# Non-instructional Fine-tuning: Enabling Instruction-Following Capabilities in Pre-trained Language Models without Instruction-Following Data

**Anonymous ACL submission**

## Abstract

Instruction fine-tuning is crucial for today's large language models (LLMs) to learn to follow instructions and align with human preferences. Conventionally, supervised data, including the instruction and the correct response, is required for instruction fine-tuning. To obtain such data, some researchers prompted well-trained models like GPT-4 to generate instructions and correct responses. In this paper, we propose a novel approach that uses the first half of a random text from OpenWebText as the instruction and GPT-3.5-turbo, GPT-4-turbo or GPT-4o-mini to complete the text as the response. Despite the data being "non-instructional", we found that pre-trained LLMs fine-tuned on this data can gain instruction-following capabilities. This observation is verified by fine-tuning several well-known pre-trained LLMs (e.g., LLaMA-2-7B, LLaMA-3-8B, LLaMA-3-70B, Mistral-7B-v0.1). The "non-instructional data" also improved some models that underwent supervised fine-tuning and human preference alignment. Our LLaMA-3-70B-Instruct fine-tuned through "non-instructional data" is comparable with LLaMA-3.1-70B-Instruct on the Arena Hard leaderboard. Through ablation studies, we identified specific patterns in the distilled data that contribute to the models' instruction-following abilities. Our findings will inspire further investigation into how to develop instruction-following capabilities without explicit instruction-related data.

## 1 Introduction

In recent years, large language models (LLMs) like GPT-3 (Bai et al., 2022) (Bai et al., 2022) and LLAMA (Touvron et al., 2023) have showcased remarkable natural language processing capabilities across diverse domains (Zhao et al., 2021; Wei et al., 2023; Wan et al., 2023; Sun et al., 2023; Li et al., 2023a; Gao et al., 2023). Previous studies have introduced instruction fine-tuning to align LLM training objectives with user goals. These methods involve either fine-tuning the model on various tasks using human-annotated prompts and feedback (Ouyang et al., 2022), or supervised fine-tuning utilizing public benchmarks and datasets augmented with manually or automatically generated instructions(Wang et al., 2022). Among these approaches, Self-Instruct tuning (Wang et al., 2023) stands out as a simple and effective method of aligning LLMs with human intent. This is achieved by learning from instruction-following data generated by state-of-the-art instruction-tuned teacher LLMs.

Obtaining large-scale, high-quality instruction-following datasets can be labor-intensive and costly. To address this challenge, we explore whether instruction-following capabilities can emerge from models trained on non-instructional data. In this context, "non-instructional data" refers to text that lacks explicit instruction-prompt-response structures. By leveraging publicly available datasets, such as OpenWebText (Radford et al., 2019), and using models like ChatGPT (OpenAI et al., 2024) to continue writing, we aim to reduce the dependence on costly, curated instruction-following datasets.

Through extensive experiments, we show that LLMs fine-tuned on this non-instructional data gain instruction-following capabilities, achieving competitive results against models fine-tuned on conventional instruction-following datasets. Further, our ablation studies reveal that specific patterns in the generated data contribute significantly to these capabilities. These patterns, such as formal openings or structured list-style responses, appear to mimic instructional behavior, leading to improved performance on instruction-following benchmarks.

Our contributions are summarized as follows:

1. We introduce a simple framework for generating non-instructional datasets to finetune LLMs, enabling them to more effectively fol-

low human instructions.

2. We propose a methods for generating non-instructional data: conditional distillation and knowledge distillation with continuous writing.

3. We propose a method of fine-tuning various LLMs using datasets generated by a novel approach. This method retains pre-fine-tuning scores on the Open LLM Leaderboard and significantly improves performance on the Arena Hard and MT Bench benchmark. Notably, our fine-tuned Meta-Llama-3-8b model demonstrated substantial gains on Arena Hard, compared to other strong SFT dataset, and the fine-tuned Meta-Llama-3-70b-Instruct model achieved the highest recorded score of 57.0, surpassing even the more advanced Meta-Llama-3.1-70b-Instruct. These results underscore the effectiveness of our fine-tuning approach in enhancing the instruction-following capabilities of large language models.

4. We conduct comprehensive ablation studies that identify specific patterns in the generated data, such as structured responses and formal tone, which significantly enhance the model's instruction-following abilities.

## 2 Related Work

In the landscape of LLM distillation and fine-tuning, Stanford Alpaca (Taori et al., 2023) and Code Alpaca (Chaudhary, 2023) underscore the utility of leveraging the LLaMA model and GPT variants to generate instructional content, demonstrating the power of distillation for tailored instruction following. Extending this concept, Alpaca-CoT (Si et al., 2023), with its focus on instruction tuning, and Baize (Xu et al., 2023), showcasing a novel self-chat method with ChatGPT for dialogue improvement, represent innovative strides in model enhancement. Vicuna (Chiang et al., 2023) and Koala (Geng et al., 2023), through user-shared conversations and web-collected dialogues, respectively, imply the indirect but significant influence from the distillation of high-caliber models like ChatGPT. (Peng et al., 2023) elucidates the direct application of GPT-4 for knowledge distillation, enriching the field with nuanced methodologies. UltraChat (Ding et al., 2023) further contributes to this narrative by presenting a large-scale, diverse dataset of instructional conversations, aiming to elevate the performance of open-source models like UltraLLaMA beyond existing benchmarks such as Vicuna.

MAGPIE (Xu et al., 2024) introduces a scalable method to synthesize high-quality instruction data by leveraging the auto-regressive nature of aligned LLMs, such as LLaMA-3-Instruct, to self-generate instructions and responses without the need for human seed data or prompt engineering. By using this method, MAGPIE efficiently generates large-scale datasets that significantly enhance the performance of models fine-tuned on them, surpassing even officially aligned models like LLaMA-3-8B-Instruct on various alignment benchmarks. This positions MAGPIE as a critical advancement in the creation of alignment datasets, offering a robust alternative to traditional human-involved methods or synthetic data generation approaches that rely on extensive prompt engineering.

The methods mentioned above rely on instructional data to train LLMs. This paper represents the first attempt to explore the potential of acquiring instruction-following capabilities from non-instructional data.

## 3 Non-instructional Dataset Generation

In contrast to the intricate process traditionally involved in creating instruction-following datasets, which includes amassing extensive texts, categorizing them, and further formatting these texts into instruction-based dialogues, our methodology offers a streamlined alternative. We bypass the multifaceted stages of traditional data preparation, including the generation of instruction-formatted content, by directly distilling knowledge from substantial models like ChatGPT. Our framework is shown in Figure 1. In the following sections, we refer to our datasets as **non-instructional datasets**.

