Automated Data Curation for Robust Language Model Fine-Tuning

Anonymous ACL submission

Abstract

Large Language Models have become the de facto approach to sequence-to-sequence text generation tasks, but for specialized tasks/domains, a pretrained LLM lacks specific capabilities to produce accurate or well-formatted responses. Supervised fine-tuning specializes a LLM by training it on dataset of example prompts with desired target responses, but real-world data tends to be noisy. While many fine-tuning algorithms exist, here we consider a *data-centric AI* perspective on LLM fine-tuning, studying how to *systematically* curate the training dataset to improve the LLM produced via *any* fine-tuning algorithm. We introduce an automated data curation pipeline CLEAR (**C**onfidence-based **L**LM **E**valuation **A**nd **R**ectification) for instruction tuning datasets, that can be used with any LLM and fine-tuning procedure. CLEAR estimates which training data is low-quality and either filters or corrects it. Quality scoring and the decision to filter or correct are based on LLM-derived confidence estimates, which ensure algorithmic modifications to the dataset remain suitably conservative. Unlike existing data curation techniques, CLEAR is a comprehensive framework that can improve a dataset (and trained model outputs) without additional fine-tuning computations. Importantly, we don't assume access to a stronger LLM than the model being fine-tuned (e.g. relying on GPT-4 when fine-tuning GPT-3.5), to see whether CLEAR can meaningfully improve the capabilities of any LLM. Our experiments reveal that CLEAR consistently improves the performance of fine-tuned models across many datasets and models (like GPT-3.5 and Llama2).

1 Introduction

Large Language Models (LLMs) pretrained on an internet-scale text corpus have shown remarkable capabilities in generating helpful human-like text (Brown et al., 2020; Touvron et al., 2023). However, the efficacy of LLMs in specialized domains or tasks is heavily reliant on the process of *instruction tuning* (i.e. supervised fine-tuning, or alignment), where pretrained models are further trained using datasets that well-represent the domain (Wei et al., 2022). Here we specifically consider sequence-to-sequence training datasets of (prompt, target response) pairs. After training, we feed the LLM new prompts from the same domain and want it to produce responses that match the corresponding targets.

Since LLMs are billion parameter neural networks that indiscriminately absorb patterns and information across a massive dataset, the quality of these instruction tuning datasets is paramount to effective fine-tuning (Zhou et al., 2023; Xu et al., 2023). Unfortunately, most real-world instruction tuning datasets are noisy, containing many examples that are low-quality in various ways: the target response may be inaccurate, poorly written, the prompt may be nonsensical/incomplete/vague, or the two may be unrelated due to data entry/processing mistakes. Such flawed training data leads to fine-tuned models whose outputs are incorrect, irrelevant, biased, poorly formatted, or flawed in other ways. Finding and fixing low-quality data can be challenging in large datasets without extensive manual labor.

While most machine learning research iterates over modeling strategies (architectures, loss functions, training algorithms, ...) for a fixed dataset to produce better results, the emerging science of *Data-centric AI* asks how we can systematically iterate on the dataset while holding the modeling strategy fixed to produce better results (Mazumder et al., 2022). Success in real-world AI projects typically requires both approaches. Since many existing fine-tuning algorithms have been proposed (Zhang et al., 2023), we follow the spirit of data-centric AI and propose CLEAR, a comprehensive and automated data curation pipeline to enhance

the effectiveness of instruction tuning datasets for any LLM and fine-tuning algorithm.

Our CLEAR pipeline involves two stages: **Auto-Filter** and **Auto-Correct** which together offer a holistic solution to improving data quality for fine-tuning purposes. The Auto-Filter stage removes data that is confidently low-quality from the dataset *without any LLM fine-tuning*. It is already able to significantly improve the dataset, such that we can produce better fine-tuned LLMs without any extra LLM fine-tuning computation. For settings where one is able to fine-tune the LLM multiple times, the Auto-Correct stages uses the current fine-tuned LLM to revise certain examples that can be confidently improved, such that fine-tuning the LLM again on the resulting corrected dataset yields improved performance.

Algorithmic modifications to a dataset are generally dangerous and unlikely to improve performance unless done with extreme care. Filtering too much data limits the number of examples to learn from, and editing data can introduce various biases or amplify flaws in existing model outputs. Thus, all data modifications in CLEAR are conservatively applied based on careful measures of *confidence*. Specifically, we rely on BSDetector (Chen and Mueller, 2023), a method that can be used with any LLM to obtain trustworthy confidence scores for its own outputs as well as estimating the confidence that given outputs (e.g. target responses) are good. CLEAR only filters data that is confidently identified as low-quality, and only revises data where the LLM-corrected response is confidently identified to be better than the current dataset response. Our experiments reveal this careful treatment of confidence to be vital for developing a data filtering + correction solution that is effective across diverse instruction-tuning datasets without any manual modifications.

