BARE: Combining Base and Instruction-Tuned Language Models for Better Synthetic Data Generation

Alan Zhu [*1] **Parth Asawa** [*1] **Jared Quincy Davis** [2 3] **Lingjiao Chen** [2 4] **Boris Hanin** [5 3] **Ion Stoica** [1]
Joseph E. Gonzalez [1] **Matei Zaharia** [1]

Abstract

As the demand for high-quality data in model training grows, researchers and developers are increasingly generating synthetic data to tune and train LLMs. A common assumption about synthetic data is that sampling from instruct-tuned models is sufficient; however, these models struggle to produce diverse outputs—a key requirement for generalization. Despite various prompting methods, in this work we show that achieving meaningful diversity from instruct-tuned models remains challenging. In contrast, we find base models without post-training exhibit greater diversity, but are less capable at instruction following and hence of lower quality. Leveraging this insight, we propose **Base-Refine** (**BARE**), a synthetic data generation method that combines the diversity of base models with the quality of instruct-tuned models through a two-stage process. With minimal few-shot examples and curation, **BARE** generates diverse and high-quality datasets, improving downstream task performance. We show that fine-tuning with as few as 1,000 **BARE**-generated samples can reach performance comparable to the best similarly sized models on LiveCodeBench tasks. Furthermore, fine-tuning with **BARE**-generated data achieves a 101% improvement over instruct-only data on GSM8K and a 18.4% improvement over SOTA methods on RAFT.

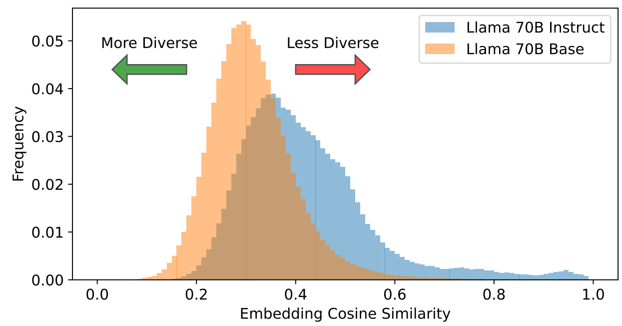

Figure 1. Histogram of pairwise embedding similarities for 1000 Llama-3.1-70B-Base vs Instruct generations of grade school math problems. The base distribution is further to the left, indicating lower similarity and hence higher diversity. Note that frequency is of the cosine similarity score.

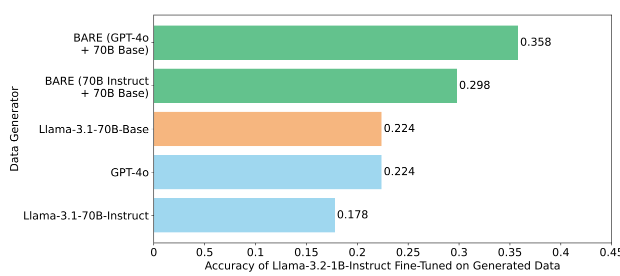

Figure 2. Accuracy of a Llama-3.2-1B-Instruct model fine-tuned on 4 different sets of synthetically generated math problems, evaluated on a randomly selected $n = 500$ subset of GSM8K. Training with **BARE**-generated data outperforms both Base and Instruct generations.

1. Introduction

As Large Language Models (LLMs) grow in size and capability, the demand for high-quality, diverse data in model training is outpacing human-generated data, necessitating the use of synthetically generated data (Villalobos et al., 2024). Consequently, we expect the compute spent on curating and generating data for model training to increase significantly in the coming years, especially in low-data domains where high-quality data is scarce. For example, it is common to use LLMs to generate synthetic data for a variety of tasks such as math, code, function calling, general reasoning, etc. (Yu et al., 2024; Guo et al., 2024; Patil et al., 2023; Liu et al., 2024).

The ease of data generation has led many dataset creators

[*]Equal contribution [1]UC Berkeley [2]Stanford University [3]Foundry [4]Microsoft Research [5]Princeton University. Correspondence to: Alan Zhu <aczhu@berkeley.edu>, Parth Asawa <pgasawa@berkeley.edu>.

(Samvelyan et al., 2024; Zhang et al., 2024c; NovaSky Team, 2025) and model creators (Dubey et al., 2024; Qwen et al., 2025; Nvidia et al., 2024; Guan et al., 2025; NovaSky Team, 2025) to turn to synthetic data sampled from instruct-tuned language models as a replacement for tasks where human-generated data is lacking.

Challenges with Instruct-Tuned Models. Though instruct-tuned models have become more and more capable at complex tasks resulting in higher quality generations, there have been discussions of how accepted methods for post-training can lead to mode collapse (Shumailov et al., 2024; Wong et al., 2024; Lambert et al., 2024). Mode collapse refers to an inability for models to generate diverse outputs to queries that don't have a single response, yet synthetic data is primarily useful when it is both high quality and diverse (Chen et al., 2024; Raventós et al., 2023).

Intuitively, one might expect a variety of prompting techniques that explicitly encourage diversity to address this gap. For example, prompting methods we baseline against include conditioning a generation on all previous ones and prompting for it to be distinct, assuming different personas for different generations, or asking for multiple distinct entries in a single call (Zhang et al., 2024b; Naik et al., 2023; Fröhling et al., 2024). However, we find that the amount of meaningful diversity from such methods is limited and cannot fully address the gap introduced by post-training. On an open-ended generation task like email generation, such prompting methods do not meaningfully improve dataset diversity (Table 2) or trained model accuracy (Figure 9).

Base vs. Instruct-Tuned Models. An alternative approach is to leverage base models, which are not constrained by post-training biases and thus produce outputs that better represent diversity present in real-world data (OpenAI, 2024b). More quantitatively, as shown in Figure 1, base models generate outputs with noticeably lower pairwise cosine similarity (mean similarity: 0.313) compared to instruct-tuned models (mean similarity: 0.421). Here we use cosine similarity as a quantitative surrogate for diversity — a dataset with greater cosine similarity has more similar items and therefore less diversity. The improved diversity in base model generations can improve downstream task performance, with Figure 2 showing that the accuracy of Llama-3.2-1B-Instruct on GSM8K increases to 22.5% from 17.8% when fine-tuning on the Llama-3.1-70B-Base generated dataset of word problems rather than the Llama-3.1-70B-Instruct generated dataset.

BARE. In this work, we observe that diversity is critical for synthetic data generation when used in downstream training and fine-tuning tasks. We find that while the data generated from a base model may be more diverse, it also tends to be of lower quality (Figure 5). This can counteract the gains from diversity and hinder downstream training. Our key insight is that by combining base and instruct-tuned models, one can improve the diversity of a dataset while controlling the quality of individual data entries.

To this end, we introduce **Base-Refine** (**BARE**) — a novel approach that leverages the diverse data generated by the base models and refines it with instruct-tuned models. In a variety of settings, we find this two-stage process enhances diversity without compromising quality, enabling the generation of datasets that improve downstream performance.

We evaluate fine-tuning with **BARE**-generated data on contemporary tasks, such as the recently introduced Retrieval-Augmented Fine-Tuning (RAFT) method (Zhang et al., 2024c). RAFT generates synthetic question/answer pairs starting from just a corpus to fine-tune a RAG model, but we find that the diversity of data it generates is low. By replacing the data generator in RAFT with **BARE**, we show we can achieve up to a 18.4% fine-tuned accuracy lift over the existing work that solely uses instruct-tuned models.

As another example, **BARE** improves the generation of synthetic math training data. We generated grade school math problems similar to GSM8K problems with several methods, shown in Figure 2. We find that refining Llama-3.1-70B-Base data generations with Llama-3.1-70B-Instruct improves a fine-tuned model's accuracy on GSM8K from 22.4% (base) to 29.8% (refined). Likewise, using GPT-4o as a generator out of the box (22.4%) does not perform as well as using GPT-4o as a refiner of generations from Llama-3.1-70B-Base (35.8%).

To summarize, our contributions are:

1. We quantitatively investigate the quality and diversity of base and instruct-tuned models for synthetic data generation across various sampling methods to motivate better system design. In Section 3, we find that base models tend to produce more diverse responses whereas instruct-tuned models offer higher quality.
2. Using these insights, in Section 4, we propose **Base-Refine** (**BARE**), a practical new method for generating synthetic data. We show in Section 5 that **BARE** consistently improves fine-tuned model performance over various baselines, including SOTA generation methods, and with only 1000 samples can lead to fine-tuned models comparable to SOTA models of similar sizes.

