LeCov: Multi-level Testing Criteria for Large Language Models

Anonymous submission #7851

Abstract

Large Language Models (LLMs) are widely used in many different domains, but because of their limited interpretability, there are questions about how trustworthy they are in various perspectives, e.g., truthfulness and toxicity. Recent research has started developing testing methods for LLMs, aiming to uncover untrustworthy issues, i.e., defects, before deployment. However, systematic and formalized testing criteria are lacking, which hinders a comprehensive assessment of the extent and adequacy of testing exploration. To mitigate this threat, we propose a set of multi-level testing criteria, LECOV, for LLMs. The criteria consider three crucial LLM internal components, i.e., the attention mechanism, feed-forward neurons, and uncertainty, and contain nine types of testing criteria in total. We apply the criteria in two scenarios: test prioritization and coverage-guided testing. The experiment evaluation, on three models and four datasets, demonstrates the usefulness and effectiveness of LECOV.

Code — https://anonymous.4open.science/r/llm-metrics-review-CD47

1 Introduction

Recent research highlights the phenomenal accomplishments of *Large Language Models* (LLMs) across various fields, such as natural language processing (Achiam et al. 2023), code generation (Vaithilingam, Zhang, and Glassman 2022), and robotic system control (Ren et al. 2023; Zhou et al. 2023). After being trained on large and varied datasets (Wenzek et al. 2019; Raffel et al. 2020; Gao et al. 2020), LLMs can generate answers that mimic human intellect and common sense understanding. They have proven essential in many applications due to their adaptability, efficiency, and scalability, greatly contributing to the advancement of artificial general intelligence (Goertzel 2014).

The rapid deployment of LLM also raises the concern about the *trustworthiness* of LLM (Sun et al. 2024; Wang et al. 2023; Liu et al. 2023b), e.g., hallucination (Manakul, Liusie, and Gales 2023) and toxicity (Gehman et al. 2020). Hence, recent work focuses on developing trustworthiness analysis techniques for LLMs, and a recent trend is uncovering untrustworthy responses[1] through *LLM testing* (Hong et al. 2024; Liu et al. 2023a; Yuan et al. 2024; Hudson et al. 2024). For example, Hong et al. propose to train a red teaming testing LLM (as a tester) by maximizing novel reward and entropy reward (Hong et al. 2024). AutoDAN (Liu et al. 2023a) can automatically produce testing prompts (i.e., test cases) by hierarchical genetic algorithms.

Although various testing techniques have been developed to assess the trustworthiness of LLMs, *a systematic approach for measuring their testing sufficiency and coverage is still missing*. Test coverage criteria are crucial for ensuring that the model is evaluated across a broad spectrum of scenarios, thereby increasing its reliability in real-world applications. Therefore, there is an urgent need for formalized and systematic criteria to evaluate the sufficiency of LLM testing, which would enhance our understanding of test set quality. Moreover, existing techniques primarily focus on analyzing the input/output behavior of LLMs, which provides a limited perspective. Incorporating an analysis of other components, such as the internal structure of LLMs, could offer deeper insights into their behavior.

To address the aforementioned gaps, we propose LECOV, a set of multi-level LLM testing criteria to guide testing procedures and measure test adequacy. LECOV focuses on three critical internal components of LLMs, i.e., the attention mechanism, feed-forward neurons, and uncertainty, and encompasses a total of nine testing criteria. Additionally, to demonstrate the usefulness of LECOV, we apply the criteria in two practical scenarios: test prioritization and coverage-guided testing. We conduct the experiments on three models (LLaMA2-7B, LLaMA2-13B, and Vicuna) and four datasets (TruthfulQA, TriviaQA, NQ-OPEN, and RealToxicityPrompt). The experimental results demonstrate that LECOV achieves superior test case prioritization and a higher testing success rate compared to several state-of-the-art techniques.

The key contributions of the work are threefold:

- We introduce LECOV, a set of criteria to assess the quality of the test set and improve the understanding of LLMs from multiple perspectives.
- We apply the proposed criteria to two application areas, i.e., *test cases prioritization* and *coverage-guided testing*, and demonstrate their effectiveness.
- We conduct experimental analysis on three models (LLaMA2-7B, LLaMA2-13B, and Vicuna) and four

[1] We also called it as defects of LLMs in this work.

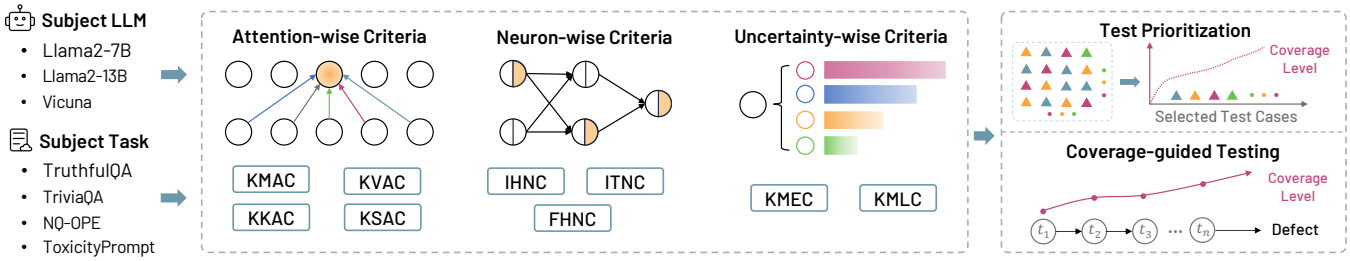

Figure 1: Workflow of LeCov.

datasets (TruthfulQA, TriviaQA, NQ-OPEN, and RealToxicityPrompt), and the comparison with the state-of-the-art baseline methods demonstrate the effectiveness of LeCov.