2 Related Work

2.1 Data Curation for ML

Data curation has been key in real-world deployment of classical machine learning models, with a broad spectrum of methods developed to address dataset mislabeling, outliers, noise, duplicates, and other data issues (Mazumder et al., 2022). Algorithmic strategies such as noise identification and removal (Northcutt et al., 2021), active learning for data prioritization (Settles, 2009), crowdsourcing for labeling (Snow et al., 2008), and self-supervised learning for error detection (Lan et al., 2020) have demonstrated how to produce better models by producing better data. These foundational strategies were mostly designed for classic machine learning tasks like classification with less complex datasets than instruction tuning for LLMs.

2.2 Instruction Fine-tuning

Significant research has been conducted into instruction tuning to specialize/improve LLMs (Kumar et al., 2016; Raffel et al., 2020; Efrat and Levy, 2020; Li and Liang, 2021; Wei et al., 2022; Wang et al., 2023a). FLAN (Wei et al., 2022) is a popular approach that employs a 137 billion parameter pre-trained language model, which is fine-tuned using instructions on more than 60 NLP datasets verbalized in natural language instruction templates. Wang et al. (2023a) showed how various instruction-tuning datasets can induce specific skills in a model, though no single dataset (or their combination) provides optimal performance across all assessments. Contrary to previous efforts aimed at creating a general foundation model capable of generalizing across a wide range of unseen tasks, here our aim is to train the best possible LLM for a specific narrow task.

2.3 Data Curation for Instruction Fine-tuning

The quality of training data in text generation applications has such significance that previous instruction tuning datasets were often crafted by hand (Khashabi et al., 2020; Ye et al., 2021; Wei et al., 2022; Wang et al., 2023b; Honovich et al., 2023). Wang et al. (2023b) introduced automated techniques by using GPT-3 (Brown et al., 2020) to produce 52,000 unique instructions not directly linked to specific tasks. This innovation opened new avenues for creating instruction datasets by extracting knowledge from teacher models. Following the introduction of Meta LLaMa (Touvron et al., 2023), there was a significant increase in the availability of open-source instruction tuning datasets and large language models (LLMs). Alpaca (Taori et al., 2023) introduces a self-instruct method that enables the autonomous creation of instructions, thereby minimizing the reliance on manual input. Vicuna (Chiang et al., 2023) capitalizes on the varied data accessible via ShareGPT, which naturally includes a wide array of data types and structures. WizardLM (Xu et al., 2023) employs an evolutionary approach to refine and diversify instructions, increasing their complexity and variability. Ultra-