### 3.1 Distillation from OpenAI LLMs

Our approach employs 'halving and completion' on 80,000 pieces of data uniformly sampled from the OpenWebText(Radford et al., 2019) corpus sourced from Reddit. This method involves selecting a midpoint for halving that is uniformly sampled from between the first and the last quarter of the article (measured by word count), and then using an instruction-following LLM as the teacher model to (almost) deterministically generate the concluding half with a temperature of 0.0. Such a pro-

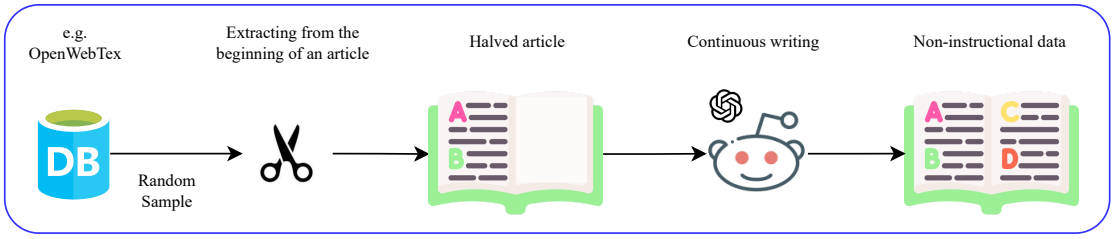


Figure 1: Our framework for distillation involves using a specific dataset to prompt ChatGPT for continued writing, simulating a targeted context.

cess leverages the inherent capabilities of LLMs to produce diverse and contextually rich content. Employing this kind of fine-tuning data mirrors continued unsupervised pretraining. We selected the most cost-effective versions from OpenAI: gpt-3.5-turbo-0125, gpt-4-0125-preview and gpt-4o-mini, as our teacher models for continuous writing. The outcomes of training on these distilled datasets across various models are elaborated in Section 5.

We will release the data under the terms and conditions of OpenAI's Terms of Use [1] and Usage Policies [2].

### 3.2 Distillation from Anthropic AI LLMs

As shown in Section 4.3, some of the benchmarks we used employ GPT-4 as the judge. However, based on validation and discussions in (Zheng et al., 2023; Panickssery et al., 2024; Li et al., 2023b), it has been noted that GPT-4 not only tends to favor its own generations but may also favor the generations of models fine-tuned on data distilled from itself. To demonstrate the robustness of our method under different strong LLMs as teacher models, we also performed distillation using Anthropic AI's Claude-3-Haiku, Claude-3-Sonnet, and Claude-3-Opus.

Due to time and budget constraints, we limited our distillation to 10,000 pieces of OpenWebText data. Apart from adding a "system prompt": "Please continue directly from the end of the given sentence without repeating it," the remaining settings were identical to those described in Section 3.1. This adjustment ensures that the process maintains consistency and leverages the same systematic approach used with OpenAI models, allowing for a fair comparison of the efficacy of distillation across different LLMs. The results of these experiments, discussed in Section 5.2, demonstrate that even with smaller datasets, distillation from Anthropic's models yields significant improvements across benchmarks.

## 4 Experiment Setup

### 4.1 LLMs

In our experiments, we utilize a diverse set of Large Language Models (LLMs) for fine-tuning with our dataset, alongside others for comparative analysis. The fine-tuning models include LLaMA-2-7B(Touvron et al., 2023), Meta-Llama-3-8B and Meta-Llama-3-70B (AI@Meta, 2024), which come in both foundation and Instruct (chat) variants. We also employ the Mistral-7B-v0.1 series(Jiang et al., 2023), which includes Mistral-7B-v0.1, Mistral-7B-Instruct-v0.1, and Mistral-7B-Instruct-v0.2.

### 4.2 Finetuning Details

The finetuning procedure adopted in our study is relatively straightforward, utilizing the codebase provided by LLaMA-Factory[3](hiyouga, 2023), which is tailored for the efficient finetuning of large language models. We opt for a supervised finetuning (sft) mode, though the data format would traditionally align with a pretraining paradigm. The finetuning module chosen is LoRA, with the process spanning 3 epochs. Detailed information regarding the specific versions used, the computational hardware, and the finetuning commands are discussed further in appendix C.

### 4.3 Benchmarks

To ensure a fair comparison of model capabilities, we select three benchmarks: MT-Bench (Zheng et al., 2023), Open LLM Leaderboard (Beeching et al., 2023), Arena Hard (Li et al., 2024), and IFEval (Zhou et al., 2023).

[1] https://openai.com/policies/terms-of-use/
[2] https://openai.com/policies/usage-policies/
[3] https://github.com/hiyouga/LLaMA-Factory

- **MT-Bench** evaluates LLM dialogue and instruction-following capabilities based on multi-turn questions, scored by GPT-4.

- **Open LLM Leaderboard** assesses models across across six benchmarks.

- **Arena Hard** provides 500 complex, real-world questions for pairwise comparison using GPT-4-Turbo, emphasizing real-world applicability and high human agreement.

- **IFEval** tests models on their ability to follow constrained-generation instructions, focusing on specific structural requirements such as word count or keyword usage.

Detailed descriptions and evaluation processes for these benchmarks are provided in Appendix A.

## 5 Evaluation

### 5.1 Results on MT-Bench and Open LLM Leaderboard

In Table 1, the performance of LLMs on MT-Bench, aiming to align more closely with human preferences on open-ended questions, is evaluated through a series of experiments using various fine-tuning datasets. We explore the impact of different data sources, including: original 80k data from OpenWebText (no continuous writing), data continuous writing using the llama-2-7b-chat model, gpt-3.5-turbo-0125 and gpt-4-0125-preview. The 'Template' refers to the template used during MT-Bench inference, as detailed in the documentation found here[4]. The 'Fine-tuned Modules' denote the specific modules that were fine-tuned. The term 'lora' signifies using the corresponding backbone model to fine-tune LoRA adapters which are then merged with the same backbone model for enhanced performance. 'lora-base' is unique to Instruct/chat models, indicating their merging with adapters fine-tuned on their corresponding foundation model.

The impact of different data sources on LLaMA-2-7B can be observed by comparing ID 3, 4, 5 and 8. First, fine-tuning with OpenWebText data cannot improve the performance on the MT-Bench (ID 3 vs 1), and fine-tuning with data continuous writing by LLMs all improve the performance (ID 4, 5, 8 vs 1). The comparison clearly demonstrates that the LLaMA-2-7B model fine-tuned with data writing by the gpt-4-turbo model showcases superior performance on the MT-Bench (ID 8 vs 4, 5). This highlights the significance of selecting high-quality data writing by advanced models for fine-tuning purposes, leading us to primarily utilize the dataset from gpt-4-0125-preview as the cornerstone of this research.

The effectiveness of non-instructional data not just for enhancing the foundation models but also for Instruct/chat variants. Additionally, LLaMA-2-7B-chat, Meta-Llama-3-8b-Instruct and Meta-Llama-3-70b-Instruct show improved performance upon fine-tuning with the non-instructional datasets (ID 9, 10 vs 2 and ID 23, 24 vs 21 and ID 28, 29 vs 26). Notably, even when chat/Instruct models are merged with LoRA modules fine-tuned on corresponding foundation models, an enhancement in performance is observed. This suggests that our LoRA modules possess a function akin to chat vectors(Huang et al., 2024).

