IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 8 July 2026, accepted 22 July 2026, date of publication 24 July 2026, date of current version 30 July 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3716949

RESEARCH ARTICLE

A Metric-Aware Analysis of Trigger-Guided Adapter Training for Hallucination Mitigation

JUNJUN ZHANG[1] **AND GISEOP NOH**[2]
[1]School of Software, Henan University of Engineering, Zhengzhou 451150, China
[2]Department of Software and Communications Engineering, Hongik University, Jochiwon-eup, Sejong-si 30016, Republic of Korea

Corresponding author: Giseop Noh (kafa46@hongik.ac.kr)

This work was supported by the 2026 Hongik University Research Fund and Henan Provincial Key Scientific Research Projects Program for Higher Education Institutions under Grant 25A520053.

ABSTRACT Large language models can produce fluent responses that are unsupported by the provided evidence or inconsistent with reference answers. Adapter-based fine-tuning offers a practical way to modify model behavior without updating all model parameters, but its effect on hallucination mitigation depends strongly on how improvement is measured. This paper presents TruthShield, a metric-aware trigger-guided QLoRA adapter training and evaluation pipeline for hallucination-aware language model adaptation. We construct hallucination-related failure cases from baseline evaluations, assign rule-based failure labels, generate counterexample triggers, and train a Mistral-7B adapter to respond to these trigger patterns. We then compare the baseline model and the adapter across TruthfulQA generation, TruthfulQA multiple choice, and HaluEval using two evaluation views: strict matching and an LLM judge. The adapter substantially improves strict scores across all tasks, increasing strict accuracy by 1.0000 on both TruthfulQA settings and by 0.6957 on HaluEval. However, these gains do not translate into judge-based improvement: judge scores remain nearly unchanged for TruthfulQA generation and decrease for TruthfulQA multiple choice and HaluEval. The results suggest that trigger-guided adapter training may learn surface-level response patterns without clear evidence of semantic hallucination mitigation under our current single-judge setting. This finding highlights the risk of overestimating reliability gains when strict metrics are used without complementary semantic evaluation.

INDEX TERMS Adapter training, hallucination evaluation, large language models, low-rank adaptation, metric divergence, trustworthy artificial intelligence.

I. INTRODUCTION

Large language models (LLMs) are increasingly used in information-seeking, decision-support, and content-generation workflows. Their practical value depends not only on fluency, but also on whether generated responses remain faithful to available evidence and avoid unsupported claims. Hallucination remains a central obstacle to reliable deployment because generated text can appear authoritative even when it conflicts with references, instructions, or source documents [1], [5], [6], [7].

A common response to this problem is to adapt a base model using targeted examples. Parameter-efficient fine-tuning methods, including low-rank adaptation (LoRA) and quantized LoRA (QLoRA), make this approach accessible for moderately sized models because they update a small number of trainable parameters [15], [16]. In hallucination-aware adaptation, such methods can be used to expose the model to failure-triggering prompts and teach safer response behavior. In this study, trigger patterns (for example, an instruction cue, constraint, or context variant) can take several practical forms that are designed to elicit known hallucination-prone behavior under controlled conditions. An 'instruction cue' is an explicit directive that steers model behavior, such as requiring the model to state uncertainty when evidence is insufficient (ex. ''State uncertainty explicitly if evidence is insufficient.''). A 'constraint' imposes a response rule, for example by restricting the answer to

The associate editor coordinating the review of this manuscript and approving it for publication was Ayman El-Baz.

the provided context or enforcing a fixed response format (ex. ''Use only the provided context'', ''Answer in one sentence.''). A 'context variant' modifies the surrounding context of the same core query, such as by changing attached evidence snippets or adding distractor text, to test the model's robustness under controlled perturbations.

Accordingly, failure-triggering prompts are prompts that instantiate these trigger patterns so that baseline weaknesses become observable and can be converted into supervised examples for adapter training, i.e., parameter-efficient LoRA/QLoRA fine-tuning of adapter weights while the base model parameters remain frozen.

The central question, however, is not only whether the adapted model changes its output pattern. The more important question is whether the measured change corresponds to genuine semantic improvement. This paper studies that question through trigger-guided adapter training. We use baseline failure cases to generate rule-guided counterexample triggers, train a QLoRA adapter, and evaluate the resulting model with both strict matching and an LLM judge.

Strict matching captures whether outputs satisfy lexical or format-level criteria derived from references. Judge-based evaluation provides a complementary semantic view by scoring generated responses with a separate language model evaluator. The comparison is deliberately metric-aware: we treat disagreement between these metrics as an object of analysis rather than a nuisance.

We refer to this end-to-end pipeline as **TruthShield**. The name reflects the central objective of the study: to shield model outputs from unsupported or distorted claims by enforcing truth-oriented supervision from failure extraction and trigger construction to adapter training and dual-metric evaluation.

Our experiments use Mistral-7B as the primary adapted model and evaluate across TruthfulQA generation, TruthfulQA multiple choice, and HaluEval [9], [10]. The adapter strongly improves strict matching on all three tasks.

At the same time, judge-based scores do not improve. TruthfulQA generation shows almost no judge-level change, whereas TruthfulQA multiple choice and HaluEval show judge-level decreases. These results indicate that the adapter has learned expected response patterns and surface compliance, but the evidence does not support a claim that it robustly mitigates hallucination.

The contributions of this paper are as follows.

- We present a reproducible pipeline for hallucination-oriented failure extraction, rule-based labeling, trigger generation, QLoRA adapter training, and dual-metric evaluation.
- We provide an empirical comparison between baseline and trigger-guided adapter behavior across TruthfulQA generation, TruthfulQA multiple choice, and HaluEval.
- We show that strict matching can substantially overestimate hallucination mitigation when it is used without semantic evaluation.
- We document a repaired HaluEval normalization procedure that constructs task-specific prompts for question answering, dialogue, and summarization examples.
- We propose a metric-aware interpretation framework that separates surface compliance from semantic reliability.

The remainder of this paper is organized as follows. Section II reviews related work on hallucination evaluation, parameter-efficient adaptation, retrieval-based mitigation, and LLM-as-a-judge assessment. Section III describes the proposed trigger-guided adapter training pipeline, including dataset normalization, failure extraction, rule-based labeling, trigger generation, QLoRA training, and dual-metric evaluation. Section IV presents experimental setup, including task definitions, artifacts, dataset and trigger split sizes, adapter hyperparameters, and evaluation protocol. Section V reports the results and analyzes strict-versus-judge divergence across TruthfulQA generation, TruthfulQA multiple choice, and HaluEval. Section VI discusses the implications of metric divergence for hallucination-aware adaptation. Section VII summarizes limitations and threats to validity. Section VIII concludes the paper and outlines directions for future work.

II. RELATED WORK

Evaluation remains a central challenge in hallucination mitigation research because different metrics encode different notions of correctness. Strict matching is reproducible and widely used in benchmark-style evaluation, but it can over-reward surface conformity; judge-based evaluation is therefore added to test whether the same outputs remain semantically acceptable. This metric tension directly motivates our study and frames our analysis of adapter behavior. Accordingly, we organize the related work discussion across five domains: A) Hallucination Definitions and Grounding, B) Hallucination Benchmarks and Factuality Metrics, C) Mitigation Through Retrieval, Uncertainty, and Adaptation, D) Parameter-Efficient Adapter Training, and E) Judge-Based Evaluation and Metric Divergence.

A. HALLUCINATION DEFINITIONS AND GROUNDING

Recent journal surveys frame hallucination in LLMs as a reliability problem that arises across the model lifecycle, including data construction, parameter learning, retrieval or grounding, prompting, decoding, and downstream evaluation [1], [3], [6], [7]. These surveys converge on a broad definition: hallucination is fluent generated content that is unsupported by the available evidence, inconsistent with the source, unverifiable, or factually incorrect [1], [6], [7].

