ReFF: Reinforcing Format Faithfulness in Language Models Across Varied Tasks

Jiashu Yao[1], Heyan Huang[1], Zeming Liu[2], Haoyu Wen[1], Wei Su[1], Boao Qian[1], Yuhang Guo[1]*

[1]School of Computer Science and Technology, Beijing Institute of Technology
[2]School of Computer Science and Engineering, Beihang University
{yaojiashu, hhy63, haoyuwen, weisu, qianboao, guoyuhang}@bit.edu.cn, zmliu@buaa.edu.cn

Abstract

Following formatting instructions to generate well-structured content is a fundamental yet often unmet capability for large language models (LLMs). To study this capability, which we refer to as format faithfulness, we present FORMATBENCH, a comprehensive format-related benchmark. Compared to previous format-related benchmarks, FORMATBENCH involves a greater variety of tasks in terms of application scenes (traditional NLP tasks, creative works, autonomous agency tasks), human-LLM interaction styles (single-turn instruction, multi-turn chat), and format types (inclusion, wrapping, length, coding). Moreover, each task in FORMATBENCH is attached with a format checker program. Extensive experiments on the benchmark reveal that state-of-the-art open- and closed-source LLMs still suffer from severe deficiency in format faithfulness. By virtue of the decidable nature of formats, we propose to Reinforce Format Faithfulness (REFF) to help LLMs generate formatted output as instructed without compromising general quality. Without any annotated data, REFF can substantially improve the format faithfulness rate (e.g., from 21.6% in original LLaMA3 to 95.0% on caption segmentation task), while keep the general quality comparable (e.g., from 47.3 to 46.4 in F1 scores). Combined with labeled training data, REFF can simultaneously improve both format faithfulness (e.g., from 21.6% in original LLaMA3 to 75.5%) and general quality (e.g., from 47.3 to 61.6 in F1 scores). We further offer an interpretability analysis to explain how REFF improves both format faithfulness and general quality.

Code & Datasets — https://github.com/BITHLP/ReFF

1 Introduction

Recent years have witnessed a significant upsurge in the development and deployment of large language models (LLMs) (Brown et al. 2020; Touvron et al. 2023; Achiam et al. 2023). With their exceptional zero-shot and few-shot capabilities, LLMs have revolutionized the paradigm of language-related tasks, where a question can be understood and solved to the best without task-specific supervision (Radford et al. 2019).

The zero- and few-shot prompting paradigms have introduced a new problem in task solving procedure, namely, the specification of the output format. To elaborate, LLMs' task solving paradigm mandates that users must devise an output format and include it in the prompt as a request for LLMs to adhere to. The format specification holds significant importance in tasks of wide concern, for example:

- Various natural language processing (NLP) tasks, such as named entity recognition, text-to-data conversion, and syntactic parsing, hold rigorous format requirements.
- Creative works, such as poems, intrinsically possess rigorous forms, including acrostic and numerous others.
- LLM-based autonomous agents need strict format adherence to avoid system crashes or dangerous behaviors.

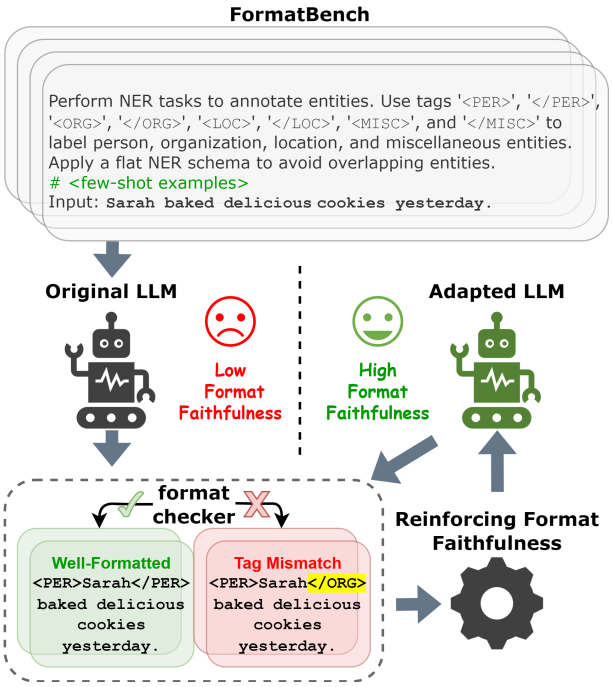

Figure 1: The overall framework of this work. The queries in FORMATBENCH are forwarded to an LLM to generate corresponding responses, whose format correctness are labelled by a format checker. The queries, generated responses, and the format labels are utilized in REFF process to iteratively obtain an adapted LLM with higher format faithfulness.