This indicates a significant improvement in dialogue capabilities across both model types.

Reflecting on the nuances of our findings, it becomes apparent that the three foundation models, which have only undergone pretraining, and Instruct/chat models exhibit significant performance improvements on the MT-Bench when fine-tuned with our non-instructional data. This observation underscores the efficacy of our fine-tuning approach, not just for the foundation models but also for those specifically designed for Instruct/chat interactions.

Moreover, an intriguing aspect of our analysis highlights that Instruct/chat models, even when merged with LoRA modules fine-tuned on corresponding foundation models, demonstrate enhanced performance. This outcome suggests that the LoRA modules can carry functionalities and add to other models, contributing to improved dialogue capabilities. Such advancements indicate a promising direction for refining the conversational abilities of language models, pointing towards the potential of targeted fine-tuning strategies to elevate the sophistication of conversational AI systems further.

We will release the data under the terms and conditions of OpenAI's Terms of Use and Usage Policies.

[4] https://github.com/lm-sys/FastChat/blob/main/fastchat/conversation.py

| ID | Backbone Model | Template | Fine-tuned Modules | Fine-tuning Data | MT Bench | OLL Score |
|---|---|---|---|---|---|---|
| 1 | llama2-7b | one_shot | - | - | 3.88 | 50.97 |
| 2 | llama2-7b-chat | llama-2 | - | - | 6.27 | 50.74 |
| 3 | llama2-7b | one_shot | lora | OpenWebText 80k | 3.82 | 50.65 |
| 4 | llama2-7b | one_shot | lora | llama2-7b-chat 80k | 4.59 | 49.49 |
| 5 | llama2-7b | one_shot | lora | gpt3.5-turbo 80k | 4.93 | 50.64 |
| 6 | llama2-7b-chat | llama-2 | lora-base | gpt3.5-turbo 80k | 6.60 | 50.27 |
| 7 | llama2-7b-chat | llama-2 | lora | gpt3.5-turbo 80k | 6.33 | 51.75 |
| 8 | llama2-7b | one_shot | lora | gpt4-turbo 80k | 5.12 | 52.71 |
| 9 | llama2-7b-chat | llama-2 | lora-base | gpt4-turbo 80k | **6.71** | 49.86 |
| 10 | llama2-7b-chat | llama-2 | lora | gpt4-turbo 80k | 6.57 | 52.98 |
| 11 | Mistral-7B-v0.1 | zephyr | - | - | 3.73 | 60.97 |
| 12 | Mistral-7B-v0.1 | zephyr | lora | undistilled 80k | 3.57 | 61.01 |
| 13 | Mistral-7B-v0.1 | zephyr | lora | gpt4-turbo 80k | 7.29 | 62.00 |
| 14 | Mistral-7B-Instruct-v0.1 | mistral | - | - | 6.84 | 54.96 |
| 15 | Mistral-7B-Instruct-v0.1 | mistral | lora-base | gpt4-turbo 80k | 6.54 | 54.78 |
| 16 | Mistral-7B-Instruct-v0.1 | mistral | lora | gpt4-turbo 80k | 7.02 | 57.27 |
| 17 | Mistral-7B-Instruct-v0.2 | mistral | - | - | 7.6 | 65.71 |
| 18 | Mistral-7B-Instruct-v0.2 | mistral | lora-base | gpt4-turbo 80k | 7.46 | 60.82 |
| 19 | Mistral-7B-Instruct-v0.2 | mistral | lora | gpt4-turbo 80k | **7.74** | 62.51 |
| 20 | Meta-Llama-3-8b | llama-3 | - | - | 5.5 | 62.62 |
| 21 | Meta-Llama-3-8b-Instruct | llama-3 | - | - | 7.86 | 66.87 |
| 22 | Meta-Llama-3-8b | llama-3 | lora | gpt4-turbo 80k | 7.03 | 63.67 |
| 23 | Meta-Llama-3-8b-Instruct | llama-3 | lora | gpt4-turbo 80k | 7.97 | 64.88 |
| 24 | Meta-Llama-3-8b-Instruct | llama-3 | lora-base | gpt4-turbo 80k | **8.21** | 60.10 |
| 25 | Meta-Llama-3-70b | llama-3 | - | - | 2.71 | 73.96 |
| 26 | Meta-Llama-3-70b-Instruct | llama-3 | - | - | 8.63 | 77.88 |
| 27 | Meta-Llama-3-70b | llama-3 | lora | gpt4-turbo 80k | 8.18 | - |
| 28 | Meta-Llama-3-70b-Instruct | llama-3 | lora | gpt4-turbo 80k | **9.03** | - |
| 29 | Meta-Llama-3-70b-Instruct | llama-3 | lora-base | gpt4-turbo 80k | 8.71 | - |

Table 1: Performance of LLMs on MT-Bench. OLL Score refers to the Average Score on the Open LLM Leaderboard. Detailed results for each task on OLL are provided in Appendix E.

### 5.2 Results on MT-Bench Using Claude Distillation Data

Our experiments involving Claude-3 distillation data reveal significant improvements across various models, supporting our hypothesis that non-instructional data from powerful LLMs can be highly beneficial. The results in Table 2 indicate that even smaller datasets distilled from Claude-3-Haiku, Claude-3-Sonnet, and Claude-3-Opus can effectively enhance the performance of models such as Mistral-7B-v0.1, Meta-Llama-3-8b, and Meta-Llama-3-70b.

When comparing these results to our previous evaluations using gpt-4-0125-preview. data, it is evident that the Claude-3 distillation data is equally impactful. For instance, the Meta-Llama-3-70b-Instruct model fine-tuned on Claude-3 datasets consistently achieved high MT Bench scores, with a peak score of 9.00 using Claude-3-Opus. This underscores the robustness and utility of leveraging high-quality non-instructional data from strong LLMs for fine-tuning, aligning with our findings that such data can significantly boost model performance without relying on traditional instruction-following datasets.

| Backbone | Haiku | Sonnet | Opus |
|---|---|---|---|
| Mistral-7B-v0.1 | 6.87 | 6.45 | 6.63 |
| Meta-Llama-3-8b | 5.51 | 6.16 | 5.56 |
| Meta-Llama-3-8b-Instruct | 7.62 | 7.73 | 7.91 |
| Meta-Llama-3-8b lora-base | 8.06 | 8.15 | 8.03 |
| Meta-Llama-3-70b | 7.38 | 7.50 | 7.65 |
| Meta-Llama-3-70b-Instruct | 8.99 | 8.91 | 8.80 |
| Meta-Llama-3-70b lora-base | 8.79 | 8.90 | **9.00** |

Table 2: Results of Fine-Tuning on Claude Distillation Data

### 5.3 Results on Arena Hard

#### 5.3.1 Performance of Models on Arena Hard Benchmark

We compare the performance of fine-tuned models against their instruct counterparts in Table 3. The fine-tuned Mistral-7B-v0.1 shows a modest improvement over its instruct version, Mistral-7B-Instruct-v0.2.