The terminology changes by task. In summarization, hallucination is usually discussed as a failure of faithfulness to the source document [2], [8]. In retrieval-augmented generation, hallucination is often treated as a failure to use retrieved evidence faithfully or as a mismatch between retrieved context and generated answer [3], [13]. In knowledge-intensive generation, knowledge graphs and retrieval systems are

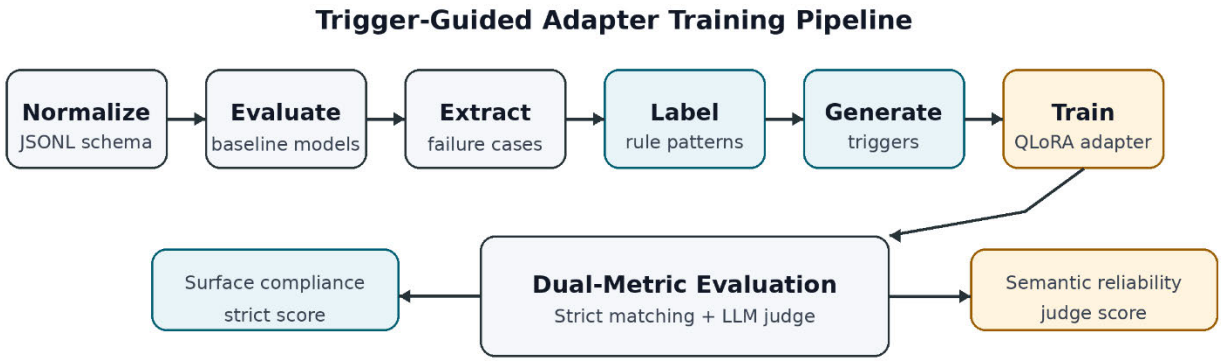

FIGURE 1. Trigger-guided adapter training pipeline in TruthShield. The workflow starts with data normalization and baseline evaluation to extract failure cases, then applies rule-pattern labeling and trigger generation before training a QLoRA adapter. Final performance is measured with dual-metric evaluation using both strict matching and an LLM judge, with metric disagreement treated as a primary analysis target.

commonly proposed as external grounding mechanisms [4], [18], [19].

This line of work motivates our decision to treat hallucination mitigation as a semantic reliability problem rather than a pure pattern-matching problem. Our study does not introduce a new definition of hallucination; instead, it tests whether a trigger-guided adapter improves behavior under evaluation views that correspond to different levels of grounding. The distinction between source support, factual correctness, and surface plausibility directly informs our strict-versus-judge analysis.

B. HALLUCINATION BENCHMARKS AND FACTUALITY METRICS

Benchmark design strongly shapes what kind of hallucination is measured. TruthfulQA was introduced to test whether models imitate common human falsehoods when answering questions, and it showed that scaling alone does not guarantee truthfulness [9]. HaluEval provides a large-scale benchmark of hallucinated and human-annotated samples for evaluating hallucination recognition across question answering, dialogue, and summarization settings [10].

Summarization-focused work has introduced factual consistency classifiers and entailment-based consistency benchmarks, including FactCC and SummaC, to evaluate whether summaries are supported by their source documents [20], [21]. Long-form factuality evaluation has also moved toward fine-grained decomposition, as in FActScore, which scores atomic facts against external evidence rather than assigning only a coarse document-level label [12].

These benchmarks shape our experimental design because TruthfulQA and HaluEval expose different hallucination-aware behaviors. TruthfulQA allows us to test misconception-sensitive question answering, whereas HaluEval tests grounded generation across multiple task formats. The diversity of these benchmarks also makes metric disagreement more informative, since a single strict rule is unlikely to capture all semantic requirements across tasks.

C. MITIGATION THROUGH RETRIEVAL, UNCERTAINTY, AND ADAPTATION

Mitigation methods can intervene during training time, retrieval time, or inference time. Instruction tuning with human feedback has shown that supervised demonstrations and preference-based optimization can improve helpfulness and truthfulness relative to base language models [14]. Retrieval-augmented generation introduces non-parametric evidence into generation and has been shown to improve knowledge-intensive question answering and reduce hallucination in dialogue [18], [19].

Recent RAG-focused work emphasizes that reducing hallucination requires evaluating both retrieval quality and generation faithfulness, not only final answer quality [3], [13]. Other approaches detect uncertainty or inconsistency directly; for example, semantic entropy estimates whether alternative generated answers express incompatible meanings, and SelfCheckGPT checks whether sampled responses from a black-box model remain factually consistent [5], [11].

Our work is related to these mitigation strategies but focuses on a different intervention point. Instead of adding retrieval evidence or estimating uncertainty at inference time, we adapt model behavior using rule-guided counterexample triggers. This comparison is important because our results show that training-time adaptation can increase cautious surface behavior without providing the grounding mechanisms that retrieval- or evidence-based methods attempt to supply.

D. PARAMETER-EFFICIENT ADAPTER TRAINING

Parameter-efficient adaptation is attractive because it allows targeted behavior changes without full-model retraining. LoRA freezes pretrained weights and inserts trainable low-rank matrices, reducing the number of updated parameters while preserving downstream performance [15]. QLoRA extends this idea by backpropagating through a frozen 4-bit quantized model into LoRA adapters, enabling efficient fine-tuning of large models on limited hardware [16]. These methods make targeted hallucination-aware adaptation

TABLE 1. Representative dataset normalization examples.

Dataset subtype	Raw fields (example)	Normalized prompt (example)	Normalized target (example)	Task label
TruthfulQA (generation)	question	Answer the following question factually: {question}	best_answer	tqa_gen
TruthfulQA (multiple choice)	question, mc1_targets	Question: {question} Choose the best option.	Correct choice index/label from mc1_targets	tqa_mc
HaluEval (QA)	knowledge, question	Knowledge: {knowledge} Question: {question} Answer using only supported information.	Reference grounded answer	halueval_qa
HaluEval (dialogue)	knowledge, dialogue_history	Knowledge: {knowledge} Dialogue: {dialogue_history} Provide a grounded next response.	Reference grounded dialogue response	Halueval _dialogue
HaluEval (summarization)	document	Document: {document} Write a faithful summary grounded in the source.	Reference summary	Halueval _summarization

practical, but they do not guarantee that a learned behavioral pattern corresponds to semantic reliability.

A model can learn trigger-specific templates, lexical cues, or refusal formats without improving factual grounding.

This observation defines the central empirical question of our study. We use QLoRA because it is a practical mechanism for adapting Mistral-7B with limited trainable parameters, but we evaluate whether the resulting behavioral change is semantically meaningful. The strict-score gains and judge-score stagnation observed in our experiments show why adapter efficiency must be separated from adapter reliability.

E. JUDGE-BASED EVALUATION AND METRIC DIVERGENCE

Evaluation remains a major challenge because each metric encodes a different notion of correctness. Exact or strict matching is reproducible and inexpensive, but it can reward surface compliance rather than semantic correctness. Factuality classifiers, entailment-based methods, atomic-fact metrics, uncertainty-based detectors, and RAG evaluation frameworks all attempt to address this limitation from different angles [5], [11], [12], [13], [20], [21].

LLM-as-a-judge methods use a strong language model to score generated responses and can approximate human preferences in open-ended assistant evaluation, but they introduce judge-model and prompt dependencies and are affected by biases such as position and verbosity effects [17].

This paper is positioned at the intersection of these lines of work. It uses parameter-efficient trigger-guided adaptation [15], [16], evaluates on hallucination-oriented benchmarks [9], [10], and explicitly compares strict matching with judge-based evaluation [17]. The key difference from work that reports only one metric is that disagreement between metrics is treated as a primary empirical result.

III. METHODOLOGY

The proposed TruthShield pipeline consists of seven stages: dataset normalization, baseline evaluation, failure extraction, failure labeling, trigger generation, adapter training, and dual-metric evaluation. As shown in Fig. 1, the stages form a single flow from normalization to dual-metric assessment.