2. Related Work

Synthetic Data. Synthetic data is commonly used to train state of the art models like Llama 3.3 (Dubey et al., 2024), Qwen 2.5 (Qwen et al., 2025), Nemotron 4 (Nvidia et al., 2024), and o3-mini (Guan et al., 2025). However, prior work has shown its usage poses risks such as model collapse, where iterative training on low-diversity data shifts the generation distribution toward a high-probability mean, degrading both performance and diversity (Shumailov et al., 2024; Shimabucoro et al., 2024; Guo et al., 2023). Recent work has pointed towards diversity in training data improving downstream performance, though doesn't consider the quality of the data in tandem (Chen et al., 2024). In contrast, the **BARE** pipeline is designed with both of these objectives in mind, producing diverse high-quality data to support model training.

Generation Methods. Current methods to improve LLM generation diversity include prompting, sampling, and multi-stage generation.

Prompting methods leverage instruction-following by requesting distinct responses in single calls or varying prompts (e.g., assigning distinct 'personalities' or randomizing few-shot examples) (Zhang et al., 2024b; Naik et al., 2023; Fröhling et al., 2024; Chen et al., 2024; Li et al., 2022). However, compared to **BARE**, these methods depend on pre-existing data or curated prompts, limiting scalability.

Sampling methods like temperature scaling and nucleus sampling (Holtzman et al., 2020) are widely used, but often prioritize token-level randomness over semantic diversity. Indeed, methods such as logit suppression (Chung et al., 2023) can enhance diversity but may require significant manual refinement to maintain quality.

Multi-Stage generation methods vary in approach. SimpleStrat (Wong et al., 2024) explicitly generates strata of the solution space and then conditions responses on each, effective for short-answer responses (often a single word or phrase), though limiting applicability to broader synthetic data generation. More complex frameworks require extensive human setup or curation to seed the diversity with topics, making them less scalable (Lambert et al., 2024; Li et al., 2024a; 2023). In contrast, **BARE**, is a practical two-step pipeline that we show works generally with minimal human involvement.

Prior work has studied differences in base and instruct models in calibration (OpenAI, 2024b) and agentic environments (Li et al., 2024b), though no work to our knowledge has leveraged base models for synthetic data generation.

Evaluating Synthetic Data. Synthetic data utility is typically assessed by downstream performance, though to motivate better systems, we additionally study diversity and entry-wise quality.

Token-level metrics like self-BLEU (Zhu et al., 2018) are common, though embedding-based approaches (e.g., BERTScore (Zhang et al., 2019), Sentence-BERT (Reimers and Gurevych, 2019)) better capture semantic diversity rather than token diversity, which is our primary focus.

Lastly, while dataset-wide quality is often measured via downstream performance, assessing individual synthetic samples remains underexplored. In Section 3, we introduce an entry-wise quality measure to evaluate sample realism, ensuring robust synthetic data generation.

3. Motivation

Synthetic data generation should result in a diverse dataset with high-quality entries. Thus we motivate our design of **BARE** with investigations on the diversity and entry-wise quality of synthetically generated data from base and instruct-tuned models, independent of downstream utility.

3.1. Diversity & Quality

Diversity. Following Tevet and Berant (2021), we use the average neural similarity score to measure the diversity of a generated dataset. Specifically, we use OpenAI's `text-embedding-3-small` (OpenAI, 2024a) to generate embeddings and use cosine similarity to calculate similarity scores, as recommended by OpenAI. We calculate pairwise cosine similarity scores for items in a generated dataset and analyze the resulting distribution of similarities. A lower average similarity indicates a more diverse dataset.

Entry-wise quality. We seek a metric measuring the quality of individual entries in the generated datasets. To this end, we propose the *indistinguishability rate* (IR) of the entries as a quality metric. We believe that we are the first to propose this metric for the quality of an individual data entry. Specifically, we measure how often a strong LLM (e.g. GPT-4o) fails to identify the synthetically generated entry as being of lower quality when combined with $n = 3$ real dataset entries. A high IR indicates that generated data closely matches properties of real-world data and the discriminator is randomly guessing, while a low IR indicates the generated data are out-of-distribution from real world data. An example IR prompt is available in Appendix B.

3.2. Experimental Setup

Models. To investigate diversity and quality differences between instruct-tuned and base models, we evaluate a synthetic dataset generated using Llama-3.1-70B-Instruct and Llama-3.1-70B-Base (Dubey et al., 2024). We additionally use GPT-4o (OpenAI, 2024c) as an instruct-tuned model with strong instruction-following capabilities.

To allow fair comparisons and simulate a low-data setting we always provide three few-shot examples.

Domains. We examine a variety of domains covering many tasks. Our data generation domains are:

- Enron Emails (Klimt and Yang, 2004) generating training data for classifying emails as spam or legitimate. We ensure class-balanced synthetic data by explicitly conditioning each generation on a uniform class distribution.
- 20 Newsgroups (Pedregosa et al., 2011) generating training data for classifying Usenet messages into one of 20 newsgroup sources. The model generates classes along with content, allowing the generation method to determine the class distribution of the synthetic dataset.
- Retrieval-Augmented Fine-Tuning (RAFT) (Zhang et al., 2024c), a domain and generator-agnostic synthetic data generation framework for fine-tuning data in RAG tasks. Here, Q/A pair data generation needs to be conditioned on contexts mimicking retrieval results. We use:
  - HotpotQA (Yang et al., 2018), a general Wikipedia-based short-answer task.
  - PubMedQA (Jin et al., 2019), a medical abstract-based yes/no/maybe question-answering task.
- GSM8K (Cobbe et al., 2021), generating grade-school math word problems and solutions for fine-tuning.
- LiveCodeBench's Test Output Prediction (LCB TOP) (Jain et al., 2024), generating coding questions and answers on predicting test case outputs given a natural language description of an algorithm's expected behavior and test input for fine-tuning.

For the classification tasks of Enron and Newsgroups, we generate a dataset of size $n = 500$. For the generative model fine-tuning tasks of HotpotQA, PubMedQA, GSM8K, and LCB TOP, we generate a larger dataset of size $n = 1000$.

We sample at a default temperature of 0.7 for base models and the highest temperature at which we experimentally found data generation is still coherent for instruct models, which is 1.0 for Llama models and 1.2 for GPT-4o. However, for Enron, we sample from GPT-4o at a temperature of 1.0 to maintain generation coherence.

Note we do not measure diversity in the RAFT domains as we expect noise due to conditioning on different documents/-contexts. For Enron, due to the relatively large differences between spam and legitimate emails, only similarities between emails of the same type are calculated.

3.3. Investigation Results

Diversity. Looking at the pairwise cosine similarity distributions of the embeddings of the generated dataset in

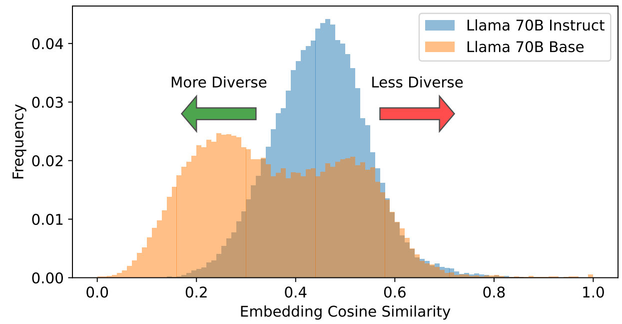

Figure 3. Distribution of pairwise embedding cosine similarities for Llama-3.1-70B-Instruct and Llama-3.1-70B-Base generations for Enron spam emails. Base model distribution have more density in the low-similarity region and less density in the high-similarity region, indicating greater diversity. Note that frequency is of the cosine similarity score.

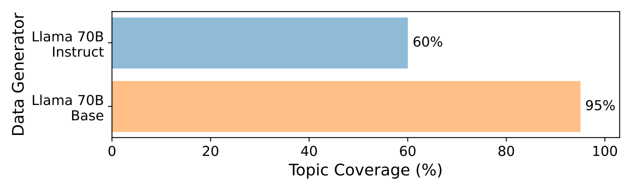

Figure 4. Topic coverage of Llama-3.1-70B-Instruct and Llama-3.1-70B-Base generations for Newsgroups. Coverage after $n = 500$ generations. The base model covers all topics except one whereas the instruct model covers only 60% of the topics.