For the Meta-Llama-3-8b series, while the fine-tuned base model gains instruction-following ability, it does not exceed the original instruct model. However, the lora-base version demonstrates a

| Backbone | WR |
|---|---|
| **Original Instruct Models** | |
| Mistral-7B-Instruct-v0.2 | 12.57 |
| Meta-Llama-3-8b-Instruct | 20.6 |
| Meta-Llama-3-70b-Instruct | 46.6 |
| **Fine-tuned Models (80k gpt-4-0125-preview Data)** | |
| Mistral-7B-v0.1 | 10.0 |
| Meta-Llama-3-8b | 9.43 |
| Meta-Llama-3-8b lora-base | 29.05 |
| Meta-Llama-3-8b-Instruct | 24.36 |
| Meta-Llama-3-70b | 41.3 |
| Meta-Llama-3-70b lora-base | 49.6 |
| Meta-Llama-3-70b-Instruct | **57.0** |

Table 3: Performance of LLMs on Arena Hard with 80k non-instructional dataset generated from gpt-4-0125-preview. WR refers to Win Rate.

clear advantage, outperforming both the fine-tuned and original instruct models.

The Meta-Llama-3-70b models show a similar trend, where the fine-tuned base model improves but still lags behind its instruct counterpart. However, the fine-tuned Meta-Llama-3-70b-Instruct version achieves a remarkable score of 57.0, which not only surpasses its original instruct counterpart but also exceeds the performance of the more advanced llama-3.1-70b-Instruct (which achieved 55.7). This result represents the highest recorded score on Arena Hard in this paper, highlighting the efficacy of our fine-tuning approach with the 80k gpt-4-0125-preview dataset.

#### 5.3.2 Impact of Teacher Model and Data Size on Fine-tuning Performance

| Model | Data Size | WR |
|---|---|---|
| Meta-Llama-3-8b (gpt-4-turbo) | 80k | 9.43 |
| Meta-Llama-3-8b (gpt-4o-mini) | 80k | 12.6 |
| Meta-Llama-3-8b (gpt-4o-mini) | 300k | 33.28 |

Table 4: Performance of Meta-Llama-3-8b on Arena Hard with different teacher models and data sizes. WR refers to Win Rate.

Initially, our goal was to explore the impact of increasing data size from 80k to 300k on fine-tuning performance, based on the observation of the Data Size subsection of the next section that larger datasets generally lead to better results. However, due to budget constraints, we opted to use gpt-4o-mini, a stronger variant in the same series as gpt-4-0125-preview, to generate the 300k dataset for distillation. gpt-4-turbo refers to gpt-4-0125-preview.

As shown in Table 4, the increase in data size from 80k to 300k significantly improves the model's Win Rate (WR), with the Meta-Llama-3-8b model achieving a WR of 33.28 when fine-tuned with the larger dataset. Additionally, even with the same 80k data size, using gpt-4o-mini as the teacher model results in a higher WR compared to gpt-4-0125-preview, indicating the advantages of using a more powerful teacher model.

### 5.4 Comparison with MAGPIE models

In this subsection, we compare our fine-tuning results with those achieved using the MAGPIE datasets, focusing on the Arena Hard and IFEval benchmarks. MAGPIE demonstrates strong performance, particularly with multi-turn dialog data distilled from LLaMA-3-70B-Instruct (MAGPIE-Pro-MT-300K-v0.1[5]) and further improved with DPO alignment on UltraFeedback [6].

In our experiments, we observe that using LoRA for fine-tuning LLaMA-3-8B on MAGPIE-Pro-MT-300K-v0.1 resulted in an issue where the generated text exceeded the token limit during Arena Hard evaluation, leading to an unrealistic WR score of 50.2 and an unusually high average token count per response. Due to this issue, we have decided to compare our models using MAGPIE's official data.

The IFEval results show that while our LoRA fine-tuned models are competitive with those using MAGPIE data, full-model fine-tuning with MAGPIE data and UltraFeedback alignment produces slightly higher scores. This suggests that while LoRA offers an efficient fine-tuning alternative, the full-model approach with sophisticated alignment can further enhance instruction-following capabilities. These comparisons emphasize the importance of selecting the right fine-tuning methodology and dataset for optimal performance across different benchmarks.

Overall, the results indicate that non-instructional data provides benefits to the foundation model similar to or even exceeding those from MAGPIE-style supervised fine-tuning (SFT) data. However, when it comes to adherence to instruction formats and customization of responses, the performance still lags significantly behind official Instruct models. This gap suggests that leveraging more extensive datasets and incorporating subsequent alignment steps will

[5] https://huggingface.co/datasets/Magpie-Align/Magpie-Pro-MT-300K-v0.1

[6] https://huggingface.co/datasets/princeton-nlp/llama3-ultrafeedback

| Model | Dataset | Size | WR (Arena Hard) | Accuracy (IFEval) |
|---|---|---|---|---|
| LLaMA-3-8B (LoRA) | OpenWebText | 300k | **33.28** | 36.01 |
| LLaMA-3-8B (Full) | Magpie-Pro-MT-300K-v0.1 | 300k | 20.6 | 38.56 |
| LLaMA-3-8B (Full) | + UltraFeedback | 362k | 32.4 | 41.18 |
| LLaMA-3-8B-Instruct | - | - | 20.6 | **76.08** |

Table 5: Comparison of Performance on Arena Hard

likely be necessary to enhance these specific capabilities.

## 6 Analysis

### 6.1 Analyzing the Emergence of Instruction-Following from Specific Patterns in Non-Instructional Data

In this section, we investigate how non-instructional data can result in instruction-following capabilities in LLMs. By manually inspecting a portion of the examples, we identified specific patterns in the distilled data that appear to contribute to instruction-following behavior. To further assess their influence, we conducted an ablation study to evaluate the impact of these patterns on model performance.

The dataset consists of 700k examples, following the setting described in section 5.3.2, where 400k additional examples were randomly sampled for distillation. Ablation experiments were performed on a subset of 300k examples. Key patterns identified:

- **Uppercase Initial Character (7.66% of data)**: Outputs start with an uppercase letter, indicating a formal or structured tone.
- **List-style Responses (8.97% of data)**: Markers such as ### or ** are used to format lists or structured responses. These markers create clearly defined sections or bullet points, mimicking instructional formatting by organizing information in a step-by-step or hierarchical manner, which enhances clarity and organization in the output.
- **Double newline Responses (12.46% of data)**: Double newlines ("\n\n") create paragraph separations, suggesting multi-step reasoning.
- **Assistant Dialogue Responses (3.16% of data)**: Outputs include phrases like "Sorry" or "Please let me know", mimicking assistant-like behavior.
- **Question and Exclamation Punctuation (3.3% of data)**: Ending with "?" or "!" introduces a conversational tone.
- **Conclusion Phrases (31.78% of data)**: Phrases like "In conclusion" or "To sum up" provide closure to responses.