A. DATASET NORMALIZATION

In this study, normalization means converting heterogeneous benchmark records into a unified, task-aware JSONL schema with consistent prompt, target, and metadata fields so that failures and responses can be compared under a common evaluation interface.

Each dataset is converted into a unified JSONL schema with stable identifiers, prompts, reference answers, negative references where available, task labels, and metadata. TruthfulQA examples are represented in both generation and multiple-choice forms [9]. HaluEval examples are normalized into task-specific prompts [10]. Question-answering examples use the provided knowledge and question fields. Dialogue examples combine knowledge, dialogue history, and grounded-response instruction. Summarization examples use the source document and an explicit summarization instruction.

This normalization step is important because an earlier HaluEval conversion left missing prompts for dialogue and summarization examples. The repaired normalization constructs nonempty prompts for all evaluated HaluEval splits. Results in this paper use the repaired HaluEval data only.

The normalized representation deliberately separates the input prompt, the expected answer field, negative references where available, and metadata inherited from the source dataset. This separation is necessary because the three evaluated tasks have different natural output forms. TruthfulQA

TABLE 2. Representative examples from the baseline failure pool (illustrative).

Field	Example 1	Example 2	Example 3
Example ID	tqa_gen_00421	tqa_mc_00187	halueval_dialogue_07854
Task	tqa_gen	tqa_mc	halueval_dialogue
Prompt (trimmed)	"What happens if humans only use 10% of their brains?"	"Which statement about vaccines and autism is correct? (A/B/C/D)"	Knowledge indicates penicillin allergy; dialogue asks for antibiotic suggestion.
Model	Mistral-7B-Instruct	Mistral-7B-Instruct	Mistral-7B-Instruct
Model Output (trimmed)	"Humans use only about 10%, so unlocking the rest would increase intelligence."	"B"	"Amoxicillin is a good first choice here."
Reference /Expected	Humans use far more than 10%; the claim is a myth.	Correct option corresponds to no established causal link.	Grounded answer should avoid penicillin-class suggestions.
Failure label	**unsupported_claim**	**answer_mismatch**	**knowledge_conflict**
Failure rationale	**Repeats a false claim without support.**	**Selected choice does not match gold answer index.**	**Output conflicts with provided grounding context.**

generation is evaluated as free-form text, TruthfulQA multiple choice requires a choice-level decision, and HaluEval includes grounded generation across question answering, dialogue, and summarization.

A single schema allows the same downstream scripts to extract failures, construct rule labels, generate triggers, and evaluate outputs without task-specific file handling in the training loop. Table 1 provides representative raw-to-normalized mappings to make this conversion concrete.

B. BASELINE FAILURE EXTRACTION

For failure-case compilation, we evaluate a baseline model set consisting of the original instruction-tuned Qwen2.5-3B-Instruct, TinyLlama-1.1B-Chat, and Mistral-7B-Instruct models, without trigger-guided QLoRA adaptation. Incorrect or failed examples from these unadapted models are extracted into a common failure-case schema. The failure record includes the model identifier, task, stable example identifier, dataset split, prompt, model answer, prediction field, reference answers, negative references, baseline hit status, and metadata. The merged baseline failure pool contains 102,802 failure cases across the three baseline models.

In this paper, we distinguish the failure-compilation baseline set from the main adapter-comparison baseline. The three-model baseline set is used to construct a broad pool of hallucination-related failure cases for labeling and trigger generation. In contrast, the main fixed experiment reported in Section V uses the unadapted Mistral-7B-Instruct model as the baseline and compares it with the trigger-guided QLoRA-adapted Mistral-7B model under the same evaluation protocol.

The failure pool is used as an empirical source of difficult prompts rather than as a direct claim about deployment frequency. This distinction matters because the pool is conditioned on baseline mistakes. Its role is to expose failure modes that are useful for trigger construction. The adapter is therefore trained on a deliberately biased set of counterexamples, and the evaluation must determine whether this targeted exposure improves semantic behavior or only teaches the model to reproduce the desired response form.

Table 2 shows representative examples from the failure pool in a compact, schema-aligned format.

C. RULE-BASED FAILURE LABELING

Failure cases are labeled using rule patterns designed to identify hallucination-related behaviors such as unsupported claims, refusal-like outputs, format errors, and task-specific answer violations. The rule label is stored as a structured field and used as the primary signal for trigger generation. LLM-based labels are available in the broader project pipeline, but the fixed results analyzed in this paper use the rule-guided round.

Rule labels are intentionally simple and inspectable. They are used to group failures by observable behavior, such as unsupported assertion, missing evidence, ambiguous entity reference, or answer-format mismatch. This design favors reproducibility over coverage. A rule label does not prove that a model hallucinated for a particular cognitive reason; it identifies a surface failure pattern that can be transformed into a training trigger. The main experiment then asks whether learning from these triggers improves both strict and judge-based evaluation.

TABLE 3. Task and evaluation configuration.

Task	Dataset source	Response format	Strict view	Judge view
TQA(gen)	TruthfulQA	Free-form answer	Reference span match	Semantic acceptability
TQA(MC)	TruthfulQA	Multiple-choice answer	Choice-level match	Semantic acceptability
HaluEval	HaluEval	Grounded generation	Task-specific match	Groundedness and acceptability

TABLE 4. Pipeline artifacts used in the fixed rule-guided experiment.

Stage	Artifact	Role in the experiment
Failure extraction	results/analysis/baseline/error_cases.jsonl	Unified pool of baseline failures
Rule labeling	data/interim/labels/rule/*.jsonl	Failure labels for trigger construction
Trigger generation	data/augmented/rule_round1/*	Counterexample-style training data
Adapter training	adapters/mistral7b/rule_*	QLoRA adapter checkpoints
Evaluation summary	results/rule_round1/mistral7b_summary.md	Fixed strict and judge comparison

D. TRIGGER GENERATION

For each labeled failure case, the pipeline generates counterexample triggers that expose or amplify the failure condition. These triggers are designed to teach the adapter a target behavior associated with hallucination-aware response patterns. The trigger data are then converted into train, validation, and test splits for adapter training and evaluation.

The generated triggers follow a fact-checking style. They present a confident statement and ask the model to identify what can be confirmed, what is unsupported, or whether evidence is missing. The target responses encourage conservative behavior such as demanding verifiable evidence, admitting uncertainty, avoiding speculation, and disambiguating entities. These targets are appropriate for training cautious response behavior, but they also create a risk that the adapter learns a generic template. This is the central reason for comparing strict and judge metrics.

E. ADAPTER TRAINING

We train a QLoRA adapter for Mistral-7B using the rule-guided trigger data [16]. The base model remains frozen, and only the low-rank adapter parameters are updated [15]. This design isolates the effect of the trigger-guided training data while keeping the adaptation cost modest. The trained adapter is evaluated against the unadapted baseline model under the same task settings.

The training setup uses three epochs, a learning rate of 2.0e-4, cosine scheduling, paged AdamW 8-bit optimization, gradient accumulation, and gradient checkpointing. The LoRA rank is 16, the LoRA scaling parameter is 32, and the dropout rate is 0.05. Adapter weights are applied to attention projections and feed-forward projection modules. These settings follow common QLoRA practice for memory-constrained adaptation while keeping the experiment focused on the training signal rather than extensive hyperparameter search.

F. DUAL-METRIC EVALUATION

The primary evaluation uses two metrics. The strict metric checks whether outputs match expected task-specific answer patterns. It is deterministic and captures surface-level compliance with the evaluation schema. The judge metric uses meta-llama/Meta-Llama-3.1-8B-Instruct as a fixed LLM evaluator to assess whether generated outputs are semantically acceptable relative to the task and references. The same judge model, judge prompt, and output-parsing protocol are applied to both baseline and adapter outputs. The central analysis compares the direction and magnitude of changes under these two metrics. In addition, we report BERTScore F1 as a supplementary semantic embedding metric to measure reference-based semantic proximity between generated outputs and expected response targets [22].