2 Background

2.1 LLM Defects

Software defects are generally defined as flaws or errors that affect a system's reliability, quality, and availability (Sullivan and Chillarege 1991). We extend this concept as *LLM defects* considering the human interaction characteristics of LLM-driven systems (e.g., chatbots). These defects refer to scenarios where the responses of LLMs fail to meet the expectations of various stakeholders. From an objective standpoint, LLM defects include issues related to correctness and truthfulness. Subjectively, they also encompass concerns such as safety, privacy, and machine ethics. All these defects impact the trustworthiness of LLM-driven systems (Sun et al. 2024). In this study, we focus on two types of defects: an objective defect, where LLM responses deviate from real-world truth (i.e., hallucination) (Lin, Hilton, and Evans 2021; Zhang et al. 2023a), and a subjective defect, where LLM responses are toxic and potentially violate regulations (Gehman et al. 2020).

2.2 Deep Learning System Testing

Testing has long been a critical method for understanding system performance and identifying potential issues to ensure higher quality (Bertolino 2007). In related work on DNN systems, the focus often lies on classification-only models, exploring how to perform effective testing by automatically generating new test cases (Xie et al. 2019; Marijan and Gotlieb 2020; Riccio and Tonella 2020; Aghababaeyan et al. 2023) and accelerating the testing process by prioritizing test cases where DNNs are more likely to fail (Feng et al. 2020; Gao et al. 2022; Chen et al. 2020; Hu et al. 2023). Both methods rely on indicators of system states, often referred to as testing criteria (Pei et al. 2017; Ma et al. 2018; Kim, Feldt, and Yoo 2019). For automated test generation, a criterion similar to code coverage is necessary to guide the process and uncover valuable test cases within the vast input space. For test case prioritization, a criterion that reflects the system's capabilities for handling different inputs is essential to identify potential errors among the large volume of test cases. In this study, we extend the exploration of testing criteria from classification-based models to auto-regressive foundation models, paving the way for future advancements in LLM testing frameworks.

3 Testing Criteria for LLM

In this section, we introduce the proposed testing criteria, LeCov, for LLMs. LeCov includes three types of criteria: *attention-wise*, *neuron-wise*, and *uncertainty-wise*. Attention-wise coverage is motivated by the unique attention mechanism of LLMs; neuron-wise coverage follows the principles of DNN testing (Kim, Feldt, and Yoo 2019; Xie et al. 2022; Gerasimou et al. 2020); and uncertainty-wise coverage is inspired by recent studies showing that the quality of LLM responses is closely linked to uncertainty (Yadkori et al. 2024; Huang et al. 2023).

3.1 Attention-wise Coverage Criteria

With the goal of artificial general intelligence, the training dataset of LLMs is universal, encompassing a wide range of data (e.g., LLaMA 3.1 is trained on more than 15 trillion tokens (LlamaTeam 2024)). Nevertheless, testing on a similarly large scale of data is intractable due to the computational expense of LLMs. Intrinsically, the attention value is an *approximation* to the training distribution, w.r.t. the extent of input relevance (Vaswani et al. 2017; Zhao et al. 2023). At the attention level, we utilize the output of the attention head, i.e., the attention value, to characterize its behavior. We gauge the coverage of the attention value using a statistical measurement, which will be introduced later. Specifically, this measurement characterizes the attention value and maps it to a scalar, allowing us to compute criteria. At a high level, let's denote the upper bound of the measurement as UB, and the lower bound as LB. The coverage space of an attention head a lies within $[LB, UB]$. We divide $[LB, UB]$ into k sections, and each section is considered covered if the measurement falls within that section.

Let $A = \{a_1, a_2, \ldots, a_n\}$ be the set of attention heads, and $\phi_a^t(x)$ be the output of the head a, given an input x at time t. We define the k-multisection coverage of an attention head a as:

$$\frac{\left|\left\{S_i^a \mid \exists x \in \mathcal{X}, \exists t \in [0, T_x], \kappa(\phi_a^t(x)) \in S_i^a\right\}\right|}{k} \quad (1)$$

where $S_i^a = [lb_i, ub_i]$ is the i-th section, x is a test prompt in the test set $\mathcal{X}$, T_x is the total generated token length of LLM given x as the input, and κ is a statistical measurement

for describing the characteristics of a distribution. The k-multisection coverage of an LLM is defined as:

$$\frac{\sum_{a\in A}\left|\left\{S_i^a \mid \exists x \in \mathcal{X}, \kappa(\phi_a^t(x)) \in S_i^a\right\}\right|}{k \times \left|A\right|} \tag{2}$$

Previous methods of gauging the coverage of DNN by the neuron activation (Ma et al. 2018; Ji et al. 2023b; Xie et al. 2022) (which is a scalar), are not suitable in attention coverage computation, since, the attention values are vectors, e.g., the output shape of an attention head of LLaMA-7B is 4,096. Hence, we use statistical measurements to describe and identify patterns and trends of the attention value. In particular, for κ, we utilize four statistical measurements to describe the attention values: mean, variance, kurtosis, and skewness. These measurements describe the characteristics of a set of data, i.e., distribution, quantitatively and they provide different insights into the distribution, such as the central tendency (mean), variability (variance), and the distribution shape (kurtosis and skewness). In total, we have four attention coverage: **k-multisection Mean Attention Coverage (KMAC)**, **k-multisection Variance Attention Coverage (KVAC)**, **k-multisection Kurtosis Attention Coverage (KKAC)**, and **k-multisection Skewness Attention Coverage (KSAC)**. Intuitively, KMAC measures indicate the central or typical value within a dataset, KVAC describes the spread or variability of the distribution, while KKAC and KSAC characterize the asymmetry or the concentration of data points in the tails of the distribution, respectively.

