Hallu-TRIG: Triggering Input-Conflicting Hallucinations in Large Language Models with Semantic-guided Metamorphic Relations

Anonymous ACL submission

Abstract

While Large Language Models (LLMs) have developed rapidly across a range of natural language processing (NLP) tasks, they have also raised concerns about their security and reliability. One such concern is Input-Conflicting Hallucination (ICH), a type of hallucination that conflicts with user input. Since the data annotation in NLP tasks is expensive and labor-intensive, existing ICH attack methods have adopted Metamorphic Testing to bypass the oracle problem. However, these attacks are black-box methods that are restricted to question answering tasks, limited to a few Metamorphic Relations (MRs), and easily defended by decoder-only LLMs. In response, we propose HALLU-TRIG, a simple yet effective grey-box method that constructs six semantic-guided MRs to generate attack cases, and proposes a diversity-guided test case prioritization method to enhance its efficiency. We evaluate HALLU-TRIG on four NLP datasets and three popularly used target LLMs. As a result, the designed MRs achieve higher hallucination trigger rates than existing state-of-the-art baselines, and the diversity-guided prioritization can efficiently trigger ICHs with less time.

1 Introduction

With dramatic evolution in recent years, Large Language Models (LLMs) such as BART (Lewis et al., 2019), GPT4 (OpenAI, 2023), Claude (Anthropic, 2023) and LLaMA 3 (LlamA, 2024) have made unprecedented progress in natural language understanding (Hendrycks et al., 2020) and generation (Zhang et al., 2024), leading popular applications in resolving various downstream tasks. As LLMs gain prominence in academia, industry, and daily use, their security and reliability have become an increasingly important topic. Input-conflicting hallucination (ICH), a type of hallucination that conflicts with source input provided by users, has grown challenging due to the free-form and lengthy nature of content generated by LLMs (Zhang et al., 2023; Huang et al., 2023). Therefore, it is necessary to reveal the ICHs in LLMs.

In ICH attacks for LLMs, a key step is verifying the correctness of LLMs' outputs, also known as the oracle problem in Software Engineering (Chen et al., 2020). This typically relies on manual inspection and is complex and time-consuming to annotate the generations of LLMs precisely (Zhong et al., 2021; Guerreiro et al., 2022; Dziri et al., 2022; Dale et al., 2022). Metamorphic Testing (MT) is a solution to bypass the oracle problem. Given an existing case (called the source case) and the LLM under test, MT generates a new case (called the follow-up case) based on a Metamorphic Relation (MR) and constructs a Metamorphic test case Pair (MP). The MR is the core component of MT, it defines how to generate a follow-up case from the source case and the expected relation between the outputs of the two cases. After executing the MP on target LLM, MT inspects the relation between the outputs and MR. Once an ICH is triggered if the two outputs violate the MR.

Existing studies have proposed a few MT-based methods to reveal the violations in LLMs, which are ICHs through our analysis. Chen et al. (Chen et al., 2021) propose QAAskeR to generate new questions and answers based on the synthesized pseudo facts derived from original questions and generated answers. Shen et al. (Shen et al., 2022) proposed QAQA to construct MPs by inserting redundant sentences searched from the training set into source cases. Liu et al. (Liu et al., 2022) propose QATest to generate follow-up cases by grammatical component-based, sentence structural-based, and adversarial perturbation-based transformations. However, these attacks: (i) are restricted to question answering tasks, (ii) only consider the syntax information and generate low-quality follow-up cases, (iii) focus on attacking two types of LLMs, i.e. UnifiedQA (Khashabi et al., 2020) and AL-

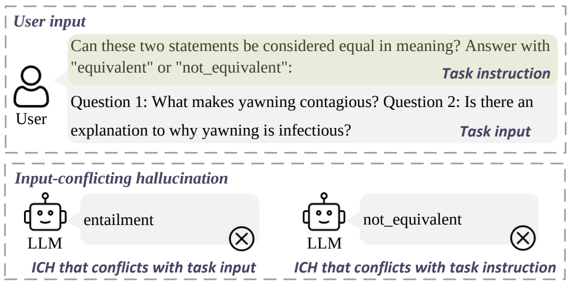

User input
Can these two statements be considered equal in meaning? Answer with "equivalent" or "not_equivalent":
Task instruction
User
Question 1: What makes yawning contagious? Question 2: Is there an explanation to why yawning is infectious?
Task input
Input-conflicting hallucination
LLM
entailment
LLM
not_equivalent
ICH that conflicts with task input
ICH that conflicts with task instruction

Figure 1: An example of the format of user input and input-conflicting hallucinations appearing in a general LLM response.

BERT (Lan et al., 2020), and have poor attack efficacy on decoder-only LLMs, and (iv) are black-box without utilizing the hidden outputs to identify ICH-sensitive MPs and facilitate the attack process.

To address the aforementioned limitations, we propose a grey-box attacking method named HALLU-TRIG to effectively and efficiently trigger the ICHs in LLMs. HALLU-TRIG consists of two components: a multi-level MP construction module and a diversity-based MP prioritization module. In the first module, we design 4 MRs at the character level, one MR at the word level, and one MR at the semantic level. All the MRs can be applied to general NLP datasets and LLMs to reveal potential ICHs. Besides, we consider the syntax and semantic information to ensure the quality of generated cases. In the second module, inspired by the test case prioritization method in traditional software and DNN testing (Cao et al., 2013; Xie et al., 2022), we involve two diversity metrics, i.e. Jensen-Shannon divergence (JS) and Hallengle Distance (HD), to perform MP prioritization and trigger ICHs efficiently with the prioritized MP sequences. Note that the diversity of each MP is calculated based on the hidden outputs of the decoder blocks in target LLMs. To measure the performances of the diversity-based MP prioritization, we propose the Average Percentage of Hallucination Triggered (APHT) and normalized APHT (NAPHT). In summary, our primary contributions are as follows:

- Effective MRs for triggering ICHs in LLMs. We propose six semantic-guided MRs that can be applied to different types of LLMs and NLP tasks. These MRs span multiple granularity levels, i.e. character, word, and semantic.
- Efficient grey-box MP prioritization method. We make the first attempt to identify the ICH-sensitive MPs with LLMs' hidden outputs and prioritize them with two diversity metrics. We propose a new performance metric to measure the performance of our prioritization method.
- Extensive evaluation of HALLU-TRIG. We conduct the experiments on several popularly used NLP tasks and LLMs. The results indicate that our method can both effectively and efficiently trigger ICHs in target LLMs.

2 Background and Related Work

2.1 Input-conflicting Hallucination in LLMs

ICH occurs when the content generated by LLMs deviates from the input provided by the users. As shown in Figure 1, it can happen in two ways: the LLM's responses contradict the task inputs or the LLM's responses contradict the task instructions. This means the LLM misunderstands the users' intents or representations of the context. Although ICH is relatively easy for users to identify according to the contexts, evaluating it within LLMs is particularly challenging since LLM generates content in a free and lengthy format.