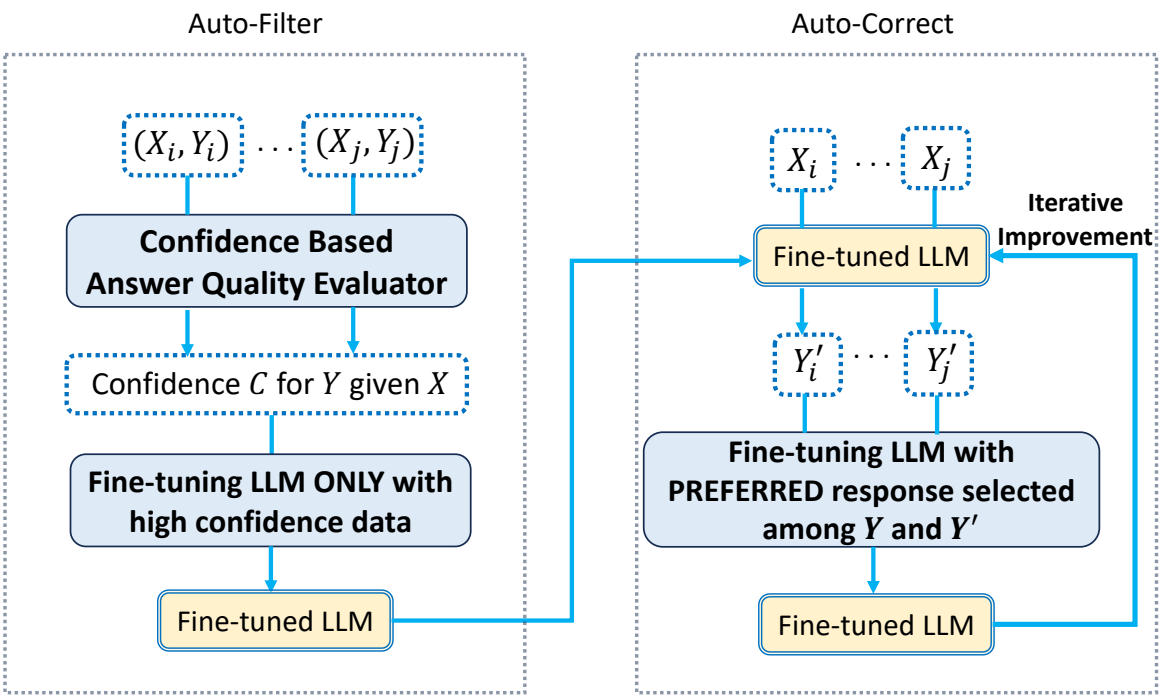

Figure 1: Pipeline of CLEAR.

Chat (Ding et al., 2023) sets up different scopes and systematically produces numerous instructions within each defined scope to improve task-specific performance. Meanwhile, LIMA (Xu et al., 2023) selects a thousand high-quality data samples strategically, showing notable improvements in LLM performance. Li et al. (2023a) introduces the instruction following metric to autonomously select good samples from vast open-source datasets.

However, much of this existing LLM fine-tuning research has focused on distilling teacher models such as ChatGPT that are more powerful than the LLM being fine-tuned (Taori et al., 2023; Chiang et al., 2023). Many existing LLM-based data curation techniques also utilize more powerful LLMs for the data curation process than the LLM being fine-tuned. In contrast, we aim to produce the best LLMs for specific tasks, in which even the most advanced LLMs like GPT-4 struggle to perform. Thus all data curation throughout this paper is performed *using the same LLM* as is being fine-tuned, to truly assess how effectively this data curation is able to boost LLM performance beyond the frontier.

3 CLEAR: Auto Data Curation Pipeline

When having instruction tuning datasets $I = \{(x_i, y_i)_{i=1}^n\}$ specific to a certain domain, our goal is to fine-tune the language model to improve its comprehension and execution of instructions. However, in practical scenarios, the instruction tuning data might contain noise, such as some mislabeling issue, which arise during the data collection phase. Moreover, this flawed data could adversely impact the training process and degrade the model's performance. The critical challenge lies in detecting these mislabeling errors and rectifying the labels accordingly. Inspired by this situation, we introduce an automated data curation pipeline that comprises two main components: **Auto-Filter** and **Auto-Correct**, which aim to detect problematic labels and rectify them, respectively. Auto-Filter employs a confidence-based answer quality evaluator (Chen and Mueller, 2023), to assign confidence scores to each pair of data. The language model is then fine-tuned only on the high confidence data. This fine-tuned LLM is utilized to assist in correcting mislabeled issues within the dataset. It achieves this by comparing the response generated by fine-tuned LLM with the original response and selecting the more preferable answer. Subsequently, the LLM is updated with this curated data, and this cycle of refinement and correction can be repeated (show in Figure 1).