Figure 3 (and recalling Figure 1), the base distribution is consistently shifted to the left, indicating that the base model generations are more diverse and correspond with a decrease in average similarity (Table 1). The higher diversity of base models is more strikingly shown in the coverage of generated classes for Newsgroups (Figure 4), where over 500 generations the Instruct model covers only 14 of the 20 classes. In contrast, the Base model covers every class except 1 (`talk.religion.misc`).

This trend is reflected in Table 1, with all domains except one showing results for base models with higher diversity. Upon inspection, we attribute the reversal in trend in LCB TOP to the base model repeating phrases from the examples in the prompt. This is related to potential issues with the quality of base model generations, which we discuss below.

Sampling Methods

We further explore data generation using several different sampling methods. As base models struggle to follow complex instructions, only instruct models are used for methods beyond independent sampling.

Table 1. Average pairwise embedding cosine similarity of Llama-3.1-70B-Instruct vs. Llama-3.1-70B-Base generated data. Generations from Base are almost always more diverse than Instruct, despite always sampling at a lower temperature (0.7 vs 1.0).

DATASET	INSTRUCT	BASE
ENRON	0.450	**0.350**
NEWSGROUPS	0.256	**0.162**
GSM8K	0.421	**0.313**
LCB TOP	**0.389**	0.468

Table 2. Average pairwise embedding cosine similarity of various prompting techniques. GPT-4o is used as the prompting generator due to the need for strong instruction following capabilities; Llama models frequently derailed. Llama-3.1-70B-Base generations are almost uniformly more diverse than any prompting method, except for persona prompting on GSM8K, where it is comparable.

METHOD	ENRON	GSM8K
GPT-4O INDEPENDENT	0.574	0.427
GPT-4O PERSONA	0.580	**0.308**
GPT-4O SEQUENTIAL	0.511	0.398
GPT-4O IN-ONE	0.363	0.347
GPT-4O DYNAMIC FEWSHOT	0.511	0.463
LLAMA-3.1-70B-BASE	**0.350**	*0.313*

- **Independent (temperature) Sampling**: The model generates n samples to form a dataset.
- **Persona Prompting**: The model is instructed to respond as a predefined persona (Fröhling et al., 2024).
- **Sequential Prompting**: The model iteratively generates outputs distinct from previous ones.
- **In-One Prompting**: The model is prompted to generate k different entries in a single response (Zhang et al., 2024b).
- **Dynamic Few-Shot Examples**: Few-shot examples are randomly selected for each call (breaking the assumption of low-data conditioned generation) (Li et al., 2022).

Table 2 shows that base models generally yield higher diversity than almost all prompting methods on the average embedding distance metric in two domains. One exception is persona prompting on GSM8K — though this diversity arises more from flavor text differences due to personas rather than actual content.

We thus find that base models are generally more diverse than instruct-tuned models and that temperature increases and prompting methods are insufficient to bridge the gap, motivating our usage of base models in the first stage of **BARE**.

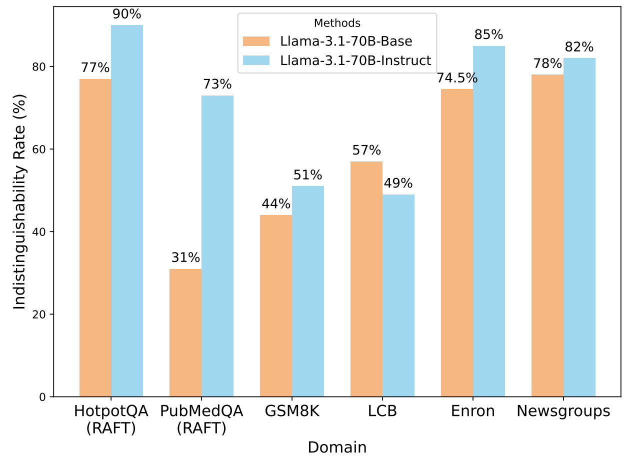

Figure 5. Indistinguishability Rate for Llama-3.1-70B-Base and Instruct model data generation methods across various datasets. Llama-3.1-70B-Instruct is almost uniformly better at appearing high-quality alongside real-world data.

Entry-wise quality. Figure 5 presents our results. In general, the instruct-tuned model has a higher IR, indicating that it is better at producing generations that resemble high-quality data. Note that some IRs are well above 75%. This outcome is not unexpected: if a model consistently generates data that aligns with the most common patterns in the real-world distribution, it becomes difficult to distinguish from actual data. Since the real-world entries often also include non-modal (less frequent) samples, a discriminator tasked with identifying lower quality data may instead misclassify these less common real-world samples as synthetic.

As mentioned above, on LCB TOP the base model repeats phrases from examples in the prompt. This leads to a higher IR as the generations copy real-world examples. Thus, while individual base generations are technically more realistic, their shortcomings are captured in the diversity metric.

We therefore find that the superior instruction follow-up capabilities of instruct-tuned models can help generate more realistic data, motivating our usage of instruct-tuned models in the second stage of **BARE**.

4. Base-Refine (BARE)

We leverage our insights to propose **BARE**, a practical synthetic data generation method combining the diversity of base models with the quality of instruct models. **BARE** uses a base model to generate an initial set of diverse but potentially lower quality data, after which an instruct-tuned model individually refines each example from the initial set – as shown in the example in Figure 6.

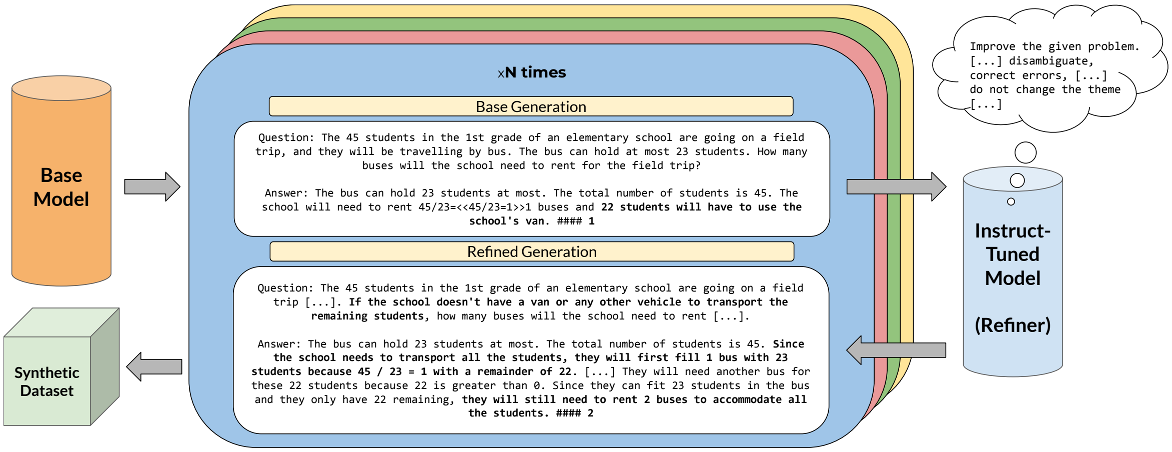

Figure 6. Instruct-tuned models provide high-quality but low-diversity data, while base models provide low-quality but high-diversity data. **BARE** independently generates a diverse initial set of data points with a base model and refines each entry individually with an instruct-tuned model to create a high-quality, high-diversity dataset. In this example of a real grade school math problem generation, the Llama-3.1-70B-Base model hallucinates in its answer to its own question. The refiner (Llama-3.1-70B-Instruct) recognizes this and disambiguates the question and corrects the reasoning.

In the base step, the base model uses a prompt with minimal few-shot examples to generate. In the refine step, the instruct-tuned model is instructed to retain the concept of the base model generation while improving it according to specific criteria (e.g., realism, correctness). This retains the overall diversity of the base model generated set while exerting greater control over the quality of the final generation.