2.2 Hallucination evaluation for LLMs

Recent hallucination evaluation methods in NLP can be categorized as follows: (i) human evaluation (Lin et al., 2021; Min et al., 2023) mainly focuses on conducting human annotations or guidelines to ensure reliable evaluation, (ii) model-based automatic evaluation (Zha et al., 2023) generally trains a proxy LLM to evaluate whether the answer of the subject model is correct, and (iii) rule-based automatic evaluation (Bang et al., 2023; Yu et al., 2023; Lee et al., 2022) uses classic classification metrics such as accuracy and F1 in classification tasks. These methods all rely on the analysis of the oracle. For these reasons, we adopt MT to bypass the oracle problem (i.e., verify whether the outputs are ICHs without ground truths) and promote revealing ICHs in LLMs.

2.3 Metamorphic Testing

MT is quite a popular technique due to its ability to effectively bypass the oracle problem (Chen et al., 2020) and uncover the defects in target software or DNNs. MT can be used both to generate test cases and verify the correctness of results. A central component of MT is a set of Metamorphic Relations (MRs), which define the expected relation between the target model, its inputs, and the outputs. Given a set of source cases, MT uses the defined MRs to generate new test cases as follow-up test cases and construct Metamorphic test case Pairs (MPs). Instead of verifying the correctness of the outputs for the source and follow-up cases in each MP, MT checks whether the outputs adhere to the corresponding MR. If an MR is violated, at least one output is hallucination.

2.4 Metamorphic Testing for LLMs

Existing studies mainly focus on detecting ICHs in UnifiedQA and ALBERT. QAAskeR (Chen et al., 2021) constructs MRs based on the synthesized pseudo facts derived from questions and answers, it may incur a range of false positives. QAQA (Shen et al., 2022) proposes MRs that do not change the answer for equivalent relations nor affect the clear inference for inferential relations. These MRs are limited to datasets with contexts and questions. QATest (Liu et al., 2022) designs MRs based on transformations on grammatical components, sentence structure, and adversarial perturbation. These MRs may greatly disturb the semantics of the follow-up cases and even change their semantics. In addition, Li et al. (Li et al., 2024) use unique logic reasoning rules to establish MRs for detecting hallucinations in LLMs, but they aim to reveal the fact-conflicting hallucination.

Different from these methods, HALLU-TRIG integers the syntax and semantic information to guarantee the quality of generated cases, and aims to effectively and efficiently reveal ICHs in popularly used open-source LLMs. Besides, we involve the diversity metrics to boost the trigger of ICHs.

3 Hallu-TRIG

3.1 Overview

Figure 2 shows the overview of HALLU-TRIG, which can be divided into two modules: a multi-level MP construction module and a diversity-guided MP prioritization module. In the first module, we propose semantic-guided MRs at different granularities and use them to generate follow-up cases and construct MPs. Specifically, we propose four MRs at the character level, one MR at the word level, and one MR at the semantic level. In the second module, we select a decoder block from each target LLM as the *compute unit*, adopt two diversity metrics to calculate the diversity of each MP with the outputs of the *compute unit*, and sort all MPs in descending order. Refer to (Cao et al., 2013), MPs with high diversity are considered ICH-sensitive and are prioritized to be input into target LLMs to check the results.

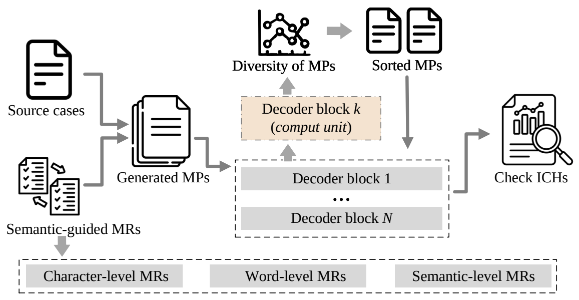


Figure 2: The overview of HALLU-TRIG.

3.2 Problem Formulation

Oracle and ICH. Generally, the test oracles for different types of tasks vary. The downstream tasks of LLMs are primarily divided into classification tasks and generation tasks. Given the human oracle $\mathcal{O}$, the source case x, and a target LLM θ, we design an MR $\mathcal{M}$ and generate a follow-up case x'. The function $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$, where $y = f_\theta(x)$ defines the mapping of LLM θ's input to its output. The function $\mathcal{M} : \mathcal{X} \rightarrow \mathcal{X}'$, where $x' = \mathcal{M}(x)$ defines the mapping of the source case and follow-up case.

In classification tasks such as duplicate sentence detection (DSD), the output of LLM is the concrete label (e.g., "equivalent" or "not_equivalent") of the input. For $\forall x' \in \mathcal{M}(\mathcal{X})$, the oracle of target LLM θ can be defined as $\mathcal{O}(x') = \mathcal{O}(x)$. Then x' is an ICH if it satisfies the requirements:

$$f_\theta(x') \neq f_\theta(x), \text{ s.t. } f_\theta(x) = \mathcal{O}(x) \quad (1)$$

In generative tasks such as reading comprehension (RC), due to the lack of a standardized format in LLM outputs, we refer to previous study (Shen et al., 2022) and determine their accuracy by comparing the semantic similarity Sim between the output $f_\theta(x)$ and the ground truth label $\mathcal{A}$.

$$Sim(f_\theta(x), \mathcal{A}) = \frac{emb(f_\theta(x)^T \cdot emb(\mathcal{A})}{\|emb(f_\theta(x))\|\|emb(\mathcal{A})\|} \quad (2)$$

where $emb(\cdot)$ is the embedding function integrated in Phrase-BERT (Wang et al., 2021) and $Sim(\cdot)$ is the cosine similarity function. For $\forall x' \in \mathcal{M}(\mathcal{X})$, the oracle of target LLM f_θ is defined as follows:

$$\mathcal{O}(x') = \mathcal{O}(x), \text{ s.t. } Sim(f_\theta(x), f_\theta(x')) \geq \gamma \quad (3)$$

where γ is the similarity threshold. In this scenario, x' is an ICH if it satisfies the requirements:

$$Sim(f_\theta(x), f_\theta(x')) < \gamma, \text{ s.t. } f_\theta(x) = \mathcal{O}(x) \quad (4)$$