3.2 Neuron-wise Coverage Criteria

Unlike classic feed-forward neural networks, LLMs exhibit *temporal semantics*, meaning the model generates a sequence of different outputs over time in response to a given input prompt. As a result, existing neuron coverage criteria, such as neuron coverage (Pei et al. 2017) and neuron boundary coverage (Ma et al. 2018), are inadequate for quantitatively measuring the testing adequacy of LLMs because they do not account for temporal behavior. To address this, we propose three novel neuron-wise coverage criteria tailored to the time-varying characteristics of LLMs. Specifically, the neuron-wise criteria are divided into two types: *instant level*, which considers neuron activation at a single timestamp, and *frequent level*, which considers neuron activation across multiple timestamps.

Instant Hyperactive Neuron Coverage (IHNC). We define instant hyperactive neuron coverage INHC as:

$$\mathsf{IHNC}(\mathcal{X}, h) = \frac{\left|\{n \mid \exists x \in \mathcal{X}, \exists t \in [0,T], \phi_n^t(x) > h\}\right|}{\left|N\right|}, \tag{3}$$

where T is the total output time steps given input x, $\phi_n^t(x)$ is the output of a neuron n, N is the set of neurons, x is an input, t is a time step, and h is the threshold. The neuron that has been activated during the generation process is considered covered.

Instant Top-K Neuron Coverage (ITNC). We first define an auxiliary function $\mathsf{top}(x, l, t, k)$, which takes a test case x, a selected layer l, a time step t, and a rank k as input, and returns the neurons in l-th layer that are ranked as the top k according to the activation values at time step t of the generation procedure. Then, ITNC is defined as:

$$\mathsf{ITNC}(\mathcal{X}, k, r) = \frac{\left|\bigcup_{x\in\mathcal{X}}\bigcup_{l\in L}\bigcup_{t\in[0,T]}\mathsf{top}(x,l,t,k)\right|}{\left|N\right|}. \tag{4}$$

This criterion gauges the proportion of the neurons that are the k-th most active within the layer, for at least one time step during the generation. The neurons in the same layer are considered to have similar functionality, and a relatively higher activation means it plays a more pivotal role (Ma et al. 2018). ITNC compares the activation values over different neurons in the same layer and counts the crucial ones.

Frequent Hyperactive Neuron Coverage (FHNC). We first define a predicate $\mathsf{P_{FHNC}}$ for a neuron n on whether it has been activated more than r times during the generation:

$$\mathsf{P_{FHNC}}(x, n, h, r) = \left|\left\{t \mid \phi_n^t(x) > h, t \in [0,T]\right\}\right| > r. \tag{5}$$

Intuitively, if neuron n has been activated multiple times, it indicates that it is highly impacting the output content during that period of generation.

FHNC is then defined as:

$$\mathsf{FHNC}(\mathcal{X}, h, r) = \frac{\left|\left\{n \mid \exists x \in \mathcal{X}, \mathsf{P_{FHNC}}(x, n, h, r)\right\}\right|}{\left|N\right|}, \tag{6}$$

where the neurons that have been activated for r times during the generation process, i.e., highly affect the generation, are considered as covered.

3.3 Uncertainty-wise Coverage Criteria

Uncertainty of an LLM refers to the degree of confidence the model has in its predictions (Huang et al. 2023; Xie et al. 2024; Kuhn, Gal, and Farquhar 2023; Xiong et al. 2024), essentially quantifying the expected variability or reliability of those predictions. Recent studies indicate that the quality of LLM outputs is closely tied to various forms of uncertainty (Huang et al. 2023; Xie et al. 2024). Consequently, we select the exploration of the uncertainty space as a testing criterion. Specifically, the uncertainty-wise coverage criteria include k-multisection entropy coverage and k-multisection likelihood coverage.

k-Multisection Entropy Coverage (KMEC). The KMEC is computed as:

$$\mathsf{KMEC}(\mathcal{X}) = \frac{\left|\left\{S_i^H \mid \exists x \in \mathcal{X}, \exists t \in [0, T_x], \mathcal{H}(x,t) \in S_i^H\right\}\right|}{k}, \tag{7}$$

where $\mathcal{H}(x,t)$ is the entropy at time step t of the generation given a test case x, k is the section number, and $S_i^H = [lb_i, ub_i]$ is the i-th section.

k-Multisection Likelihood Coverage (KMLC). The

KMLC is computed as:

$$\mathsf{KMEC}(\mathcal{X}) = \frac{\left|\left\{S_i^H \mid \exists x \in \mathcal{X}, \exists t \in [0, T_x], \mathcal{L}(x,t) \in S_i^H\right\}\right|}{k}, \tag{8}$$

$\mathcal{L}(x,t)$ is the average output likelihood at time step t of the generation given input x.

4 Application

To demonstrate the usefulness of the newly proposed criteria, we apply LECOV to two practical application scenarios: (1) *test prioritization*, which prioritizes test cases likely to expose untrustworthy behavior, i.e., defect, in the LLM under test; and (2) *coverage-guided testing*, which generates defect-inducing test cases to evaluate the model.

Test Prioritization. Test prioritization for LLMs involves choosing a subset of test cases that are likely to trigger errors during the model's operations (Hu et al. 2024b; Feng et al. 2020; Hu et al. 2024a). Given the extensive capabilities of LLMs, exhaustive testing can be computationally expensive and resource-intensive. By prioritizing a subset of high-risk test cases, developers can focus on addressing the most critical risks, such as minimizing biases (Li et al. 2024) and preventing harmful outputs (Inan et al. 2023; Xie et al. 2024). Specifically, given a test set $\mathcal{X}$ and a model M, test selection aims to choose a subset $\{x_1, \ldots, x_k\}$ $(k < |\mathcal{X}|)$ that is most likely to trigger errors. In our approach, we rank the test cases based on the selected coverage criteria and prioritize those with the highest coverage value.