Importantly, few-shot examples are only required in the base step to ensure formatting (though they can also be included in the refine step), and only in small amounts. Thus, **BARE** is especially useful in domains with very limited real-world data as little human effort is required to generate a diverse high-quality synthetic dataset. In our experiments, we limit ourselves to just three few-shot examples. In addition, we intentionally use very general prompts for **BARE** to demonstrate its flexibility, underscoring the potential for even greater improvement with tailored prompts. Representative prompts are included in Appendix B.

5. Evaluation

We evaluate **BARE** for diversity, data quality, and downstream utility on the same domains and against the same baseline methods presented in Section 3. We implement **BARE** with Llama-3.1-70B-Base in the base generation stage and Llama-3.1-70B-Instruct in the refinement stage, and perform additional experiments with the Llama 3.1 8B family and with Llama 3.1 8/70B as the base model and GPT-4o as the refiner (Dubey et al., 2024; OpenAI, 2024c).

We follow generally the same sampling temperatures as in Section 3. However, for Enron and Newsgroups, we instead sample from Llama-3.1-8B-Instruct at temperature 0.7 in order to maintain coherence in generations. We always sample the refine model at a temperature of 0.7.

For evaluation of downstream utility for the generative tasks, we LoRA (Hu et al., 2021) fine-tune Llama-3.1-8B-Instruct for 4 epochs using the generated data, except for GSM8K where Llama-3.2-1B-Instruct (Dubey et al., 2024) is fine-tuned instead due to high baseline performance of the 8B model. Other fine-tuning hyperparameters are in Appendix A.1.1. The fine-tuned models are evaluated on a static $n = 100$ test set for HotpotQA and PubMedQA, a static $n = 500$ test set for GSM8K, and the full $n = 442$ test set for LCB TOP. In this section, we will focus only on the generative tasks; the downstream evaluation process for classification tasks is presented in Appendix A.1.2 and detailed results for all domains can be found in Appendix A.2.

BARE Quality & Diversity. We begin by looking at the quality/diversity trends of **BARE** when compared with previous methods. From the histograms plotted in Figure 7, we can see that **BARE** effectively does not change the similarity distribution of generated data when compared to the base model at all – it is able to successfully retain the diversity of base generations. Detailed results with average embedding similarity scores can be found in Appendix A.2.

Simultaneously though, looking at Figure 8, we can see that while retaining diversity, **BARE** leads to a monotonic increase in the IR for every domain - suggesting that it is able to lift the quality of the generations to be on par or in

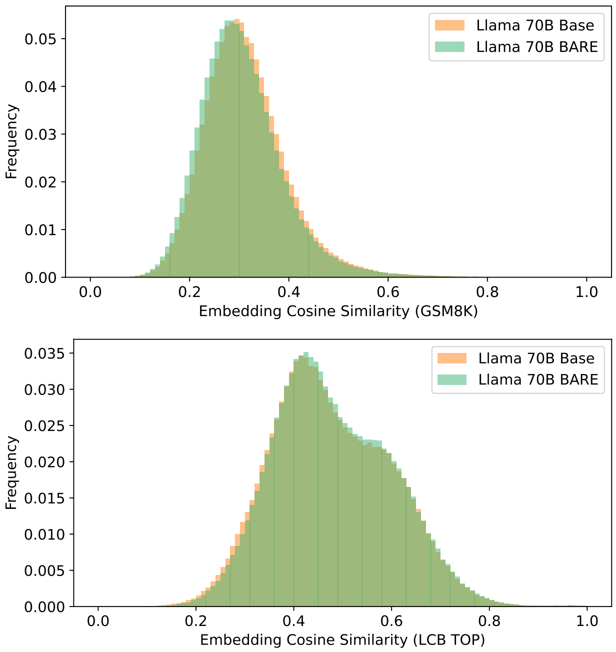

Figure 7. Distribution of pairwise embedding similarities for Llama-3.1-70B-Base and Llama 3.1 70B BARE generations for GSM8K (top) and LCB TOP (bottom). The Base and **BARE** distributions are extremely similar for both tasks, indicating that refinement retains the diversity of the base generations. Note that frequency is of the cosine similarity score.

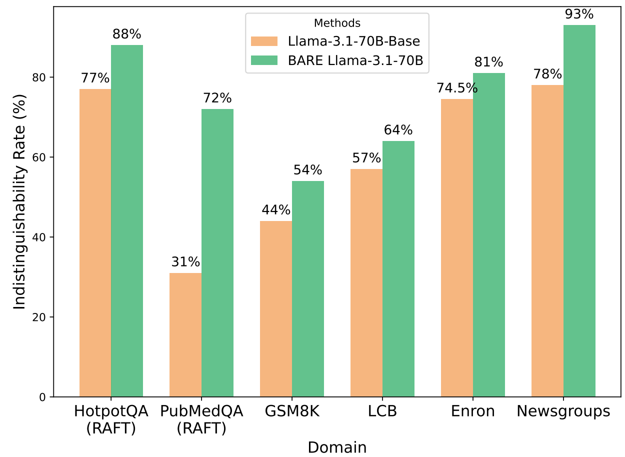

Figure 8. Indistinguishability Rate for Llama-3.1-70B-Base and **BARE** data generation across various datasets. **BARE** consistently improves the quality of data generated from base models.

some cases even surpass directly sampling from an instruct model. Combined, this indicates that **BARE** is capable of leveraging the diversity of base models and quality of instruct-tuned models in its end generations.

Fine-tuned Model Accuracy. We now show the utility of **BARE** datasets as a whole. In Figure 10, we demonstrate the accuracy of a model fine-tuned on the datasets generated using different methods. Almost uniformly, **BARE** leads to greater improvements in downstream model quality than generating from either Base or Instruct models, and across all domains **BARE**-based data leads to the highest fine-tuned model accuracy.

Impressively, in Figure 9, beyond just base and instruct sampling, we show that enhancing GPT-4o with **BARE** using only a small base model like Llama-3.1-8B-Base produces **monotonically better** fine-tuning data than all prompting methods discussed in Section 3.3 that use only GPT-4o. These clear results showcase **BARE**'s ability to out-perform the existing methods for high-quality, diverse data generation.

Focusing on the RAFT domains, **BARE** improves upon the standard SOTA pipeline for fine-tuning LLMs for RAG by up to 18.4%, as seen with the 8B family on HotpotQA (standard RAFT is implemented in our Instruct generation results). **BARE** with both model families also outperforms existing RAFT pipelines on PubMedQA. While **BARE** with the Llama 3.1 70B family does not improve upon RAFT for HotpotQA, switching out the Llama-3.1-70B-Instruct refiner for GPT-4o does lead to an improvement (Appendix A.2).

On GSM8K, **BARE** is the only method that provides useful training data. The un-trained model performance was 21.8%, and fine-tuning on **BARE** generated data achieves accuracies of 24.9% and 32.8% with the Llama 8B and 70B families, respectively. Accuracy when training with data generated by single model methods either decreased accuracy or had little difference. In fact, training on data generated by Llama-3.1-70B-Instruct led to an accuracy of just 17.8%, which **BARE** with Llama-3.1-70B-Base refined by GPT-4o outperforms by 101% (35.8% accuracy).

On LCB TOP, fine-tuning a Llama-3.1-8B-Instruct model on 1000 examples generated by **BARE** using the Llama 3.1 8B model family for just 4 epochs resulted in performance of 28.1% accuracy, comparable to the current top models of similar size on the LCB leaderboard: DeepSeekCoder 6.7B Instruct (Guo et al., 2024) at 32.7% and Magicoder$\mathcal{S}$ DS 6.7B (Wei et al., 2024) at 32.4%. While both these models perform slightly better, they used orders of magnitude more data and trained for longer than us.