Attack Goal. Given a set of source cases as input $\mathcal{S}$, each case that can be correctly recognized by target LLM f_θ constitutes a set $\mathcal{S}' \subset \mathcal{S}$. The attack goal contains two aspects: (i) *maximization of the hallucination trigger rate (HTR)*. Let $\mathcal{S}' = \{x_1, x_2, ..., x_{|\mathcal{S}'|}\}$ where x_i is the i-th source case in $\mathcal{S}'$, $|\mathcal{S}'|$ is the size of $\mathcal{S}'$, and x'_i is the corresponding follow-up case generated by MR $\mathcal{M}$. In classification tasks, the HTR of $\mathcal{M}$ on $\mathcal{S}$ can be calculated as follows:

$$\text{HTR}_{\mathcal{M}} = \frac{\sum_{i=1}^{\mathcal{S}'} [f_\theta(x'_i) \neq f_\theta(x_i)]}{|\mathcal{S}'|} \tag{5}$$

In generation tasks, it can be calculated as follows:

$$\text{HTR}_{\mathcal{M}} = \frac{\sum_{i=1}^{\mathcal{S}'} [Sim(f_\theta(x), f_\theta(x')) < \gamma]}{|\mathcal{S}'|} \tag{6}$$

We formulate the problem in this aspect as follows:

$$\mathcal{M} \leftarrow \arg\max_{\mathcal{M}} \text{HTR}_{\mathcal{M}} \tag{7}$$

Our objective is to construct an MR $\mathcal{M}$ to maximize $\text{HTR}_{\mathcal{M}}$, and (ii) *maximization of Average Percentage of Hallucination Triggered (APHT)*. Given a generated pile of MPs $\Gamma = \{\mathcal{MP}_1, \mathcal{MP}_2, ..., \mathcal{MP}_{|\Gamma|}\}$ and the prioritized sequence of MPs $\Gamma' = \left\langle \mathcal{MP}'_1, \mathcal{MP}'_2, ..., \mathcal{MP}'_{|\Gamma|} \right\rangle$, where $\mathcal{MP}_i$ is the i-th MP and $|\Gamma|$ is the total number of MPs in Γ. Inspired by the testcase ranking metrics in traditional software and DNN testing (Rothermel et al., 1999; Xie et al., 2022), we design a new rank-based metric (i.e., APHT) to evaluate the performance of our prioritization method:

$$\text{APHT}(\Gamma') = 1 - \frac{\sum_{i=1}^{m} \mathcal{RH}_i}{|\Gamma| \cdot m} + \frac{1}{2 \cdot |\Gamma|} \tag{8}$$

where m denotes the numbers of ICH-sensitive MPs in Γ', and $\mathcal{RH}_i$ is the rank of the i-th ICH-sensitive MP in Γ'. A higher APHT indicates that the ICH-sensitive cases are well identified and ranked. Furthermore, considering the reachable upper and lower bounds of APHT, we scale it into the normalized APHT (NAPHT) which is calculated as follows:

$$\text{NAPHT}(\Gamma') = \frac{\text{APHT}(\Gamma') - \text{APHT}_{min}}{\text{APHT}_{max} - \text{APHT}_{min}} \tag{9}$$

where APHT_{max} refers to the APHT that prioritizes the ICH-sensitive MPs from 1 to m, and APHT_{min} refers to the APHT that prioritizes the ICH-sensitive MPs from $|\Gamma| - m + 1$ to $|\Gamma|$. Then we formulate the second problem as follows:

$$\mathcal{D} \leftarrow \arg\max_{\mathcal{D}} \text{NAPHT}(\Gamma') \tag{10}$$

In this scenario, our objective is to utilize a diversity metric $\mathcal{D}$ to improve the effectiveness of MP prioritization, i.e., maximize $\text{NAPHT}(\Gamma')$.

3.3 Attack Methodology

Multi-level MP Construction. In our method, it is necessary to construct a series of MRs to support the process of MT and then trigger ICHs in target LLMs. We comprehensively analyze the linguistic structure of LLMs' input and construct MRs at the character, word, and semantic levels to attack target LLMs effectively.

At the character level and word level, there are two main steps: (i) find the most vulnerable words in the source cases, and (ii) perturb the vulnerable words, to generate MPs. Given a source case $x = \{w_1, w_2, ..., w_{|x|}\}$ and a BERT model ϕ, where x_i is the i-th word and $|x|$ is the number of all words in x. The corresponding output is $g_\phi(x) = \{o_1, o_2, ..., o_{|x|}\}$ where o_t is the output logit for time step t. To find the most vulnerable word in x, we replace w_i with `[MASK]` such that $x_{\backslash w_i} = \{w_1, ..., w_{i-1}, \texttt{[MASK]}, w_{i+1}, ..., w_{|x|}\}$. The output of $x_{\backslash w_i}$ is $g_\phi(x_{w_i}) = \{o'_1, o'_2, ..., o'_{|x|}\}$ where o'_t is the new output logit for time step t. The influence score for w_i is calculated as follow:

$$\mathcal{I}_{w_i} = \left| \sum_{t=1}^{|x|} o_t - \sum_{t=1}^{|x'|} o'_t \right| \tag{11}$$

We select w_i with the maximum influence score.

To perform perturbation at the character level, we randomly alter the character in the word from the input and define four MRs:

❶ **MR1: Random Character Deletion (RCDel).** RCDel generates a follow-up case by randomly deleting a single character, excluding the first and last characters.

❷ **MR2: Neighboring Character Swap (NCSwa).** NCSwa generates a follow-up case by randomly swapping the neighboring characters at once.

❸ **MR3: Random Character Insertion (RCIns).** RCIns generates a follow-up case by randomly inserting a single character between the first and the last characters.

❹ **MR4: Random Character Substitution (RCSub).** RCSub generates a follow-up case by randomly substituting a single character.

Algorithm 1 Diversity-Guided MP Prioritization

```
Input: a pile of original MPs $\mathit{\Gamma}$, compute unit $\mathcal{D}_k$.
Output: a sequence of prioritized MPs $\mathit{\Gamma}'$.
Initilize: $\mathit{\Gamma}' \leftarrow \emptyset$.
 1: for $\mathcal{MP}$ in $\mathit{\Gamma}$ do
 2:     $x, x' \leftarrow \mathcal{MP}$
 3:     $output_x = \mathcal{D}_k(x)$
 4:     $output_{x'} = \mathcal{D}_k(x')$
 5:     // Compute the execution diversity of $\mathcal{MP}$
 6:     $\Delta_{\mathcal{MP}} = Diversity(output_x, output_{x'})$
 7:     $\mathit{\Gamma}' \leftarrow \mathit{\Gamma}' \cup (\mathcal{MP}, \Delta_{\mathcal{MP}})$
 8: end for
 9: // Prioritize all the MPs based on computed diversities
10: $\mathit{\Gamma}' = Prioritization(\mathit{\Gamma}')$
```

To perform perturbation at the word level, we use BERT-MLM to generate N candidates for the vulnerable word and define one MR:

❺ MR5: Similar Word Substitution (SWSub). SWSub generates a follow-up case by replacing the original word with a single word selected from the top-k candidates.