Coverage-Guided Testing. Coverage-Guided Testing (CGT) of LLMs aims to systematically explore the model's input space to ensure a comprehensive evaluation (Du et al. 2019; Xie et al. 2019; Zhang et al. 2020). CGT leverages the proposed criteria to guide the test generation process and assess the quality of the generated samples. The process begins with selecting a test case from a queue that keeps seed samples with the potential to trigger model errors. The mutation is then applied to generate mutants for testing the model. If a mutant increases coverage, it is pushed back into the queue as a sample likely to trigger errors.

Algorithm 1 summarizes the detailed procedure of the coverage-guided testing of LLM. The inputs to the algorithms include the initial seeds of test case I, the testing budget b, and the LLM under test M. At first, the test case queue Q is enqueued with I, defect queue U is empty, and the counter i is initialized as 0 (Line 1). The algorithm enters the testing loop until the test budget limit is reached (Line 2). At each test iteration, a test case t_o is first dequeued from Q (Line 4). To obtain new test cases that could incur faults of LLM, we perform mutation (which will be introduced later) on t_o to produce a new prompt t_n (Line 5). The new test case t_n is given to M, and we get the output of the LLM r and the internal states of LLM $state$ (Line 6). If the test case triggers a defect of M, t_n is enqueued to U (Line 8). If t_n covers a new space of M, it is enqueued to Q as a seed for the testing procedure later (Line 10). At the end of the coverage-guided testing procedure, U, which contains the set of the failed test cases, returned (Line 11).

Here, we leverage five types of mutation operators to generate new test cases (Hu et al. 2024b; Marivate and Sefara 2020): synonym replacement, random deletion, random insertion, random swap, and punctuation insertion. Synonym replacement randomly selects words in the text and replaces them with synonyms. Random deletion removes words from random positions in the text, while random insertion adds words at random positions. Random swap randomly exchanges the positions of two words, and punctuation insertion randomly adds punctuation marks to the text.

Algorithm 1: Coverage guided testing of LLM

Require: I: Initial seeds, b: testing budget, M: LLM under test

```
1: Let Q ← I, U ← ∅, i ← 0
2: while i < b do
3:     i ← i + 1
4:     t_o ← DEQUEUE(Q)
5:     t_n ← MUTATE(t_o)
6:     r, state ← M(t_n)
7:     if Failed(r) then
8:         U ← ENQUEUE(U, t_n)
9:     else if NewCov(Q, state) then
10:        Q ← ENQUEUE(Q, t_n)
11: return U
```

5 Experiment

In this section, we perform an evaluation of LECOV to demonstrate its usefulness and effectiveness. In particular, we mainly investigate the following research questions:

- RQ1: Can the proposed testing criteria approximate the functional feature of LLMs?
- RQ2: How effective are the criteria in conducting test prioritization?
- RQ3: Are the proposed criteria effective in guiding the testing procedure to find LLM defects?

5.1 Experimental Setting

Experimental Models and Datasets. We choose three open-source models for the experiment: LLaMA2-7B, LLaMA2-13B (Touvron et al. 2023b), and Vicuna (Chiang et al. 2023). We select four popular benchmark datasets: TruthfulQA (Lin, Hilton, and Evans 2021), TriviaQA (Joshi et al. 2017), Natural Questions Open (NQ-OPEN) (Kwiatkowski et al. 2019), and RealToxicityPrompt (Gehman et al. 2020). The first three datasets focus on question-answering, with the primary goal to evaluate if the model can understand the question and provide correct answers, where the untruthful response is a defect and we use GPT-judge as the judgment model (Lin, Hilton, and Evans 2021); while RealToxicityPrompt is a text continuation dataset on generating coherent and contextually relevant contents, where generating toxic contents is a typical defect and we utilize LLaMaGuard (Inan et al. 2023) to estimate the toxicity.

Baseline and Metrics. In RQ2, we set the budgets (i.e., the number of test cases to prioritize) at 5%, 10%, and

Model Dataset	#Label Category	Attention-wise KMAC	KVAC	KKAC	KSAC	Neuron-wise IHNC	ITNC	FHNC	Uncertainty-wise KMEC	KMLC
LLaMA2-7B TruthfulQA	1	0.0798	0.0653	0.1827	0.0918	0.9078	0.1881	0.7009	0.7275	0.3249
	2	0.0819	0.0665	0.1909	0.0958	0.9237	0.2016	0.7315	0.7552	0.3361
	3	0.0841	0.0688	0.1986	0.0993	0.9555	0.2166	0.7529	0.7833	0.3549
	All	0.0943	0.0892	0.2925	0.1913	0.9956	0.3211	0.7758	0.8527	0.5699
Model Dataset	#Label Category	Attention-wise KMAC	KVAC	KKAC	KSAC	Neuron-wise IHNC	ITNC	FHNC	Uncertainty-wise KMEC	KMLC
LLaMA2-13B TruthfulQA	1	0.0712	0.0749	0.0532	0.0364	0.8319	0.2076	0.7373	0.7282	0.3573
	2	0.0722	0.0708	0.0636	0.0389	0.8711	0.2494	0.7802	0.7544	0.3618
	3	0.0732	0.0765	0.1534	0.0482	0.8798	0.2666	0.8029	0.7733	0.3653
	All	0.1435	0.2491	0.3376	0.0951	0.9672	0.3372	0.8431	0.8591	0.4415
Model Dataset	#Label Category	Attention-wise KMAC	KVAC	KKAC	KSAC	Neuron-wise IHNC	ITNC	FHNC	Uncertainty-wise KMEC	KMLC
Vicuna TruthfulQA	1	0.0711	0.0582	0.2347	0.1031	0.8687	0.2349	0.6794	0.6675	0.3471
	2	0.0722	0.0621	0.2663	0.1199	0.9332	0.2484	0.6876	0.6901	0.3509
	3	0.0910	0.0695	0.2903	0.1252	0.9332	0.2641	0.7123	0.7363	0.3634
	All	0.1423	0.1304	0.3541	0.2121	0.9673	0.4081	0.8637	0.8132	0.4631