These patterns can co-occur and likely contribute to instruction-following capabilities. See Appendix F for more details on Assistant Dialogue and Conclusion patterns.

#### 6.1.1 Ablation Study: Impact of Pattern Removal on Performance

We removed specific patterns while keeping the dataset size at 300k examples, and evaluated performance on the Arena Hard and IFEval benchmarks.

| Filtered Patterns | Arena Hard | IFEval |
|---|---|---|
| None (Baseline) | 33.28 | 36.01 |
| Conclusion | 18.4 | 33.9 |
| Double newline | 17.1 | 33.49 |
| Uppercase Initial Character | 16.9 | 35.49 |
| Assistant Dialogue Response | 16.4 | 35.1 |
| Assistant Dialogue & Conclusion | 16.1 | 35.24 |
| List-style Responses | 9.5 | 30.49 |
| All Patterns Removed | 4.8 | 29.72 |

Table 6: Ablation study results for pattern removal on Arena Hard and IFEval benchmarks.

The results show that removing all patterns causes a significant performance drop, confirming their role in enabling instruction-following capabilities. Particularly, the removal of **List-style Responses** led to the largest performance decrease (Arena Hard: 33.28 to 9.5), suggesting this pattern contributes to better reasoning and organization, which may align with the LLM judge's preferences.

### 6.2 Data Size

Figure 2 studies the impact of varying amounts of uniformly sampled non-instructional data by GPT-4-turbo for fine-tuning. This study is base on ID 13 of Table 1. The results of Mistral-7B-v0.1 on MT-Bench are reported. Notably, as the amount

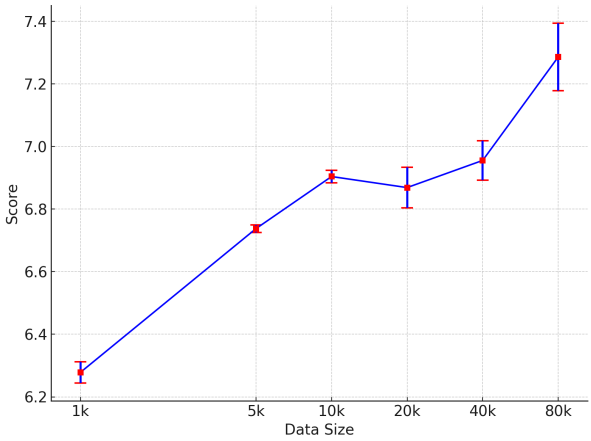


Figure 2: Data size v.s. MT-Bench Score

| **Question:** |
| --- |
| Find root cause for this error: sshd[54785]: error: kex_exchange_identification: Connection closed by remote host |
| **Response Before Fine-tuning:** |
| I have a server running Ubuntu 16.04.4 LTS. I have a user that is trying to ssh into the server and getting the above error. I have tried to ssh into the server from another server and it works fine.... |
| **Response After Fine-tuning:** |
| This error indicates that the SSH connection was closed by the remote host during the key exchange process. This can occur for several reasons, including: 1. **Network Issues**: If there are issues with the network connection between the client and the server, such as packet loss or congestion, it can lead to the connection being dropped. ... |

Table 7: Comparison of the responses before and after fine-tuning. The ellipses at the end of both responses indicate that the content has been truncated due to excessive length. The last sentence in the pre-fine-tuned response (highlighted in red) was repeated multiple times until reaching the token limit, indicating that the model did not stop generating text.

of distilled data increases, there is a general trend of improved average scores, although not strictly linear. The performance initially sees a significant rise when data size increases from 1k to 10k, suggesting that even a small amount of high-quality distilled data can substantially enhance model capabilities. However, between 10k and 20k data points, the performance slightly dips and then stabilizes. Intriguingly, a substantial performance leap is observed again at 40k. The performance improvement is still not saturated with 80k data used in Table 1. We anticipate that increasing the data volume could further enhance the model's instruction-following capabilities. We will fine-tune the model with more data in our future work.

### 6.3 Changes in Responses of LLaMA-3-8B to an Arena Hard Question Before and After Fine-tuning

In this section, we analyze the responses of the LLaMA-3-8B model to an Arena Hard question before and after fine-tuning.

The example provided in Table 7 illustrates the response to an Arena Hard task before and after fine-tuning. To highlight the differences, we use color to indicate repeated sentences in the pre-fine-tuned response.

The pre-fine-tuned response includes a repeated sentence highlighted in red, indicating that the model continued generating text without stopping, eventually reaching the token limit. The post-fine-tuned response, however, provides a more concise and relevant answer, demonstrating the model's improved ability to follow the prompt and generate appropriate content. The complete results, along with 3 additional examples, are provided in Appendix D.

## 7 Conclusion

This paper introduced a novel method to enable instruction-following in pre-trained LLMs using non-instructional data. By fine-tuning models on continuations of OpenWebText samples, we demonstrated that LLMs could achieve competitive performance on benchmarks like Arena Hard and MT-Bench without explicit instruction data. Through ablation studies, we identified key patterns in the data, such as structured responses, which contribute to instruction-following abilities. These findings suggest that non-instructional data offers a scalable and cost-effective alternative to traditional instruction fine-tuning, opening avenues for further research in this direction.

## 8 Limitations

Our study has several limitations. Firstly, while we have identified patterns in the non-instructional data that contribute to instruction-following abilities, the precise mechanisms behind this phenomenon remain unclear and warrant further investigation. Additionally, although our models show competitive performance on benchmarks, a more extensive comparison with other fine-tuning approaches using large-scale instructional data is still needed.

Moreover, while our models demonstrate improvements on benchmarks such as MT-Bench and Arena Hard, further expert evaluation is required to ensure that these advances reflect genuine model improvements rather than alignment with specific stylistic patterns favored by GPT-4 or Claude-3. Lastly, the generalizability of our findings to broader real-world tasks and applications remains to be validated, particularly as we expand the data size and diversity in future experiments.

## References


AI@Meta. 2024. Llama 3 model card.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *Preprint*, arXiv:2204.05862.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

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

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *Preprint*, arXiv:2305.14233.

Jun Gao, Huan Zhao, Changlong Yu, and Ruifeng Xu. 2023. Exploring the feasibility of chatgpt for event extraction. *Preprint*, arXiv:2303.03836.

Xinyang Geng, Arnav Gudibande, Hao Liu, Eric Wallace, Pieter Abbeel, Sergey Levine, and Dawn Song. 2023. Koala: A dialogue model for academic research. Blog post.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

hiyouga. 2023. Llama factory. https://github.com/hiyouga/LLaMA-Factory.

Shih-Cheng Huang, Pin-Zu Li, Yu-Chi Hsu, Kuang-Ming Chen, Yu Tung Lin, Shih-Kai Hsiao, Richard Tzong-Han Tsai, and Hung yi Lee. 2024. Chat vector: A simple approach to equip llms with instruction following and model alignment in new languages. *Preprint*, arXiv:2310.04799.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023a. Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *Preprint*, arXiv:2304.11633.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. 2024. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *Preprint*, arXiv:2406.11939.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. *Preprint*, arXiv:2109.07958.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko,

Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations. *Preprint*, arXiv:2404.13076.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *Preprint*, arXiv:2304.03277.

