Towards an On-device Agent for Text Rewriting

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have demonstrated impressive capabilities for text rewriting. However creating a smaller yet potent language model for text rewriting presents two formidable challenges: costly data collection and absence of emergent capabilities. In this paper we present solutions to address the above challenges. We propose an new instruction tuning method to develop a mobile text rewriting model that leverages LLM-generated data and heuristic reinforcement learning, eliminating the need for human data collection. Moreover, to bridge the performance gap from the constraint size, we propose a cascading approach based on the confidence levels which are distilled from the large server model's critiques. To evaluate the text rewriting tasks for mobile scenarios, we introduce MESSAGEREWRITEEVAL, a human-labeled benchmark that focuses on text rewriting of messages through natural language instructions. Through empirical experiments, we demonstrate that our on-device model surpasses the current state-of-the-art LLMs in text rewriting while maintaining a significantly reduced model size using public benchmark EDITEVAL and our new benchmark. We show that our proposed cascading approach improves model performance further.

1 Introduction

The process of text rewriting can be considered a form of controlled text generation (Zhang et al., 2022), where text inputs are modified based on user specifications. Various text rewriting categories have been extensively explored, including paraphrasing (Siddique et al., 2020; Xu et al., 2012), style transfer (Riley et al., 2020; Zhang et al., 2020; Reif et al., 2021), sentence fusion (Mallinson et al., 2022), and sentence compression (Mallinson et al., 2018; Stahlberg et al., 2022). The advent of Large Language Models (LLMs) (Passos et al., 2023; Brown et al., 2020; Touvron et al., 2023) has ushered in a new era for text rewriting, demonstrating unparalleled quality by harnessing pre-trained models (Shu et al., 2023). With the widespread use of mobile communications and text messaging (Hanson et al., 2010; Pennington et al., 2022), these LLMs are being integrated into text rewriting applications, enabling users to create messages that are "formal", "concise" etc. (Burke, 2023).

Despite the impressive text rewriting ability enabled by LLMs, their deployment for real-world chat messaging faces practical issues. While deploying large models on users' devices is impractical due to their size, server-based architectures introduce several drawbacks. They make it harder to preserve user privacy (Li et al., 2021), limit the models' ability to operate offline (Murshed et al., 2021), and incur higher overall compute costs (Chen et al., 2023a). Developing a compact yet potent language model presents two unique challenges, First, training smaller models requires significantly larger datasets which requires costly data collection (Kang et al., 2023). Second, the emergent capabilities of the LLM only appears after reaching a critical size (Wei et al., 2022).

In this paper, we present a systematic approach for enhancing the rewriting capability of LLMs while adhering to size constraints to ensure reasonable on-device inference speeds. We introduce a benchmark called MESSAGEREWRITEEVAL, compiled from human-donated message texts and rewrites by human with diverse language instructions. Unlike existing benchmarks for text rewriting such as EDITEVAL (Dwivedi-Yu et al., 2022) or OPENREWRITEEVAL (Shu et al., 2023) which are derived from text sourced from paragraphs or long passages, our benchmark is designed to better represent daily conversational exchanges between individuals.

Inspired by InstructGPT (Ouyang et al., 2022), we train our model using a combination of supervised fine-tuning (SFT) and reinforcement learning (RL). While InstructGPT relies heavily on human raters for both instruction data and preference data, our approach minimizes human intervention in the data collection process. To elaborate: (1) For instruction data generation, we develop a novel method based on continued generations from LLMs to generate high quality synthetic data. (2) Instead of using a reward model (Shu et al., 2023), we propose a heuristic-based reward signal for reinforcement learning that can improve the model without additional labeling. We conduct empirical investigations to assess the model's performance against the MESSAGEREWRITEEVAL and EDITEVAL benchmarks. Our proposed model outperforms its corresponding foundation model and other instruction-tuned LLMs, which validates the usefulness of the generated training data and the proposed heuristic reinforcement learning.

To further mitigate the size constraints and bridge the gap between the on-device model and the giant server-side LLMs, we propose a cascading approach to chain our on-device model with the more powerful server model. The system follows a simple yet effective principle: the server side will only be used when the on-device language model fails to provide a good response. Instead of relying on an external model to judge the quality of response (Chen et al., 2023a), we propose to add a simple suffix to the on-device model output that indicates how confident the model is in its prediction. The suffix is learned from the larger server-side LLM via distillation. Our findings demonstrate that the proposed cascading approach further enhances performance.

Our main contributions can be summarized as follows:

- We develop a powerful LLM that demonstrates superior performance compared to the state-of-the-art LLMs for text rewriting while being efficient for on-device inference. Importantly, this model's efficacy does not rely on human-labeled data collection. We devise innovative strategies to generate varied instruction datasets for rewriting, that enhance the editing and rewriting capacities of the model. Additionally, we present a heuristic-based reinforcement learning approach that eliminates the need for training the reward model.

- We design an effective cascading mechanism to connect our on device model to the server side model. We distill the critiquing ability of the server LLM to the smaller model using discriminative training, which enables efficient inference. Our cascading strategy can further improve the on-device model's performance, bringing it closer to the capabilities of the server-side model while reducing the number of server calls.

- We introduce a new benchmark, MESSAGEREWRITEEVAL, designed for research on message text rewriting and covering different types of rewrites expressed through natural language instructions: formality, elaboration, shortening, paraphrasing, and proofreading. To the best of our knowledge, no such benchmark is currently available.

2 Related Work

2.1 Text Editing

The text editing (Chuklin et al., 2022) task covers a wide range of sub-tasks such as paraphrasing (May, 2021), style transfer (Tikhonov et al., 2019), spelling and grammatical error correction (Napoles et al., 2017), formalization (Rao and Tetreault, 2018), simplification (Xu et al., 2016) and elaboration (Iv et al., 2022). Recent work has investigated a more diverse set of rewrite options (Faltings et al., 2020; Schick et al., 2022; Shu et al., 2023) by leveraging the diversity of edits in Wikipedia. While our model can take diverse prompts as input, its core strength is on rewriting messages through formalizing, shortening, elaborating, paraphrasing, and proofreading.

2.2 Instruction Tuning

Instruction tuning has been shown to improve model performance and generalization to unseen tasks (Chung et al., 2022; Sanh et al., 2022). InstructGPT (Ouyang et al., 2022) extends instruction tuning using reinforcement learning with human feedback (RLHF), which heavily relies on human raters to obtain instruction data and rankings of model outputs. The dependency on human preference data could be alleviated by reinforcement learning with AI feedback (RLAIF) (Bai et al., 2022; Shu et al., 2023), but training a separate reward model is still required. We extend this framework using a heuristic based reinforcement learning (Cheng et al., 2021) for rewriting tasks, which enables reinforcement learning without a reward model.

2.3 Distillation and Data Augmentation

Knowledge distillation (Hinton et al., 2015) has been successfully used to transfer knowledge from larger teacher models into smaller student models (Hinton et al., 2015; Tang et al., 2019; Wang et al., 2021; Smith et al., 2022; Beyer et al., 2022; Peng et al., 2023; Wu et al., 2023). The quality of distillation could be improved in a variety of ways such as using a better design of Chain-of-Thought prompts (Shu et al., 2023), combining the noisy predictions with majority vote (Arora et al., 2022), using a augmented label with reasoning (Hsieh et al., 2023), reweighting the student's loss (Iliopoulos et al., 2022) etc. Unlike previous work, we use a pre-trained LLM to generate data and also provide critique for generated output, enabling automatic filtering. Furthermore, we extend our distillation technique to perform critiques.