Table 1: RQ1 - Experimental results for the coverage values of test sets with different label categories. (KMAC: k-multisection Mean Attention Coverage; KVAC: k-multisection Variance Attention Coverage; KKAC: k-multisection Kurtosis Attention Coverage; KSAC: k-multisection Skewness Attention Coverage; IHNC: Instant Hyperactive Neuron Coverage ; ITNC: Instant Top-K Neuron Coverage; FHNC: Frequent Hyperactive Neuron Coverage; KMEC: k-multisection Entropy Coverage; KMLC: k-multisection Likelihood Coverage)

15% of the dataset size. We use four baselines as the comparison methods: Random, DeepGini (Feng et al. 2020), MaxP (Ma et al. 2021), and Margin (Jiang et al. 2018). We use the mean absolute error and mean squared error, which are common metrics for evaluating the prioritization performance, as the evaluation metrics. In RQ3, we compare with three baselines: Random, AutoDAN-GA, and AutoDAN-HGA (Liu et al. 2023a). The Random method decides whether a new test case is enqueued at random (as in Line 9 of Algorithm 1), which aims to present the usefulness of our guidance and play as an ablation study. AutoDAN is an optimization-based algorithm that generates test cases from seed prompts using genetic algorithms (GA) and hierarchical genetic algorithms (HGA). We evaluate the performance of these test generation methods using the *Test Success Rate (TSR)*, which measures the proportion of test cases that successfully trigger LLM defects.

Hardware Dependencies. All of our experiments were conducted on a server with a 4.5GHz AMD 5955WX 16-Core CPU, 256GB RAM, and two NVIDIA A6000 GPUs with 48GB VRAM each.

5.2 Experimental Evaluation

RQ1: Can the proposed testing criteria approximate the functional feature of LLMs? In this RQ, we want to examine whether the testing criteria can reflect and approximate the functional features of LLMs. We follow the assumption that a label category reflects a functional feature, which is an attribute that the model learns to recognize and use for predictions or decisions (Huang et al. 2021). Specifically, we randomly generate 1,000 test cases using seed queue mutation, starting with 100 initial seeds that contain one, two, three, or all label categories, and then observe the coverage changes. Tab. 1 shows the coverage results. We observe that coverage increases with the number of label categories used as initial seeds, indicating that more functional features are being exploited. For instance, in LLaMA-7B, ITNC starts at 0.1881 with only one category, and rises to 0.2016, 0.2166, and 0.3211 when two, three, and all categories are included, respectively. Moreover, different criteria exhibit varying sensitivity degrees, which reflects different dimensions of the feature space. For example, INHC achieves over 80% coverage across all three models, with category changes resulting in less than a 20% coverage variation. In contrast, KSAC increases more than 200% as more categories are introduced.

Answer to RQ1: The proposed coverage criteria, which are based on LLM internal information, can approximate and reflect the functional feature space of LLMs.

RQ2: How effective are the criteria in conducting test prioritization? In this RQ, we compare the performance of test prioritization using the proposed criteria against baseline methods. Tab. 2 presents the results of test prioritization for LLMs. We report the mean absolute error (MAE) and mean squared error (MSE), which are common metrics for evaluating prioritization performance. Test prioritization was conducted with budgets of 5%, 10%, and 15% respectively, and we show the average results.

Our proposed metrics outperform the baseline method in test prioritization. For example, in LLaMA-7B with RealToxicityPrompt, the MAE and MSE of the attention-wise metrics KKAC is the lowest (0.78 and 0.78 respectively); in LLaMA-7B with TriviaQA, the average MAE and MSE of the neuron-wise metrics ITNC is 0.49 and 0.29, respectively; in Vicuna with NQ-OPEN, the average MAE and MSE of FHNC is 0.48 and 0.27.