Similar to HotpotQA, on LCB TOP, **BARE** with the Llama 3.1 70B family shows less of an improvement than the 70B-instruct-only data generation method. Again, using GPT-4o as a refiner instead of Llama-3.1-70B-Instruct gives stronger results. This suggests that with the right refiner choice,

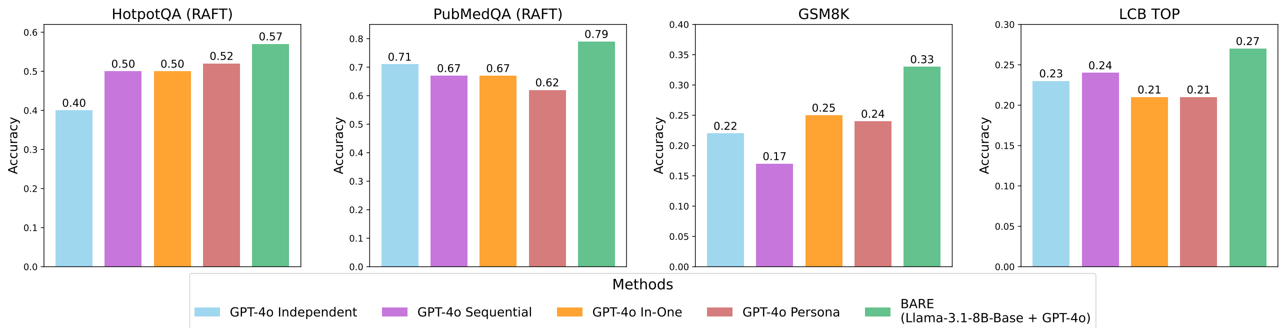

Figure 9. Fine-tuned model accuracy results on HotpotQA, PubMedQA, GSM8K, and LCB TOP for prompting methods using GPT-4o. Prompting methods have mixed effects on downstream performance when compared against standard sampling, while using GPT-4o with simply the Llama-3.1-8B-Base model in **BARE** improves consistently over all prompting methods.

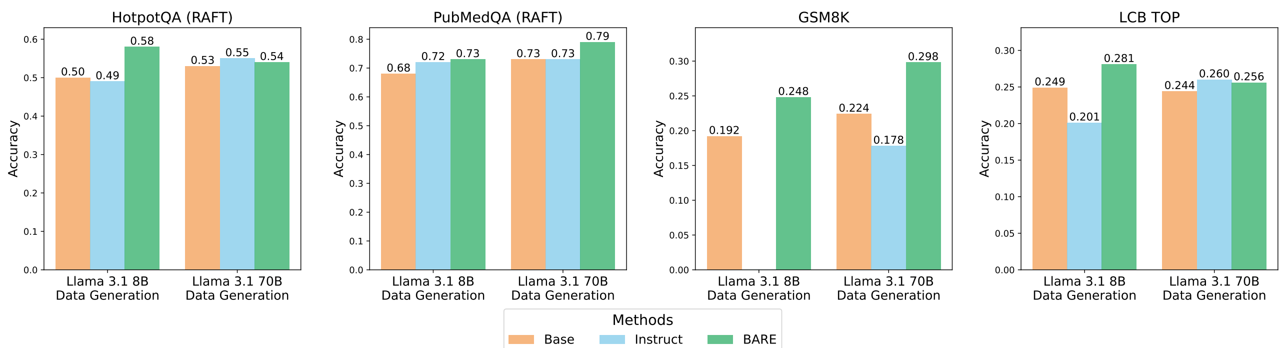

Figure 10. Accuracy of a Llama-3.1-8B-Instruct Model (HotpotQA, PubMedQA, LCB TOP) and Llama-3.2-1B-Instruct Model (GSM8K) finetuned on synthetic data generated using Base, Instruct, and **BARE** methods. Note that GSM8K 8B Instruct results are not shown as data generations were derailed.

BARE consistently provides gains. We show detailed results of refining with GPT-4o in all domains in Appendix A.2, emphasizing **BARE**'s consistently strong data generation capabilities.

We also find that temperature ablations do not meaningfully affect the utility of instruct-tuned model generations, especially when compared to gains using **BARE**. For details, see Appendix A.3.

Instruct-Instruct Ablation. A possible challenge is that the gains in performance are due to a multi-step pipeline rather than combining a base and instruct-tuned model. To demonstrate the importance of using a base model in **BARE**, we performed an ablation where an instruct-tuned model's generations are refined by an instruct-tuned model on the GSM8K task. The accuracy on the test set after switching Llama-3.1-70B-Base to Instruct in the first step drops from 29.8% to 25.4%. This trend held when GPT-4o was the refiner as well, with a drop from 35.8% to 30.8%. At the same time, the pattern of little change in the average pairwise similarity before and after refinement remains, indicating that diversity must be introduced in the first stage. Having earlier established that base models are most effective at diversity, we conclude that the use of base models is a necessary component of **BARE**. Detailed results from this ablation are in Appendix A.4.

6. Conclusion

6.1. Key Contributions

In summary, in this work we quantitatively investigated the quality and diversity of various synthetic data generation methods. Through this investigation, we find that base models are generally more diverse than instruct-tuned models while instruct-tuned models produce higher quality output than base models, validating hypotheses in the research community. These insights motivate the design of a better system to generate synthetic data, **BARE**.

Through extensive experiments, we validate the importance of each step in **BARE** and demonstrate its ability to preserve base model diversity while enhancing output quality. Moreover, by fine-tuning on **BARE**-generated data for various domains, we underscore **BARE**'s practical utility, consistently outperforming existing synthetic data generation methods on downstream tasks such as GSM8K and LiveCodeBench, in addition to RAFT, for which we set a new SOTA.

6.2. Future Work

There are a lot of exciting directions for future work. For one, **BARE** is not necessarily the only way to elicit diversity in the synthetic data generation process. We were able to achieve all of our results with essentially basic prompting, leading us to believe there is room for even more improvement if one was to, for example, fine-tune the refiner specifically for this use case or use stronger refiner models. Introducing additional stages to **BARE** could also lead to improvements (e.g., additional refinement steps). Beyond multi-stage systems, the design space for diversity is vast – in one direction, training instruct-tuned models with different objective functions that encourage entropy is another area of exploration.

Furthermore, the implications of a lack of high-quality, diverse generations from LLMs go beyond just synthetic data. For example, much of the work in inference time compute relies on models being able to generate sufficiently different possible trajectories and using these diverse trajectories (exploration) with a feedback signal to improve reasoning capabilities (Zhang et al., 2024a; Zelikman et al., 2022), yet only a few methods look explicitly at generator diversity (Wang et al., 2024).

Lastly, the same **BARE** method demonstrated in this paper for synthetic training data can also be used with no modifications to generate synthetic evaluation sets, which are especially valuable in so many real-world domains with low-data availability.

Acknowledgements

We would like to thank Justin Wong, Peter West, Raj Ammanabrolu, Alex Dimakis, Alex Xu, and Andrew Qin for their many insightful discussions about this work.

Sky Computing Lab is supported by gifts from Accenture, AMD, Anyscale, Cisco, Google, IBM, Intel, Intesa Sanpaolo, Lambda, Microsoft, NVIDIA, Samsung SDS, SAP, and VMware.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 2146752. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Hao Chen, Abdul Waheed, Xiang Li, Yidong Wang, Jindong Wang, Bhiksha Raj, and Marah I. Abdin. On the diversity of synthetic data and its impact on training large language models, 2024. URL https://arxiv.org/abs/2410.15226.

John Joon Young Chung, Ece Kamar, and Saleema Amershi. Increasing diversity while maintaining accuracy: Text data generation with large language models and human interventions. *arXiv preprint arXiv:2306.04140*, 2023.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 herd of models, 2024.

Leon Fröhling, Gianluca Demartini, and Dennis Assenmacher. Personas with attitudes: Controlling LLMs for diverse data annotation. *arXiv preprint arXiv:2410.11745*, 2024.

Melody Y. Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Helyar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, Hyung Won Chung, Sam Toyer, Johannes Heidecke, Alex Beutel, and Amelia Glaese. Deliberative alignment: Reasoning enables safer language models, 2025. URL https://arxiv.org/abs/2412.16339.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, et al. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*, 2024.

Yanzhu Guo, Guokan Shang, Michalis Vazirgiannis, and Chloé Clavel. The curious decline of linguistic diversity: Training language models on synthetic text. *arXiv preprint arXiv:2311.09807*, 2023.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *ICLR*, 2020.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code, 2024. URL https://arxiv.org/abs/2403.07974.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering, 2019. URL https://arxiv.org/abs/1909.06146.

Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *Machine Learning: ECML 2004*, volume 3201 of *Lecture Notes in Computer Science*, pages 217–226. Springer, Berlin, Heidelberg, 2004. doi: 10.1007/978-3-540-30115-8_22.

Nathan Lambert, Hailey Schoelkopf, Aaron Gokaslan, Luca Soldaini, Valentina Pyatkin, and Louis Castricato. Self-directed synthetic dialogues and revisions technical report. *arXiv preprint arXiv:2407.18421*, 2024.