*Corresponding author

Name	# Test	Application Scene			Interaction Style		Format Type			
		TradNLP	Creat	Robot	Single	Multi	Include	Wrap	Length	Code
CHEM-*	148	✗	✗	✓	✓	✗	✗	✗	✗	✓
S-BENCH	1,727	✗	✓	✗	✓	✗	✓	✗	✗	✓
IFEVAL	541	✗	✓	✗	✓	✗	✓	✓	✓	✗
FOFO	494	✗	✓	✗	✓	✗	✓	✓	✗	✓
FORMATBENCH	24,483	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 1: Comparison between FORMATBENCH and previous format-related benchmarks. FORMATBENCH features a significantly larger test set, offering a greater variety of application scenes, human-computer interaction styles, and format requirement types. The definition of each category is described in detail in Section 3.1.

To summarize, the ability to adhere to pre-defined format specifications is of utmost importance in the deployment of LLMs. This ability, which we refer to as format faithfulness, is a crucial aspect to consider in many real-world tasks.

However, there still exists two significant gaps in studies relating to format faithfulness. Firstly, current datasets related to formatting are primarily focused on one specific task, such as text-to-data (Tang et al. 2023), code generation (Skreta et al. 2023), one-turn instruction (Li et al. 2024), and specialty area documentation (Xia et al. 2024), rather than covering varied tasks. This narrow focus restricts the breadth and reliability of format faithfulness evaluation. Secondly, current adaptation approaches aimed at improving format faithfulness like prompt engineering (Skreta et al. 2023) and finetuning (Tang et al. 2023) neglect the decidable nature of format problems, i.e., whether a response adheres to format requirements can be assessed by a non-parameter format checker. This oversight can result in lower effectiveness, as demonstrated in the following experiments.

To address the gap in comprehensive benchmarks, we combine adaptation of existing datasets, online data collection, and manual data annotation, presenting FORMATBENCH. Compared to previous benchmarks, FORMATBENCH includes not only a significantly larger test set, but also a wider range of tasks in diverse application scenes, interaction styles, and format types. As a result, it obtains a comprehensive evaluation of the format faithfulness of LLMs. Extensive experiments on the benchmark reveal that FORMATBENCH poses significant challenges to even the most capable models with simple format requirements, such as selecting among admissible options in a multi-choice question.

To fill the gap in format adaptation approaches of neglecting format decidability, we propose Reinforcing Format Faithfulness (REFF), as is illustrated in Figure 1. REFF takes full advantage of the decidability of format by using a format checker to judge the format correctness of LLM generated content, and then utilizing the judged data in a reinforcement learning (RL) process to improve format faithfulness. Extensive experiments of REFF on FORMATBENCH yield highly favorable results. Without any annotated data, REFF can significantly improve the format faithfulness rate (e.g., from 21.6% in original LLaMA3 to 95.0% on caption segmentation task), while keep the general quality comparable (e.g., from 47.3 to 46.4 in F1 scores). Combined with labeled training data, REFF can simultaneously improve both format faithfulness (e.g., from 21.6% in original LLaMA3 to 75.5%) and general quality (e.g., from 47.3 to 61.6 F1 scores).

We further combine analyses and examples to explain how REFF is able to obtain highly favorable results in terms of both format faithfulness and general quality. The discussion reveals that although often being consistent and aligned, format faithfulness and general quality of LLMs may also trade off as inversely correlated metrics. As a result, solely improving format faithfulness may cause LLMs generating well-formatted but semantically irrelevant content, while REFF can combine the best to two worlds by involving both metrics.

Our main contributions are summarized as follows.

- For a comprehensive evaluation of format faithfulness, we develop FORMATBENCH, which covers a variety of tasks. Experiments show FORMATBENCH is challenging for state-of-the-art LLMs.
- We propose REFF by incorporating format checking in a reinforcement learning process. REFF is validated to be highly effective in improving format faithfulness with or without extra training data.
- We offer an interpretability analysis to explain how REFF can simutaneously improve both format faithfulness and general quality.

2 Related Work

Format-Related LLM Benchmarks In recent years, there has been significant attention paid to benchmarks and evaluation metrics in language modeling fields. Several notable benchmarks have been developed to evaluate the holistic effectiveness of LLMs (Wang et al. 2018, 2019; Liang et al. 2022; Srivastava et al. 2023). Additionally, a few benchmarks have been proposed to evaluate format-related aspects. However, previous format-related benchmarks are task-specific, failing to provide a comprehensive evaluation of overall format faithfulness. For example, CHEM-* (Skreta et al. 2023) exclusively addresses a domain-specific programming language, STRUC-BENCH (Tang et al. 2023) exclusively addresses text-to-table conversion, IFEVAL (Li et al. 2024) exclusively addresses single-turn instruction, and FOFO (Xia et al. 2024) exclusively addresses specific domain document generation. FORMATBENCH differs from these benchmarks as it covers a variety of tasks, as is shown in Table 1.