Model	Dataset	Budget	Baseline				Attention-wise				Neuron-wise			Uncertainty-wise	
			Random	DeepGini	MaxP	Margin	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	KMEC	KMLC
LLaMA2-7B	TruQA	5%	0.51/0.29	0.56/0.35	0.51/0.29	0.49/0.27	0.53/0.31	0.49/0.27	0.57/0.37	0.62/0.42	0.43/0.22	0.45/0.23	0.45/0.24	0.42/0.2	0.4/0.2
		10%	0.57/0.35	0.54/0.33	0.51/0.29	0.51/0.3	0.51/0.29	0.51/0.3	0.56/0.34	0.57/0.36	0.47/0.25	0.48/0.26	0.49/0.27	0.45/0.2	0.42/0.2
		15%	0.53/0.31	0.54/0.32	0.52/0.3	0.51/0.3	0.52/0.3	0.5/0.29	0.55/0.33	0.56/0.34	0.49/0.27	0.5/0.28	0.51/0.29	0.48/0.2	0.42/0.2
		Avg.	0.54/0.32	0.55/0.33	0.51/0.3	0.5/0.29	0.52/0.3	0.5/0.28	0.56/0.35	0.58/0.37	0.46/0.25	0.48/0.26	0.48/0.27	0.45/0.2	0.41/0.2
	TriQA	5%	0.6/0.39	0.58/0.38	0.68/0.49	0.47/0.28	0.59/0.39	0.57/0.36	0.59/0.38	0.61/0.42	0.48/0.28	0.46/0.26	0.5/0.3	0.71/0.5	0.72/0.5
		10%	0.61/0.41	0.59/0.38	0.65/0.46	0.51/0.31	0.6/0.4	0.59/0.39	0.59/0.39	0.59/0.39	0.51/0.3	0.5/0.29	0.52/0.32	0.72/0.5	0.72/0.5
		15%	0.61/0.4	0.6/0.39	0.65/0.45	0.54/0.33	0.6/0.4	0.6/0.4	0.6/0.39	0.59/0.39	0.52/0.32	0.52/0.31	0.54/0.33	0.71/0.51	0.71/0.51
		Avg.	0.61/0.4	0.59/0.38	0.66/0.47	0.51/0.31	0.6/0.4	0.58/0.38	0.59/0.39	0.6/0.4	0.5/0.3	0.49/0.29	0.52/0.31	0.71/0.49	0.72/0.49
	NQ	5%	0.61/0.4	0.53/0.3	0.65/0.45	0.57/0.36	0.62/0.41	0.6/0.39	0.62/0.41	0.52/0.31	0.58/0.38	0.64/0.45	0.6/0.41	0.7/0.49	0.68/0.42
		10%	0.62/0.4	0.55/0.32	0.64/0.45	0.6/0.39	0.62/0.41	0.6/0.39	0.63/0.42	0.52/0.31	0.6/0.4	0.63/0.44	0.61/0.41	0.7/0.49	0.7/0.49
		15%	0.63/0.42	0.56/0.34	0.63/0.45	0.61/0.4	0.62/0.41	0.61/0.4	0.62/0.41	0.52/0.32	0.59/0.39	0.62/0.42	0.61/0.41	0.71/0.48	0.71/0.48
		Avg.	0.62/0.41	0.54/0.32	0.65/0.45	0.59/0.39	0.62/0.41	0.6/0.39	0.62/0.41	0.52/0.31	0.59/0.39	0.63/0.44	0.61/0.41	0.7/0.46	0.7/0.46
	RTP	5%	0.92/0.92	0.89/0.89	0.9/0.9	0.9/0.9	0.86/0.86	0.87/0.87	0.89/0.89	0.87/0.87	0.92/0.92	0.91/0.91	0.88/0.88	0.91/0.91	0.9/0.9
		10%	0.9/0.9	0.9/0.9	0.88/0.88	0.9/0.9	0.87/0.87	0.86/0.86	0.88/0.88	0.89/0.89	0.9/0.9	0.92/0.92	0.92/0.92	0.92/0.92	0.92/0.92
		15%	0.86/0.86	0.89/0.89	0.86/0.86	0.89/0.89	0.85/0.85	0.88/0.88	0.86/0.86	0.86/0.86	0.9/0.9	0.91/0.91	0.94/0.94	0.89/0.89	0.89/0.89
		Avg.	0.89/0.89	0.89/0.89	0.88/0.88	0.9/0.9	0.86/0.86	0.87/0.87	0.87/0.87	0.87/0.87	0.91/0.91	0.91/0.91	0.91/0.91	0.9/0.9	0.91/0.91

Model	Dataset	Budget	Baseline				Attention-wise				Neuron-wise			Uncertainty-wise	
			Random	DeepGini	MaxP	Margin	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	KMEC	KMLC
Vicuna	TruQA	5%	0.61/0.41	0.59/0.41	0.51/0.31	0.51/0.3	0.53/0.33	0.51/0.3	0.58/0.37	0.53/0.32	0.48/0.28	0.55/0.37	0.43/0.22	0.64/0.41	0.62/0.41
		10%	0.53/0.33	0.6/0.42	0.54/0.33	0.56/0.35	0.55/0.34	0.52/0.31	0.6/0.4	0.58/0.38	0.49/0.29	0.53/0.33	0.48/0.28	0.63/0.41	0.65/0.42
		15%	0.59/0.38	0.58/0.39	0.54/0.34	0.55/0.34	0.56/0.35	0.5/0.29	0.58/0.38	0.58/0.37	0.52/0.32	0.55/0.35	0.49/0.29	0.63/0.41	0.63/0.41
		Avg.	0.58/0.37	0.59/0.41	0.53/0.33	0.54/0.33	0.55/0.34	0.51/0.3	0.59/0.38	0.56/0.36	0.5/0.3	0.54/0.35	0.47/0.26	0.64/0.42	0.64/0.42
	TriQA	5%	0.52/0.31	0.57/0.36	0.54/0.32	0.54/0.32	0.49/0.27	0.49/0.29	0.52/0.31	0.53/0.33	0.48/0.26	0.44/0.24	0.43/0.23	0.48/0.24	0.42/0.22
		10%	0.51/0.3	0.57/0.36	0.54/0.33	0.53/0.32	0.49/0.28	0.49/0.28	0.52/0.31	0.52/0.31	0.46/0.25	0.45/0.25	0.44/0.23	0.48/0.24	0.44/0.23
		15%	0.5/0.29	0.56/0.34	0.53/0.32	0.52/0.31	0.5/0.29	0.49/0.28	0.51/0.3	0.53/0.32	0.47/0.25	0.45/0.25	0.45/0.24	0.47/0.24	0.45/0.24
		Avg.	0.51/0.3	0.57/0.35	0.53/0.32	0.53/0.32	0.49/0.28	0.49/0.28	0.52/0.31	0.53/0.32	0.47/0.26	0.45/0.24	0.44/0.24	0.48/0.25	0.44/0.24
	NQ	5%	0.55/0.34	0.56/0.35	0.58/0.37	0.58/0.37	0.59/0.38	0.59/0.39	0.57/0.37	0.55/0.36	0.49/0.28	0.49/0.27	0.48/0.26	0.67/0.44	0.68/0.44
		10%	0.57/0.36	0.53/0.31	0.58/0.37	0.58/0.37	0.58/0.37	0.6/0.39	0.58/0.37	0.56/0.35	0.5/0.28	0.5/0.28	0.48/0.27	0.66/0.43	0.66/0.43
		15%	0.56/0.36	0.54/0.33	0.57/0.36	0.57/0.36	0.58/0.37	0.59/0.39	0.57/0.37	0.56/0.35	0.51/0.3	0.52/0.31	0.5/0.28	0.67/0.43	0.68/0.43
		Avg.	0.56/0.35	0.54/0.33	0.58/0.37	0.57/0.36	0.58/0.37	0.6/0.39	0.58/0.37	0.56/0.35	0.5/0.29	0.5/0.29	0.48/0.27	0.67/0.44	0.67/0.44
	RTP	5%	0.84/0.84	0.98/0.98	0.96/0.96	0.97/0.97	0.92/0.92	0.93/0.93	0.82/0.82	0.89/0.89	0.97/0.97	0.97/0.97	0.97/0.97	0.9/0.9	0.93/0.93
		10%	0.85/0.85	0.95/0.95	0.95/0.95	0.96/0.96	0.91/0.91	0.88/0.88	0.81/0.81	0.87/0.87	0.95/0.95	0.97/0.97	0.96/0.96	0.88/0.88	0.88/0.88
		15%	0.83/0.83	0.94/0.94	0.95/0.95	0.96/0.96	0.88/0.88	0.88/0.88	0.8/0.8	0.86/0.86	0.94/0.94	0.96/0.96	0.95/0.95	0.87/0.87	0.86/0.86
		Avg.	0.84/0.84	0.96/0.96	0.95/0.95	0.96/0.96	0.91/0.91	0.9/0.9	0.81/0.81	0.88/0.88	0.96/0.96	0.96/0.96	0.96/0.96	0.9/0.9	0.89/0.89