Haoran Li, Qingxiu Dong, Zhengyang Tang, Chaojun Wang, Xingxing Zhang, Haoyang Huang, Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, Yuxian Gu, Xin Cheng, Xun Wang, Si-Qing Chen, Li Dong, Wei Lu, Zhifang Sui, Benyou Wang, Wai Lam, and Furu Wei. Synthetic data (almost) from scratch: Generalized instruction tuning for language models, 2024a. URL https://arxiv.org/abs/2402.13064.

Margaret Li, Weijia Shi, Artidoro Pagnoni, Peter West, and Ari Holtzman. Predicting vs. acting: A trade-off between world modeling & agent modeling, 2024b. URL https://arxiv.org/abs/2407.02446.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. Making large language models better reasoners with step-aware verifier. *arXiv preprint arXiv:2206.02336*, 2022.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. Textbooks are all you need II: phi-1.5 technical report, 2023. URL https://arxiv.org/abs/2309.05463.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, and Andrew M. Dai. Best practices and lessons learned on synthetic data, 2024. URL https://arxiv.org/abs/2404.07503.

Ranjita Naik, Varun Chandrasekaran, Mert Yuksekgonul, Hamid Palangi, and Besmira Nushi. Diversity of thought improves reasoning abilities of large language models. *arXiv preprint arXiv:2310.07088*, 2023.

NovaSky Team. Sky-T1: Train your own o1 preview model within $450. https://novasky-ai.github.io/posts/sky-t1, 2025. Accessed: 2025-01-09.

Nvidia, :, Bo Adler, Niket Agarwal, Ashwath Aithal, Dong H. Anh, Pallab Bhattacharya, Annika Brundyn, Jared Casper, Bryan Catanzaro, Sharon Clay, Jonathan Cohen, Sirshak Das, Ayush Dattagupta, Olivier Delalleau, Leon Derczynski, Yi Dong, Daniel Egert, Ellie Evans, Aleksander Ficek, Denys Fridman, Shaona Ghosh, Boris Ginsburg, Igor Gitman, Tomasz Grzegorzek, Robert Hero, Jining Huang, Vibhu Jawa, Joseph Jennings, Aastha Jhunjhunwala, John Kamalu, Sadaf Khan, Oleksii Kuchaiev, Patrick LeGresley, Hui Li, Jiwei Liu, Zihan Liu, Eileen Long, Ameya Sunil Mahabaleshwarkar, Somshubra Majumdar, James Maki, Miguel Martinez, Maer Rodrigues de Melo, Ivan Moshkov, Deepak Narayanan, Sean Narenthiran, Jesus Navarro, Phong Nguyen, Osvald Nitski, Vahid Noroozi, Guruprasad Nutheti, Christopher Parisien, Jupinder Parmar, Mostofa Patwary, Krzysztof Pawelec, Wei Ping, Shrimai Prabhumoye, Rajarshi Roy, Trisha Saar, Vasanth Rao Naik Sabavat, Sanjeev Satheesh, Jane Polak Scowcroft, Jason Sewall, Pavel Shamis, Gerald Shen, Mohammad Shoeybi, Dave Sizer, Misha Smelyanskiy, Felipe Soares, Makesh Narsimhan Sreedhar, Dan Su, Sandeep Subramanian, Shengyang Sun, Shubham Toshniwal, Hao Wang, Zhilin Wang, Jiaxuan You, Jiaqi Zeng, Jimmy Zhang, Jing Zhang, Vivienne Zhang, Yian Zhang, and Chen Zhu. Nemotron-4 340b technical report, 2024. URL https://arxiv.org/abs/2406.11704.

OpenAI. New embedding models and API updates. https://openai.com/index/new-embedding-models-and-api-updates/, 2024a.

OpenAI. GPT-4 technical report, 2024b. URL https://arxiv.org/abs/2303.08774.

OpenAI. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/, 2024c.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs, 2023. URL https://arxiv.org/abs/2305.15334.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. In *Journal of Machine Learning Research*, volume 12, pages 2825–2830, 2011.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression, 2023. URL https://arxiv.org/abs/2306.15063.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL https://arxiv.org/abs/1908.10084.

Mikayel Samvelyan, Sharath Chandra Raparthy, Andrei Lupu, Eric Hambro, Aram H. Markosyan, Manish Bhatt, Yuning Mao, Minqi Jiang, Jack Parker-Holder, Jakob Foerster, Tim Rocktäschel, and Roberta Raileanu. Rainbow teaming: Open-ended generation of diverse adversarial prompts, 2024. URL https://arxiv.org/abs/2402.16822.

Luísa Shimabucoro, Sebastian Ruder, Julia Kreutzer, Marzieh Fadaee, and Sara Hooker. LLM see, LLM do: Guiding data generation to target non-differentiable objectives, 2024. URL https://arxiv.org/abs/2407.01490.

Ilia Shumailov, Zakhar Shumaylov, and Yue et al. Zhao. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024. doi: 10.1038/s41586-024-07566-y.

Guy Tevet and Jonathan Berant. Evaluating the evaluation of diversity in natural language generation. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 326–346, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.25. URL https://aclanthology.org/2021.eacl-main.25/.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. Will we run out of data? limits of llm scaling based on human-generated data, 2024. URL https://arxiv.org/abs/2211.04325.

Evan Wang, Federico Cassano, Catherine Wu, Yunfeng Bai, Will Song, Vaskar Nath, Ziwen Han, Sean Hendryx, Summer Yue, and Hugh Zhang. Planning in natural language improves LLM search for code generation. *arXiv preprint arXiv:2409.03733*, 2024.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Empowering code generation with OSS-instruct. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 52632–52657. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wei24h.html.

Justin Wong, Yury Orlovskiy, Michael Luo, Sanjit A. Seshia, and Joseph E. Gonzalez. SimpleStrat: Diversifying language model generation with stratification, 2024. URL https://arxiv.org/abs/2410.09038.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering, 2018. URL https://arxiv.org/abs/1809.09600.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models, 2024. URL https://arxiv.org/abs/2309.12284.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning, 2022. URL https://arxiv.org/abs/2203.14465.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. Accessing GPT-4 level mathematical olympiad solutions via Monte Carlo tree self-refine with LLaMa-3 8B. *arXiv preprint arXiv:2406.07394*, 2024a.

Tianhui Zhang, Bei Peng, and Danushka Bollegala. Improving diversity of commonsense generation by large language models via in-context learning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9226–9242, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.540. URL https://aclanthology.org/2024.findings-emnlp.540.

Tianjun Zhang, Shishir G. Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E. Gonzalez. RAFT: Adapting language model to domain specific RAG, 2024c. URL https://arxiv.org/abs/2403.10131.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. BERTscore: Evaluating text generation with BERT. *arXiv preprint arXiv:1904.09675*, 2019.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100, 2018.

A. Additional Results

A.1. Downstream Evaluation - Additional Details

A.1.1. Fine-Tuning Task Hyperparameters

We list below the fine-tuning hyperparameters that were used in common for HotpotQA, PubMedQA, GSM8K, and LCB TOP. Learning rate was determined independently for each domain via learning rate sweeps (across orders of magnitude); each sweep gave the same optimal learning rate.

- Learning Rate: 0.001
- LoRA α: 16
- LoRA Rank: 8
- LoRA Dropout: 0.0

A.1.2. Classification Task Setup

The generated data is used to train a BERT-based classifier (Devlin et al., 2018) for 2 epochs on Enron and 9 epochs on Newsgroups. The trained models are evaluated on a static test set with $n = 500$ examples for each domain.

A.2. Core Experiment Results - All Domains

This appendix contains diversity, IR, and downstream performance results for all core experiments: generation with Llama 3.1 8B and 70B Base and Instruct models, **BARE** with Llama 3.1 models of both families, and **BARE** with the use of GPT-4o.

Note that HotpotQA RAFT and PubMedQA RAFT diversity results present here were not presented in Table 1 as we believe the numbers are noisy and not fit for drawing conclusions, due to the use of 100 different simulated retrieval contexts that generation was conditioned on (as required by RAFT). Not only does this introduce noise to the similarity calculation, but the strong instruction following capability of instruct models allow them to better leverage the inherent diversity in different prompts. However, for completeness, we report the values in the tables in this appendix.