Task	Format Requirements	Bad Cases
NER	legal flat NER schema	<PER>Sarah**</ORG>** baked delicious cookies yesterday.
CapSeg	$\leq$ 42 characters per line $\leq$ 2 lines per block	The shimmering lake reflected the colors of **the setting sun.** <eob> The shimmering lake <eol> reflected the colors **<eol>** of the setting sun. <eob>
MTT	adherence to translation rules	src: Das Exanthem des M. Still ist ein Symptom von hoher Sensitivität. rule: "Exanthem" should be translated into "rash" The **exanthema** of Still's disease is a symptom of high sensitivity.
XDL	successful compilation	<!-- a piece of XDL code that **doesn't pass compilation** -->

Table 2: Core format requirements for four tasks in FORMATBENCH, and examples of corresponding wrong responses. Parts of the response that do not meet the format requirements are shown by bolding.

Format-Related LLM Adaptations Before the era of LLMs, controllable text generation (CTG) has been proposed to steer a model to generate desired texts according to given control conditions (Prabhumoye, Black, and Salakhutdinov 2020; Zhang et al. 2023). However, CTG methods usually adopt specially designed finetuning schema or modify the sampling procedure in decoding steps (Miao et al. 2019; Qin et al. 2022; Kumar, Paria, and Tsvetkov 2022), which are intricate for LLMs. Recently, several works adopt prompt engineering (Skreta et al. 2023) or finetuning (Tang et al. 2023) to improve format following ability, but neglect the decidable nature of format problems. Unlike previous work, our proposed REFF is designed based on the decidability of formats, and significantly outperforms previous approaches with or without extra training data.

3 FORMATBENCH

FORMATBENCH is a collection of tasks with formatting requirements, as shown in the example in Figure 1. In this section, we will firstly introduce the variety that FORMATBENCH covers, then outline the benchmark construction, and finally define the metrics associated with the benchmark.

3.1 Variety

FORMATBENCH endeavors to conduct a comprehensive evaluation of format faithfulness. To this end, it covers a variety of application scenes, human-computer interaction styles, and format requirement types.

Application Scene (1) The rigorous output format is necessary for various traditional NLP tasks. (2) In currently prevalent creative tasks, users often devise a format and ask LLMs to follow. (3) Promising LLM-based autonomous agents need to adhere a pre-defined format to interact with the environment. FORMATBENCH combines the all three scenes.

Interaction Style Apart from traditional format specifications in (1) single-turn instructions, FORMATBENCH also involves evaluating format faithfulness in (2) multi-turn interactions, where an LLM iteratively receiving observations and choosing actions to meet the changing format requirements.

Format Types Inspired by previous works (Li et al. 2024; Xia et al. 2024), FORMATBENCH focuses on four aspects of format specifications, namely keyword inclusion, tag wrapping, length constraints, and coding. (1) Inclusion involves certain words to be included or excluded. (2) Wrapping involves enclosing a span of text with pre-defined tags or characters. (3) Length constrains the count of generated content, such as characters or sentences. (4) Coding here refers to more complex format structures that requires a compilers that synthesize the full text.

As is shown in Table 1, FORMATBENCH covers a variety of format-related tasks, and culminating a larger amount of test data compared to previous benchmarks.

3.2 Construction

Source Aiming at cover a variety of format-related tasks, the data construction in FORMATBENCH (Figure 2) involves adaptation of existing datasets, collection of web data, and manual annotation. The task descriptions, data annotation, and quality control are detailed in Appendix A.

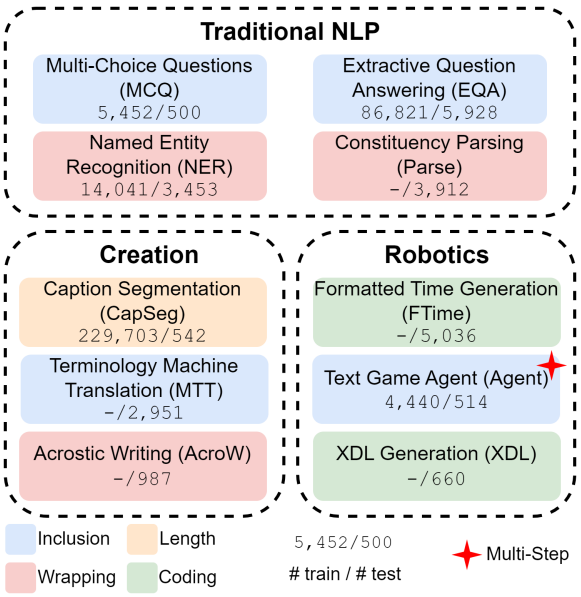

Figure 2: Tasks included in FORMATBENCH with their corresponding groups and data sizes.