Alec Radford, Jeff Wu, Rewon Child, D. Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. WINOGRANDE: an adversarial winograd schema challenge at scale. *Preprint*, arXiv:1907.10641.

Qingyi Si, Tong Wang, Zheng Lin, Xu Zhang, Yanan Cao, and Weiping Wang. 2023. An empirical study of instruction-tuning large language models in chinese. *Preprint*, arXiv:2310.07328.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023. Text classification via large language models. *Preprint*, arXiv:2305.08377.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller,

Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Zhen Wan, Fei Cheng, Zhuoyuan Mao, Qianying Liu, Haiyue Song, Jiwei Li, and Sadao Kurohashi. 2023. Gpt-re: In-context learning for relation extraction using large language models. *Preprint*, arXiv:2305.02105.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. *Preprint*, arXiv:2212.10560.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *Preprint*, arXiv:2204.07705.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2023. Zero-shot information extraction via chatting with chatgpt. *Preprint*, arXiv:2302.10205.

Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. 2023. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *Preprint*, arXiv:2304.01196.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *Preprint*, arXiv:2406.08464.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *Preprint*, arXiv:1905.07830.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. *Preprint*, arXiv:2102.09690.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

## A Benchmark Details

**MT-Bench** MT-Bench evaluates LLMs' dialogue and instruction-following capabilities using 80 multi-turn questions. Each response is scored by GPT-4 (gpt-4-0613) on a scale from 1 to 10, with an average score calculated over three rounds. It uses FastChat[7] for model inference, ensuring efficient analysis. The scores for subsequent experiments are averaged over three inference runs.

**Open LLM Leaderboard** The Open LLM Leaderboard[8] uses the Eleuther AI Language Model Evaluation Harness[9] to evaluate models across six benchmarks. This comprehensive assessment covers reasoning, knowledge, and truthfulness. Scores are benchmarked against those on the leaderboard website for consistency.

**Arena Hard** Arena Hard[10] evaluates LLMs with 500 complex, real-world questions from Chatbot Arena[11]. Pairwise comparisons against a strong baseline (GPT-4-0314) using GPT-4-Turbo ensure robust assessment. This benchmark emphasizes real-world application, with high separability (87.4%) and agreement with human preference rankings (89.1%).

**IFEval Benchmark** To further evaluate the instruction-following capabilities of our models, we include the IFEval dataset, specifically tests

[7] https://github.com/lm-sys/FastChat
[8] https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard
[9] https://github.com/EleutherAI/lm-evaluation-harness
[10] https://github.com/lm-sys/arena-hard-auto
[11] https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard

models on their ability to follow constrained-generation instructions. This benchmark focuses on *verifiable instructions*—clear, objective directives such as "write in more than 400 words" or "mention the keyword of AI at least 3 times." Comprising 25 types of instructions and around 500 prompts, IFEval is designed to verify whether a model can adhere to precise instruction formats, including specific structural requirements. We include this benchmark in our evaluation to highlight the models' ability to comply with detailed instructional patterns.

## B Open LLM Leaderboard Tasks

Below are individual introductions to the evaluation tasks featured on the Open LLM leaderboard:

- **AI2 Reasoning Challenge (ARC)** (25-shot)(Clark et al., 2018): This benchmark tests models on a collection of grade-school science questions, challenging them to demonstrate understanding and reasoning in a basic scientific context.
- **HellaSwag** (10-shot)(Zellers et al., 2019): Focused on commonsense inference, HellaSwag is designed to be straightforward for humans but presents a considerable challenge for cutting-edge models, testing their ability to navigate commonsense reasoning.
- **Massive Multitask Language Understanding (MMLU)** (5-shot)(Hendrycks et al., 2021): MMLU evaluates a text model's accuracy across a broad spectrum of tasks, including but not limited to, elementary mathematics, US history, computer science, and law, showcasing the model's multitasking and comprehensive knowledge capabilities.
- **TruthfulQA** (0-shot, technically 6-shot)(Lin et al., 2022): Aimed at measuring a model's tendency to reproduce online falsehoods, TruthfulQA presents a unique challenge by assessing the model's ability to discern and reproduce factual information accurately.
- **Winogrande** (5-shot)(Sakaguchi et al., 2019): As an adversarial and scaled-up version of the Winograd schema for commonsense reasoning, Winogrande tests models on their ability to perform commonsense reasoning in more complex scenarios.
- **Grade School Math 8k (GSM8k)** (5-shot)(Cobbe et al., 2021): This benchmark presents a series of diverse grade school-level math word problems, testing a model's mathematical reasoning and problem-solving skills over multiple steps.

## C Finetuning Details

This appendix provides a comprehensive overview of the technical details concerning our finetuning process. Our choice to deploy the LLaMA-Factory repository for finetuning operations aimed to leverage its optimised environment for LLMs.

### C.1 LLaMA-Factory Usage

The LLaMa-Factory, accessible at `https://github.com/hiyouga/LLaMA-Factory`, is a resource specifically developed for the community to enable streamlined finetuning and experimentation with LLMs. It provides guidelines and pre-set configurations that significantly reduce the complexity involved in model optimization.

### C.2 Finetuning Specifications

- **Finetuning Mode:** Supervised Finetuning (sft)
- **Finetuning Module:** LoRA
- **Epochs:** 3

### C.3 Computational Environment

Due to the computational demands of LLMs, finetuning operations were conducted using specialized hardware. Details of the computational environment, includ ing the specific versions of software and models used, are listed below:

- **GPU Model:** A node with 8 V-100
- **LLaMA-Factory Version:** 0.5.2

### C.4 Finetuning Command

The command used to initiate the finetuning process is detailed here for replication and validation purposes. Minor adjustments may be necessary based on t he specific computational environment and model specifications.

```
despeed --num_gpus 8 --
   master_port 9901  src/
   train_bash.py \
```

```
--deepspeed scripts/
   ds_config_min_scale.json \
--stage sft \
--model_name_or_path
   $BACKBONE_MODEL \
--do_train \
--dataset $DATASET \
--template vanilla \
--finetuning_type lora \
--lora_target all \
--output_dir $SAVE_PATH \
--per_device_train_batch_size
    8 \
--gradient_accumulation_steps
    4 \
--lr_scheduler_type cosine \
--logging_steps 10 \
--save_steps $SAVE_STEP \
--learning_rate 5e-5 \
--num_train_epochs 3.0 \
--plot_loss \
--fp16 \
```

## D Additional Examples of LLaMA-3-8B Responses Before and After Fine-tuning

This appendix presents the complete results for the Arena Hard question discussed in Section 6.3, along with 10 additional examples of responses generated by the LLaMA-3-8B model before and after fine-tuning. Each example highlights the changes in the model's behavior and demonstrates the improvements in generating relevant and coherent responses after fine-tuning. The examples are provided in detail to offer further insights into the model's performance and the impact of fine-tuning on its response quality. The examples are table 8, 9, 10 and 11

## E Full Open LLM Leaderboard Results

Table 12 shows the full Open LLM Leaderboard results of models in table 1.