Given a source case x, the perturbed follow-up case x', and w_i and w_i' refer to the words before and after perturbation, respectively. Considering their naturalness and semantics, with $perplex(x') = \sqrt[|x'|]{\Pi_{i=1}^{|x'|} P(w_i'|x'_{\setminus w_i'})}$ and the similarity function $Sim'(\cdot)$ proposed in *SimCSE* (Gao et al., 2021), we evaluate the quality of x' as follows:

$$Nat(x, x') = |perplex(x) - perplex(x')| \quad (12)$$

$$Score_{x'} = -\lambda_1 Nat(x, x') + \lambda_2 Sim'(x, x') \quad (13)$$

where λ_1 and λ_2 are weights, x' is accepted with score greater than or equal to φ (i.e., $Score_{x'} \geq \varphi$).

At the semantic level, we use the NLTK (Bird et al., 2001) tool to construct the syntactic parse tree of the source case and manipulate it to ensure the naturalness and semantic accuracy of the generated follow-up case (Loper and Bird, 2002). The MR in this scenario is defined as:

❻ MR6: Semantic Negation (SNeg). SNeg generates a follow-up case by negating the copular verb, auxiliary verb, etc. (See Appendix A for more details of SNeg.)

3.4 Diversity-Guided MP Prioritization

Referring to traditional software and DNN testing (Chen et al., 2004; Xie et al., 2022; Cao et al., 2013), we prioritize the MPs with the guidance of execution diversity. Given a LLM $\theta = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_{|\theta|}\}$, the $\mathcal{MP}_i = (x_i, x_i')$, and the generated pile of MPs $\mathit{\Gamma}$, where $\mathcal{D}_k$ denotes the k-th decoder block and $|\theta|$ denotes the total number of decoder blocks in θ.

The first key component of this module is the diversity metrics. In HALLU-TRIG, we introduce two accurate and cheap implementation diversity metrics that work effectively on DNNs (Xie et al., 2022), i.e. *Jensen-Shannon divergence (JS)* and *Hellinger distance (HD)*. JS measures the difference between two distributions based on *Kullback-Leibler divergence (KL)*, which is symmetric and bounded. To calculate the JS-based diversity of $\mathcal{MP}_i$ and $\mathcal{D}_k$, we first calculate its KL-based diversity as follows:

$$\Delta_{KL}(x_i||x_i') = \sum_{n_j^k \in \mathcal{L}_k} \mathcal{D}_k(x_i, n_j^k) \ln \frac{\mathcal{D}_k(x_i, n_j^k)}{\mathcal{D}_k(x_i', n_j^k)} \quad (14)$$

where $\mathcal{D}_k(x_i, n_j^k)$ and $\mathcal{D}_k(x_i', n_j^k)$ denotes the outputs of the j-th neuron n_j^k in last full connected layer $\mathcal{L}_k$ of $\mathcal{D}_k$ for x_i and x_i', respectively. Then the corresponding JS-based diversity can be calculated as follows:

$$\Delta_{JS} = \frac{\Delta_{KL}(\mathcal{D}_k(x_i)||M) + \Delta_{KL}(\mathcal{D}_k(x_i')||M)}{2} \quad (15)$$

where $M = \frac{1}{2}(\mathcal{D}_k(x_i) + \mathcal{D}_k(x_i'))$, and $\mathcal{D}_k(x_i)$ and $\mathcal{D}_k(x_i')$ is the final output of $\mathcal{D}_k$ for x_i and x_i', respectively. The HD-based diversity calculates the divergence between two distributions using the Hellinger integral. This type of diversity metric is calculated as follows:

$$\Delta_{HD} = \frac{\sqrt{\sum_{n_j^k \in \mathcal{L}_k} \left(\sqrt{\mathcal{D}_k(x_i, n_j^k)} - \sqrt{\mathcal{D}_k(x_i', n_j^k)}\right)^2}}{2} \quad (16)$$

The second key component is the MP prioritization algorithm. We choose the k-th decoder block $\mathcal{D}_k$ as the *compute unit* and compute the diversity of each MP with the aforementioned metrics based on the output of $\mathcal{D}_k$. Specifically, for each $\mathcal{MP}$ in the generated pile of MPs $\mathit{\Gamma}$, we execute the source case x and the follow-up case x' in target LLM θ, extract the outputs of the *compute unit* $\mathcal{D}_k$ to calculate the diversities. After that, all the MPs are sorted in descending order. The details are shown in Algorithm 1. Generally, MPs with high execution diversity are regarded as ICH-sensitive (Cao

Target Model	Dataset	Attack Method							
		QAQA	QATest	MR1	MR2	MR3	MR4	MR5	MR6
Vicuna-13B	MNLI	–	8.72%	12.15%	12.11%	11.09%	14.58%	38.58%	52.36%
	QQP	–	9.68%	9.65%	9.27%	9.17%	17.89%	34.00%	82.69%
	SQuAD2	12.58%	16.97%	15.43%	15.59%	17.95%	24.41%	56.51%	53.61%
	NarrativeQA	19.28%	17.79%	19.33%	19.88%	21.25%	26.84%	62.10%	64.72%
LLaMA-3-8B	MNLI	–	10.30%	14.09%	13.59%	14.98%	19.40%	40.78%	57.13%
	QQP	–	11.80%	13.16%	9.81%	14.39%	18.10%	39.88%	82.90%
	SQuAD2	17.13%	22.09%	20.94%	20.69%	22.95%	30.63%	54.68%	52.24%
	NarrativeQA	19.93%	26.15%	28.18%	26.87%	27.22%	32.53%	66.20%	65.45%
BART-large	MNLI	–	18.91%	17.22%	18.85%	19.01%	18.68%	38.67%	57.13%
	QQP	–	24.09%	24.66%	23.44%	24.91%	24.79%	56.66%	91.12%
	SQuAD2	18.25%	25.85%	30.05%	34.66%	32.59%	33.55%	59.13%	57.31%
	NarrativeQA	–	–	–	–	–	–	–	–

Table 1: The HTR of MRs: HALLU-TRIG vs. SOTA baselines across models and datasets.

Target Model	MNLI	QQP	SQuAD2	NarrativeQA
Vicuna-13B	99%	99%	98%	97%
LLaMA-3-8B	98%	99%	97%	97%
BART-large	97%	98%	97%	–

Table 2: The TPR across models and datasets.