Format For each task, we define the format requirements based on previous literature and rough consensus. Some examples and their corresponding cases that fail to satisfy them are listed in Table 2, and all specific format requirements

are listed in Appendix A. Moreover, we construct a corresponding format checker for each task in FORMATBENCH. A format checker is a program, that given an input query and a response, determines whether the response adheres to the format requirements of the task. Formally, given a query q and a generated response r, the format checker is defined as:

$$\mathcal{F}(q,r) = \begin{cases} 1 & \text{if } r \text{ fits the format,} \\ -1 & \text{otherwise.} \end{cases} \quad (1)$$

3.3 Metrics

We associate two metrics to FORMATBENCH, namely, format faithfulness rate and general quality. Format faithfulness rate evaluates to what extent can an LLM $\mathcal{M}$ follows the format specifications, by calculating the format checker pass-rate across the set all samples D:

$$FFR = \mathbb{E}_{q \in D}[\mathbb{1}(\mathcal{F}(q, \mathcal{M}(q)) = 1)]. \quad (2)$$

General quality of the generated responses is also evaluated, as a response is considered effective only if it is both faithful in format and correct in content. Due to the heterogeneity among all tasks in FORMATBENCH, we respectively define the general quality metrics (e.g., BLEU, F1, accuracy) for each task, as described in Appendix A.

4 Reinforcing Format Faithfulness

4.1 Algorithm

Format problems have a decidable nature, where a format checker can easily discriminate whether generated texts adhere to format requirements or not. However, previous methods fail to fully take advantage of this feature.

We find that the format dicidability perfectly fits the reinforcement learning paradigm, where an environment provides rewards for the action given by an agent. To be specific, in a RL-based format faithfulness adaptation, LLMs can be viewed as agents that generating structured texts as actions, which are rewarded by an format checker environment.

In doing so, we proposed REFF to use reinforcement learning for format faithfulness adaptation by rewarding models for correct formats and penalizing incorrect ones.

Algorithm 1: REFF

Input: query set Q, format checker $\mathcal{F}$, LLM $\mathcal{M}$, # epoch n
Output: adapted LLM $\mathcal{M}'$

```
Let M' ← M
for epoch in [1, 2, ..., n] do
    for q in Q do
        r ← M'(q)                          // response generation
        s ← F(q, r)                        // format checking, s ∈ {−1, 1}
        M' ← step(M', q, r, s)             // PPO stepping
    end for
end for
return M'
```

The process of REFF is shown in Algorithm 1, where the $step()$ is the function of reinforcement learning from human feedback style (RLHF-style) stepping aimed at updating the LLM given the action and the reward. The used RLHF-style loss function (Ziegler et al. 2019; Ouyang et al. 2022) differs from the original proximal policy optimization (PPO) (Schulman et al. 2017), in that it additionally adds a Kullback-Leibler (KL) penalty from the original model to prevent the adapted model from shifting too far. The algorithm generally shares the same procedure with RLHF, except that the computation of rewards is different. Specifically, RLHF often uses a pre-trained reward model, while REFF relies on format checkers to compute the rewards. We offer a rigorous math representation of REFF algorithm in Appendix B.

4.2 Settings

Considering the data availability in various real-world scenarios, we set three settings for RL in REFF, whose accessible data groups (query set Q in Algorithm 1) are list in Table 3.

Settings	Test Queries	Train Queries	Train Labels
REFF-tst	✔	✗	✗
REFF-trn	✗	✔	✗
REFF-trn-ft	✗	✔	✔

Table 3: Data used for RL in three settings of REFF.

Test-Only REFF When there exists no extra training data, LLMs can use queries in the test set as the query set Q. Notably, no label of the test set is available to the model in this setting. However, this setting only applies to the offline scenarios, where LLMs handle a batch of queries and generate all responses subsequently.

Train-Only REFF w./wo. Finetuning Train-only setting can be applied in an online scenario, where the queries are processed and responsed one by one, as the adaptation of LLMs only involves training queries as the query set Q. Additionally, considering that a training set often includes both queries and labels, we further study a train-only with finetuning setting, where the reinforcement process is implemented after finetuning on the training set.

5 Experiments

5.1 Baselines

There are two groups of baselines we implement to compare with REFF, namely, refinement and finetuning.