2.4 LLM Cascades

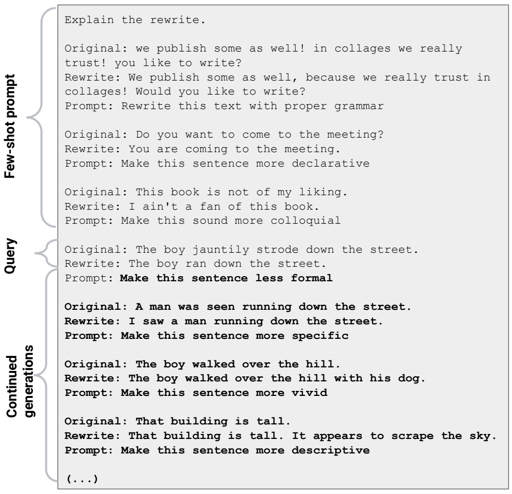
Language model cascades have been investigated in many previous works (Li et al., 2020; Cai et al., 2023; Wu et al., 2022; Dohan et al., 2022). Frugal GPT (Chen et al., 2023a) proposed several strategies for using multiple LLMs to minimize the inference cost. For the cascaded design, the regression score from DistillBert (Sanh et al., 2019) is used for deciding whether or not the model response is adequate. Although our approach achieves a similar goal, it does not require an extra model. We incorporate this capability into the language model in a single pass text generation step by using the suffixes of the generation (Thoppilan et al., 2022).

3 Methods

Our approach follows the "supervised fine-tuning (SFT) + reinforcement learning (RL)" paradigm (Ouyang et al., 2022), but does not require any human labeling or preference data collection. We first discuss our approach to generate synthetic training data for supervised fine-tuning. We then present our heuristic reward and RL process. Finally, we describe our cascading method.

3.1 Supervised Fine-tuning

We follow existing works to leverage the document level edit data from Wikipedia (Schick et al., 2022; Shu et al., 2023). In pilot studies, we observed that using this data alone cannot provide adequate short form, message like data for training our on-device models. To generate in-domain data efficiently, we propose a data generation approach based on continued generation by off-the-shelf LLMs, which can then be filtered using LLMs. The details of the training data are provided in Section A.5.


Few-shot prompt
Query
Continued generations

```
Explain the rewrite.

Original: we publish some as well! in collages we really
trust! you like to write?
Rewrite: We publish some as well, because we really trust in
collages! Would you like to write?
Prompt: Rewrite this text with proper grammar

Original: Do you want to come to the meeting?
Rewrite: You are coming to the meeting.
Prompt: Make this sentence more declarative

Original: This book is not of my liking.
Rewrite: I ain't a fan of this book.
Prompt: Make this sound more colloquial

Original: The boy jauntily strode down the street.
Rewrite: The boy ran down the street.
Prompt: Make this sentence less formal

Original: A man was seen running down the street.
Rewrite: I saw a man running down the street.
Prompt: Make this sentence more specific

Original: The boy walked over the hill.
Rewrite: The boy walked over the hill with his dog.
Prompt: Make this sentence more vivid

Original: That building is tall.
Rewrite: That building is tall. It appears to scrape the sky.
Prompt: Make this sentence more descriptive

(...)
```

Figure 1: Paired dataset from the continued generations of the LLM. Bolded text includes a generated prompt for the query and the continued generations, which contains samples of Source, Rewrite, and Prompt.

3.1.1 Synthetic Paired Dataset from Continued Generations

To collect more shorter-form and message-like data, we leverage the few-shot capability of pre-trained LLMs. Figure 1 shows an example of the initial prompts and demonstrates how the LLM is continuing to generate diverse examples from a given query, which is sampled from a small seed query set. The continued generations enable efficient generation of diverse paired data.

3.1.2 An LLM guided data selection

To further improve the quality of our synthetic dataset, we propose to use LLMs to critique the generated data. We leverage the few-shot Chain-of-Thoughts (CoT) reasoning of the off-the-shelf LLM to judge whether the response is following the instruction of the prompt to rewrite the original sentence in a good manner. We provide detailed prompt samples in Table 15. We also leverage the self-consistency (Wang et al., 2022a) approach to improve the accuracy. Specifically, we only keep the data when it is approved by all LLM judges.

3.1.3 Generative Fine-tuning

Given a pre-trained decoder-only language model, we fine-tune it using the collected instruction tuning dataset. The input is formed by concatenating the <instruction> and the <source> with a newline, while the output is the <target>.

3.2 Heuristic based Reinforcement Learning

The reinforcement learning part is typically called Reinforcement Learning with Human Feedback (RLHF) (Ziegler et al., 2019) as human labelers are heavily involved in training the reward model. In this section, we introduce a novel approach to improve alignment through heuristics without any human labeling.

3.2.1 Heuristic Reward

The intuition is that a few common heuristics can yield high quality rewrites. We propose to use the following heuristics as reward signals.

Natural Language Inference (**NLI**) (Bowman et al., 2015) scores over the source-prediction pair. Given a "premise" and a "hypothesis", NLI scores the probability that the "hypothesis" is correct given the "premise". In the context of LLMs, NLI score estimates whether the LLM's output prediction preserves meaning and factuality given the source text. We use the off-the-shelf NLI predictor from (Honovich et al., 2022), denoted as nli.

Reversed NLI. NLI score where the premise and the hypothesis are reversed, denoted $rnli$.

Length Ratio. The ratio of the number of tokens in the LLM output text to that in the source text, denoted $length\_ratio$.

Edit Distance Ratio (Edit Ratio). Edit distance (Levenshtein, 1966) measures the minimum number of token-level edits (insertions, deletions and substitutions) to convert a source text into a target text. We use the relative edit distance between the prediction and source text, computed as the ratio of the edit distance to the length of the source text. The edit ratio, denoted as $edit\_ratio$, represents the proportion of the source text that has been modified.

N-gram frequency. Text generation can easily get stuck in undesirable sentence-level loops with decoding (Xu et al., 2022). We propose measuring the N-gram frequency to detect potential loops in the generated output – if the frequency of a certain N-gram is too high, we introduce a constant negative reward to penalize it. We denote the output of this algorithm as $ngram\_reward$.

We formulate the final reward as a weighted combination of all the signals above in equation (1). For different rewriting tasks, the coefficient σ_i should be designed to reflect the expectation of the rewrites. For instance, the expectation for "shorten" is higher nli value (a larger positive σ_1) and lower $length\_ratio$ (a negative σ_3). We share the choice of hyper-parameter σ_i in Appendix Table 8.

$$\begin{aligned} Reward = \sigma_1 nli + \sigma_2 rnli + \sigma_3 length\_ratio \\ + \sigma_4 edit\_ratio + \sigma_5 ngram\_reward \end{aligned} \tag{1}$$

3.2.2 Reinforcement Learning

We further refine the fine-tuned model by employing reinforcement learning (Ouyang et al., 2022), guided by the heuristics provided. The prompts for reinforcement learning are collected from the LLM during training data generation. For each prompt in the train set, we first use LLM's fewshot ability to classify the prompt into the rewrite types. During the reinforcement learning, this rewrite type will be fed to the "heuristic reward" module to generate the reward, which will be finally optimized through PPO (Schulman et al., 2017).

3.3 Critique Distillation and Model Cascade

We apply a simple cascade mechanism whereby the on-device model serves as the first gate to the incoming rewrite request, and the large server side model is invoked only when the on-device rewrite is deemed low quality. Towards this goal, we need to answer two questions. First, how to enable the on-device model to do "self-critique", which is challenging given its small size and the complexity of the task. Second, how do we make the process more efficient without additional inference steps. We next present our suffix based distillation approach as a solution to the above questions. We leverage the off-the-shelf LLM as a critiquer and distill its knowledge as an extra "suffix" in the data into the on-device model. The approach is summarized in Figure 2.