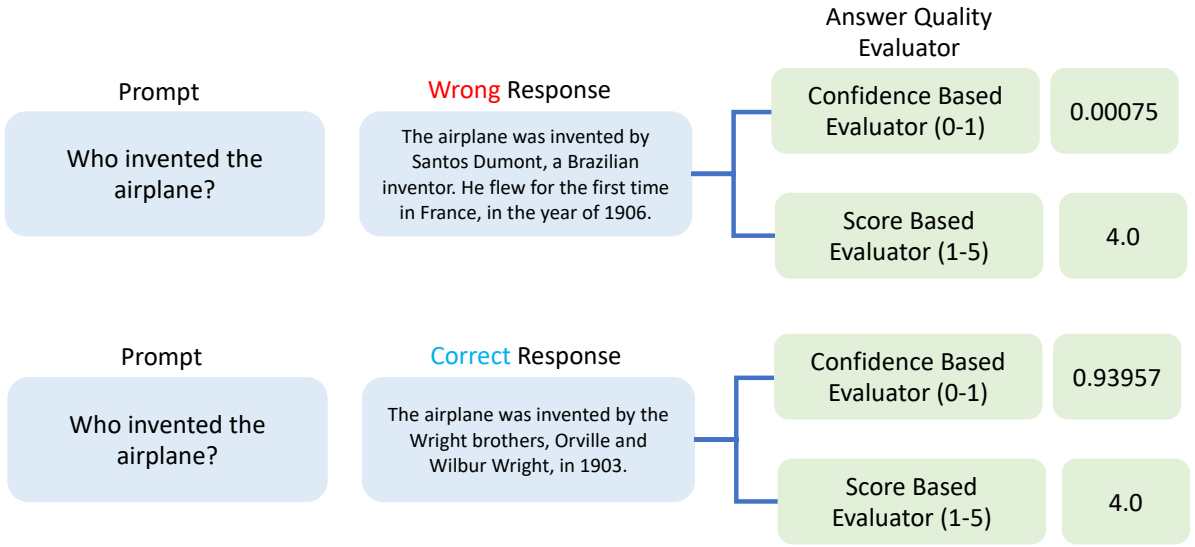

Figure 2: In this comparison, we assess two types of answer quality evaluators: confidence-based and score-based. The confidence-based evaluator provides a confidence score within the range of 0 to 1, where higher values indicate greater confidence. The score-based evaluator queries GPT-3.5-Turbo using a specific prompt shown in Table 5 and assigns scores within the range of 1 to 5, with higher scores indicating better quality. For the wrong response (above figure), the confidence-based evaluator assigns a very low confidence score, while the score-based evaluator assigns a score of 4.0. Conversely, for a correct answer (below figure), the confidence-based evaluator assigns a very high confidence score, while the score-based evaluator still assigns a score of 4.0. This suggests that the confidence-based evaluator is more reliable and trustworthy than the score-based evaluator.

4 Auto-Filter

In our work, we explore a novel approach to evaluating the quality of responses in instruction tuning tasks, diverging from the conventional method of using large language models like ChatGPT for assigning quality scoring through input-output pair analysis shown in Table 5. Our methodology employs a confidence-based evaluator, specifically BSDetector as introduced by Chen and Mueller (2023), which assesses the confidence that a response is good in terms of two factors: *observed consistency* and *self-reflection certainty*. BSDetector uses our same LLM to generate multiple candidate responses to a given prompt (via temperature sampling), then evaluates the semantic alignment between these candidate responses and the target response in the dataset (via natural language inference), and additionally integrates direct LLM evaluations of the target response (directly prompting the LLM to report its confidence). The resulting confidence estimates account for both aleatoric and epistemic uncertainty, without requiring any modification/training of the LLM. Our subsequent experiments reveal that this confidence-based approach to assessing answer quality is more precise compared to traditional LLM-ratings based answer quality evaluation (see Figure 2).

Given a instruction fine-tuning datasets comprising input-output pairs $\{(x_i, y_i)_{i=1}^n\}$, we use BSDetector with the base pretrained LLM (before it is fine-tuned) to estimate a confidence score C_i for each pair (x_i, y_i). We then filter out data pairs with low confidence scores, below a predefined threshold γ:

$$F = \{(x_i, y_i) | c_i > \gamma\}.$$

Subsequently, we fine-tune the large language model on the remaining training data F.

4.1 Auto-Correct

In the automated filtering stage, we exclude data characterized by low confidence, indicative of a discrepancy between the input and its corresponding response. This raises the immediate question of how to rectify the inaccuracies in these mismatched data pairs. The most direct approach involves substituting the incorrect response with one generated by another large language model. Nonetheless, for certain specialized fields, the utilization of even the most advanced LLMs, such as GPT-4, may still result in inadequate capabilities in managing these specific responses. Having fine-tuned the language model with data of high confidence, we then

explore the utilization of this enhanced LLM for automatic corrections. Given that the fine-tuned LLM now exceeds the capabilities of other models, such as GPT-4, we proceed to generate responses $(x_i, y'_i)_{i=1}^n$ through queries to this fine-tuned model. The subsequent step involves determining whether to adopt the newly generated response y' or to retain the original response y. In this approach, we engage the LLM such as GPT-4 to determine the preferable choice among y and y', recognizing that while the GPT-4 may not directly provide correct answers to specific queries, it possesses the capacity to evaluate and identify the superior response among two candidate response (Burns et al., 2023). Following the process of label correction, we achieve an improved quality dataset for instruction fine-tuning. Subsequently, this curated dataset is used to further improve the LLM fine-tuning. The prompt we used to determine the preferable choice among y and y' shown in Table 1.