Refinement There exists many works on prompt engineering about augmenting LLMs with internal reflections (Wei et al. 2022; Madaan et al. 2023) to refine their initial content. Among them, a recent paper focuses on generating well-structured codes (Skreta et al. 2023). Inspired by this, we take refinement as a general prompt schema for improving format faithfulness on all tasks. Generally, an LLM iteratively polishes the output format according to error information from the format checker. Optionally, we further augment the refinement process with LLM internal thoughts following the CoT (Wei et al. 2022) and ReAct (Yao et al. 2023) prompting.

Models	MCQ	EQA	NER	Parse	CapSeg	MTT	AcroW	FTime	Agent	XDL	avg.
GPT-3.5	99.0	89.7	95.3	36.2	45.8	56.0	44.5	95.4	71.0	5.5	63.8
LLaMA3	97.0	89.6	84.8	0.2	21.6	52.3	1.7	99.4	88.3	13.3	54.8
Gemma	98.0	90.0	82.0	5.5	28.2	50.9	2.0	98.8	91.4	0.0	54.7
Qwen1.5	96.6	91.3	71.9	4.6	24.7	55.3	0.9	99.0	88.1	10.3	54.3
Mistral	96.0	91.1	86.2	1.6	34.1	40.4	6.9	99.4	86.8	0.0	54.2
Mistral-inst	96.0	89.5	77.5	1.4	32.1	54.2	3.0	97.2	79.0	0.0	53.0
LLaMA2	97.4	86.9	83.9	0.3	25.5	39.9	0.1	99.8	73.7	8.9	51.6
LLaMA	89.6	88.5	74.1	0.3	22.1	29.7	0.0	72.8	81.7	42.4	50.1
Falcon	83.4	77.8	62.7	0.1	26.9	20.5	0.0	32.0	63.2	45.5	41.2
Falcon-inst	83.2	55.4	26.0	0.0	22.7	11.9	0.1	35.9	35.4	54.5	32.5

Table 4: Format faithfulness rate (%) of original models on FORMATBENCH.

Finetuning The abilities of LLMs can be further adapted according to specific goals by finetuning (Zhao et al. 2023). Specifically, recent work (Tang et al. 2023) finetunes LLMs to generate well-structured data on specific tasks. Inspired by these works, we propose to conduct finetuning on LLMs to improve the overall format faithfulness.

5.2 Experimental Setup

Models We conduct a comprehensive evaluation on format faithfulness with FORMATBENCH across many state-of-the-art open-source LLMs sizing about 7B, including LLaMA-7B, LLaMA-2-7B, LLaMA-3-8B, Qwen-1.5-7B, Falcon-7B, Falcon-7B-Inst, Mistral-7B-v0.3, Mistral-7B-Inst-v0.3, Gemma-7B. We further compare these models to closed-source GPT-3.5 (gpt-3.5-turbo-instruct). Note that we only use instruction models, as all tasks in FORMATBENCH are instruction tasks, where instruction prompting styles are more suitable and flexible than chat ones. In adaptation experiments including refinement, finetuning, and REFF, we use LLaMA-3-8B as the base model, as it exhibits a favorable format faithfulness in the original model evaluation.

Adaptation Implementation We use `trl` (von Werra et al. 2020) library to implement the finetuning and the RLHF-style PPO of REFF. More information about the implementation of adaptation methods are detail in Appendix D.

Hyper-Parameters To ensure the robustness and reliability of the results, we try to use default and commonly-used hyper-parameters, and keep them consistent among different experiments. Here we list several key points, and the detailed hyper-parameters are outlined in Appendix D.

- In generation, we adopt greedy decoding in all experiments for a fair and efficient comparison.
- We use LoRA (Hu et al. 2021) in all LLM adaptation experiments with a consistent configuration $r = 16$.
- In fintuning, we use a constant learning rate $2e-5$ and train for 3 epochs with 256 instances per batch.
- In reinforcement learning, we set target of KL divergency to be 6, use a constant learning rate $1.41e-5$, and train for 3 epochs with 32 instances per batch.

5.3 Original Model Results

We evaluate the models using the prompts in Appendix C.

Results The format faithfulness results of original LLMs on FORMATBENCH are presented in Table 4 (general quality in Appendix E). We find the benchmark to be both discriminating and challenging for LLMs. Firstly, it can be observed that stronger model like GPT-3.5 does exhibits better faithfulness to format, as its format faithfulness rates surpasses those of other smaller open-source models. Secondly, it is validated that format tasks are still highly challenging for even the most capable models.

Outliers Moreover, there are some intriguing exceptions found in the results, where smaller models like Falcon-inst demonstrate superior faithfulness compared to GPT-3.5 in XDL task (54.5% versus 5.5%), as shown in Table 4. We try to explain this phenomenon in Section 6 by studying the relation between format faithfulness and general quality.