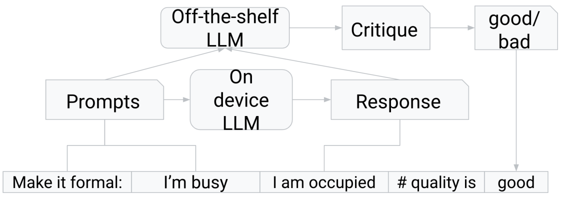


Figure 2: The illustration of distillation for self-critiques. The final sentence with "quality is good" as suffix will be used as training data for discriminative training.

3.3.1 Critique Distillation from LLMs

Similar to reinforcement learning, we prepare unpaired prompt data sampled from continued generation of LLMs. The responses are generated by our model. Then the (prompt, response) pairs are fed to the off-the-shelf LLM to decide whether they are acceptable or not. We leverage the Chain-of-Thought (CoT) reasoning along with the self-consistency approach. We use the prompts shown in Appendix Table 15

3.3.2 Discriminative Fine-tuning

Although generative fine-tuning with the larger LLM's response can make it possible to perform self-critiquing for the small models, "generation" and "self-critique" will be two separate text generation steps, resulting in increased inference times. To fuse the two steps, we transform generative fine-tuning into discriminative fine-tuning (Thoppilan et al., 2022). This is done by concatenate a label ("good"/"bad") to the response with some predefined delimiter. In this way, we can generate the suffix data distilled from the critique provided by the off-the-shelf LLM. Finally we finetune the ondevice model with the suffix data along with original generations.

3.3.3 Cascading

Once the model is finetuned with suffix data, it can use the suffix score, i.e. probability of outputting "good", to decide whether to cascade. Specifically, after decoding some text we compare the "suffix score" s (which is a probability between 0 and 1) and some pre-defined threshold γ. If $s > \gamma$, the on-device model is deemed confident; Otherwise, the model relies on the server side model.

4 Experiment Settings

4.1 Model Training Setting

Our pre-trained checkpoint is PaLM 2-XXS[1]. We leverage pre-trained PALM 2-L as the off-the-shelf LLM for data generation, LLM filtering, and critique distillation. The training hyper-parameters for instruction tuning and reinforcement learning are listed in the Appendix Section A.4.

4.2 Evaluation Datasets

4.2.1 MessageRewriteEval

To evaluate the model performance in the on-device messaging scenario, we introduce MessageRewriteEval, a novel evaluation dataset specifically designed for message-level rewrite assessments. All text message pairs are sourced from real-life, human-written daily use cases and evaluated by human raters for data quality. To ensure comprehensive evaluation, these pairs encompass five text rewrite tasks: *Formalize*, *Paraphrase*, *Shorten*, *Elaborate*, and *Proofread*. Each text pair in the dataset consists of three components: *source*, *target*, and *instruction*. The task distribution statistics and example instructions are provided in Appendix Section A.1. The data collection guidelines are given in Appendix Section A.2.

4.2.2 EditEval

Besides the on-device messaging scenario, we evaluate the model performance on more general text rewriting tasks. We use the public rewrite benchmark EditEval [2] (Dwivedi-Yu et al., 2022) which covers rewriting task at both sentence and paragraph levels. The detailed description of the different datasets in this benchmark can be found in Appendix Section A.3.

4.3 Automatic Evaluation Metrics

We employ various metrics to evaluate the model's quality:

NLI (Bowman et al., 2015) and **Reversed NLI** (Section 3.2.1).

Edit Distance Ratio (Edit Ratio) (Section 3.2.1).

SARI (Xu et al., 2016) is an n-gram based metric that measures the similarity of a prediction to both the source and reference texts. The scores of add, retain and delete operations are computed by averaging n-gram scores. The SARI metric is obtained using an arithmetic average of the F1 scores of add and retain operations and the precision of the delete operation.

BLEU (Papineni et al., 2002) is computed as a geometric mean of n-gram precisions of different orders.

Update-ROUGE (Updated-R) (Iv et al., 2022) is a modified version of ROUGE (Lin and Hovy, 2003) that specifically computes ROUGE-L on the updated sentences rather than the full text.

Success Rate We use the LLM to assess whether or not the response follows the instruction (i.e. "good" or not). Although a binary classification might be too coarse grained for evaluating rewrite quality, it is a very intuitive and straightforward

[1] We follow the size notations in PaLM 2 tech report (Passos et al., 2023). Model size **XXS** is over 20 times smaller than model size **S** and over 5 times smaller than model size **XS**.

[2] https://github.com/facebookresearch/EditEval

metric to show the merit of cascading. The LLM prompts are provided in Appendix Table 15.

On-device Inference Ratio For cascading experiments, A higher ratio means a smaller percent of server calls.

4.4 Baselines

Since it is designed for on-device application, our model has a compact size in comparison to other LLMs. In choosing baseline models, we prioritize the ones that are similar in size to ours. We choose the state-of-the-art pre-trained models **PaLM 2** (Passos et al., 2023), **LLaMA** (Touvron et al., 2023) and the instruction tuned models **Alpaca** (Taori et al., 2023), **Vicuna** (Chiang et al., 2023), **Flan-PaLM 2** (Passos et al., 2023) as our baseline models. We also provide **Alpaca-PaLM 2** for comparison. The Alpaca's instruction dataset is finetuned using a PaLM 2 baseline checkpoint.

For a fair comparison, we leveraged in-context learning with CoT few-shot prompting (we share the details in Appendix Section A.11) to instruct the model to provide reasonable responses for the pre-trained models since they are not instruction tuned. In contrast, for the instruction tuned LLMs including ours, we use zero-shot settings. For cascading, we note that constructing a powerful large language model is not within the the scope of this study. Therefore, our experiments utilize the 175B InsGPT (Ouyang et al., 2022) as the server model.

4.5 Human Evaluation

We follow the same human evaluation setup as the RewriteLM paper (Shu et al., 2023). 300 examples are randomly sampled from MESSAGEREWRITEEVAL for human evaluation with five language experts. A 3-point Likert scale (0-Bad, 1-Medium, 2-Good) is used for the following features: 1) **Instruction Success**: whether the output text follows the given instruction. 2) **Content Preservation**: whether the essential content of the input text are kept in the output text, independent from style or quality. 3) **Factuality**: whether the output content is accurate and truthful. 4) **Coherency**: whether the output text is non-ambiguous, and logically coherent written, independent from the input text. 5) **Fluency**: whether the output text is written with good clarity, correct grammar, and style. The detailed rating guideline is in Appendix A.8.

5 Results

5.1 Performance of the On-device Model

To show that our approach can generally enhance the model's rewriting ability, we first report performance of our SFT model and RL model on EDITEVAL. And then we evaluate the same SFT model and RL model on MESSAGEREWRITEEVAL. We present latency and memory metrics for on-device inference in Appendix A.9.

5.1.1 Results on EditEval

Table 1 summarizes the results. The metrics of the baseline models are directly obtained from the EditEval paper (Dwivedi-Yu et al., 2022). We list only those models whose sizes are similar to our on-device models; Nevertheless, our model is substantially smaller than these models. We provide SARI values for each dataset and extra Update-R scores for the two datasets relevant for the paragraph update task.

The results show that our on-device model with size **XXS** outperformed other models on most of the tasks despite being much smaller. For the fluency, coherence, paraphrase, simplification and paragraph update tasks, our model wins by a large margin. Heuristic reinforcement learning generally boosts the model's performance on all tasks.

5.1.2 Results on MESSAGEREWRITEEVAL

The automatic evaluation results for the MESSAGEREWRITEEVAL dataset are shown in Table 2. We first examine results of three sets of models: pre-trained LLMs, Instruction-Tuned LLMs and our on-device Instruction-Tuned LLMs.