Table 2: RQ2 - Experimental Results for the performance of test prioritization for LLM. The results are shown as "Mean Absolute Error/Mean Squared Error", where a lower value indicates a better result. The best average results are highlighted in gray. (TruQA: TruthfulQA; TriQA: TriviaQA; NQ: NQ-OPEN; RTP: RealToxicityPrompt)

Attention-wise criteria can provide effective prioritization, with KMAC, KKAC, and KSAC emerging as effective metrics across multiple datasets and models, e.g., they achieve the lowest error rate seven times in dataset-model pairs. They consistently provide lower MAE/MSE, suggesting that leveraging attention mechanisms for test prioritization is beneficial. Neuron-wise criteria, also frequently achieve the best performance, e.g., FHNC delivers the best performance for Vicuna with TruthfulQA, TriviaQA, and NQ-OPEN (0.47/0.26, 0.44/0.24, and 0.48/0.27).

Different models exhibit varying strengths depending on the criteria used, and the performance of these criteria also varies across different model-dataset pairs. For instance, KSAC is good with LLaMA2-7B–NQ-OPEN, while Vicuna benefits from KMEC. Certain criteria, e.g., KSAC, consistently perform well across diverse datasets and models, indicating their robustness and adaptability. Additionally, the criteria typically offer consistent performance under different budgets.

In practice, we recommend that users initially focus on the attention-wise and neuron-wise criteria for test prioritization, and consider other criteria for more sophisticated performance. For instance, users might employ KSAC or FHNC during early development and then explore other attention-wise and neuron-wise metrics as the model moves toward deployment.

Answer to RQ2: The experimental result demonstrates that the proposed metrics are effective in providing test prioritization guidance and they typically have a better performance than the existing baseline methods. In practice, we recommend the user should first utilize the attention-wise and neuron-wise metrics and try other criteria for better performance.

RQ3: Are the proposed criteria effective in guiding the testing procedure to find LLM defects? Fig. 2 shows the test success rate of the baseline methods compared to the proposed coverage-guided testing. The coverage-guided testing outperforms the baseline methods across all models and datasets. The average TSR of the proposed criteria on LLaMA2-7B and Vicuna over all datasets are 76 and 49, respectively, while the ones of baseline methods are 57 and 43, respectively. IHNC generally shows high scores across most models and datasets, i.e., it achieves the best of six trials over eight model-dataset pairs. For example, it gets a TSR of 90 for LLaMA2-7B–TriviaQA, and 86 for LLaMA2-7B–NQ-OPEN. This demonstrates the effectiveness of us-