5.4 Adapted Model Results

In order to adapt LLMs to alleviate format unfaithfulness, we propose REFF by incorporating a format checker into a reinforcement learning process. We further offer three settings in Table 3, namely REFF-tst, REFF-trn, and REFF-trn-ft to cover different application scenarios.

We study the adaptation approaches in test-only setting with four tasks including NER, CapSeg, MTT, and XDL. The corresponding examples are listed in Table 2. These four tasks are chosen for two reasons, (1) they fully covers the application scenes and format types in our proposed taxonomy shown in Figure 2, and (2) their format requirement difficulties are moderate as shown in Table 4. In the train-only setting, we choose the NER and CapSeg tasks as the other two tasks are not attached with training data.

REFF-tst The results of REFF-tst and other baselines in test-only setting are shown in Table 5. Comparing REFF to other approaches, it is obvious that REFF can significantly improve the format faithfulness rate while keep the general quality comparable without any annotated data. Notably, the format faithfulness of REFF exceeds not only its LLaMA3 baselines, but also GPT-3.5, validating the high effectiveness

Models	Format Faifulness Rate (↑)				General Quality (↑)		
	NER	CapSeg	MTT	XDL	NER	CapSeg	MTT
GPT-3.5	95.3	45.8	56.0	5.5	94.3	40.6	30.9
+ refine	96.1	62.5	73.1	5.9	94.3	42.0	31.7
+ refine*	96.7	72.3	83.8	5.9	94.2	23.0	31.5
LLaMA3	84.8	21.6	52.3	13.3	88.3	47.3	32.2
+ refine	85.0	21.8	61.8	13.3	88.3	47.2	17.7
+ refine*	85.6	33.0	69.2	13.3	88.5	21.8	13.7
ReFF-tst-NER	96.7	20.7	58.5	17.6	91.4	**47.8**	33.0
ReFF-tst-CapSeg	87.6	95.0	52.0	15.0	87.9	46.4	31.3
ReFF-tst-MTT	88.8	21.6	**98.2**	13.0	88.4	47.6	31.0
ReFF-tst-XDL	86.2	21.4	51.0	**52.6**	89.0	47.3	31.9
ReFF-tst	**99.7**	**100.0**	97.2	14.8	**93.1**	40.9	**35.3**

Table 5: Format faithfulness rate (%) and general quality (F1 for NER and CapSeg, BLEU-4 for MTT) in the test-only setting. Best results among LLaMA3-based models are bolded. REFF-tst-[task] refers to the model adapting with exclusively corresponding dataset, while REFF-tst mixes and shuffles all four datasets. The asterisk symbol denotes refinement with internal thoughts. The general quality of XDL is not evaluated due to the need for a high level of expert knowledge, as detailed in Appendix A.

of our proposed REFF approach. Moreover, comparing REFF trained on one specific task (REFF-tst-[task]) to that trained on mixed data (REFF-tst), we can find that the catastrophic forgetting phenomenon is not significant in format-related reinforcement learning, as the format faitfhfulness rate of one task doesn't significantly drops when combined with data from other tasks.

REFF-trn When test data is not available beforehand, while there exists training data for adaptation, REFF-trn succeeds in improving format faithfulness to the extent that REFF-tst does, as is shown in Table 6. These results proves the format faithfulness improvement is robust, and does not result from overfitting to the test set.

REFF-trn-ft By combining reinforcement for improving format faithfulness and finetuning for improving general quality, REFF-trn-ft obtains highly favorable results on both metrics, as is shown in Table 6.

Models	FF Rate (↑)		GQ (↑)	
	NER	CapSeg	NER	CapSeg
GPT-3.5	95.3	45.8	94.3	40.6
+ refine	96.1	62.5	94.3	42.0
+ refine*	96.7	72.3	94.2	23.0
LLaMA3	84.8	21.6	88.3	47.3
+ finetune	99.0	38.2	**95.9**	**63.6**
ReFF-trn	**99.8**	**99.8**	92.6	40.9
ReFF-trn-ft	99.2	75.5	95.2	61.6

Table 6: Format faithfulness rate (%) and general quality (F1) in the train-only setting. Best results are bolded. The asterisk symbol denotes refinement with internal thoughts.

Outliers Moreover, similar to the exceptions in the original model results, there also exists outliers in adapted model results. As shown in Table 5, the general quality drops drastically with refinement on LLaMA3 MTT task (from 32.2 to 13.7 in BLEU-4), and REFF-tst-XDL obtains an unusually higher format faithfulness rate (52.6%) on XDL task than its counterpart trained with all four tasks (14.8%). We will also give an explanation to the outliers in Section 6.