Edit Ratio measure of token-level different between texts, We empirically observed that a larger Edit Ratio does not always correlate with better rewrite performance, as it often arises from hallucinations. In terms of SARI, BLEU, and Update-R metrics, our on-device size models outperform LLaMA, Alpaca-7B and Vicuna-7B, despite having a much smaller size. We also compare our results to Alpaca-PaLM 2 and Flan-PaLM 2, which share the same base architecture and model size. The fact that our model achieves much better SARI, BLEU, and Update-R scores validates the effectiveness of our approach. Moreover, the gap in performance between the SFT and RL models shows that our heurisic reinforment learning is very effective. We performed three independent training runs of the RLed model and present the average and standard

		JFL	ITR$_{FLU}$	ITR$_{CLA}$	ITR$_{COH}$	STS	TRK	AST	WNC	FRU		WFI	
		SARI	SARI	SARI	SARI	SARI	SARI	SARI	SARI	SARI	Update-R	SARI	Updated-R
Copy	Size	26.7	32.3	29.5	31.3	21.1	26.3	20.7	31.9	29.8	0	33.6	-
T0++ (Sanh et al., 2022)	11B	34.7	35.5	37.6	32.7	28.4	32.9	28.2	29.3	12.6	3.7	4.4	8.1
PEER-11 (Schick et al., 2022)	11B	55.8	**52.1**	32.5	32.7	28.2	32.1	29.5	**54.5**	**39.6**	31.4	**34.9**	20.4
Tk (Wang et al., 2022b)	3B	31.8	32.4	**38.4**	33.8	30.2	32.8	29.9	31.1	12.6	3.6	1.3	4.5
T0 (Sanh et al., 2022)	3B	42	24.6	32.6	22.2	34.3	34.4	32.3	22.3	14.2	9.6	5.1	16.3
PEER-3 (Schick et al., 2022)	3B	55.5	51.4	32.1	32.1	28.6	32.5	30.5	53.3	39.1	30.9	34.4	18.7
PaLM 2 (Passos et al., 2023)	S	36.07	22.68	28.79	27.82	34.45	34.32	35.92	25.2	24.28	26.39	11.41	20.42
Flan PaLM 2 (Passos et al., 2023)	XS	30.03	36.01	34.81	33.17	31.91	34.32	31.4	27.75	15.19	5.34	6.86	3.12
Flan PaLM 2 (Passos et al., 2023)	XXS	34.43	30.12	34.08	31.32	29.25	33.06	35.92	17.5	13.6	2.75	4.78	0.97
Alpaca PaLM 2	XXS	29.33	17.01	24.42	23.81	32.59	31.56	33.46	28.11	23.53	14.22	6.5	3.72
SFT (Ours)	XXS	58.36	37.67	33.85	36.03	37.49	38.88	**41.95**	32.35	35.44	47.49	22.03	32.36
SFT + heuristic RL (Ours)	XXS	**61.1**	40.26	34.81	**37.33**	**38.25**	**40.21**	**41.95**	35.28	35.81	**49.49**	26.32	**40.71**

Table 1: Model Performance on EditEval (Dwivedi-Yu et al., 2022). Only models with reasonable sizes are listed. Size **XXS** is **less than half the size** of T0/Tk models. Despite their reduced sizes, our models achieve even better performance than most of the other larger models. Relative to similar-sized instruction-tuned models, our models win by a large margin.

	Size	Edit Ratio	NLI	Reversed NLI	SARI	BLEU	Update-R
InsGPT (Ouyang et al., 2022)	175B	0.18	0.91	0.88	51.14	35.0	58.91
LLaMA (Touvron et al., 2023)	7B	3.98	0.68	0.74	31.58	16.65	29.24
PaLM 2 (Passos et al., 2023)	XS	0.98	0.83	0.72	38.92	22.98	37.45
PaLM 2 (Passos et al., 2023)	XXS	1.59	0.76	0.82	31.49	18.81	31.85
Alpaca (Taori et al., 2023)	7B	0.26	0.76	0.76	42.21	24.80	45.15
Vicuna (Chiang et al., 2023)	7B	1.27	0.46	0.52	38.18	14.30	30.17
Flan-PaLM 2 (Passos et al., 2023)	XS	0.11	**0.94**	0.83	29.50	25.89	34.63
Flan-PaLM 2 (Passos et al., 2023)	XXS	0.11	0.93	0.80	27.41	17.59	15.43
Alpaca-PaLM 2	XXS	0.3	0.84	0.78	43.14	25.88	47.76
SFT	XXS	**0.17**	0.89	0.75	46.23	27.78	51.84
SFT + heuristic RL	XXS	0.16(0.01)	0.93(0.01)	**0.85(0.01)**	**47.34(0.17)**	**30.50(0.26)**	**54.97(0.61)**
SFT + heuristic RL + critique distillation	XXS	0.16	0.93	0.84	48.6	32.43	55.72
Ours + InsGPT (40% server calls)	-	0.16	0.93	**0.87**	**49.87**	**34.59**	**58.87**
Ours + InsGPT (15% server calls)	-	0.16	0.92	0.86	49.03	33.76	57.41

Table 2: Model Performance on MESSAGEREWRITEEVAL. Our models achieves best performance compared with all listed Pre-trained LLMs and Instruction-Tuned LLMs, which have either same or larger size then ours. When cascaded with InsGPT, the performance is further improved.

deviation of the performance metrics in the table. The low standard deviations across metrics suggest consistency in the RLed model's performance.

We also study the role of each heuristic by doing ablations. We summarize results in Table 3. As we can see from the table, removing any one of the proposed heuristics will reduce the overall quality of rewrites. Notably the NLI s-t and the NLI t-s play more important roles for securing good rewrite comparing to other rewards.

	Edit Ratio	NLI s-t	NLI t-s	SARI	BLEU	Update-R
heuristic RL	**0.16**	**0.93**	**0.85**	**47.34**	**30.50**	**54.97**
- Edit Dist	0.13	**0.93**	**0.85**	47.30	30.28	54.21
- Len Ratio	0.15	**0.93**	0.84	47.27	30.24	54.00
- Ngram	0.15	0.92	**0.85**	47.22	30.32	53.96
- NLI s-t	0.16	0.89	0.84	46.50	28.81	52.86
- NLI t-s	0.15	0.92	0.78	47.11	28.91	52.40

Table 3: Ablation study for the heuristic rewards. Each experiment removes one heuristic and keep the rest.

5.2 Performance of Cascading

Our cascading experiments are conducted on the top of the on-device model with RL using MESSAGEREWRITEEVAL benchmark. Here we choose it over EDITEVAL for cascading experiments as it is more aligned with the mobile cases. We first evaluate how the critique distillation is impacting the model's performance. Next we show the end-to-end cascading performance and a detailed analysis and demonstrate that our suffix score is more effective than the baseline LM score.

5.2.1 The Effect of Critique Distillation

In Table 2 we show that the model's overall performance is further improved on SARI, BLEU, and Update-R with little regression on Reversed NLI when we combine the distilled discriminative dataset with the generative dataset. This suggests that with the suffix score from critique distillation,

	Instruction Success	Content Preservation	Factuality	Coherence	Fluency	AVG
Agreement	0.620	0.748	0.687	0.714	0.710	0.696
InsGPT	**1.780**	**1.933**	**1.924**	**1.979**	**1.969**	**1.917**
Alpaca-PaLM 2	1.193	1.569	1.559	1.937	1.939	1.639
SFT	1.492	1.767	1.770	1.967	1.959	1.791
SFT + heuristic RL	1.674	1.881	1.853	1.965	1.959	1.867
Ours + InsGPT (40% server calls)	1.777	1.932	1.919	1.977	1.970	1.915

Table 4: Human Evaluation Results.

the model tends to pick sample with higher quality.