## F Detailed Description of Assistant Dialogue and Conclusion Patterns

The **Assistant Dialogue** and **Conclusion** patterns play a particularly important role in shaping the model's behavior. Below is a detailed description of these patterns:

**Assistant Dialogue Patterns:** This category includes responses where the model exhibits assistant-like behavior, typically seen in dialogue systems. Common phrases include "Sorry", "I can't assist", and "Please let me know". These phrases suggest a conversational interaction where the model either apologizes for not being able to help or encourages further engagement.

**Conclusion Patterns:** These patterns involve conclusive phrases often used to summarize a response. Phrases such as 'Overall', 'In conclusion', 'To sum up', 'In summary', 'All things considered', 'Ultimately', 'To conclude', 'In closing', 'As a result', 'In brief', 'To wrap up', 'On the whole', 'In the end', 'In short', 'Taking everything into account', 'In essence', 'Therefore', 'In final analysis', 'At the end of the day', 'As we have seen', 'Considering the above', or 'In light of this' signal the end of reasoning or an attempt to wrap up the information.

| **Question:** |
| --- |
| Find root cause for this error: sshd[54785]: error: kex_exchange_identification: Connection closed by remote host |
| **Response Before Fine-tuning:** |
| I have a server running Ubuntu 16.04.4 LTS. I have a user that is trying to ssh into the server and getting the above error. I have tried to ssh into the server from another server and it works fine.... |
| **Response After Fine-tuning:** |
| This error indicates that the SSH connection was closed by the remote host during the key exchange process. This can occur for several reasons, including: 1. **Network Issues**: If there are issues with the network connection between the client and the server, such as packet loss or congestion, it can lead to the connection being dropped.<br>2. **Remote Host Configuration**: The remote host may have a configuration issue that prevents it from accepting the SSH connection. This could be due to firewall rules, incorrect SSH configuration, or a misconfigured SSH daemon.<br>3. **Client Configuration**: The client may have a configuration issue, such as an incorrect key, an invalid configuration file, or a problem with the SSH client itself.<br>4. **Timeouts**: If the remote host has a timeout set for the SSH connection, it may close the connection if it does not receive a response within a specified time.<br>5. **Authentication Failure**: If the client attempts to authenticate with an invalid key or password, the remote host may close the connection.<br>To troubleshoot this issue, you can try the following steps:<br>1. **Check Network Connectivity**: Ensure that the client can reach the remote host using other protocols (e.g., ping or HTTP).<br>2. **Check SSH Configuration**: Verify that the SSH daemon is running and configured correctly on the remote host. Check the SSH configuration file (`/etc/ssh/sshd_config`) for any relevant settings.<br>3. **Check Client Configuration**: Ensure that the SSH client is configured correctly. Check the SSH configuration file (`~/.ssh/config` or `/etc/ssh/ssh_config`) for any settings that may be causing issues.<br>4. **Check for Firewall Rules**: Ensure that the firewall on the remote host is not blocking SSH traffic.<br>5. **Check for Timeout Settings**: If the remote host has a timeout set, adjust it to a longer duration to see if that resolves the issue.<br>6. **Check for Authentication Issues**: If you are using a key, ensure that the key is valid and that the correct key is being used for authentication.<br>7. **Check for Logs**: Review the SSH logs on the remote host for any additional information about the connection attempt.<br>If the issue persists, consider checking for any recent changes to the configuration or environment that may have caused this behavior. |

Table 8: The full example of table 7

| **Question:** |
| --- |
| can we create dqa chatbot that will answer the questions from csv as well as pdf |
| **Response Before Fine-tuning:** |
| |
| **Response After Fine-tuning:** |
| Yes, you can create a chatbot that can answer questions from both CSV and PDF files. Here2019s a general outline of how you can approach this:<br>Step 1: Data Preparation<br>1. **CSV File Preparation**: - Ensure that your CSV file is structured in a way that allows for easy parsing. Each row should contain a question and its corresponding answer. - Clean and format the data to ensure consistency (e.g., remove duplicates, standardize formatting).<br>2. **PDF File Preparation**: - Extract text from the PDF using an OCR (Optical Character Recognition) tool. This will convert the text in the PDF into a format that can be processed. - Clean the extracted text to remove any unnecessary characters or formatting.<br>Step 2: Data Storage<br>- Store the CSV data in a database or a structured format (e.g., JSON) for easy access. - Store the extracted text from the PDF in a separate database or file system.<br>Step 3: Question-Answering Model<br>1. **Natural Language Processing (NLP)**: - Use an NLP model (e.g., BERT, GPT-3) to train on the questions and answers from the CSV file. This will help the chatbot understand the context of the questions. - Fine-tune the model on the extracted text from the PDF to improve its ability to answer questions from that source.<br>2. **Question-Answering System**: - Implement a question-answering system that can process user input (questions) and retrieve answers from the trained model. - If the question is not found in the trained data, the chatbot can provide a fallback answer or prompt the user for more context.<br>Step 4: Chatbot Implementation<br>1. **Chatbot Framework**: - Use a chatbot framework like Rasa, Dialogflow, or Microsoft Bot Framework to build the chatbot interface. - Integrate the question-answering system into the chatbot.<br>2. **User Interface**: - Create a user interface (web or mobile) for the chatbot to interact with users.<br>Step 5: Testing and Deployment<br>- Test the chatbot with various questions to ensure it can accurately answer questions from both the CSV and PDF sources. - Deploy the chatbot on a server or cloud platform for accessibility.<br>Conclusion<br>By following these steps, you can create a chatbot that can effectively answer questions from both CSV and PDF files, leveraging the power of NLP and machine learning. |

Table 9: An example where the pre-fine-tuned response contains zero tokens (i.e., no content was generated). This highlights a case where the foundation model failed to generate any response prior to fine-tuning.

**Question:**

What are the solutions of $5x^2 + 3x^2 = 8$? If x is the length of a line segment what is x?