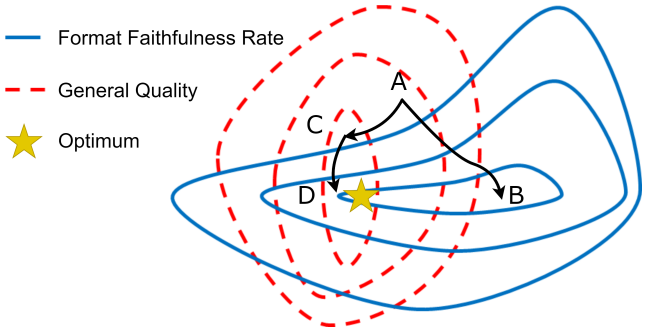

Figure 3: Conceptual contour map of format faithfulness and general quality. Inner circles indicate higher scores for both metrics. Solely improving format faithfulness (A → B) may result in an LLM with high format faithfulness but low general quality. REFF can get the best of two worlds by combining finetuning (A → C) and reinforcement (C → D).

6 Analysis

We begin our analysis by considering the outliers in the results of experiments in Section 5, including:

- Outlier 1: Although considered being more capable, GPT-3.5 does not perform as well as 7B open-source models in format faithfulness on XDL task (Table 4).
- Outlier 2: When using refinement techniquesin LLaMA3, format faithfulness improves significantly, but general quality suffers from a drastic drop on MTT task (Table 5).
- Outlier 3: REFF-tst-XDL significantly outperforms REFF-tst, its counterpart with similar training data, in format faithfulness rate on XDL task (Table 5).

Dimethyl (2RS)-2-(N-(methyloxycarbonyl)amino)-6-oxopimelate (300mg, 1.04mmol) was treated with a solution of **CH2N2** in **diethyl ether** at 0°C for 1h. Solvent was removed in vacuo and the residue purified by flash chromatography (50% EtOAc in hexane, Rf 0.5; 10% CH3CN in CH2Cl2 was also effective, Rf 0.27) to give Dimethyl (2RS, 6RS)-2-(N-(methyloxycarbonyl)amino)-6-epoxymethanopimelate as a colourless oil (273.4mg, 91%).

```
<XDL>
  <Synthesis>
    <Hardware>
      <Component id="beaker"/>
    </Hardware>
    <Reagents>
      <Reagent name="Dimethyl (2RS)-2-(N-(methyloxycarbonyl)amino)-6-
       oxopimelate"/>
      <Reagent name="CH2N2"/>
      <Reagent name="diethyl ether"/>
    </Reagents>
    <Procedure>
      .....
    </Procedure>
  </Synthesis>
</XDL>
```

```
<XDL>
  <Synthesis>
    <Hardware>
      <Component id="beaker"/>
    </Hardware>

    <Reagents>
      <Reagent name="acetic acid"/>
    </Reagents>

    <Procedure>
      .....
    </Procedure>
  </Synthesis>
</XDL>
```

Figure 4: An instance in XDL task (top), the corresponding response of REFF-tst (left), and that of REFF-tst-XDL which obtains a higher format faithfulness rate on XDL task (right). REFF-tst-XDL generates syntactically correct but irrelevant code.

These exceptions have a same pattern, that is, the discrepancy between format faithfulness and general quality. More specifically, a model that is supposed to have higher quality may exhibits poorer format faithfulness. By analyzing the three outliers and the underlying shared pattern, we summarize the relation between format faithfulness and general quality, and then explain how our proposed REFF shows highly favorable results in both metrics.

In this section, we will first offer an insightful illustration to explain the discrepancy between format faithfulness and general quality, and then discuss a specific case of Outlier 3.

6.1 Illustration

Figure 3 illustrates the conceptual relation between format faithfulness and general quality. Although being consistent in most scenarios, format faithfulness and general quality are two different metrics, and sometimes may trade off as inversely correlated indicators.

An LLM can obtain an acceptable general quality, while failing to adhere format requirements, as point A shows. Meanwhile, an LLM can be completely faithful to format, while being poor in general quality, as point B shows. Solely adapting LLMs with the supervision of format faithfulness rates may guide them from the former situation (point A) to the latter one (point B).

The discussion above give all three exceptions an explanation. Point A explains Outlier 1, where strong LLMs like GPT-3.5 generate poor-structured texts. Point B explains Outlier 3, where an LLM may gain a significant format faithfulness improvement without higher general quality. The trace from Point A to Point B explains Outlier 2, where an LLM compromises general quality to improve format faithfulness.

Fortunately, experiments in Table 6 show that our proposed REFF is able to combine the best of both worlds. In the finetuning process in REFF, an LLM is firstly adapted to a position with high general quality (point A to C). In the following, it is futher adapted for better format faithfulness by reinforcement (point C to D). With the application of KL regularization term that prevents the model from shifting too far from the original parameters (point C) in reinforcement process, the LLM after reinforcement (point D) can avoid significant decrease in general quality.