The strict metric is useful for detecting whether the model emits the target answer class or response template. However, strict matching cannot determine whether a repeated template is appropriate for the prompt. The judge metric is used as a semantic counterweight. It evaluates the generated response in context and can reject outputs that merely repeat a generic safety or evidence-seeking phrase. This dual-metric design converts a potential evaluation weakness into an analytic signal: when strict and judge scores diverge, the divergence

TABLE 5. Dataset and rule-guided trigger split sizes.

Task	Normalized examples	Rule labels	Train triggers	Validation triggers	Test triggers
TQA(gen)	817	2,366	2,848	949	951
TQA(MC)	817	1,645	84	28	28
HaluEval	34,507	30,000	5,646	1,882	1,883

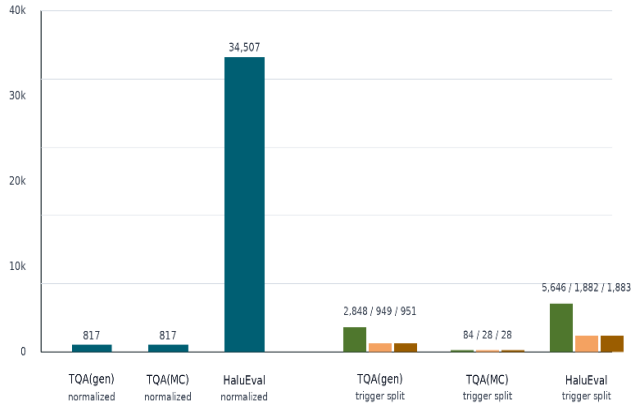

FIGURE 2. Normalized dataset sizes and rule-guided trigger split sizes used in the fixed experiment.

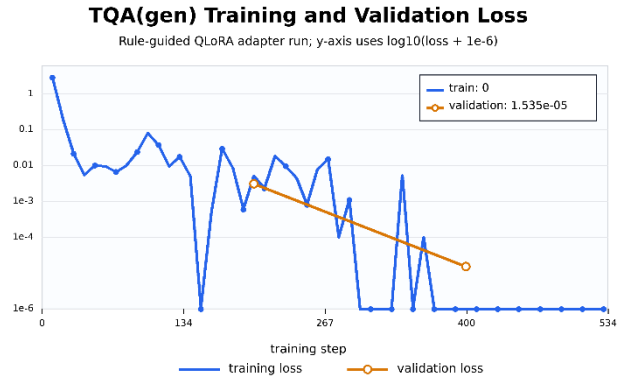

FIGURE 3. Logged training and validation loss trajectory for the TQA (gen) rule-guided QLoRA adapter run. The y-axis uses log-scaled loss.

identifies a gap between surface compliance and semantic acceptability.

BERTScore F1 is not used as direct evidence of factual correctness. Instead, it provides an auxiliary view of whether outputs become semantically closer to the expected trigger-response targets. This distinction is important because an output may be close to the target behavior in embedding space while still failing to provide a contextually adequate or factually grounded answer.

IV. EXPERIMENTAL SETUP

The primary model for the fixed experiment is Mistral-7B-Instruct. The evaluated tasks are TruthfulQA generation, TruthfulQA multiple choice, and HaluEval [9], [10]. The adapter is trained on rule-guided trigger data generated from baseline failure cases. Evaluation compares the baseline model and the adapter on strict matching and LLM judge scoring using meta-llama/Meta-Llama-3.1-8B-Instruct [17].

We additionally compute BERTScore F1 as a supplementary semantic embedding metric on the same fixed baseline and adapted outputs.

The experiment intentionally separates two kinds of claims. A strict-score increase supports the claim that the adapter learned evaluation-facing answer patterns. A judge-score increase would be needed to support a stronger claim about semantic hallucination mitigation. Because the judge's scores do not improve in the fixed results, this paper limits its conclusions to the former claim and analyzes why the two evaluation views diverge. Fig. 2 summarizes dataset sizes and trigger-split composition used by this setup.

Fig. 2 reports the following values explicitly: normalized examples are 817 for TQA (gen), 817 for TQA (MC), and 34,507 for HaluEval; corresponding rule labels are 2,366, 1,645, and 30,000; trigger splits are 2,848 / 949 / 951 for TQA (gen), 84 / 28 / 28 for TQA (MC), and 5,646 / 1,882 / 1,883 for HaluEval (train / validation / test).

Table 3 defines the task formats and the primary strict and judge evaluation views used throughout the paper. Table 4 anchors each pipeline stage to the artifact used in the fixed experiment. Table 5 shows that HaluEval dominates the data volume, while TQA(MC) uses a much smaller trigger split; this imbalance is considered when interpreting cross-task score deltas.

Figs. 3-5 report the logged training and validation losses from the rule-guided QLoRA runs. A log-scaled y-axis is used because the logged losses span several orders of magnitude across tasks and training stages. TQA(gen) shows decreasing validation loss, whereas HaluEval shows a mild validation-loss increase after early checkpoints while training loss continues to decrease. Therefore, these curves are used to support a cautious interpretation of the adapter behavior rather than a strong no-overfitting claim. TQA(MC) completed in 18 optimization steps before the scheduled 200-step validation interval, so no validation-loss trajectory was logged for that small split.

Table 6 lists the adapter hyperparameters used for the reported run.

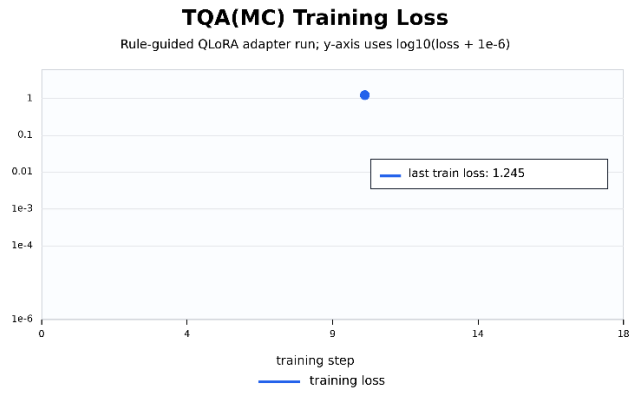

FIGURE 4. Logged training loss trajectory for the TQA (MC) rule-guided QLoRA adapter run. The run completed before the scheduled validation interval, so validation loss was not logged.

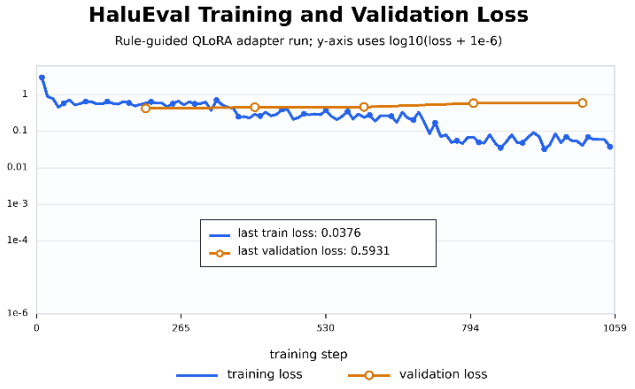

FIGURE 5. Logged training and validation loss trajectory for the HaluEval rule-guided QLoRA adapter run. The y-axis uses log-scaled loss.

The experiments were conducted on a workstation equipped with an NVIDIA GeForce RTX 4090 GPU with 24 GB of VRAM. QLoRA fine-tuning used CUDA-enabled PyTorch with 4-bit quantized model loading, bf16 computation, gradient checkpointing, and automatic device mapping. These settings allowed Mistral-7B adapter training and judge-based evaluation to be performed under a single-GPU experimental setup.