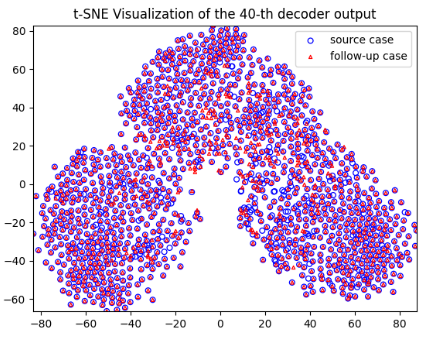


Figure 3: Comparison of distributions between the source and follow-up cases on MNLI and Vicuna-13B.

et al., 2013; Xie et al., 2022). Their outputs will therefore be prioritized for inspection to identify potential ICHs efficiently.

4 Experiments

4.1 Experimental Setup

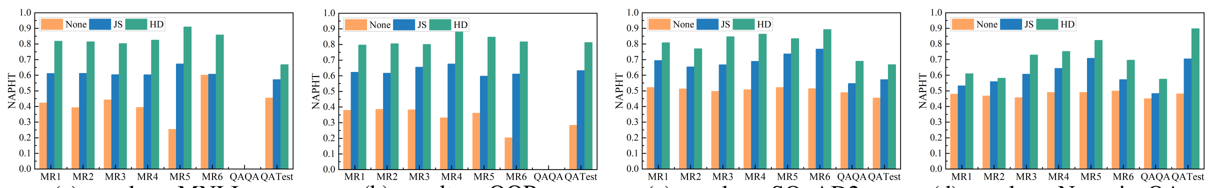
Downstream Tasks and Datasets. We conduct a comprehensive evaluation of HALLU-TRIG by designing experiments across both classification and generation tasks. Referring to existing studies (Yuan et al., 2024; Duan et al., 2024; McCoy et al., 2019; Shen et al., 2022; Chen et al., 2021), we adopt three representative NLP tasks with corresponding datasets: (i) three-category natural language inference (NLI) task based on MNLI (Williams et al., 2017), (ii) binary classification duplicate sentence detection (DSD) task based on QQP (Wang et al., 2018), and (iii) generative reading comprehension (RC) task based on SQuAD2 (Rajpurkar et al., 2018) and NarrativeQA (Kočiskỳ et al., 2018). (See Appendix A)

Target Models. For a comprehensive evaluation, we conduct experiments on three generative open-source LLMs. We pick the representative LLMs with state-of-the-art (SOTA) performances on general NLP tasks: (i) *Vicuna*: the latest version of Vicuna-13b-1.5-16k (Vicuna-13B for short), (ii) *LLaMA*: the latest version of Meta-LLaMA-3-8B (LLaMA-3-8B for short), and (iii) *BART*: the version of BART-large. (See Appendix A)

Attack Baselines. We compare HALLU-TRIG with two SOTA MT-based attack methods, and use the most effective MR in each attack method as the baselines for a fair comparison: (i)*QAQA* (Shen et al., 2022): Inserts another input from the training set as redundant sentence into a given source case to generate a follow-up case, and (ii) *QATest* (Liu et al., 2022): Perturbs the source case with typos to generate follow-up case.

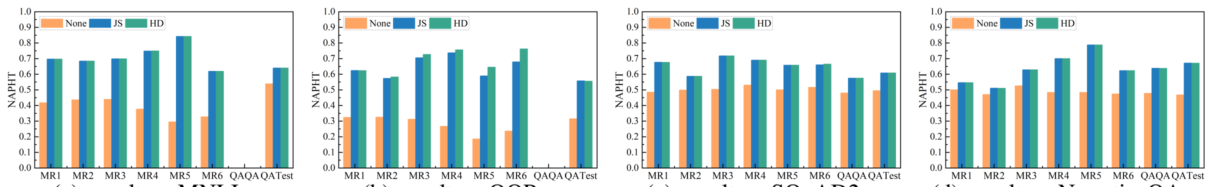
Evaluation Metrics. We use hallucination trigger rate (HTR) and true positive rate (TPR) to evaluate the effectiveness of the proposed MRs, and use NAPHT to assess the efficiency of the MP prioritization method.

Implementation Details. We conduct the experi-



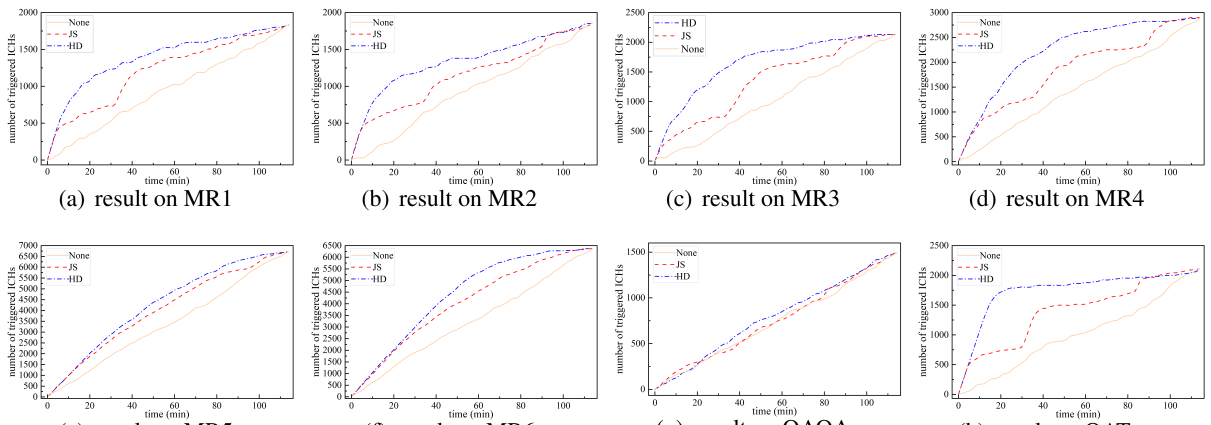
(a) result on MNLI (b) result on QQP (c) result on SQuAD2 (d) result on NarrativeQA

Figure 4: Prioritization performance of different diversity metrics under each MR on Vicuna-13B.



(a) result on MNLI (b) result on QQP (c) result on SQuAD2 (d) result on NarrativeQA

Figure 5: Prioritization performance of different diversity metrics under each MR on LLaMA-3-8B.



(a) result on MR1 (b) result on MR2 (c) result on MR3 (d) result on MR4

(e) result on MR5 (f) result on MR6 (g) result on QAQA (h) result on QATest

Figure 6: Comparison of ICH triggered over time: unsorted vs. sorted MPs on SQuAD2 and Vicuna-13B.

ments with the Ubuntu 18.04 system with 32-core 2.1GHz Xeon CPU, 196GB RAM and 4 NVIDIA GeForce RTX 3090 24G GPUs. We generate 5 and 1 candidates for each source case at the character and word level, respectively. At the word level, we set the N and k as 30 and 5, respectively. Through manual inspection and analysis of the experiment results, we set the threshold of cosine similarity γ and quality φ, weight λ_1 and λ_2 as 0.76, 0.2, 0.5, and 0.5, respectively. MRs in HALLU-TRIG are applied to *Premise* or *Question 1* in classification tasks, and to the questions in generation tasks.