6.2 Examples of XDL Task

To explain why in Outlier 3 the XDL task falls into the point B (high format faithfulness, low general quality), we take a closer look at the example in Figure 4. In the example, REFF-tst-XDL sneakily passes the format checker by generating short and simple well-formatted code (high format faithfulness) that is irrelevant to the instruction (low general quality). The phenomenon is an typical instance of mode collapse in RLHF, characterized by a reduced diversity in produced samples (Casper et al. 2023). RL-based adaptation techniques that may suffer from the mode collapse still have some room for substantially improving the performance in XDL task, as they require the original LLM to produce a number of diverse correctly formatted responses for rewarding.

7 Conclusion

In this paper, we aim to conduct a comprehensive evaluation of format faithfulness and enhance it without compromising the general quality of LLMs. In doing so, we firstly propose FORMATBENCH, a format-related benchmark that covers a variety of tasks. FORMATBENCH is shown to be highly discriminating and challenging for state-of-the-arts LLMs. Subsequently, by utilizing the decidable nature of formats, we incorporate format checking procedures into reinforcement learning to propose REFF. Extensive experiments validate the high effectiveness of REFF in simultaneously enhancing both format faithfulness and general quality. Finally, we provide an interpretability analysis to elucidate the reasons behind REFF's effectiveness by exploring the relationship between format faithfulness and general quality.

Acknowledgements

We'd like to thank all the anonymous reviewers for their diligent efforts in helping us improve this work. This work is supported by the National Natural Science Foundation of China (Grant No. U21B2009) and Beijing Institute of Technology Science and Technology Innovation Plan (Grant No. 23CX13027).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Casper, S.; Davies, X.; Shi, C.; Gilbert, T. K.; Scheurer, J.; Rando, J.; Freedman, R.; Korbak, T.; Lindner, D.; Freire, P.; et al. 2023. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*.

Hu, E. J.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2021. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Kumar, S.; Paria, B.; and Tsvetkov, Y. 2022. Gradient-based constrained sampling from language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2251–2277.

Li, S.; Yan, J.; Wang, H.; Tang, Z.; Ren, X.; Srinivasan, V.; and Jin, H. 2024. Instruction-following Evaluation through Verbalizer Manipulation. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 3678–3692.

Liang, P.; Bommasani, R.; Lee, T.; Tsipras, D.; Soylu, D.; Yasunaga, M.; Zhang, Y.; Narayanan, D.; Wu, Y.; Kumar, A.; et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

Miao, N.; Zhou, H.; Mou, L.; Yan, R.; and Li, L. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6834–6842.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744.

Prabhumoye, S.; Black, A. W.; and Salakhutdinov, R. 2020. Exploring controllable text generation techniques. *arXiv preprint arXiv:2005.01822*.

Qin, L.; Welleck, S.; Khashabi, D.; and Choi, Y. 2022. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35: 9538–9551.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Skreta, M.; Yoshikawa, N.; Arellano-Rubach, S.; Ji, Z.; Kristensen, L. B.; Darvish, K.; Aspuru-Guzik, A.; Shkurti, F.; and Garg, A. 2023. Errors are Useful Prompts: Instruction Guided Task Programming with Verifier-Assisted Iterative Prompting. *arXiv preprint arXiv:2303.14100*.

Srivastava, A.; Rastogi, A.; Rao, A.; Shoeb, A. A. M.; Abid, A.; Fisch, A.; Brown, A. R.; Santoro, A.; Gupta, A.; Garriga-Alonso, A.; et al. 2023. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Tang, X.; Zong, Y.; Zhao, Y.; Cohan, A.; and Gerstein, M. 2023. Struc-Bench: Are Large Language Models Really Good at Generating Complex Structured Data? *arXiv preprint arXiv:2309.08963*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

von Werra, L.; Belkada, Y.; Tunstall, L.; Beeching, E.; Thrush, T.; Lambert, N.; and Huang, S. 2020. TRL: Transformer Reinforcement Learning. https://github.com/huggingface/trl.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 353–355.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35: 24824–24837.

Xia, C.; Xing, C.; Du, J.; Yang, X.; Feng, Y.; Xu, R.; Yin, W.; and Xiong, C. 2024. FOFO: A Benchmark to Evaluate LLMs' Format-Following Capability. *arXiv preprint arXiv:2402.18667*.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*.

Zhang, H.; Song, H.; Li, S.; Zhou, M.; and Song, D. 2023. A survey of controllable text generation using transformer-based pre-trained language models. *ACM Computing Surveys*, 56(3): 1–37.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Ziegler, D. M.; Stiennon, N.; Wu, J.; Brown, T. B.; Radford, A.; Amodei, D.; Christiano, P.; and Irving, G. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.