5.2.2 End-to-end Performance

The on-device model's reliance on the server model is controlled by the threshold γ. As shown in Table 2, the performance of the cascaded models lies between the on-device and the server side model. With a higher number of server calls, we obtain higher SARI, BLEU, and Update-R, as expected. With 40% server calls, the overall performance is already quite close to the full server model. We also profiled the latency of it and did more analysis in Appendix A.10.

5.2.3 Suffix Score vs LM Score

We now provide more insight into our cascading approach with suffix score. We vary the threshold γ from 0 to 1 to measure Success Rate as a function of the On-device Inference Ratio. The trade-off between the two metrics is shown in Figure 3. To demonstrate the efficacy of the distilled suffix score derived from larger LLM critiques as a reliable indicator of output quality, we compare it with an LM score, representing the likelihood of the generated text. As shown in Figure 3, "suffix score with 1 sample" is outperforming "LM score with 1 sample" by large margin. This indicates that given a text output, suffix score offers higher quality estimates. As a result, when sampling multiple outputs (8 samples), suffix score can accurately select the decoded candidate with the highest quality, which greatly improves performance. In contrast, the LM score stays almost unchanged when increasing the number of samples, showing that it is less helpful.

5.3 Human Evaluation

In Table 4 we show the human evaluation results that align with the auto metric results shown in Table 2. The inter-annotator agreements, quantified using the Fleiss kappa coefficient (Fleiss 1971), demonstrate the reliability of the evaluations. There is a huge gain from SFT after heuristic RL. With 40% server-side calls (GPT 3.5), the model gains another big performance boost very close to the server-side model. Our SFT model's superior performance compared to Alpaca-PaLM 2 highlights the benefits of our training data over the Alpaca dataset. For coherence and fluency, all models achieve scores over 1.93 with strong ability to generate unambiguous and logic coherent text. The results suggest that the automatic metrics and human evaluation are quite consistent.

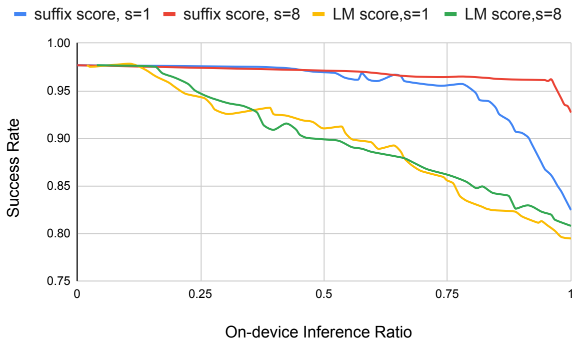


Figure 3: Comparing Suffix Score with LM scores when cascading our model with InsGPT.

6 Conclusion

In this paper we provided an effective approach to build an on-device rewrite model that does not rely on human-labeled data or preference data. We introduced MESSAGEREWRITEEVAL, a new human-labeled benchmark that focuses on text rewriting for messages through natural language instructions. We also developed an efficient and effective cascading approach using distillation of critiques. Through experiments, from both automatic metrics and human evaluations, we demonstrated that our on-device model outperforms the current state-of-the-art models in text rewriting despite having a much smaller size. Furthermore, cascading our model with the server side model can further boost its performance.

7 Limitations

Our paper experiments is based on PALM 2, whose technique details is not open sourced. Thus we can only share a rough and relative size compared to all baselines but can not disclose the exact number of parameters. Besides the authors' affiliation is not permitted to run LLaMA 2 models due to Meta's license, thus we can not disclose its metrics as our baselines.

8 Ethical Discussion

Our work does not collect any user information nor produces any harmful output. We mention it helps improving privacy as on-device model does local inference and thus reduce the chance of privacy leaking.

References

Simran Arora, Avanika Narayan, Mayee F Chen, Laurel J Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher Ré. 2022. Ask me anything: A simple strategy for prompting language models. *arXiv preprint arXiv:2210.02441*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. 2022. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10925–10934.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Dave Burke. 2023. Express yourself on android, with help from ai. https://blog.google/products/android/new-android-features-generative-ai/.

Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. 2023. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023a. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Yu-Hui Chen, Raman Sarokin, Juhyun Lee, Jiuqiang Tang, Chuo-Ling Chang, Andrei Kulik, and Matthias Grundmann. 2023b. Speed is all you need: On-device acceleration of large diffusion models via gpu-aware optimizations.

Ching-An Cheng, Andrey Kolobov, and Adith Swaminathan. 2021. Heuristic-guided reinforcement learning. *Advances in Neural Information Processing Systems*, 34:13550–13563.

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.

Aleksandr Chuklin, Aliaksei Severyn, Daniil Mirylenka, Eric Emil Malmi, Felix Stahlberg, Jakub Adamek, Jonathan Stephen Mallinson, Sebastian Krause, Shankar Kumar, and Yue Dong. 2022. Text generation with text-editing models. In *Proceedings of NAACL 2022*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

David Dohan, Winnie Xu, Aitor Lewkowycz, Jacob Austin, David Bieber, Raphael Gontijo Lopes, Yuhuai Wu, Henryk Michalewski, Rif A Saurous, Jascha Sohl-Dickstein, et al. 2022. Language model cascades. *arXiv preprint arXiv:2207.10342*.

Jane Dwivedi-Yu, Timo Schick, Zhengbao Jiang, Maria Lomeli, Patrick Lewis, Gautier Izacard, Edouard Grave, Sebastian Riedel, and Fabio Petroni. 2022. Editeval: An instruction-based benchmark for text improvements. *arXiv preprint arXiv:2305.01645*.

Felix Faltings, Michel Galley, Gerold Hintz, Chris Brockett, Chris Quirk, Jianfeng Gao, and Bill Dolan. 2020. Text editing by command. *arXiv preprint arXiv:2010.12826*.

Georgi Gerganov. 2023. Port of facebook's llama model in c/c++. https://github.com/ggerganov/llama.cpp. Accessed: 2023-03-15.

Trudy L Hanson, Kristina Drumheller, Jessica Mallard, Connie McKee, and Paula Schlegel. 2010. Cell phones, text messaging, and facebook: Competing time demands of today's college students. *College teaching*, 59(1):23–30.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. True: Re-evaluating factual consistency evaluation. In *Proceedings of the Second DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering*, pages 161–175.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*.

Fotis Iliopoulos, Vasilis Kontonis, Cenk Baykal, Gaurav Menghani, Khoa Trinh, and Erik Vee. 2022. Weighted distillation with unlabeled examples. *Advances in Neural Information Processing Systems*, 35:7024–7037.

Robert Iv, Alexandre Passos, Sameer Singh, and Ming-Wei Chang. 2022. Fruit: Faithfully reflecting updated information in text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3670–3686.

Junmo Kang, Wei Xu, and Alan Ritter. 2023. Distill or annotate? cost-efficient fine-tuning of compact models. *arXiv preprint arXiv:2305.01645*.

VI Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

Lei Li, Yankai Lin, Deli Chen, Shuhuai Ren, Peng Li, Jie Zhou, and Xu Sun. 2020. Cascadebert: Accelerating inference of pre-trained language models via calibrated complete models cascade. *arXiv preprint arXiv:2012.14682*.

Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. 2021. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 human language technology conference of the North American chapter of the association for computational linguistics*, pages 150–157.

Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2022. Edit5: Semi-autoregressive text-editing with t5 warm-start. *arXiv preprint arXiv:2205.12209*.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2018. Sentence compression for arbitrary languages via multilingual pivoting. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2453–2464, Brussels, Belgium. Association for Computational Linguistics.

Philip May. 2021. Machine translated multilingual sts benchmark dataset.