Table 3. Average pairwise embedding cosine similarity, IR, and downstream F1 results on a randomly selected static $n = 500$ subset of Enron. A BERT model with a classification head was trained for 2 epochs on the generated data ($n = 500$). Only pairwise similarities for generations within the same class (spam or legitimate) were calculated.

Generation Method	Average Embedding Similarity	IR	Downstream F1
Llama 3.1 8B Instruct	0.500	86.0%	0.753
Llama 3.1 70B Instruct	0.450	85.0%	0.848
Llama 3.1 8B Base	0.368	63.5%	0.790
Llama 3.1 70B Base	0.350	74.5%	0.819
BARE Llama 3.1 8B	0.413	85.0%	**0.872**
BARE Llama 3.1 70B	0.406	82.0%	0.771
BARE GPT-4o + Llama 3.1 8B Base	0.379	84.5%	**0.872**
BARE GPT-4o + Llama 3.1 70B Base	0.356	88.5%	0.846

Table 4. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on on a randomly selected static $n = 500$ subset of Newsgroups. A BERT model with a classification head was trained for 9 epochs on the generated data ($n = 500$).

Generation Method	Average Embedding Similarity	IR	Downstream Accuracy
Llama 3.1 8B Instruct	0.271	85%	26%
Llama 3.1 70B Instruct	0.246	82%	30%
Llama 3.1 8B Base	0.155	58%	41%
Llama 3.1 70B Base	0.162	78%	29%
BARE Llama 3.1 8B	0.162	91%	40%
BARE Llama 3.1 70B	0.134	93%	**49%**
BARE GPT-4o + Llama 3.1 8B Base	0.131	81%	44%
BARE GPT-4o + Llama 3.1 70B Base	0.285	87%	47%

Table 5. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on a randomly selected static $n = 100$ subset of HotpotQA RAFT. A Llama-3.1-8B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$). The baseline performance of the Llama-3.1-8B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row.

Generation Method	Average Embedding Similarity	IR	Downstream Accuracy
Baseline Performance	–	–	33%
Llama 3.1 8B Instruct	0.214	76%	49%
Llama 3.1 70B Instruct	0.216	90%	55%
Llama 3.1 8B Base	0.221	62%	50%
Llama 3.1 70B Base	0.209	77%	53%
BARE Llama 3.1 8B	0.217	77%	**58%**
BARE Llama 3.1 70B	0.210	88%	54%
BARE GPT-4o + Llama 3.1 8B Base	0.214	78%	57%
BARE GPT-4o + Llama 3.1 70B Base	0.205	89%	56%

Table 6. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on a randomly selected static $n = 100$ subset of PubMedQA RAFT. A Llama-3.1-8B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$). The baseline performance of the Llama-3.1-8B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row.

Generation Method	Average Embedding Similarity	IR	Downstream Accuracy
Baseline Performance	–	–	53%
Llama 3.1 8B Instruct	0.376	63%	72%
Llama 3.1 70B Instruct	0.373	73%	73%
Llama 3.1 8B Base	0.396	39%	68%
Llama 3.1 70B Base	0.603	31%	73%
BARE Llama 3.1 8B	0.377	47%	73%
BARE Llama 3.1 70B	0.367	72%	**79%**
BARE GPT-4o + Llama 3.1 8B Base	0.385	57%	72%
BARE GPT-4o + Llama 3.1 70B Base	0.474	40%	63%

Table 7. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on a $n = 500$ randomly selected static subset of GSM8K. A Llama-3.2-1B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$) instead of the 3.1 8B model due to its high no-training performance. The baseline performance of the Llama-3.2-1B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row. Llama-3.1-8B-Instruct generation results are not reported due to data generation derailing.

Generation Method	Average Embedding Similarity	IR	Downstream Accuracy
Baseline Performance	–	–	21.8%
Llama 3.1 8B Instruct	N/A	N/A	N/A
Llama 3.1 70B Instruct	0.421	51%	17.8%
Llama 3.1 8B Base	0.310	23%	19.2%
Llama 3.1 70B Base	0.313	44%	22.4%
BARE Llama 3.1 8B	0.310	27%	24.8%
BARE Llama 3.1 70B	0.305	54%	29.8%
BARE GPT-4o + Llama 3.1 8B Base	0.295	60%	32.8%
BARE GPT-4o + Llama 3.1 70B Base	0.302	64%	**35.8%**

Table 8. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on the full LCB TOP set. A Llama-3.1-8B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$). The baseline performance of the Llama-3.1-8B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row.

Generation Method	Average Embedding Similarity	IR	Downstream Accuracy
Baseline Performance	–	–	18.6%
Llama 3.1 8B Instruct	0.416	51%	20.6%
Llama 3.1 70B Instruct	0.389	49%	26.0%
Llama 3.1 8B Base	0.468	36%	24.9%
Llama 3.1 70B Base	0.477	57%	24.4%
BARE Llama 3.1 8B	0.462	47%	**28.1%**
BARE Llama 3.1 70B	0.481	64%	25.6%
BARE GPT-4o + Llama 3.1 8B Base	0.459	72%	26.7%
BARE GPT-4o + Llama 3.1 70B Base	0.471	68%	27.4%

A.3. Independent Sampling Temperature Ablations - HotpotQA, PubMedQA, and LCB TOP

This appendix contains diversity, IR, and downstream performance results for our temperature ablation experiments. We perform a temperature sweep for Llama-3.1-8B-Instruct generation with $t = 0.5, 0.7, 1.0$. We find that while adjusting the temperature can improve downstream performance, in general the gains are small relative to gains by using **BARE**.

Table 9. Temperature ablations with independent sampling from Llama-3.1-8B-Instruct. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on a randomly selected static $n = 100$ subset of HotpotQA RAFT. A Llama-3.1-8B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$). The baseline performance of the Llama-3.1-8B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row.

Generation Method	Average Embedding Similarity	IR	Downstream Accuracy
Baseline Performance	–	–	33%
Llama 3.1 8B Instruct ($t = 1.0$)	0.214	76%	49%
Llama 3.1 8B Instruct ($t = 0.7$)	0.216	77%	50%
Llama 3.1 8B Instruct ($t = 0.5$)	0.220	83%	42%
Llama 3.1 8B Base ($t = 1.0$)	0.221	62%	50%
BARE Llama 3.1 8B (All $t = 0.7$)	0.217	77%	**58%**

Table 10. Temperature ablations with independent sampling from Llama-3.1-8B-Instruct. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on a randomly selected static $n = 100$ subset of PubMedQA RAFT. A Llama-3.1-8B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$). The baseline performance of the Llama-3.1-8B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row.

Generation Method	Average Embedding Similarity	IR	Downstream Accuracy
Baseline Performance	–	–	53%
Llama 3.1 8B Instruct ($t = 1.0$)	0.376	62%	72%
Llama 3.1 8B Instruct ($t = 0.7$)	0.375	71%	72%
Llama 3.1 8B Instruct ($t = 0.5$)	0.377	73%	**75%**
Llama 3.1 8B Base ($t = 1.0$)	0.396	39%	68%
BARE Llama 3.1 8B (All $t = 0.7$)	0.377	47%	73%

Table 11. Temperature ablations with independent sampling from Llama-3.1-8B-Instruct. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on the full LCB TOP set. A Llama-3.1-8B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$). The baseline performance of the Llama-3.1-8B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row.

GENERATION METHOD	AVERAGE EMBEDDING SIMILARITY	IR	DOWNSTREAM ACCURACY
BASELINE PERFORMANCE	–	–	18.6%
LLAMA 3.1 8B INSTRUCT ($t = 1.0$)	0.365	33%	20.1%
LLAMA 3.1 8B INSTRUCT ($t = 0.7$)	0.416	51%	20.6%
LLAMA 3.1 8B INSTRUCT ($t = 0.5$)	0.450	53%	22.9%
LLAMA 3.1 8B BASE ($t = 1.0$)	0.468	36%	24.9%
BARE LLAMA 3.1 8B (ALL $t = 0.7$)	0.462	47%	**28.1%**

A.4. BARE First Stage Ablations - GSM8K

This appendix contains diversity, IR, and downstream performance results for our ablation replacing the first stage of **BARE** with an instruct-tuned model, specifically Llama-3.1-70B-Instruct. We refine using Llama-3.1-70B-Instruct and GPT-4o, and investigate the change in downstream performance compared to standard **BARE** (using Llama-3.1-70B-Base in the first stage). Note that dataset diversity is unchanged compared to direct generation from Llama-3.1-70B-Instruct, that IR improves after refinement, and that downstream performance is consistently worse than standard **BARE**.