5 Experiments

Datasets. We evaluate the effectiveness of our data curation process across three supervised fine-tuning (sequence to sequence text generation) datasets. **SQuAD** (Rajpurkar et al., 2016): prompts are articles and target responses are answers to questions created by crowdworkers based on a collection of Wikipedia articles, with each answer being a specific text fragment or span from the related article. **Emails** [1]: prompts are emails and target responses include categorizing the email into one of seven predefined themes by examining the email's subject and body content and also vary based on the email's length (whether the email content is short, medium, or long affects how the response is written). **DROP** (Dua et al., 2019): prompts are articles and target responses are answers to reading comprehension questions that require discrete reasoning over paragraphs (correctly answering requires resolving references in a question, perhaps to multiple places in the article, and performing basic operations over the references like addition, counting, or sorting).

To study our approach for noisy data that requires curation, we perturbed 20% of each training dataset (not the corresponding test set). For the emails dataset, the perturbation was to randomly swap target responses across different examples. To perturb a subset of the SQuAD and DROP datasets, where the answers are contained within the provided context, we randomly selected a different sentence from the context as the target response.

Evaluation metrics. For each dataset, our LLM fine-tuning performance evaluation focuses on two metrics (computed over a fixed held out test set): how often the model's response format adheres to a valid JSON structure and how often the model's responses are correct. For each model produced via a fine-tuning method, we report the proportion of model responses that are in valid JSON format, and the accuracy of model responses (which is computed via the proportion of exact matches to target reference responses, since we expect a well-supervised model to able to match the types of target responses it was fine-tuned on).

Baseline Methods. Our study also evaluates the following non fine-tuning methods: **Zero-shot** on GPT-3.5-turbo/GPT-4.0/Llama-2-7b-chat is directly querying these foundation models. **Few-shot** on GPT-3.5-turbo/GPT-4.0/Llama-2-7b-chat is directly asking the foundation models using in-context learning. For the fine-tuning methods, we use the full model fine-tuning on Llama-2-7b-chat and GPT-3.5 Turbo fine-tuning API. **Fine-tuning on the noisy data** refers to the process of fine-tuning the model on the original datasets without any data curation. **Auto-Filter** involves fine-tuning the model on newly created datasets, where data with low confidence levels are eliminated. This procedure sets a median value as the confidence score threshold, excluding any data falling below this threshold. **Auto-Correct** represents the strategy of fine-tuning the model using data that has undergone a correction process. This involves first applying the fine-tuned LLM to generate responses for each query. The next step is to compare the generated responses with the original ones, select the more preferable response from the two, and proceed to fine-tune the model on this curated set of responses. Importantly, the fine-tuning routine stays exactly the same when evaluating different data curation strategies – we only alter the dataset itself, not the model or training algorithm.

Other Details. We study the effectiveness of data curation strategies across two different fine-tuning methods. On the Llama-2-7b-chat model, we con-

[1] https://huggingface.co/datasets/neelblabla/enron_labeled_emails_with_subjects-llama2-7b_finetuning

Please review two answers carefully and select the one that you believe is superior. Consider factors such as accuracy, completeness, relevance to the question.

Question: `[. . . ]`

You are provided with two responses to the same question:

`[The Start of Answer A]`: `[. . . ]` `[The End of Answer A]`

`[The Start of Answer B]`: `[. . . ]` `[The End of Answer B]`

Please provide a brief reasoning you used to derive it. After providing your explanation, output your final verdict by strictly following this format: "`[[A]]`" if Answer A is better, "`[[B]]`" if Answer B is better, and "`[[C]]`" for a tie.

Table 1: Prompt (Zheng et al., 2023) used to determine the preferable choice among y and y'.