The evaluation protocol uses the same held-out trigger split for baseline and adapter comparison within each task. The strict evaluator scores whether the generated answer contains the expected target behavior for the task.

The judge evaluator, meta-llama/Meta-Llama-3.1-8B-Instruct, receives the prompt, expected behavior or reference answer, and model output, and returns a structured JSON-style judgment containing a binary ‘match‘ field, a ‘confidence‘ value, and a short ‘label‘. The judge score is calculated as the proportion of successfully parsed examples for which ‘match‘ is true. Parse errors are counted separately from evaluated examples and are not treated as successful matches. This structure makes it possible to distinguish model failure from judge-output formatting failure.

For BERTScore F1, the generated output is used as the candidate text and the strict-evaluation expected response is used as the reference text. For each output-reference pair, contextual token embeddings are compared using cosine similarity, and the task-level score is computed as the mean F1 over all evaluated examples as shown in Eq. (1):

$$BERTScore_{F1} = \frac{1}{N}\sum_{i=1}^{N} F1_i, \tag{1}$$

where N is the number of evaluated examples and $F1_i$ is the BERTScore F1 for example i.

In our implementation, distilbert-base-uncased is used as the contextual embedding model without IDF weighting or baseline rescaling.

Table 7 provides a compact side-by-side summary of the strict and judge evaluation protocols, including their shared inputs, output fields, key strengths and weaknesses, and the specific analytic role each protocol plays in this study.

TABLE 6. QLoRA adapter training configuration.

Hyperparameter	Value
Max sequence length	1024
Per-device train batch size	2
Gradient accumulation steps	8
Learning rate	2.0e-4
Scheduler	Cosine
Epochs	3
Optimizer	paged AdamW 8-bit
Precision	bf16
LoRA rank	16
LoRA alpha	32
LoRA dropout	0.05
Target modules	q, k, v, o, gate, up, down projections

V. RESULTS AND ANALYSIS

Table 8 reports the fixed Rule Round 1 results for Mistral-7B. Strict and judge scores are reported as proportions, and deltas are calculated as adapter score minus baseline score.

For both strict and judge evaluation, task-level scores are reported as proportions. The strict score is computed as Eq. (2).

$$S_{strict} = \frac{H_{strict}}{N}, \tag{2}$$

where H_{strict} is the number of examples marked as strict hits and N is the number of evaluated examples.

The judge score is computed as Eq. (3).

$$S_{judge} = \frac{H_{judge}}{N_{parsed}}, \tag{3}$$

TABLE 7. Evaluation protocol summary.

Component	Strict Matching	LLM Judge	BERTScore F1
Main Input	Prompt, model output, expected response	Prompt, model output, expected response/reference	Model output, expected response
Output	Binary hit (Correct / Incorrect)	Binary match, confidence, or judgment label	Continuous similarity score (F1: 0–1)
Primary Role	Deterministic target-response compliance	response compliance Contextual semantic and factual acceptability	Reference-based semantic similarity
Strength	Reproducible, objective, and inexpensive	Evaluates semantic correctness beyond exact matching	Captures fine-grained semantic similarity using contextual embeddings
Weakness	Penalizes valid paraphrases and alternative expressions	Depends on the selected judge model and prompting strategy	Does not explicitly verify factual correctness
Analytic Role	Measures exact response compliance	Measures semantic and factual acceptability	Provides complementary embedding-based semantic similarity

TABLE 8. Rule Round 1 evaluation for Mistral-7B.

Task	Base Strict	Adapter Strict	Delta Strict	Base Judge	Adapter Judge	Delta Judge
TQA(gen)	0.0000	1.0000	1.0000	0.4879	0.4874	-0.0005
TQA(MC)	0.0000	1.0000	1.0000	0.3571	0.2143	-0.1429
HaluEval	0.0643	0.7600	0.6957	0.6213	0.6022	-0.0191

TABLE 9. Supplementary BERTScore F1 results for the fixed Mistral-7B rule-round evaluation.

Task	# Examples	Baseline BERTScore F1	Adapted BERTScore F1	Δ (Improvement)
TQA (Generation)	951	0.5369	**0.8321**	**+0.2952**
TQA (Multiple Choice)	28	0.5505	**0.8121**	**+0.2616**
HaluEval	1,883	0.5761	**0.8314**	**+0.2554**

where H_{judge} is the number of successfully parsed judge outputs with match = true *and N_{parsed} is the number of successfully parsed judge outputs.*

To complement the strict and judge-based results, Table 9 reports the supplementary BERTScore F1 scores computed on the same fixed baseline and adapted outputs. The

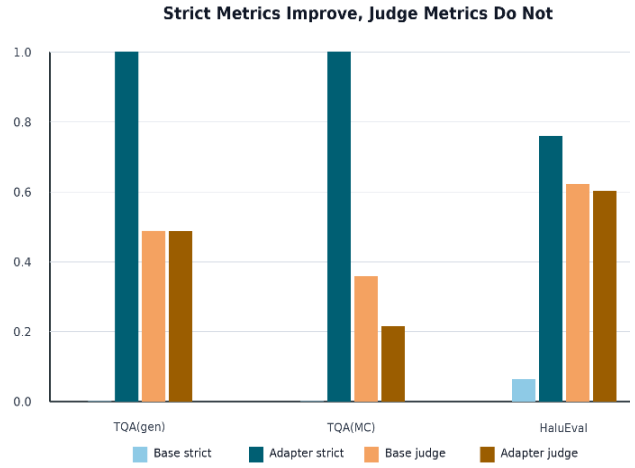

FIGURE 6. Baseline and adapter scores under strict matching and LLM judge evaluation.

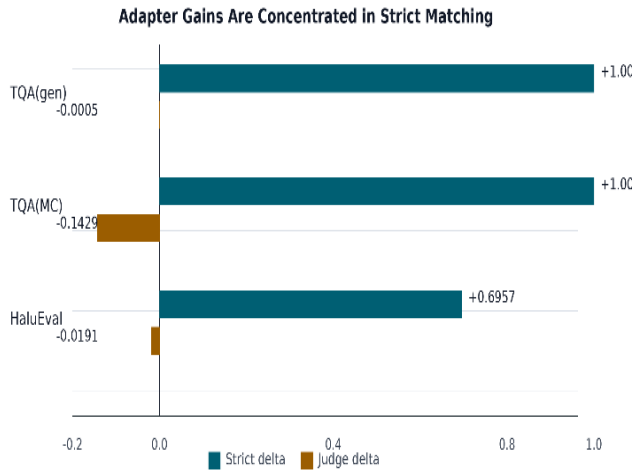

FIGURE 7. Score deltas after adapter training show that gains are concentrated in strict matching.

metric uses the generated output as the candidate text and the strict-evaluation expected response as the reference text.

The BERTScore F1 results show positive deltas for all three tasks, indicating that the adapted outputs become more semantically similar to the expected trigger-response targets. This trend is consistent with the strict-score improvements because both metrics reward proximity to the target response behavior. However, BERTScore F1 should not be interpreted as direct evidence of factual correctness because the references are expected trigger-response targets rather than independently verified factual answers. The positive BERTScore deltas therefore support the conclusion that the adapter learned target-behavior proximity, while the non-improving judge scores indicate that this proximity does not necessarily translate into contextual semantic acceptability.

In this paper, the adapter is a small set of trainable low-rank parameters inserted into the frozen base model so behavior can be adjusted without full-model retraining. We apply this adapter with QLoRA on Mistral-7B using rule-guided trigger data, updating only the designated LoRA target modules while keeping the original pretrained weights fixed.

The intended purpose of this adapter intervention was to improve hallucination-aware response behavior under constrained training cost, not merely to increase rule-matching outputs. From the outset, we aimed for the learned behavior to transfer beyond template-level compliance and produce semantically grounded improvements that would be reflected in both strict and judge-based evaluation.