MG Sarwar Murshed, Christopher Murphy, Daqing Hou, Nazar Khan, Ganesh Ananthanarayanan, and Faraz Hussain. 2021. Machine learning at the network edge: A survey. *ACM Computing Surveys (CSUR)*, 54(8):1–37.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Alex Passos, Andrew Dai, Bryan Richter, Christopher Choquette, Daniel Sohn, David So, Dmitry (Dima) Lepikhin, Emanuel Taropa, Eric Ni, Erica Moreira, Gaurav Mishra, Jiahui Yu, Jon Clark, Kathy Meier-Hellstern, Kevin Robinson, Kiran Vodrahalli, Mark Omernick, Maxim Krikun, Maysam Moussalem, Melvin Johnson, Nan Du, Orhan Firat, Paige Bailey, Rohan Anil, Sebastian Ruder, Siamak Shakeri, Siyuan Qiao, Slav Petrov, Xavier Garcia, Yanping Huang, Yi Tay, Yong Cheng, Yonghui Wu, Yuanzhong Xu, Yujing Zhang, and Zack Nado. 2023. Palm 2 technical report. Technical report, Google Research.

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

Natalie Pennington, Amanda J Holmstrom, and Jeffrey A Hall. 2022. The toll of technology while working from home during covid-19. *Communication Reports*, 35(1):25–37.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 129–140.

Emily Reif, Daphne Ippolito, Ann Yuan, Andy Coenen, Chris Callison-Burch, and Jason Wei. 2021. A recipe for arbitrary text style transfer with large language models. *arXiv preprint arXiv:2109.03910*.

Parker Riley, Noah Constant, Mandy Guo, Girish Kumar, David Uthus, and Zarana Parekh. 2020. Textsettr: Few-shot text style extraction and tunable targeted restyling. *arXiv preprint arXiv:2010.03802*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2022. Multitask prompted training enables zero-shot task generalization. In *ICLR 2022-Tenth International Conference on Learning Representations*.

Timo Schick, Jane Dwivedi-Yu, Zhengbao Jiang, Fabio Petroni, Patrick Lewis, Gautier Izacard, Qingfei You, Christoforos Nalmpantis, Edouard Grave, and Sebastian Riedel. 2022. Peer: A collaborative language model. *arXiv preprint arXiv:2208.11663*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Lei Shu, Liangchen Luo, Jayakumar Hoskere, Yun Zhu, Canoee Liu, Simon Tong, Jindong Chen, and Lei Meng. 2023. Rewritelm: An instruction-tuned large language model for text rewriting. *arXiv preprint arXiv:2305.15685*.

AB Siddique, Samet Oymak, and Vagelis Hristidis. 2020. Unsupervised paraphrasing via deep reinforcement learning. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1800–1809.

Ryan Smith, Jason A Fries, Braden Hancock, and Stephen H Bach. 2022. Language models in the loop: Incorporating prompting into weak supervision. *arXiv preprint arXiv:2205.02318*.

Felix Stahlberg, Aashish Kumar, Chris Alberti, and Shankar Kumar. 2022. Conciseness: An overlooked language task. In *Proceedings of the Workshop on Text Simplification, Accessibility, and Readability (TSAR-2022)*, pages 43–56, Abu Dhabi, United Arab Emirates (Virtual). Association for Computational Linguistics.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

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.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Alexey Tikhonov, Viacheslav Shibaev, Aleksander Nagaev, Aigul Nugmanova, and Ivan P Yamshchikov. 2019. Style transfer for texts: Retrain, report errors, compare with rewrites. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3936–3945.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? gpt-3 can help. *arXiv preprint arXiv:2108.13487*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022a. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022b. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *arXiv preprint arXiv:2204.07705*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. 2023. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *arXiv preprint arXiv:2304.14402*.

Tongshuang Wu, Michael Terry, and Carrie Jun Cai. 2022. Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts. In *Proceedings of the 2022 CHI conference on human factors in computing systems*, pages 1–22.

Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. 2022. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. *Advances in Neural Information Processing Systems*, 35:3082–3095.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Wei Xu, Alan Ritter, William B Dolan, Ralph Grishman, and Colin Cherry. 2012. Paraphrasing for style. In *Proceedings of COLING 2012*, pages 2899–2914.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*.

Yi Zhang, Tao Ge, and Xu Sun. 2020. Parallel data augmentation for formality style transfer. *arXiv preprint arXiv:2005.07522*.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *CoRR*, abs/1909.08593.

A Appendix

A.1 MessageRewriteEval Data

Statistics of the MessageRewriteEval are located in Table 5. For every task and the complete dataset, we offer the following details: sample counts; the average word length for instruction (Ins), source (Sou), and target (Tar); the average length ratio (Len Ra) of the target over the source; and the Edit Ratio (Edit Ra, refer to Section Automatic Evaluation Metrics). All these statistical measurements are based on words. Additionally, NLI scores between the source and the golden target are available in both directions: from source to target and from target to source. Besides, samples of the instructions for each task in MessageRewriteEval are presented in Table 6.

							NLI	
	Size	Ins	Sou	Tar	Len Ra	Edit Ra	s-t	t-s
Formalize	177	5.42	8.86	12.3	1.3	0.26	0.79	0.83
Shorten	221	5.33	9.65	5.92	0.6	0.21	0.9	0.88
Elaborate	206	5.76	9.42	29.27	3.1	0.15	0.95	0.8
Paraphrase	151	3.83	9.58	10.83	1.1	0.21	0.9	0.88
Proofread	280	11.64	10.88	10.24	0.94	0.12	0.95	0.96
All	1035	6.92	9.79	13.54	1.38	0.18	0.91	0.88

Table 5: Statistics of MessageRewriteEval.

Task	Instruction Examples
Formalize	Make the text formal. Make this sentence more formal. Formalize the text. Rewrite this sentence in a more formal way.
Shorten	Make the text more concise. Rewrite this text in concise language. Make the text shorter. Make this sound more concise
Elaborate	Make this more verbose. Expand this text. Rephrase this sentence in a more expand style. Make the text more elaborated.
Paraphrase	Rewrite this sentence. Rephrase the text. Paraphrase the following text. Rewrite, reword and reorganize. way.
Proofread	Fix the grammar error or spelling error of the following text. Correct the following sentence if there is any spelling or grammar error. Please proofread this sentence.

Table 6: The instruction samples for each task of MessageRewriteEval.

A.2 Data Guidelines

During the data donation and review process for MessageRewriteEval, the follow guideline is provided:

- Content should be preserved in target from source.
- For certain rewrite task, the target should follow the requirement in the instruction.
- *Formalize*: the target should be more formal compared to source including: (1) formal vocabulary, (2) impersonal expression and (3) standard grammatical forms.
- *Shorten*: the target is simpler, more concise compared to source preserving the tone and format from the source.
- *Elaborate*: the target extend the source with more relevant information and ideas but the same tone and format as the source. The relevant information should not be made up facts.
- *Paraphrase*: the target changes the wording of the source while preserving the content, tone, format and verbosity.
- *Proofread*: the target fixes the grammar and wording errors in the source text.

A.3 EditEval Dataset

According to EditEval license page[3], it is permitted with the following: Commercial use, Modification, Distribution, and Private use.

The rewrite task and dataset information in EditEval benchmark can be found in Table 7. The two datasets for *Updating* task are paragraph level, while the rest datasets are all sentence level.

Task	Dataset	Abbrev.	Size
Fluency	JFLEG	JFL	747
Fluency	ITERATOR	ITR_{FLU}	203
Clarity	ITERATOR	ITR_{CLA}	342
Coherence	ITERATOR	ITR_{COH}	76
Simplification	ASSET	AST	359
Simplification	TurkCorpus	TRK	359
Paraphrasing	STS	STS	97
Neutralization	WNC	WNC	1000
Updating	FRUIT	FRU	914
Updating	WAFER-INSERT	WFI	4565

Table 7: EditEval Dataset Statistics

[3] https://github.com/facebookresearch/EditEval/blob/main/LICENSE

Rewrites	NLI σ_1	Reverse NLI σ_2	Length Ratio σ_3	Edit Dist σ_4	Ngram Freq σ_5
Formalize	1.0	1.0	0.0	0.4	1.0
Shorten	1.0	0.4	-0.2	0.4	1.0
Elaborate	0.4	1.0	0.5	0.4	1.0
Paraphrase	1.0	1.0	0.0	0.4	1.0
Proofread	1.0	1.0	0.0	0.0	1.0

Table 8: The choice of $sigma_i$. For formalize and paraphrase, the length ratio is not considered important while for proofread/grammar correction, we apply the additional logic that the length ratio should be close to 1.