	Training Data or Prompting	Model	SQuAD Valid JSON (%)	SQuAD Accuracy (%)	Email Valid JSON (%)	Email Accuracy (%)	DROP Valid JSON (%)	DROP Accuracy (%)
Pretrained Model (No Fine-Tuning)	Zero-Shot	GPT-3.5	99.85	66.65	93.5	23.25	99.50	33.40
		GPT-4.0	99.90	75.93	100.0	48.25	100	39.80
		Llama-2	94.90	51.85	2.0	3.50	84.20	16.80
	One-Shot	GPT-3.5	99.20	69.50	99.0	38.75	99.60	40.80
		GPT-4.0	100.0	79.40	98.0	48.0	100.0	43.0
		Llama-2	24.65	9.70	17.25	19.50	32.0	4.90
	Three-Shot	GPT-3.5	87.60	61.20	95.75	47.0	98.50	41.80
		GPT-4.0	99.94	<u>80.08</u>	100.0	<u>49.75</u>	99.0	<u>46.10</u>
		Llama-2	13.10	2.55	1.75	5.75	20.60	4.60
Fine-Tuning	Original Data	Llama-2	92.45	49.86	99.30	50.67	99.30	44.70
	Auto-Filter Data	Llama-2	96.90	59.86	100.00	49.67	100.0	47.40
	Auto-Correct Data	Llama-2	96.90	71.44	99.67	52.33	100.0	50.50
	Original Data	GPT-3.5	97.90	64.50	100.0	43.0	100.0	56.80
	Auto-Filter Data	GPT-3.5	99.20	81.51	100.0	46.67	100.0	71.70
	Auto-Correct Data	GPT-3.5	100.0	**81.90**	100.0	**56.33**	100.0	**73.0**

Table 2: We present the results for non fine-tuning baselines and with fine-tuning across various Large Language Models. For the non fine-tuning approach, we highlight the top-performing results by underlining them. For the fine-tuning method, we emphasize the best results by making them bold. Across three datasets, it's evident that fine-tuned results surpass those of the non fine-tuning approach, even outperforming the most advanced LLM, GPT-4, in a three-shot setting. This underscores the critical need for fine-tuning in specific domains. Within the fine-tuning approach, we note a significant improvement from utilizing noisy data, particularly when only half of the data is used. Further improvements are observed when fine-tuning with corrected labeled data for both llama-2-7b and GPT-3.5 turbo fine-tuning API.

duct full model fine-tuning, in which all parameters of the neural network are updated via the Adam optimizer. We set the batch size at 128, and train for 3 epochs, using a learning rate of 1×10^{-5} with an accompanying cosine learning rate schedule. For the GPT-3.5 Turbo model, we use OpenAI's fine-tuning API. The exact training algorithm/hyperparameters used remain undisclosed to us, but this API has been observed to be highly effective for LLM fine-tuning. When evaluating

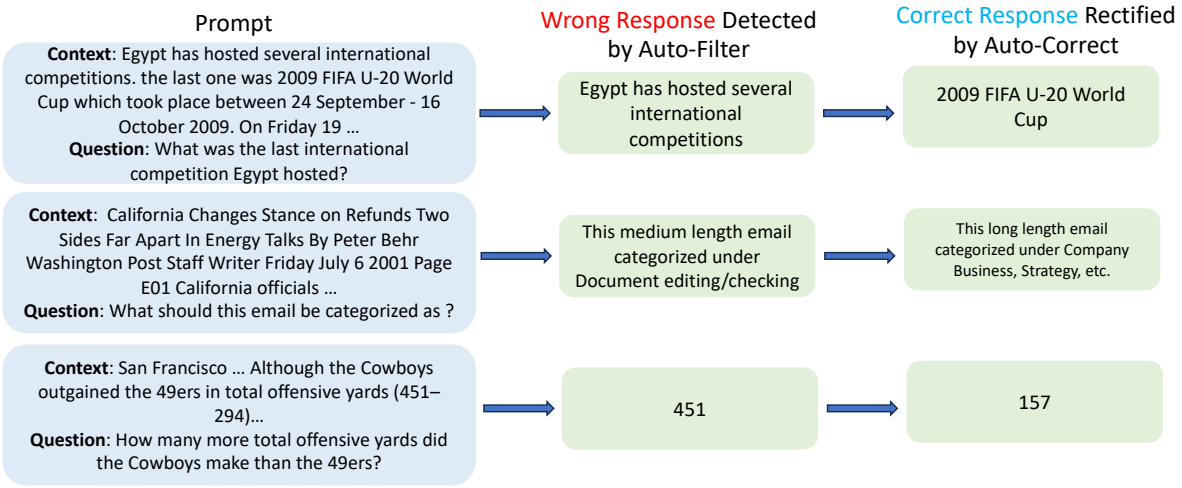

Figure 3: We show a bad target response detected in the Auto-Filter stage and then rectified in the Auto-Correct stage, from each of our three datasets.

outputs from our models at test time, we perform all text generation without any sampling, and limit the maximum number of new tokens to 512.