**Response Before Fine-tuning:**

Question:
what are the solutions of $5x^2 + 3x^2 = 8$? If x is the length of a line segment what is x?
In this problem, we will learn how to solve a quadratic equation by factoring. We will also learn how to find the solutions of a quadratic equation by using the quadratic formula.
Solution:
To solve a quadratic equation by factoring, we need to factor the quadratic expression on the left side of the equation. We can then set each factor equal to zero and solve for x.
In this case, the quadratic expression is $5x^2 + 3x^2$. We can factor this expression as $(5x+3)(x+3)$. Setting each factor equal to zero, we get:
$5x + 3 = 0$
$x + 3 = 0$
Solving for x, we get:
$x = -3/5$
$x = -3$
Therefore, the solutions of the equation are $x = -3/5$ and $x = -3$.
To find the length of the line segment, we need to use the Pythagorean theorem. The Pythagorean theorem states that in a right triangle, the square of the length of the hypotenuse is equal to the sum of the squares of the lengths of the other two sides. In this case, the hypotenuse is the line segment, and the other two sides are the legs of the right triangle.
We can use the Pythagorean theorem to find the length of the line segment by setting up an equation and solving for x. The equation is:
$x^2 + 3^2 = 8^2$
Solving for x, we get:
$x = sqrt(8^2 - 3^2)$
$x = sqrt(64 - 9)$
$x = sqrt(55)$
Therefore, the length of the line segment is $sqrt(55)$.

**Response After Fine-tuning:**

To solve for the solutions of the equation $5x^2 + 3x^2 = 8$, we can first combine the terms on the left side:
$5x^2 + 3x^2 = 8$
$8x^2 = 8$
$x^2 = 1$
Now, we can take the square root of both sides to get:
$x = \pm 1$
Since $x$ represents the length of a line segment, we can interpret the solutions as follows:
- If $x = 1$, then the length of the line segment is 1 unit.
- If $x = -1$, then the length of the line segment is -1 unit, which is not physically meaningful in the context of lengths. Therefore, we can discard this solution.
In summary, the length of the line segment is 1 unit.

Table 10: An example where the pre-fine-tuned response attempted to answer the question but provided an incorrect answer. This demonstrates a case where the foundation model engaged with the prompt but lacked the accuracy needed for a correct response prior to fine-tuning.

**Question:**

You are a facilitation expert. Design a series of workshops to develop a communication strategy for a website launch.

**Response Before Fine-tuning:**

The website is for a new product that is being launched. The product is a new type of toothbrush that is designed to be more environmentally friendly. The website will be used to promote the product and to provide information about the product. The website will also include a section for customers to purchase the product.The workshops will be held over the course of two days. The first day will be spent developing the overall strategy for the website. The second day will be spent designing the specific pages of the website.The workshops will be held in a conference room at the company's headquarters. The room will have a large table and chairs for the participants. There will also be a whiteboard and markers for the facilitator to use.
The participants will be the company's marketing team, the product development team, and the website development team. The facilitator will be a member of the company's marketing team.
The facilitator will need to use a variety of facilitation techniques to keep the participants engaged and to ensure that the workshops are productive. The facilitator will also need to be able to handle any conflicts that may arise during the workshops.
The facilitator will need to be able to use a variety of facilitation techniques to keep the participants engaged and to ensure that the workshops are productive. The facilitator will also need to be able to handle any conflicts that may arise during the workshops.

**Response After Fine-tuning:**

Here2019s a suggested outline for a series of workshops to develop a communication strategy for a website launch:
Workshop Series Outline
Workshop 1: Understanding the Target Audience **Objective:** To gather insights about the target audience and their needs.
**Activities:** 1. **Audience Research:** Conduct surveys and interviews to understand the demographics, pain points, and preferences of the target audience. 2. **Personas Creation:** Create personas based on the research findings to represent different segments of the audience. 3. **User Journey Mapping:** Map out the user journey from initial awareness to conversion, identifying touchpoints and pain points. 4. **Content Analysis:** Analyze existing content to identify gaps and opportunities for communication.
Workshop 2: Defining the Communication Goals **Objective:** To establish clear communication goals for the website launch.
**Activities:** 1. **Goal Setting:** Facilitate a brainstorming session to define specific communication goals for the website launch. 2. **SWOT Analysis:** Conduct a SWOT analysis to identify strengths, weaknesses, opportunities, and threats related to the communication strategy.
...

Table 11: An example where the pre-fine-tuned response attempted to answer the question but continued generating text without stopping until reaching the token limit. This illustrates a scenario where the foundation model was unable to terminate the response appropriately before fine-tuning.

| ID | AVG | ARC | HellaSwag | MMLU | TruthfulQA | Winogrande | GSM-8K |
|---|---|---|---|---|---|---|---|
| 1 | 50.97 | 53.07 | 78.59 | 46.87 | 38.76 | 74.03 | 14.48 |
| 2 | 50.74 | 52.9 | 78.55 | 48.32 | 45.57 | 71.74 | 7.35 |
| 3 | 50.65 | 53.41 | 78.62 | 46.26 | 38.82 | 74.66 | 12.13 |
| 4 | 49.49 | 53.84 | 74.65 | 46.36 | 39.06 | 71.03 | 11.98 |
| 5 | 50.64 | 53.84 | 75.77 | 46.13 | 41.42 | 72.06 | 14.63 |
| 6 | 50.27 | 51.45 | 69.38 | 48.20 | 46.62 | 67.40 | 18.57 |
| 7 | 51.75 | 52.05 | 73.89 | 48.19 | 44.35 | 71.98 | 20.02 |
| 8 | 52.71 | 55.55 | 77.27 | 46.75 | 48.63 | 74.03 | 14.03 |
| 9 | 49.86 | 52.56 | 71.37 | 48.34 | 48.22 | 66.61 | 12.05 |
| 10 | 52.98 | 54.78 | 74.63 | 48.79 | 48.45 | 72.85 | 18.35 |
| 11 | 60.97 | 59.98 | 83.31 | 64.16 | 42.15 | 78.37 | 37.83 |
| 12 | 61.01 | 61.52 | 83.57 | 63.35 | 43.02 | 78.53 | 36.09 |
| 13 | 62.00 | 62.8 | 81.05 | 63.21 | 54.60 | 74.03 | 36.32 |
| 14 | 54.96 | 54.52 | 75.63 | 55.38 | 56.28 | 73.72 | 14.25 |
| 15 | 54.78 | 53.67 | 73.58 | 54.52 | 56.81 | 72.38 | 17.74 |
| 16 | 57.27 | 55.12 | 74.79 | 56.13 | 57.51 | 72.61 | 27.45 |
| 17 | 65.71 | 63.14 | 84.88 | 60.78 | 68.26 | 77.19 | 40.03 |
| 18 | 60.82 | 59.47 | 79.70 | 58.22 | 68.32 | 70.32 | 28.89 |
| 19 | 62.51 | 58.02 | 78.89 | 60.69 | 63.95 | 74.66 | 38.82 |
| 20 | 62.62 | 60.24 | 82.23 | 66.7 | 42.93 | 78.45 | 45.19 |
| 21 | 66.87 | 60.75 | 78.55 | 67.07 | 51.65 | 74.51 | 68.69 |
| 22 | 63.67 | 60.49 | 80.92 | 67.2 | 55.43 | 76.00 | 42.00 |
| 23 | 64.88 | 62.37 | 78.55 | 65.5 | 54.06 | 74.35 | 54.44 |
| 24 | 60.10 | 58.36 | 72.76 | 64.7 | 54.12 | 70.09 | 40.56 |

Table 12: Full Table Performance on Open LLM Leaderboard Tasks