A.4 Hyper-parameter Setting

During supervised finetuning, SFT, we use 8 Tensor Processing Units (TPU) V3 chips for fine-tuning. The batch size is 64, and the maximum training step is 30000. We use the Adafactor optimizer (Shazeer and Stern, 2018) with a learning rate of 0.003. Both the input and output sequence lengths are set to 1024 tokens. The training dropout rate is 0.05. For reinforcement learning, we compute the heuristic reward with parameters in 8. We use same setup as fine-tuning except that the training step is 3000. During inference, the temperature is set to 0.5. Unless specifically noted, we use sampling decoding with sample number 8 for our experiments.

A.5 Training Data Stats

We share the detailed training data stats in Table 9. We splitted the data 8:1:1 as Train:Eval:Text during the training.

A.6 Training Data Samples

In Table 10 We share some samples from our training dataset following the method described in Section 3.1.1.

A.7 MessageRewriteEval Samples with Model Outputs

We share some samples from our MessageRewriteEval in Table 11. At the same table, we share the outputs from two models, both finetuned on PaLM2 XXS. The first one is finetuned with Alpaca dataset (Alpaca PaLM2) and the second one is finetune with our synthetic dataset, the statistic numbers of these two models can be found in Table 1 and Table 2.

A.8 Human Evaluation Guideline

We follow the same human evaluation guideline as the RewriteLM paper (Shu et al., 2023).

Instruction Success: The ability of the model to adhere to the given instruction is evaluated in this criterion. It is:

- Score 2 (Fully/Mostly Followed): if the model output entirely adheres to the provided instructions, demonstrating a clear understanding and implementation of the given task. Or the output mostly adheres to the instructions, with minor deviations or errors.

- Score 1 (Partially Followed): if the model output shows some adherence to the instructions but deviates significantly in certain aspects or fails to completely implement them, leading to partial fulfillment of the task.

- Score 0 (Not Followed/Mostly Ignored): if the model output largely ignores the provided instructions, making it evident that the task has not been understood or implemented properly. Or despite some slight adherence, the output largely deviates from the intended task as per the instructions.

Content Preservation: The essential content and meaning of the reference is preserved in the rewrite, independent of its style or the quality of the writing. It is:

- Score 2 (Fully/Mostly Preserved): if the rewrite is an excellent representation of the content in the reference, with no omissions. Or the rewrite mostly matches the content of the reference, but one or two elements of the meaning have been lost.

- Score 1 (Half Preserved): if some of the content is present in the rewrite but approximately the same amount is missing.

- Score 0 (Not Preserved/Mostly Lost): if the rewrite is entirely unrelated to the reference. Or despite some slight similarities, the rewrite is hard to recognize as being based on the reference.

Factuality: The rewrite only provides as much information as is present in the reference, without adding anything. It is not misleading and does not make any false statements (unless these were also present in the reference).

- Score 2 (Fully/Mostly faithful): Everything in the rewrite is grounded in the reference. Or the

	Size	Inst Len	Src Len	Tar Len	Len Ratio	Edit Dist	Edit Ratio	Rouge1
All	155676	7.25	30.06	34.47	1.10	20.06	0.30	93.86
Long form	24384	6.85	118.86	141.09	1.20	115.44	0.97	60.95
short form	131292	7.33	13.60	14.69	1.08	2.35	0.18	99.97

Table 9: Instruction Tuning Training Data Statistics.

	Comment	Source	Target
Formalize	Make this sound more formal	The meeting will be at 8 p.m.	The conference will commence at eight in the evening.
Shorten	Make this sound more concise	70 per cent of the total market share.	70% of the market.
Elaborate	Elaborate the sentences.	Sunny. High near 15C. Winds SSW at 10 to 15 km/h.	The weather is sunny. The high temperature is near 15C. Wind comes from SSW at 10 to 15 km/h.
Paraphrase	Rephrase the text	If someone is an enemy of my enemy, then that person is my friend.	The enemy of my enemy is my friend.
Proofread	Please proofread this sentence	I can help uou my love	I can help you, my love.

Table 10: Samples of training dataset generated following the method in Section 3.1.1

rewrite says something that is not mentioned in the reference or contradicts the reference, but it is not an important addition or it is hard to say whether the statement is true or false.

- Score 1 (Partly faithful): The rewrite adds significant factual statements to the reference. These may be inaccurate or otherwise not based on the reference, but do not entirely undermine the faithfulness of the rewrite as a whole.

- Score 0 (Not/Slightly faithful): The rewrite is mostly wrong, made up, or contradicts what is in the reference text."

Coherence: The rewrite is coherent if, when read by itself (without checking against the reference), it's easy to understand, non-ambiguous, and logically coherent. On the other hand, the rewrite is not coherent if it's difficult to understand what it is trying to say.

- Score 2 (Good): The whole of the rewrite is mostly fluent and easy to read, independent of any reference content. Some specific parts of the rewrite could be more naturally phrased, but overall it is fairly clear and easy to understand.

- Score 1 (Neutral): The rewrite is comprehensible, though not on the first read or only with some effort.

- Score 0 (Bad): The rewrite is very hard to understand, except by checking against the reference.

Fluency: The rewrite is considered fluent if it follows all the rules of its language, including spelling, grammar and punctuation. It reads as though it was written by someone who speaks English as their first language.

- Score 2 (Flawless/Good): The rewrite is grammatically correct, contains no spelling errors, and follows all other linguistic rules. An average English speaker would not see anything that looks "wrong". Or there are just one or two linguistic errors or non-standard formulations, but nothing serious.

- Score 1 (Flawed): The rewrite contains a number of errors of different types, but these errors, even when taken together, do not make the text significantly harder to understand.

- Score 0 (Poor): The rewrite contains a large number of errors, so that some sections of the text are hard to understand, but other parts are more manageable.

A.9 On-device Inference Metrics

To demonstrate the effectiveness of running our models using limited resources, we obtain benchmark numbers on popular mobile phones to obtain two primary metrics: **Inference Latency per Token**, measured in milliseconds, and **Memory Consumption**, quantified in gigabytes during model operation. We introduce an inference engine utilizing OpenCL that harnesses the computational capabilities of on-device GPUs. We adopt similar optimizations reported in (Chen et al., 2023b) and further devise special kernels tailored for our

	Comment	Source	Alpaca PaLM2	Ours
Formalize	Make this sound more formal.	He was sleepy.	He had been sleeping for a long time.	He was drowsy.
Shorten	Rewrite this text more concise.	I am not in mood of going to dinner.	I have no desire to eat dinner.	I don't want to go to dinner.
Elaborate	make this more verbose.	I'll be in Lisbon in May	I will be in Lisbon during the peak tourist season, which runs from May to August.	I'll be in Lisbon in May. I'm really looking forward to it!
Paraphrase	Rewrite the text another way.	No one wants to come with me tonight.	Not a single person is willing to join me tonight.	I don't have anyone to go with tonight.
Proofread	Please proofread this sentence.	It was allmess...	It was all right.	It was all a mess ...

Table 11: Samplse from MESSAGEREWRITEEVAL, and the outputs from models trained by Alpaca vs our training dataset.