6 Results

Table 2 presents the results of all baselines and our methodology. In the non fine-tuning parts, GPT-4.0 stands out as the superior LLM, demonstrating the strongest performance across three datasets. It is observed that few-shot learning outperforms zero-shot learning. However, in the case of Llama-7B-chat, few-shot learning exhibits inferior results compared to zero-shot learning, attributed to the smaller model's heightened sensitivity to the selection of demonstrations. Regarding the fine-tuned models, it is noted that employing the entire dataset without selective curation can significantly degrade model performance. Conversely, fine-tuning with only half of the data, refined through automatic filtering, yields better results than utilizing the complete, uncurated dataset. Moreover, data that has undergone Auto-Correct further enhances model performance. Remarkably, the fine-tuned model outperforms even the most advanced model, GPT-4 in a three-shot setting. This highlights that even the most powerful LLMs may lack the capability to adequately address specific domain challenges. Additionally, we present a case study in Figure 3 where, for each dataset, the wrong responses identified by Auto-Filter were subsequently corrected by Auto-Correct, covering three different datasets.

6.1 Using Confidence based answer quality evaluator in Auto-Filter

In this section, we aim to highlight the advantages of using a confidence-based answer quality evaluator in the Auto-Filter procedure. We compare with score based evaluator, which directly prompts the GPT-3.5-turbo to rate the input-output pairs (Li et al., 2023b) and return the score from 1 to 5. The prompt we used is depicted in Table 5.

After assigning scores to each data pair, we discard 50% data with lowest scores. Subsequently, we fine-tune the model exclusively on the high-scoring data. This approach is compared against our confidence-based evaluator across various datasets. The fine-tuning process is applied to GPT-3.5 API. We additionally consider results based on randomly selected 50% of the data. The results of this comparison are presented in the table 3. The confidence-based evaluator is either on par with or exceeds the performance of the score-based evaluator and random selection, which demonstrates the benefits of confidence based evaluator over score based. As discussed in Chen and Mueller (2023), the confidence-based method involves directly asking the Large Language Model to assess whether a response is accurate or not. Additionally, it takes into account observed consistency, enabling it to provide a more precise evaluation of the response quality.

6.2 Using fined-tuned LLM in Auto-Correct

In the Auto-Correct process, we choose the preferred response by comparing the original response

Evaluator	SQuAD		Email		DROP	
	Valid JSON (%)	Accuracy (%)	Valid JSON (%)	Accuracy (%)	Valid JSON (%)	Accuracy (%)
Random	97.50	62.90	100.0	43.0	100.0	65.20
Score-based Evaluator	99.50	78.40	100.0	39.67	100.0	73.00
confidence-based Evaluator	99.20	81.51	100.0	46.67	100.0	71.70

Table 3: In the Auto-Filtering procedure, we use random, score-based evaluator and confidence-based evaluator to select 50% data and fine-tune the GPT-3.5 Turbo model. We show the performance of each data filtering method across three datasets.

Model used to generate the candidate response	SQuAD		Email		DROP	
	Valid JSON (%)	Accuracy (%)	Valid JSON (%)	Accuracy (%)	Valid JSON (%)	Accuracy (%)
GPT-3.5 Turbo	99.20	77.80	100.0	6.0	100.0	63.00
Fine-tuned LLM	100.0	81.90	100.0	56.33	100.0	73.0

Table 4: In the Auto-Correct procedure, we fine-tune the GPT-3.5 Turbo model on preferred response either from fine-tuned LLM or original response. Similarly, we apply this process between responses generated by the GPT-3.5 Turbo and the original response. The effectiveness of each approach is assessed across three datasets.

with that generated by fine-tuned LLMs. Consider an alternative approach where we obtain responses from a different, advanced LLM, such as ChatGPT, and then choose the more preferable response between the original and ChatGPT's. Specifically, we use GPT-3.5 Turbo to generate responses to all prompts, and then select the preferable one using the prompt in Table 1. The findings, shown in Table 4, illustrate that our method still exceeds the efficacy of choosing responses from GPT-3.5 Turbo. This emphasizes the importance of employing fine-tuned LLMs to produce candidate responses in the Auto-Correct process. As demonstrated in Table 2, the inferior performance in a zero-shot setting already suggests that relying on other LLMs for generating candidate responses might not be a viable strategy. Furthermore, considering that we use a model from the GPT family to select the preferable response, there could be an inherent bias towards responses it generates itself.