Table 12. Average pairwise embedding cosine similarity, IR, and downstream accuracy results on a $n = 500$ randomly selected static subset of GSM8K. A Llama-3.2-1B-Instruct model was fine-tuned for 4 epochs on the generated data ($n = 1000$). The baseline performance of the Llama-3.2-1B-Instruct model on the evaluation set prior to any fine-tuning is reported in the first row.

GENERATION METHOD	AVERAGE EMBEDDING SIMILARITY	IR	DOWNSTREAM ACCURACY
BASELINE PERFORMANCE	–	–	21.8%
LLAMA 3.1 70B INSTRUCT	0.421	51%	22.4%
LLAMA 3.1 70B INSTRUCT SELF-REFINE	0.422	63%	25.4%
GPT-4O REFINING LLAMA 3.1 70B INSTRUCT	0.421	70%	30.8%
BARE LLAMA 3.1 70B	0.305	54%	29.8%
BARE GPT-4O + LLAMA 3.1 70B BASE	0.302	64%	**35.8%**

B. GSM8K Prompt Examples

In this appendix, we provide exact prompts used for the GSM8K domains, representative of those used throughout this work. Examples are formatted for inclusion in the prompts in the "{examples}" fields, with "EXAMPLE START" and "EXAMPLE END" delimiters for the base prompt. **BARE** uses the standard Base Prompt in the base generation step.

Static Few-shot Examples

Example 1

Question: Alice has 20 quarters. She wants to exchange them for nickels and so she goes to the bank. After getting back from the bank, she discovers that 20% of the nickels are iron nickels worth $3 each. What is the total value of her money now?
Answer: A quarter is worth five nickels because .25 / .05 = $\ll .25/.05 = 5 \gg 5$. She gets 100 nickels from the bank because 20 x 5 = $\ll 20 * 5 = 100 \gg 100$. 20 of the nickels are iron nickels because 100 x .20 = $\ll 100 * .20 = 20 \gg 20$. 80 of the nickels are regular because 100 - 20 = $\ll 100 - 20 = 80 \gg 80$. The iron nickels are worth $60 because 20 x 3 = $\ll 20 * 3 = 60 \gg 60$. The regular nickels are worth $4 because 80 x .05 = $\ll 80 * .05 = 4 \gg 4$. Her money is now worth $64 because 60 + 4 = $\ll 60 + 4 = 64 \gg 64$. #### 64

Example 2

Question: A church has 120 members. 40% are adults. The rest are children. How many children more children are there than adults?
Answer: There are 48 adults because 120 x .4 = $\ll 120 * .4 = 48 \gg$ 48. 60% of members are children because 100 - 40 = $\ll 100 - 40 = 60 \gg$ 60. There are 72 children because 120 x .6 = $\ll 120 * .6 = 72 \gg$ 72. There are 24 more children than adults because 72 - 48 = $\ll 72 - 48 = 24 \gg$ 24. #### 24

Example 3

Question: Lisa is looking to attempt a World Record. She has decided to try and match Joey Chestnut's record of eating 75 full hotdogs, buns included, in 10 minutes. Halfway through the time Lisa has eaten 20 hotdogs. How many hotdogs will she have to eat per minute to at least tie Joey Chestnut's record?
Answer: Joey Chestnut ate 75 hotdogs to claim the record and Lisa has eaten 20 hot dogs so far, so she still needs to eat $75 - 20 = \ll 75 - 20 = 55 \gg 55$ hotdogs to tie Joey Chestnut. Lisa has a 10-minute time period to eat the hotdogs and half the time has already passed, which means Lisa has $10/2 = \ll 10/2 = 5 \gg 5$ minutes left until the competition is over. If she needs to eat 55 hotdogs to tie Joey Chestnut and there are 5 minutes left in the competition period, then she needs to eat $55/5 = \ll 55/5 = 11 \gg 11$ hot dogs per minute to have a chance of tying for a win. #### 11

Base Prompt

Here are a few examples of grade school math word problems that require performing a sequence of elementary calculations using basic arithmetic operations. A bright middle school student should be able to solve each problem. The numerical answer is provided at the end of each example after ####.

{examples}

EXAMPLE START

Instruct Few-shot Prompt

Provide an example of a grade school math word problem that requires performing a sequence of elementary calculations using basic arithmetic operations. A bright middle school student should be able to solve each problem. Problems require no concepts beyond the level of early Algebra. You must first specify the question, then provide the very concise reasoning and answer. Provide your example in the following format:

```
Question:  [question]
Answer:  [answer]
```

Provide only the question and answer in the given format. Note how the numerical answer is provided after #### after each brief reasoning for a question. Here are some examples:

{examples}

Now it's your turn. Start your response with the question.

Refine Prompt

Improve the given grade school math word problem. Edit the problem or answer to be more similar in style to the examples, and disambiguate as necessary, in addition to correcting any errors. Do not change the theme of the problem. A bright middle school student should be able to solve each problem. Problems require no concepts beyond the level of early Algebra. Note how the numerical answer is provided after `####` after each brief reasoning for a question. Provide your edited problem in the following format:

```
Question:  [question]
Answer:  [answer]
```

Provide only the question and answer in the given format. Here are some examples of categories and problems on those categories:

{examples}

Now it's your turn. Here is the question and answer for you to edit:
Question:
{question}
Answer:
{answer}

Provide only the improved question and answer in the given format. Do not include any commentary or notes. Start your response with the question.

Sequential Prompt

Generate a new grade school math word problem that requires performing a sequence of elementary calculations using basic arithmetic operations. A bright middle school student should be able to solve each problem. Problems require no concepts beyond the level of early Algebra. Here are the previously generated examples:

{examples}

Your new problem should:

1. Be different from the previous examples
2. Follow the same format and style as prior problems

Note how the numerical answer is provided after `####` after each brief reasoning for a question. Provide only the question and answer in the given format here:

```
Question:  [question]
Answer:  [answer]
```

Start your response with the question.

In One Prompt

Provide {num} examples of problems that might be grade school math word problems that require performing a sequence of elementary calculations using basic arithmetic operations. A bright middle school student should be able to solve each problem. Problems require no concepts beyond the level of early Algebra. You must first specify the question then provide the brief reasoning and answer. Note how the numerical answer is provided after #### after each brief reasoning for a question. Provide your examples in the following format:

```
Question: [question]
Answer: [answer]
```

Here are some examples:

{examples}

Now it's your turn. Generate {num} different problems following this format. Your question should be different in content from the examples. Make sure to only provide only the question and answer. Start each example with the question. Delimit the end of an example with the phrase "END OF EXAMPLE" (all caps) on a new line.

Persona Prompt

{persona_description}

Provide an example of a grade school math word problem that requires performing a sequence of elementary calculations using basic arithmetic operations. A bright middle school student should be able to solve each problem. Problems require no concepts beyond the level of early Algebra. You must first specify the question, then provide the very concise reasoning and answer. Provide your example in the following format:

```
Question: [question]
Answer: [answer]
```

Provide only the question and answer in the given format. Note how the numerical answer is provided after #### after each brief reasoning for a question. Here are some examples:

{examples}

Now it's your turn. Start your response with the question.

Indistinguishability Rate Prompt

System Prompt:

You are an expert at evaluating question and answer pairs for grade school math word problems.

You will be shown $\{k\}$ examples. Each example consists of some context, a question, and an answer. All but one of them is generated from a high quality AI while one of them is of low quality.

Your task is to identify which example (1, 2, ..., $\{k\}$) appears to be of low quality. Consider factors like:

1. Differing natural language patterns and flow
2. Differing question structure, clarity, and difficulty
3. Context and specificity
4. Any subtle artifacts or unnatural patterns

Analyze each example carefully and explain your reasoning. End with 'Answer: [Question Number]' where Question Number is 1, 2, ..., $\{k\}$.

—

User Prompt:

Here are $\{k\}$ examples. One of them is of low quality. Please identify which one:

{questions}

Analyze each example and explain which one you think is of low quality. End with 'Answer: [Question Number]'.