4.2 Effectiveness of Semantic-guided MRs

Attack Effectiveness. For each dataset, we used the whole dataset to calculate the HTR, and randomly sampled 100 ICHs for manual review to calculate the TPR. Table 1 and Table 2 show the results of HTR and TPR, respectively. In most scenarios, the proposed MRs have higher HTR than the SOTA baselines, especially the sentence-level and semantic-level MRs. Besides, the TPR of HALLU-TRIG on each dataset and target model is no less than 97%. These indicate that HALLU-TRIG can effectively trigger ICHs in target LLMs the triggered ICHs are almost true positives.

Quality of MPs. We apply the t-SNE technique (Van der Maaten and Hinton, 2008) to the final decoder outputs of target models to visualize and analyze the distribution of source and follow-up cases. The rationale is that the reported bugs are meaningful for the system if the generated cases are distributed consistently with the original cases (Berend et al., 2020). Figure 3 depicts the distribution of MPs on a subset of MNLI in Vicuna-13B (See Appendix A for results on LLaMA-3-8B

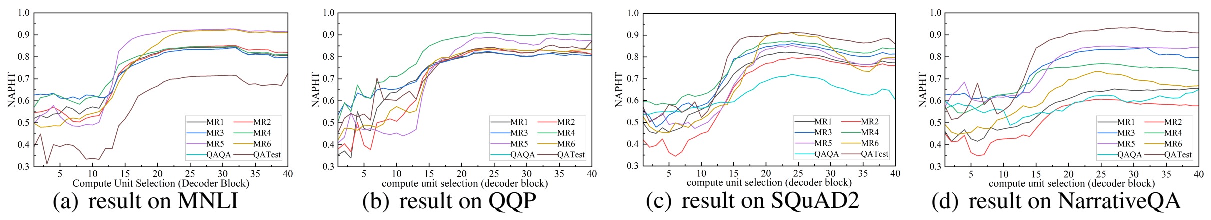


Figure 7: HD-based MP prioritization using different decoder blocks as the *compute unit* on Vicuna-13B.

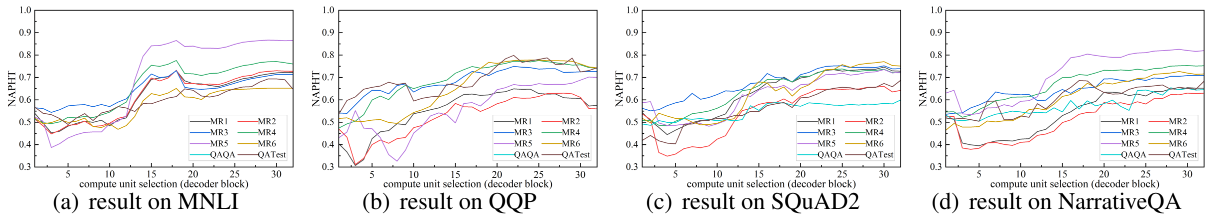


Figure 8: HD-based MP prioritization using different decoder blocks as the *compute unit* on LLaMA-3-8B.

and BART-large). We can observe that the distribution within all the MPs is consistent, indicating the high quality of the generated follow-up cases.

4.3 Effectiveness of MP Prioritization

Evaluation with NAPHT. In this study, we set the implementation of no prioritization as the baseline (None), and select the middle decoder block in target models as the *compute unit*. In other settings, our prioritization method is also effective (see more details in Section 4.4). Figure 4 and Figure 5 shows the results on Vicuna-13B and LLaMA-3-8B, respectively (See Appendix A for the result on BART-large). We find that applying our prioritization method with any diversity metric can increase the NAPHT compared to no prioritization under each MR on all datasets and models. This demonstrates the effectiveness of our prioritization method. Additionally, HD can guide MP prioritization better than JS.

Analysis on Efficiency. We also execute the unsorted and sorted MP sequences, and record the number of triggered ICHs over time (min) to further analyze the efficiency of the prioritization method. Figure 6 depicts the result on SQuAD2 and Vicuna-13B. Intuitively, compared with the unsorted MP sequences, The sorted MP sequence can trigger more ICHs within the same time and requires less time to trigger the same number of ICHs.

4.4 Ablation Study

In this study, we aim to investigate the impact of the selection of the *compute unit* $\mathcal{D}_k$, which is directly reflected through changes in NAPHT. Figure 7, and Figure 8 shows the result on Vicuna-13B and LLaMA-3-8B, respectively (See Appendix A for the result on BART-large). From these figures, the value of NAPHT increases with the increment of k in most scenarios. Thus we can summarize that the efficiency of MP prioritization is improved as k increases, and a larger k tends to achieve a better performance in prioritizing the ICH-sensitive MPs in Γ'. In particular, the NAPHT of these MRs approach maximum when $\mathcal{D}_k$ is set to a middle position. Therefore, we can set the intermediate position in target models as the *compute unit* to save diversity computing time while achieving near-optimal prioritization performance.

5 Conclusion

In this paper, we introduce HALLU-TRIG, a grey-box method to trigger the ICHs in the popularly used open-source LLMs. It attacks the target models at the character, sentence, and semantic levels with six semantic-guided MRs. To boost the revealing of ICHs, we use the hidden outputs in target LLMs to identify the ICH-sensitive MPs and make the first attempt to prioritize the generated MPs with two diversity metrics. Besides, we propose a new metric to measure the performance of the MP prioritization method. Through a comprehensive evaluation, HALLU-TRIG outperforms the SOTA baselines in terms of attack effectiveness, attack quality, and generalizability. The MP prioritization method can stably improve the attack efficiency both for HALLU-TRIG and the SOTA baselines.

6 Limitations

In this study, we mainly discuss the limitations of HALLU-TRIG in terms of threat to internal validity and threat to external validity.

Threat to internal validity primarily lies in two aspects: (i) the effectiveness of the proposed MRs and NAPHT; (ii) the configuration sensitivity of HALLU-TRIG and the adopted diversity metrics. For the first concern, the MRs are constructed based on the common ideas of text mutation in NLP (Li et al., 2018; Gao et al., 2018; Garg and Ramakrishnan, 2020; Morris et al., 2020; Jin et al., 2019), and the NAPHT follows the NAPVD and NAPFD in traditional software and DNN testing. We perform additional manual inspections to ensure their effectiveness in Section 4.2 and Section 4.3.

For the second concern, the performance of the MP prioritization method has been confirmed on each diversity metric and target model under different configurations. Note that **we only need to extract the hidden output from a single decoder block of the target model to perform MP prioritization on the CPU**, which is entirely feasible as long as the target model can be executed.