7 Conclusion

This paper presents CLEAR, a pipeline for curating data that can detect and rectify errors in datasets. It employs a confidence-based evaluator for assessing the quality of each question-answer pair, selecting only those with high confidence for fine-tuning a large language model. Subsequently, the fine-tuned LLM is utilized to address and correct issues of mislabeling. Following this data curation process, we observe substantial enhancements in the performance of the fine-tuned LLM. One open question is how can automated data curation be effectively integrated with other data augmentation techniques to further enhance model performance? For instance, creating additional synthetic datasets in situations where available data is scarce, and how to integrate these synthetic dataset with the original dataset to further enhance the performance of the fine-tuned large language model.

Limitations

While our automated data curation pipeline presents a significant advancement in enhancing the

quality of instruction tuning datasets for large language models (LLMs), it is important to acknowledge its limitations. The pipeline's current framework does not explicitly account for the possibility of biases within the original dataset or those introduced during the automated curation process. Since the model's performance and the quality of its output are contingent upon the data it was trained on, any inherent biases could be perpetuated or amplified through successive iterations of fine-tuning and correction.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeff Wu. 2023. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *CoRR*, abs/2312.09390.

Jiuhai Chen and Jonas Mueller. 2023. Quantifying uncertainty in answers from any language model via intrinsic and extrinsic confidence assessment. *CoRR*, abs/2308.16175.

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.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 3029–3051. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2368–2378. Association for Computational Linguistics.

Avia Efrat and Omer Levy. 2020. The turking test: Can language models understand instructions? *CoRR*, abs/2010.11982.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 14409–14428. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1896–1907. Association for Computational Linguistics.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1378–1387. JMLR.org.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2023a. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. *CoRR*, abs/2308.12032.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023b. Self-alignment with instruction backtranslation. *CoRR*, abs/2308.06259.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language*

Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, pages 4582–4597. Association for Computational Linguistics.

Mark Mazumder, Colby Banbury, Xiaozhe Yao, Bojan Karlaš, William Gaviria Rojas, Sudnya Diamos, Greg Diamos, Lynn He, Alicia Parrish, Hannah Rose Kirk, et al. 2022. Dataperf: Benchmarks for data-centric ai development. *arXiv preprint arXiv:2207.10062*.

Curtis Northcutt, Lu Jiang, and Isaac Chuang. 2021. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv e-prints*, page arXiv:1606.05250.

Burr Settles. 2009. Active learning literature survey.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast - but is it good? evaluating non-expert annotations for natural language tasks. In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 254–263. ACL.

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, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023a. How far can camels go? exploring the state of instruction tuning on open resources. *CoRR*, abs/2306.04751.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13484–13508. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *CoRR*, abs/2304.12244.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. Crossfit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7163–7189. Association for Computational Linguistics.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

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. *CoRR*, abs/2306.05685.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: less is more for alignment. *CoRR*, abs/2305.11206.

A Prompt for Score-based Answer Quality Evaluator

Below is an instruction from an user and a candidate answer. Evaluate whether or not the answer is a good example of how AI Assistant should respond to the user's instruction. Please assign a score using the following 5-point scale: 1: It means the answer is incomplete, vague, off-topic, controversial, or not exactly what the user asked for. For example, some content seems missing, numbered list does not start from the beginning, the opening sentence repeats user's question. Or the response is from another person's perspective with their personal experience (e.g. taken from blog posts), or looks like an answer from a forum. Or it contains promotional text, navigation text, or other irrelevant information. 2: It means the answer addresses most of the asks from the user. It does not directly address the user's question. For example, it only provides a high-level methodology instead of the exact solution to user's question. 3: It means the answer is helpful but not written by an AI Assistant. It addresses all the basic asks from the user. It is complete and self contained with the drawback that the response is not written from an AI assistant's perspective, but from other people's perspective. The content looks like an excerpt from a blog post, web page, or web search results. For example, it contains personal experience or opinion, mentions comments section, or share on social media, etc. 4: It means the answer is written from an AI assistant's perspective with a clear focus of addressing the instruction. It provide a complete, clear, and comprehensive response to user's question or instruction without missing or irrelevant information. It is well organized, self-contained, and written in a helpful tone. It has minor room for improvement, e.g. more concise and focused. 5: It means it is a perfect answer from an AI Assistant. It has a clear focus on being a helpful AI Assistant, where the response looks like intentionally written to address the user's question or instruction without any irrelevant sentences. The answer provides high quality content, demonstrating expert knowledge in the area, is very well written, logical, easy-to-follow, engaging and insightful. Please first provide a brief reasoning you used to derive the rating score, and then write "Score: " in the last line.

Input: []

Response: []

Table 5: The prompt (Li et al., 2023b) used to request the LLMs to assess the quality score of input-output pairs.