	S23		Pixel 7	
	8-bits	4-bits	8-bits	4-bits
P. Parsing (ms)	1.2	1.2	4.2	4.2
Decoding (ms)	48.0	35.0	67.7	55.6
Memory (Gb)	1.6	0.9	1.7	1.0

Table 12: Benchmark results of our model. The average latency per token for the prompt parsing and decoding phases are reported in milliseconds. The last row shows the total memory consumption in gigabytes.

on-device Instruct-oriented models. To accommodate models within constrained memory capacities, we employ 8-bit post-training quantization as the standard setting for reporting quality metrics. The latency/memory numbers of both 8- and lower-bit quantized model are presented to compare with commonly adopted configurations. We note that the quality implication of lower-bit quantization and quantization-aware training is beyond the scope of this paper.

Table 12 presents the performance benchmarks of our inference engine on both the Samsung S23 and Pixel 7 Pro. These evaluations were conducted using 1024 input tokens and decoding over 100 tokens. Results for both 8-bit and 4-bit quantized models are provided. It is noteworthy that, on the S23, the mean latency per token during the prompt parsing phase is 1.2ms (equivalent to >800 tokens/second), with the decoding latency being 35ms (29 tokens/second). To the best of our knowledge, the latency of our model on a cell phone is greatly faster than the reported numbers (i.e. 18 - 22 tokens/second) benchmarked on Macbook M1 Pro 32GB Ram for a 7B Llama model with 4-bits quantization (Gerganov, 2023).

A.10 Impact of cascading to the inference latency

We profiled the latency by comparing the cascading method with the InsGPT model in Table 13. As most LLMs are hosted on server, we use the InsGPT as base for evalution, and we can achieve over 96% performance with 74% less latency or over 97.5% performance with 52% less latency.

	Sari	Bleu	Update-R	Latency
InsGPT	51.14	35	58.91	100%
Our Model	48.6/95.0%	32.43/92.7%	55.72/94.6%	13%
15% InsGPT	49.03/95.9%	33.76/96.5%	57.41/97.5%	26%
40% InsGPT	49.87/97.5%	34.59/98.8%	58.87/99.9%	47.8%

Table 13: Quality vs Latency Tradeoff.

A.11 Few-shot Prompts for Pre-trained LLM Inference

The few-shot prompt in Table 14 is used to facilitate the pre-trained LLM inference for text rewriting tasks. This is only used for evaluation of our baseline Pre-trained LLMs.

A.12 Few-shot CoT Prompts for LLM Critique

The few-shot CoT prompts are used in order to leverage the off-shelf LLMs capability to judge the quality of rewrite. The prompt sample is in Table 15.

Few-shot Prompt
Q: make the text more formal: Just ninety-seven more words and I’m done. Why am I misspelling “ninety?” It seems like I shouldn’t misspell it. I should get it right at least once in this paper. There I did it! That gets me within sixty words of being done. It’s funny how I am panicking about my spelling of “panicking.” Wow, that was really not a great sentence. This whole thing has been a strange journey of enlightenment and glee. Yes, I said “glee.” Sixteen more words, and I’ve done it! Keep going, Mitch! Wait, am I not supposed to use my name? Or am I? This is crazy. A: I have to write an additional 97 words before my task is complete. I need to inquire as to why I have misspelled “ninety.” It is a word that I should have the ability to spell correctly. This paper should feature the word’s correct spelling at least once. It is amusing that I am rendered anxious due to my inability to spell “anxious” correctly. Surprisingly, the sentence that I recently produced was not high caliber. This task has led me down an odd path of enlightenment and glee. I assure you I said ”glee.” I need to write 16 more words until my task is complete. Maintain your focus, Mitch. Can I state my name? This is a deranged undertaking. Q: make the text more concise: You will also want to get all your legal documents in order. This can range from health and safety certificates to contracts from all employees to contractor’s insurance. For a restaurant, you will likely need a large insurance coverage to allow for any claims that could arise from either staff or customers. A: Get your legal documents in order, from health and safety certificates to contracts with employees to contractor’s insurance. For a restaurant, you will need large insurance coverage to allow for any claims from staff or customers. Q: Please elaborate the following sentence: Next, you should have a menu, or an idea of the cuisine you want to serve at the establishment. You can sit down with some professional contracted chefs to design a menu or you can hire a chef and work on a menu together with the person who will run your kitchen. You will also want to interview people for other major roles within the restaurant, like waiters, sous-chefs, porters, bartenders and hosts. A good manager is key to running the establishment, so you want to make sure you have done your homework and found a suitable candidate that can successfully start your restaurant and maintain it at the level that you desire. A: It would be best to have a menu or a framework for the cuisine you want to serve at your restaurant. You can meet with chefs contracted to create a menu, or you can hire a chef and develop a menu together with the person overseeing your kitchen. It would be best if you were inclined to interview people for other primary roles in your establishment. These roles include servers, sous chefs, bartenders, and hosts. A quality manager is vital to running a restaurant. Ensure you have researched and discovered a qualified candidate who can successfully open your restaurant and maintain it at the desired level. Q: Rewrite, reword and reorganize: The issue we need to be aware of is that we can get frustrated trying to find that passion. Though they may work in areas that seem to fit their desires, some think they should wait for the perfect spot before they engage. The idea of “doing something you don’t completely love” is repulsive and a waste of time. A: The issue we need to be cognizant of is that we can get exasperated trying to find that passion. Though they may work in areas that seem to fit their desires, some think they should wait for the perfect spot before they engage. The notion of “doing something you don’t completely love” is revolting and a not worth of one’s time. Q: Please proofread the following sentence: lan om using itjust tobcovwr our car in the winter. A: I plan on using it just to cover our car in the winter. Q:

Table 14: The few-shot CoT prompts for pre-trained LLM inference.

CoT Prompts
Judge whether the #Response rephrases #Context and complete the rewriting task in #Comment. Choose among two choices: GOOD, BAD. #Comment: Make the text more formal. #Context: Do we want to go to movie now? This one should be pretty good. #Response: Want to go to movie? It should be a great one. #Choose (GOOD) or (BAD): BAD #Explanation: Response is not more formal than Context. #Comment: Simplify the text. #Context: Ric Flair had a match against Mitch of the Spirit Squad. All five members of the Spirit Squad were present, so Flair brought out Rowdy Roddy Piper, Money Inc., and Arn Anderson as his backup. Flair's allies kept the Squad in check, enabling Flair to win the match. #Response: Ric Flair defeated Mitch of the Spirit Squad with help from Rowdy Roddy Piper, Money Inc., and Arn Anderson. #Choose (GOOD) or (BAD): GOOD #Explanation: Response is shorter than Context Response preserves overall meaning. #Comment: Elaborate the following text. #Context: Iuter X Vanguard collaboration T-shirt by Giorgio Di Salvo. Octopus print. All Iuter apparel is Made in Italy. #Response: This T-shirt is part of the collaboration between Iuter and Vanguard. It is designed by Giorgio Di Salvo and features an octopus print. All Iuter apparel is Made in Italy. #Choose (GOOD) or (BAD): GOOD #Explanation: Response rephrases and elaborates the context with preserved meaning. #Comment: Paraphrase the source text. #Context: He likes the dogs a lot, according to his parents. #Response: He is fond of the dogs. #Choose (GOOD) or (BAD): BAD #Explanation: Response did not preserve all the meaning of Context. The fact "according to his parents" is missing in Response. #Comment: Fix the grammar and spelling error if there is any. #Context: Native is very fortunate. #Response: Native people are very fortunate. #Choose (GOOD) or (BAD): GOOD #Explanation: Response fix the grammar errors in the Context. #Comment: {comment} #Context: {input} #Response: {output_best} #Choose (GOOD) or (BAD):

Table 15: The few-shot CoT prompt samples for LLM critique. "GOOD" indicates the response is following the instruction of the comment to rewrite the source (context).