In summary, we find that the general setup can effectively and stably work on various MRs, metrics, and target LLMs.

Threat to external validity is mainly about the representativeness of the tasks, datasets, and target models adopted in our experiments. To counter this, we adopt three tasks (i.e., NLI, DSD, and RC), four datasets (i.e., MNLI, QQP, SQuAD2, and NarrativeQA), and three LLMs (i.e., BART-large, Vicuna-13B and LLaMA-3-8B), which are all widely used in the state-of-the-art research in NLP (Zhu et al., 2023; Peng et al., 2023).

References

Anthropic. 2023. Introducing Claude. https://www.anthropic.com/index/introducing-claude.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, and 1 others. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

David Berend, Xiaofei Xie, Lei Ma, Lingjun Zhou, Yang Liu, Chi Xu, and Jianjun Zhao. 2020. Cats are not fish: Deep learning testing calls for out-of-distribution awareness. In *Proceedings of the 35th IEEE/ACM international conference on automated software engineering*, pages 1041–1052.

Steven Bird, Edward Loper, and Ewan Klein. 2001. NLTK. https://www.nltk.org/.

Yuxiang Cao, Zhi Quan Zhou, and Tsong Yueh Chen. 2013. On the correlation between the effectiveness of metamorphic relations and dissimilarities of test case executions. In *2013 13th International Conference on Quality Software*, pages 153–162. IEEE.

Songqiang Chen, Shuo Jin, and Xiaoyuan Xie. 2021. Testing your question answering software via asking recursively. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 104–116. IEEE.

Tsong Y Chen, Shing C Cheung, and Shiu Ming Yiu. 2020. Metamorphic testing: a new approach for generating next test cases. *arXiv preprint arXiv:2002.12543*.

Tsong Yueh Chen, DH Huang, TH Tse, and Zhi Quan Zhou. 2004. Case studies on the selection of useful relations in metamorphic testing. In *Proceedings of the 4th Ibero-American Symposium on Software Engineering and Knowledge Engineering (JIISIC 2004)*, pages 569–583. Citeseer.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, and 1 others. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

David Dale, Elena Voita, Loïc Barrault, and Marta R Costa-jussà. 2022. Detecting and mitigating hallucinations in machine translation: Model internal workings alone do well, sentence similarity even better. *arXiv preprint arXiv:2212.08597*.

Haonan Duan, Adam Dziedzic, Nicolas Papernot, and Franziska Boenisch. 2024. Flocks of stochastic parrots: Differentially private prompt learning for large language models. *Advances in Neural Information Processing Systems*, 36.

Nouha Dziri, Hannah Rashkin, Tal Linzen, and David Reitter. 2022. Evaluating attribution in dialogue systems: The begin benchmark. *Transactions of the Association for Computational Linguistics*, 10:1066–1083.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. *arXiv preprint arXiv:2004.01970*.

Nuno M Guerreiro, Elena Voita, and André FT Martins. 2022. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. *arXiv preprint arXiv:2208.05309*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*, 2:10.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *arXiv preprint arXiv:2005.00700*.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale N Fung, Mohammad Shoeybi, and Bryan Catanzaro. 2022. Factuality enhanced language models for open-ended text generation. *Advances in Neural Information Processing Systems*, 35:34586–34599.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Ningke Li, Yuekang Li, Yi Liu, Ling Shi, Kailong Wang, and Haoyu Wang. 2024. Halluvault: A novel logic programming-aided metamorphic testing framework for detecting fact-conflicting hallucinations in large language models. *arXiv preprint arXiv:2405.00648*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Zixi Liu, Yang Feng, Yining Yin, Jingyu Sun, Zhenyu Chen, and Baowen Xu. 2022. Qatest: A uniform fuzzing framework for question answering systems. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, pages 1–12.

Meta LlamA. 2024. LLaMA 3. https://llama.meta.com/llama3/.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.

John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. *arXiv preprint arXiv:2005.05909*.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

Gregg Rothermel, Roland H Untch, Chengyun Chu, and Mary Jean Harrold. 1999. Test case prioritization: An empirical study. In *Proceedings IEEE International Conference on Software Maintenance-1999 (ICSM'99).'Software Maintenance for Business Change'(Cat. No. 99CB36360)*, pages 179–188. IEEE.

Qingchao Shen, Junjie Chen, Jie M Zhang, Haoyu Wang, Shuang Liu, and Menghan Tian. 2022. Natural test generation for precise testing of question answering software. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, pages 1–12.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Shufan Wang, Laure Thompson, and Mohit Iyyer. 2021. Phrase-bert: Improved phrase embeddings from bert with an application to corpus exploration. *arXiv preprint arXiv:2109.06304*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Xiaoyuan Xie, Pengbo Yin, and Songqiang Chen. 2022. Boosting the revealing of detected violations in deep learning testing: A diversity-guided method. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, pages 1–13.

Jifan Yu, Xiaozhi Wang, Shangqing Tu, Shulin Cao, Daniel Zhang-Li, Xin Lv, Hao Peng, Zijun Yao, Xiaohan Zhang, Hanming Li, and 1 others. 2023. Kola: Carefully benchmarking world knowledge of large language models. *arXiv preprint arXiv:2306.09296*.

Lifan Yuan, Yangyi Chen, Ganqu Cui, Hongcheng Gao, Fangyuan Zou, Xingyi Cheng, Heng Ji, Zhiyuan Liu, and Maosong Sun. 2024. Revisiting out-of-distribution robustness in nlp: Benchmarks, analysis, and llms evaluations. *Advances in Neural Information Processing Systems*, 36.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. Alignscore: Evaluating factual consistency with a unified alignment function. *arXiv preprint arXiv:2305.16739*.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. 2024. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, and 1 others. 2023. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and 1 others. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *arXiv preprint arXiv:2104.05938*.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, and 1 others. 2023. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv preprint arXiv:2306.04528*.

A Appendix

A.1 MR6: SemanticNegation

Since the transformation under MR6 is not semantic-preservation, we define the semantic relations on each task here to ensure the soundness of MR6. Given a source case x, a generated follow-up case x', and a target model θ.

- In MNLI, if the output of x is "neutral", then $f_\theta(x') = f_\theta(x)$, otherwise $f_\theta(x') \neq f_\theta(x)$.
- In QQP, $f_\theta(x') = f_\theta(x)$;
- In SQuAD2, if the output of x is "unanswerable", then the output of x' is "unswerable", otherwise $f_\theta(\mathcal{X}') \neq f_\theta(x)$.
- In NarrativeQA, $Sim(f_\theta(x), f_\theta(x')) < 0.76$.