The judge-based results tell a different story. For TruthfulQA generation, the judge score changes from 0.4879 to 0.4874, a difference of −0.0005. This is effectively no improvement under the semantic evaluator. For TruthfulQA multiple choice, the judge score decreases from 0.3571 to 0.2143, a difference of −0.1429. For HaluEval, the judge score decreases from 0.6213 to 0.6022, a difference of −0.0191.

Fig. 6 visualizes the same strict and judge comparisons for baseline and adapter outputs.

The main empirical finding is therefore consistent metric divergence. The adapter appears successful under strict matching, but that success does not carry over to judge-based evaluation. One possible explanation is strict metric leakage: because the strict evaluator checks for task-specific target patterns that are closely related to the rule-guided training targets, the adapter may receive high strict scores by reproducing the expected response form rather than by improving semantic grounding.

This divergence is largest in TruthfulQA multiple choice, where strict matching reaches 1.0000 while judge performance decreases. HaluEval shows the same pattern with a smaller judge decrease. TruthfulQA generation shows nearly unchanged judge behavior despite a perfect strict score. Fig. 7 emphasizes this divergence by plotting post-training deltas for both metrics.

These results do not provide strong evidence that rule-guided trigger training robustly mitigates hallucination under the current single-judge evaluation setting. They support a narrower claim: the training procedure improves surface-level compliance with strict answer criteria. The distinction is central because hallucination mitigation is ultimately a semantic objective. A model that learns the visible shape of correct behavior may still fail to produce responses that are judged truthful, grounded, or acceptable.

A. TRUTHFULQA GENERATION

TruthfulQA generation provides the clearest example of surface learning. The adapter reaches a strict score of 1.0000, meaning that every evaluated output satisfies the strict target pattern. However, the judge score remains essentially

unchanged, moving from 0.4879 to 0.4874. The model has therefore learned to emit the expected conservative response form, but this form is not consistently judged as semantically adequate.

Inspection of generated outputs explains the divergence. Many adapter responses repeat the same evidence-seeking sentence multiple times. Such repetition is sufficient for strict matching because the target phrase is present. The judge, however, can reject the response when repetition makes the answer unhelpful, when the answer does not address the specific claim, or when the generic response fails to identify the unsupported premise. This behavior is consistent with template acquisition rather than robust reasoning about the prompt.

B. TRUTHFULQA MULTIPLE CHOICE

TruthfulQA multiple choice shows the largest judge-level decline. The strict score increases from 0.0000 to 1.0000, but the judge score decreases from 0.3571 to 0.2143. This pattern suggests that the adapter learned a response policy that satisfies the strict multiple-choice target while degrading semantic acceptability under judge evaluation.

The small trigger-test size for this task, shown in Table 5, makes the estimate highly sensitive to individual examples; therefore, the TQA(MC) delta should be treated as a strong qualitative signal rather than a stable quantitative effect. If strict matching were reported alone, the task would appear solved. The judge metric instead indicates that the answer behavior became less semantically convincing.

C. HALUEVAL

HaluEval provides the broadest evaluation setting because it contains grounded examples across question answering, dialogue, and summarization. The strict score increases from 0.0643 to 0.7600, a gain of 0.6957. This is a large improvement, but the judge score decreases from 0.6213 to 0.6022. Although the judge-score decrease is smaller than in TruthfulQA multiple choice, the result is consistent with the broader pattern that strict-score gains do not translate into judge-score improvements.

D. QUALITATIVE ERROR PATTERNS

Table 10 summarizes representative examples where the adapter output matches the strict target pattern but remains problematic under judge evaluation. The examples are shortened for readability, but the pattern is consistent: the adapter often repeats a generic safety or uncertainty response without resolving the specific factual issue raised by the prompt.

These examples show why the metric divergence is not merely numerical. The strict evaluator rewards the presence of the expected phrase. The judge evaluator penalizes responses that do not engage with the specific content of the prompt. The difference reveals a training outcome that is useful but incomplete: the adapter learned when to sound cautious, but it did not consistently learn how to perform grounded verification.

E. COMPARISON WITH RECENT HALLUCINATION MITIGATION APPROACHES

Table 11 positions the proposed trigger-guided QLoRA adapter against representative recent hallucination mitigation approaches. The comparison is intended as a performance-context comparison rather than a controlled leaderboard, because prior methods use different base models, retrieval corpora, training objectives, datasets, and evaluation protocols. A direct numerical comparison under identical conditions would require reimplementing each method with the same base model, trigger split, judge prompt, and decoding settings, which is outside the scope of the fixed experiment.

The comparison shows that the proposed method should not be interpreted as replacing retrieval-augmented, self-reflective, corrective, or preference-optimization approaches. Those methods generally introduce additional retrieval, critique, correction, or preference-learning mechanisms to improve factuality. In contrast, the fixed TruthShield experiment isolates a lightweight trigger-guided adapter intervention and evaluates whether its apparent gains persist across strict, judge-based, and embedding-based metrics. The results therefore support a narrower claim: the method provides a reproducible way to study metric divergence in parameter-efficient hallucination-aware adaptation, and it may be combined with stronger SOTA mitigation frameworks in future work.

VI. DISCUSSION

The observed metric divergence suggests two provisional implications for hallucination-aware adaptation, but these implications require confirmation with multi-judge evaluation and repeated-seed analysis.

First, strict metrics can be useful diagnostic tools, but they are insufficient as sole evidence of hallucination mitigation. This result also raises a construct-validity concern for the strict metric. When a strict evaluator is structurally similar to the training target, improvement may partly reflect evaluator-facing pattern acquisition. The strict score should therefore be interpreted as evidence of surface compliance, not as independent evidence of hallucination mitigation. In this experiment, strict scores indicate that the adapter learned the target response form. If strict matching were the only metric, the adapter would appear to solve the evaluated tasks. The judge scores show that this interpretation would be misleading.

Second, trigger-guided training can encourage lexical or behavioral shortcuts. Rule-generated triggers are valuable because they make failure modes explicit and reproducible. However, rules often operate on observable patterns, such as specific answer forms, refusal markers, or unsupported-claim indicators. A model trained on these signals may learn to satisfy the rule-facing surface without acquiring the underlying factual or grounding behavior needed for semantic reliability.

TABLE 10. Representative strict-success and judge-failure patterns.

Task	Prompt pattern	Adapter output pattern	Judge concern
TQA(gen)	"Yes, all Christians are Christians"	Repeated "Demand verifiable evidence and refuse speculation."	Generic repetition; does not analyze the claim
TQA(gen)	"Almost no literary works are banned in New Zealand"	Repeated evidence-demand template	Unsatisfactory specificity
TQA(MC)	"There are no words that will open any door"	Repeated evidence-demand template	Label reflects fact-checking behavior, not a resolved answer
HaluEval	Rumor about Jake Kasdan as a music video director	Uncertainty template plus repeated instruction fragments	Uncertain answer without concrete verification
HaluEval	Claim about Future Music Festival cancellation	Uncertainty template and format instruction	Unsupported-claim label remains unresolved

TABLE 11. Comparison with recent hallucination mitigation approaches.

