_i - \mathbf{x}_i) + r_g\phi_g(\mathbf{g} - \mathbf{x}_i) - r_w\phi_w(\mathbf{g}_w - \mathbf{x}_i)]$, where normalization term $C = r_v\phi_v + r_p\phi_p + r_g\phi_g + r_w\phi_w$
- 12 model weights $\mathbf{x}_i \leftarrow \mathbf{x}_i + \lambda\mathbf{v}_i$
- 13 **if** $f(\mathbf{x}_i | \{\mathbf{m}_i\}_{i=1}^n) > f(\mathbf{g} | \{\mathbf{m}_i\}_{i=1}^n)$, $\mathbf{g} \leftarrow \mathbf{x}_i$; **if** $f(\mathbf{x}_i | \{\mathbf{m}_i\}_{i=1}^n) < f(\mathbf{g}_w | \{\mathbf{m}_i\}_{i=1}^n)$, $\mathbf{g}_w \leftarrow \mathbf{x}_i$; **if** $f(\mathbf{x}_i | \{\mathbf{m}_i\}_{i=1}^n) > f(\mathbf{p}_i | \{\mathbf{m}_i\}_{i=1}^n)$, $\mathbf{p}_i \leftarrow \mathbf{x}_i$
- 14 **end**
- 15 **end**
- 16 **return** \mathbf{g}

Algorithm 2: Adversarial Swarms

Input: Initial evaluation data \mathcal{D} , test taker models $\{\mathbf{m}_i\}_{i=1}^n$, evaluation objective $f(\mathbf{x} | \{\mathbf{m}_i\}_{i=1}^n) \rightarrow \mathcal{R}$ where \mathbf{x} is a data generator (§2.2). Hyperparameters: max iteration \mathcal{K} , patience c .

- 1 train a swarm of initial data generators $\{\mathbf{x}_i\}_{i=1}^N$ on \mathcal{D} (§2.1) (lines 2-6, Alg.1)
- 2 initialize the data swarm $\{\mathbf{x}_i\}_{i=1}^N$ and model swarm $\{\mathbf{m}_i\}_{i=1}^n$
- 3 **for** $k = 1$ **to** \mathcal{K} **do**
- 4 **if** the two global bests \mathbf{g}_{data} and \mathbf{g}_{model} did not change in the last c iterations **then break**
- 5 $\{\mathbf{x}_i\}_{i=1}^N \leftarrow \text{PSO-step}^2(\{\mathbf{x}_i\}_{i=1}^N, \{f(\mathbf{x}_i | \{\mathbf{m}_j\}_{j=1}^n)\}_{i=1}^N)$ (lines 10-13, Alg.1)
- 6 generate joint dataset $\mathcal{D}_{joint} = \bigcup_{i=1}^N \text{sample}(\mathbf{x}_i)$
- 7 $\{\mathbf{m}_i\}_{i=1}^n \leftarrow \text{PSO-step}^2(\{\mathbf{m}_i\}_{i=1}^n, \{\text{performance}(\mathbf{m}_i | \mathcal{D}_{joint})\}_{i=1}^n)$
- 8 **end**
- 9 **return** global bests \mathbf{g}_{data} and \mathbf{g}_{model}