Dataset	Format	Train/Test
MNLI	Etailment/Neutral/Contradiction	392,702/19,647
QQP	Equivalent/Not_equivalent	363,870/40,431
SQuAD2	Extractive	130,319/11,873
NarrativeQA	Abstractive	32,747/3,461

Table 3: The format and statistics of the testing datasets

A.2 Downstream Tasks and Datasets

The details about our datasets are shown as follows:

- **MNLI** dataset is a large-scale dataset composed of sentence pairs with textual entailment annotations, where each sentence pair contains a *Premise* and a *Hypothesis*. It is designed to evaluate a model's ability to understand and infer the relationship between the *Premise* and *Hypothesis*. Based on MNLI, NLI is a three-category task, containing three labels: "entailment", "neutral" and "contradiction".
- **QQP** dataset is a collection of question pairs from the community question-answering website Quora, annotated with whether the pairs are semantically equivalent. Based on QQP, DSD is a binary classification task, with labels of "equivalent" and "not_equivalent".
- **SQuAD2** is an extractive reading comprehension dataset in which the contexts, questions, and answers are collected from Wikipedia articles by crowdworkers. Each question's answer in the dataset is a segment or a certain span from the context. SQuAD2 also includes over 50,000 adversarially unanswerable questions, with corresponding answers marked as "unanswerable".
- **NarrativeQA** is an abstractive reading comprehension dataset that consists of stories in the form of books and movie scripts. It contains questions that require deep understanding and reasoning over the entire narrative to answer correctly. The answer to each question is not limited to specific spans of text within the story.

Table 3 shows the statistics of these four datasets. In particular, the testing set in MNLI contains the matched and mismatched testing sets.

A.3 Target Models

In this work, we aim to attack the generative encoder-decoder and decoder-only LLMs and consider BART, LLaMA, and Vicuna as the target LLMs. Specifically, we chose BART-large, the latest development Meta-LLaMA-3-8B, and the version of Vicuna-13b-1.5-16k, where Vicuna-13b-1.5-16k is fine-tuned on LLaMA 2. We select these three LLMs for the following reasons: (1) the target LLMs should be open-source since we need to extract the hidden outputs from them; (2) the target LLMs should be scalable and deployable within our limited computational resources; (3) the target LLMs should achieve state-of-the-art performances on general NLP tasks. As far as we know, these LLMs satisfy the above three requirements and the latter two LLMs achieve over 90% capability of Bard/ChatGPT (Chiang et al., 2023). Hence, we consider them the best-fit subject LLMs to deliver representative results and insights. To facilitate the evaluation of model performance, we refer to PromptBench (Zhu et al., 2023) and adopt different task instructions for tasks to guide the model in generating content in a fixed format (shown in Table 5). In our experiments, all the target models are fine-tuned with LoRa (Hu et al., 2021). Through an empirical study, we fine-tune the target models with different sizes of datasets and find that: (i) BART-large achieves the best performances with 1500 cases, and (ii) Vicuna-13B and LLaMA-3-8B achieve the best performances with 600 cases. Table 4 depicts the accuracy of the fine-tuned target models. Note that the accuracy of each fine-tuned BART-large model on NarrativeQA dataset is less

Model	MNLI	QQP	SQuAD2	NqrrativeQA
Vicuna-13B	85.41%	84.90%	81.60%	80.60%
LLaMA-3-8B	83.87%	83.10%	78.70%	81.40%
BART-large	82.06%	82.50%	76.60%	–

Table 4: Performances of fine-tuned target LLMs

Task	Task instruction
NLI	Assess the connection between the following sentences and classify it as "entailment", "neutral", or "contradict-ion":
DSD	Can these two statements be considered equal in mean-ing? Answer with "equivalent" or "not_equivalent":
RC	Discover the best answer of the question based on the context. If the context doesn't include an answer, respo-nd with "unanswerable":

Table 5: Task instructions used to fine-tune target LLMs

than 35%, and we do no conduct any experiment in this scenario.

A.4 Results

A.5 Distribution of MPs on Target Models

On the randomly sampled subset of MNLI, Figure 9 and Figure 10 depicts the distribution of MPs on LLaMA-3-8B and BART-large.

A.6 MP Prioritization

Figure 11 shows the performance of MP prioritization method on BART-large.

A.7 Ablation study on BART-large

Figure 12 depicts the impact of the selection of *compute unit* on BART-large.

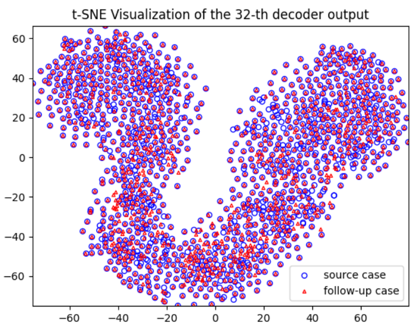


Figure 9: Distribution between the source cases and follow-up cases in LLaMA-3-8B

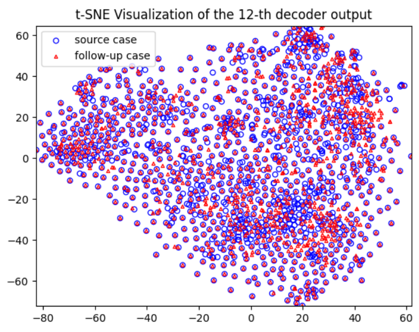
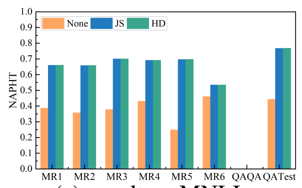
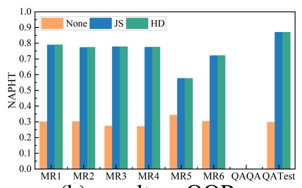


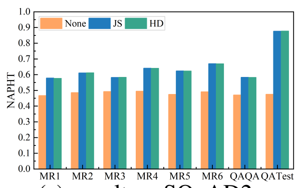
Figure 10: Distribution between the source cases and follow-up cases in BART-large



(a) result on MNLI

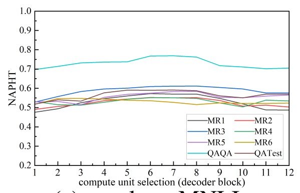


(b) result on QQP

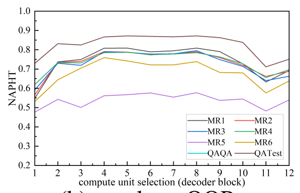


(c) result on SQuAD2

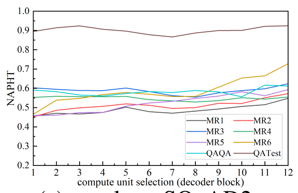
Figure 11: Prioritization performance of different diversity metrics under each MR on BART-large



(a) result on MNLI



(b) result on QQP



(c) result on SQuAD2

Figure 12: HD-based MP prioritization using different decoder blocks as the *compute unit* on BART-large.