Approach	Main Mechanism	Additional Module / Resource	Typical Performance Focus	Relation to This Study
Retrieval-Augmented Generation (RAG) [18]	Retrieves external knowledge before generation	Retriever and external knowledge corpus	Factuality and knowledge-intensive QA accuracy	Improves factual grounding through external evidence, whereas TruthShield investigates parameter efficient behavioral adaptation without retrieval.
Self-RAG [23]	Adaptive retrieval, generation, and self-reflection	Retriever and reflection-token training	Factuality, citation accuracy, and retrieval-aware generation quality	Introduces retrieval and self-critique mechanisms, while TruthShield focuses on lightweight trigger-guided adapter training without additional retrieval or reflection modules.
Corrective RAG (CRAG) [24]	Evaluates retrieval quality and performs corrective retrieval	Retrieval evaluator, web/search extension, and document filtering	Robustness against incomplete or noisy retrieved evidence	Addresses retrieval failure through adaptive correction, whereas TruthShield analyzes the transferability of trigger-guided adapter behavior beyond exact target-response matching.
Preference-Based Alignment [14]	Optimizes model behavior using preference or feedback signals	Preference dataset and reward/optimization pipeline	Human preference alignment and response quality	Directly optimizes preferred responses, whereas TruthShield employs rule-guided trigger targets and explicitly investigates the divergence among multiple evaluation metrics.
TruthShield (Ours)	Trigger-guided QLoRA adapter training	No external retrieval or preference optimization module	Strict matching, LLM judge, and BERTScore evaluation under the same held-out trigger split	Provides a lightweight, metric-aware analysis of parameter-efficient adaptation and complements retrieval- and preference-based hallucination mitigation approaches.

The results also show why hallucination mitigation experiments should report complementary metrics. Strict matching provides reproducibility and comparability. Judge evaluation provides a broader semantic check. Neither view is complete on its own, but disagreement between them is informative. In this study, disagreement reveals that the adapter's

TABLE 12. Interpretation of metric combinations in hallucination-aware adaptation.

Strict change	Judge change	Interpretation
Positive	Positive	Evidence consistent with both surface and semantic improvement
Positive	Neutral or negative	Surface compliance without demonstrated semantic gain
Neutral or negative	Positive	Possible semantic improvement not captured by strict patterns
Neutral or negative	Neutral or negative	No observed mitigation benefit

apparent improvement is concentrated in format-level compliance.

The findings also clarify what trigger-guided training is useful for. It can be valuable as a controlled intervention for probing whether a model can learn specific cautionary behaviors. It can also be useful for building datasets of adversarial or counterexample-style prompts. The limitation is that these benefits are not identical to factual improvement. A model can learn to say that evidence is required without actually checking evidence, identifying the unsupported subclaim, or producing a grounded alternative answer.

A practical implication is that hallucination-aware fine-tuning should include evaluation examples that punish generic caution. For example, an evaluation set should be distinguished between an answer that simply refuses to speculate and an answer that explains which part of a claim is verifiable, which part is unsupported, and what evidence would be required. Without this distinction, training may converge on safe-sounding templates that score well under narrow pattern matching.

Improving judge-based assessment scores would likely require changing the training signal rather than simply increasing the number of trigger examples. Future trigger targets should require claim-level verification: identifying unsupported subclaims, separating verifiable from unverifiable content, using available evidence when present, and producing grounded alternative answers when the original claim is unsupported. In addition, repetition control through target deduplication, more diverse response templates, and decoding constraints may improve judge acceptability by reducing generic or repetitive cautionary outputs.

Another implication concerns reporting. Studies that use rule-derived training data should report at least one metric that is not structurally similar to the rules used to create the data. In this experiment, the strict metric is close to the training target, whereas the judge metric provides a different view. The disagreement between them is therefore not a failure of the experiment; it is the main evidence about the adapter's behavior. The fixed rule-guided experiment falls into the second row of Table 12.

VII. LIMITATIONS

This paper has several limitations. First, the fixed adapter results focus on Mistral-7B as the primary adapted model. Baseline failure extraction includes multiple models, but the reported adapter comparison is not yet a multi-adapter, multi-model study. A stronger generalization claim would require training and evaluating adapters for several base models under identical trigger-generation and judge-evaluation protocols.

Second, the training round analyzed here uses rule-guided triggers. LLM-guided trigger variants are part of the broader pipeline but are not included in the fixed result claim. Rule-guided triggers provide transparency and reproducibility, but they may overrepresent failure modes that are easy to express with patterns. They may also encourage the model to imitate the target phrase instead of learning a richer verification procedure. The strict evaluator may also be affected by metric leakage because its matching rules are related to the rule-guided target responses used during trigger training. This limits the independence of the strict metric and makes judge-based or human evaluation necessary for validating semantic improvement.

Third, judge-based evaluation depends on the single judge model, meta-llama/Meta-Llama-3.1-8B-Instruct, and judge prompt. A different judge may change absolute scores, although the strict-versus-judge comparison remains a useful diagnostic design. Future work should evaluate judge robustness by using multiple judge models, prompt variants, and calibration examples. Human annotation would also help determine whether judge failures correspond to real semantic inadequacy or to evaluator-specific preferences.

Fourth, the study does not claim exhaustive coverage of hallucination types. The rule labels and trigger generation process emphasize failure modes represented in the evaluated datasets and baseline outputs. They do not cover all forms of factual error, temporal drift, numerical reasoning error, citation fabrication, or multi-hop evidence synthesis.

Fifth, the qualitative examples show repetitive output behavior, but the current experiment does not isolate whether repetition is caused primarily by the training data, decoding settings, adapter overfitting, or interaction among these

TABLE 13. Threats to validity and mitigation strategies.

Threat	Effect on interpretation	Mitigation in this study	Future mitigation
Single adapted base model	Limits model-level generalization	Claims are restricted to the fixed Mistral-7B adapter	Train comparable adapters for additional model families
Rule-derived trigger data	May encourage template learning	Judge evaluation is used as a semantic countercheck	Add evidence-seeking and claim-decomposition targets
Judge-model dependence	Absolute judge scores may vary	Strict and judge metrics are reported separately	Use multiple judges and human calibration
Unequal task sizes	Deltas have different stability across tasks	Task-level results are interpreted by direction and magnitude	Report confidence intervals and repeated seeds
Repetitive generation	May depress judge scores independently of factuality	Qualitative examples identify repetition explicitly	Tune decoding and deduplicating training targets

factors. This matters because repetition may be reducible through training-target deduplication, more diverse target responses, and decoding constraints, whereas deeper semantic failures require changes to training objectives and evaluation design. Improving judge-based assessment scores would likely require targets that encourage claim-level verification rather than repeated generic cautionary phrases.

We do not report confidence intervals, repeated random seeds, or statistical significance tests; therefore, score deltas should be interpreted cautiously, especially for tasks with small test splits such as TQA(MC).

Table 13 summarizes the main threats to validity. The most important risk is construct validity: strict matching may not measure the construct of semantic hallucination mitigation. This risk is not incidental to the study; it motivates the dual-metric design. Internal validity is affected by the possibility that repetition or decoding settings contribute to judge failures. External validity is limited by the single adapted model and the specific trigger-generation procedure. Conclusion validity is limited by unequal task sizes, especially for TruthfulQA multiple choice.

These limitations constrain the conclusion. The evidence supports a metric-aware analysis of trigger-guided adapter behavior, not a broad claim that the adapter improves semantic reliability in deployment.

VIII. CONCLUSION

This paper evaluated trigger-guided QLoRA adapter training for hallucination-aware language model adaptation using strict matching and LLM judge evaluation. The adapter produced large strict-score gains on TruthfulQA generation, TruthfulQA multiple choice, and HaluEval. However, judge-based scores did not improve and decreased on two of the three tasks. The results indicate that trigger-guided adapter training can improve surface-level compliance, while evidence for semantic hallucination mitigation remains inconclusive in this experimental setting.

The broader lesson is methodological. Hallucination mitigation should not be evaluated through strict metrics alone, especially when the training data are derived from rule patterns. A metric-aware evaluation design can reveal whether an apparent improvement reflects deeper semantic reliability or only alignment with the evaluator's visible format.

Future work should extend this analysis in three directions. First, the same trigger-guided protocol should be applied to additional base models to determine whether the strict-versus-judge divergence is model-specific or common across architectures and instruction-tuned checkpoints. Second, trigger construction should move beyond generic cautionary responses toward evidence decomposition, source attribution, and claim-level verification targets. Such targets should require the model to identify unsupported subclaims, separate verifiable from unverifiable content, use available evidence when present, and provide grounded alternative answers when the original claim is unsupported.

Third, evaluation should combine strict scoring, multiple LLM judges, and human annotation so that metric disagreement can be analyzed rather than hidden by a single aggregate score. Future work will also evaluate trigger-guided adaptation in combination with retrieval-augmented, self-reflective, corrective, and preference-optimization-based hallucination mitigation methods under controlled experimental settings.

These extensions would make it possible to test whether trigger-guided adaptation can progress from surface compliance toward genuine semantic reliability.

ACKNOWLEDGMENT

The authors used ChatGPT by OpenAI for language editing and grammar refinement only. They reviewed and approved

all AI-assisted edits and took full responsibility for the scientific content, analysis, results, and conclusions of this article.

REFERENCES

[1] A. Alansari and H. Luqman, "Large language models hallucination: A comprehensive survey," *Comput. Sci. Rev.*, vol. 61, Aug. 2026, Art. no. 100970, doi: 10.1016/j.cosrev.2026.100970.
[2] S. Liu, Y. Gao, S. Li, P. Wang, and T. Wang, "A hallucination detection and mitigation framework for faithful text summarization using LLMs," *Sci. Rep.*, vol. 16, no. 1, Dec. 2025, Art. no. 1374, doi: 10.1038/s41598-025-31075-1.
[3] W. Zhang and J. Zhang, "Hallucination mitigation for retrieval-augmented large language models: A review," *Mathematics*, vol. 13, no. 5, p. 856, Mar. 2025, doi: 10.3390/math13050856.
[4] E. Lavrinovics, R. Biswas, J. Bjerva, and K. Hose, "Knowledge graphs, large language models, and hallucinations: An NLP perspective," *J. Web Semantics*, vol. 85, May 2025, Art. no. 100844, doi: 10.1016/j.websem.2024.100844.
[5] S. Farquhar, J. Kossen, L. Kuhn, and Y. Gal, "Detecting hallucinations in large language models using semantic entropy," *Nature*, vol. 630, no. 8017, pp. 625–630, Jun. 2024, doi: 10.1038/s41586-024-07421-0.
[6] Z. Lin, S. Guan, W. Zhang, H. Zhang, Y. Li, and H. Zhang, "Towards trustworthy LLMs: A review on debiasing and dehallucinating in large language models," *Artif. Intell. Rev.*, vol. 57, no. 9, Aug. 2024, Art. no. 243, doi: 10.1007/s10462-024-10896-y.
[7] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Comput. Surveys*, vol. 55, no. 12, pp. 1–38, Dec. 2023, doi: 10.1145/3571730.
[8] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, "On faithfulness and factuality in abstractive summarization," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 1906–1919, doi: 10.18653/v1/2020.acl-main.173.
[9] S. Lin, J. Hilton, and O. Evans, "TruthfulQA: Measuring how models mimic human falsehoods," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics (Volume 1: Long Papers)*, 2022, pp. 3214–3252, doi: 10.18653/v1/2022.acl-long.229.
[10] J. Li, X. Cheng, X. Zhao, J.-Y. Nie, and J.-R. Wen, "HaluEval: A large-scale hallucination evaluation benchmark for large language models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 6449–6464, doi: 10.18653/v1/2023.emnlp-main.397.
[11] P. Manakul, A. Liusie, and M. Gales, "SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 9004–9017, doi: 10.18653/v1/2023.emnlp-main.557.
[12] S. Min, K. Krishna, X. Lyu, M. Lewis, W.-T. Yih, P. Koh, M. Iyyer, L. Zettlemoyer, and H. Hajishirzi, "FActScore: Fine-grained atomic evaluation of factual precision in long form text generation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 12076–12100, doi: 10.18653/v1/2023.emnlp-main.741.
[13] S. Es, J. James, L. Espinosa Anke, and S. Schockaert, "RAGAs: Automated evaluation of retrieval augmented generation," in *Proc. 18th Conf. Eur. Chapter Assoc. Comput. Linguistics: Syst. Demonstrations*, 2024, pp. 150–158, doi: 10.18653/v1/2024.eacl-demo.16.
[14] L. Ouyang et al., "Training language models to follow instructions with human feedback," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 1–9, doi: 10.52202/068431-2011.
[15] E. J. Hu et al., "LoRA: Low-rank adaptation of large language models," in *Proc. Int. Conf. Learn. Represent.*, 2022, pp. 1–10.
[16] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient finetuning of quantized LLMs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 10088–10115, doi: 10.52202/075280-0441.
[17] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, H. Zhang, J. Gonzalez, and I. Stoica, "Judging LLM-as-a-judge with MT-bench and chatbot arena," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 46595–46623, doi: 10.52202/075280-2020.
[18] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1–11.
[19] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, "Retrieval augmentation reduces hallucination in conversation," in *Proc. Findings Assoc. Comput. Linguistics: EMNLP*, 2021, pp. 3784–3803, doi: 10.18653/v1/2021.findings-emnlp.320.
[20] W. Kryscinski, B. McCann, C. Xiong, and R. Socher, "Evaluating the factual consistency of abstractive text summarization," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2020, pp. 9332–9346, doi: 10.18653/v1/2020.emnlp-main.750.
[21] P. Laban, T. Schnabel, P. N. Bennett, and M. A. Hearst, "SummaC: Revisiting NLI-based models for inconsistency detection in summarization," *Trans. Assoc. Comput. Linguistics*, vol. 10, pp. 163–177, May 2022, doi: 10.1162/tacl_a_00453.
[22] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating text generation with BERT," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–8.
[23] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," in *Proc. Int. Conf. Learn. Represent.*, 2024, pp. 1–12.
[24] S.-Q. Yan, J.-C. Gu, Y. Zhu, and Z.-H. Ling, "Corrective retrieval augmented generation," 2024, *arXiv:2401.15884*.

JUNJUN ZHANG received the B.S. degree in engineering from Nanyang Institute of Technology, China, in 2008, the M.S. degree in computer technology from Zhengzhou University, China, in 2017, and the Ph.D. degree in computer engineering from Cheongju University, Republic of Korea, in 2023. From 2009 to 2016, she was a full-time Lecturer in information engineering with Zhengzhou Vocational College of Technology, China. From 2017 to 2019, she was a Lecturer with the School of Software, Zhengzhou University of Industrial Technology, China. Since 2024, she has been an Associate Professor with the School of Software, Henan University of Engineering, China. She is a key member of Henan Engineering Technology Research Center of Intelligent Transportation Video Image Perception and Recognition. She has published over 15 research papers, holds two national invention patents, and has led or participated in several government- and industry-funded projects in artificial intelligence and software engineering. Her research interests include deep learning, pattern recognition, multimodal data fusion, and intelligent transportation.

GISEOP NOH received the B.S. degree in industrial engineering from Korea Air Force Academy, South Korea, the M.S. degree in computer science from the University of Colorado Denver, USA, in 2009, and the Ph.D. degree in computer science from Seoul National University, South Korea, in 2014. From 1994 to 2005, he was an Officer with Republic of Korea Air Force, where he was responsible for the operation and maintenance of advanced weapon computer systems. He was with the Defense Acquisition Program Administration on aviation electronics and computer projects. From 2017 to 2018, he was a Faculty Member with the Department of Computer Science, Korea Air Force Academy. From 2018 to 2024, he was a Professor with the Department of Artificial Intelligence Software, Cheongju University, South Korea. Since 2025, he has been a Professor with the Department of Software Convergence, Hongik University, South Korea, where he established and directs the DeepShark Laboratory. His research interests include machine learning, computer vision, retrieval-augmented generation (RAG) systems, graph neural networks, and AI applications in semiconductor inspection and smart manufacturing. He is a member of Korean Institute of Information Scientists and Engineers (KIISE) and Korea Information Processing Society (KIPS).