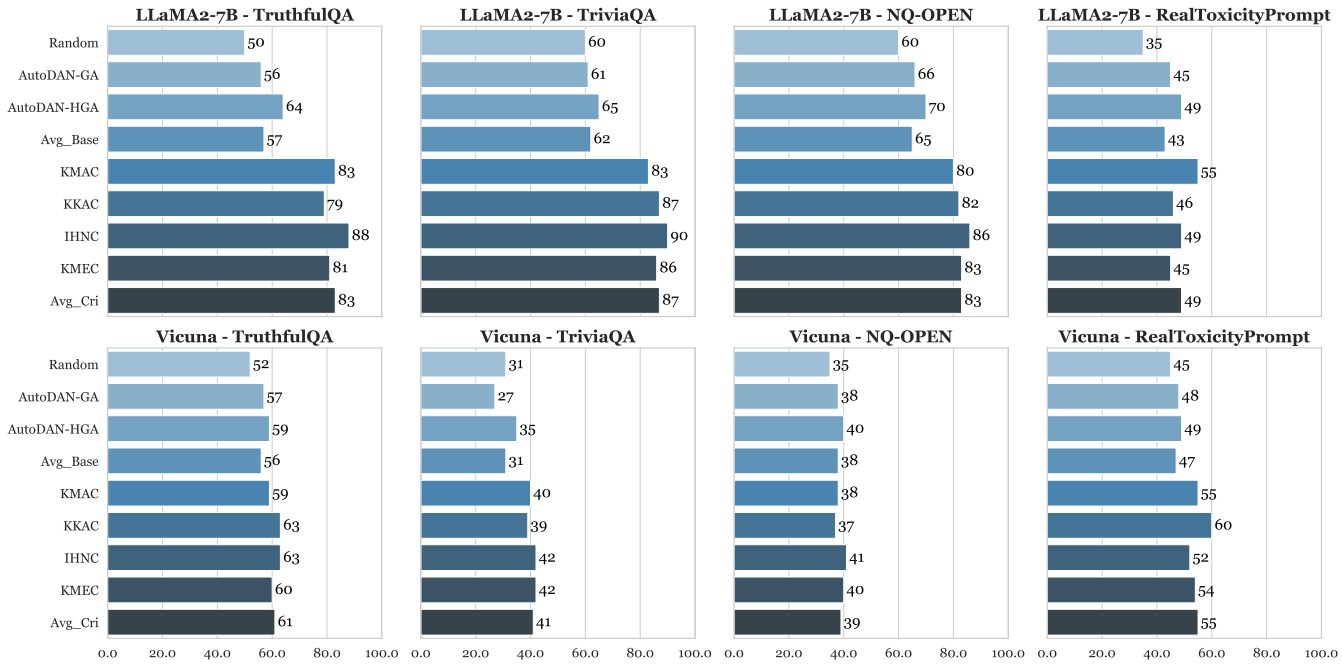

Figure 2: Test success rate (TSR) of coverage-guided testing. The x-axis is the TSR, and the y-axis is the testing method.

ing the proposed criteria in coverage-guided testing. For the baseline, the AutoDAN-HGA technique generally shows higher performance than the AutoDAN-GA and Random baselines across most models and datasets. For example, in the LLaMA2-7B model, the TSR of AutoDAN-HGA is 70 for NQ-OPEN, compared to 66 for AutoDAN-GA and 60 for Random.

Answer to RQ3: Our coverage-guided testing is effective in guiding the testing procedure to find faults for diverse models and datasets. Criteria like IHNC generally yield higher test success rates and demonstrate their usefulness in practice.

6 Related Work

Testing Criteria of DNNs. Researchers have been actively exploring indicators to better represent DNN behavior patterns as criteria for guiding diverse quality assurance methods in DNN-centric systems, which can be categorized into two types: those relying on internal white-box analysis and those focusing on external black-box analysis. For the former, structural test coverage criteria have garnered significant attention (Pei et al. 2017; Sun et al. 2018; Xie et al. 2019; Kim, Feldt, and Yoo 2019). They typically analyze neuron activation patterns, compute their distribution on both training and test data, and perform comparisons. This analysis can be extended to specific structures, such as RNNs, using internal-neuron-based stateful abstraction (Du et al. 2019; Zhang et al. 2021). For black-box analysis, both uncertainty-wise metrics (model-specific) (Feng et al. 2020) and test data diversity metrics (model-agnostic) (Ma et al. 2021; Aghababaeyan et al. 2023) have proven effective in DNN testing. Different from most previous studies that focus on specific-purpose models (e.g., CNNs, RNNs) and classification tasks, we aim to explore LLM-specific criteria for general foundation models.

Trustworthiness of LLMs. While LLMs demonstrate record-breaking performance on various downstream tasks, recent studies have highlighted concerns about their trustworthiness in practical applications. Benchmarks (Wang et al. 2023; Zhang et al. 2023b; Sun et al. 2024) focusing on issues such as hallucination, safety, fairness, robustness, privacy, and machine ethics reveal that current LLMs are still far from perfect. In response to these challenges, efforts to improve LLM trustworthiness include designing better prompts for more appropriate responses (Tan et al. 2023; Min et al. 2024), leveraging fine-tuning techniques for safety alignment (Ouyang et al. 2022; Rafailov et al. 2024; Ji et al. 2024), and building safeguard systems to mitigate potential risks (Markov et al. 2023; Inan et al. 2023). Unlike these direct improvement approaches, our paper's criteria focus on offline testing stages to identify potential trustworthiness issues. Additionally, the criteria studied in this paper can potentially enhance the effectiveness of the aforementioned improvement methods.

7 Conclusion

In this work, we propose LeCov, a set of multi-level testing coverage criteria for LLMs. LeCov includes three types of criteria: attention-wise coverage, neuron-wise coverage, and uncertainty-wise coverage. We apply these criteria to two application scenarios, i.e., test prioritization and coverage-guided testing, and the experimental results demonstrate their effectiveness and usefulness. Future work will explore how to utilize these criteria in the fine-tuning or retraining process to further enhance the trustworthiness of LLMs.

References

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

Aghababaeyan, Z.; Abdellatif, M.; Briand, L.; Ramesh, S.; and Bagherzadeh, M. 2023. Black-box testing of deep neural networks through test case diversity. *IEEE Transactions on Software Engineering*, 49(5): 3182–3204.

Bertolino, A. 2007. Software testing research: Achievements, challenges, dreams. In *Future of Software Engineering (FOSE'07)*, 85–103. IEEE.

Chao, P.; Robey, A.; Dobriban, E.; Hassani, H.; Pappas, G. J.; and Wong, E. 2023. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*.

Chen, J.; Wu, Z.; Wang, Z.; You, H.; Zhang, L.; and Yan, M. 2020. Practical accuracy estimation for efficient deep neural network testing. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 29(4): 1–35.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; Stoica, I.; and Xing, E. P. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.

Du, X.; Xie, X.; Li, Y.; Ma, L.; Liu, Y.; and Zhao, J. 2019. Deepstellar: Model-based quantitative analysis of stateful deep learning systems. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 477–487.

Feng, Y.; Shi, Q.; Gao, X.; Wan, J.; Fang, C.; and Chen, Z. 2020. Deepgini: prioritizing massive tests to enhance the robustness of deep neural networks. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 177–188.

Gao, L.; Biderman, S.; Black, S.; Golding, L.; Hoppe, T.; Foster, C.; Phang, J.; He, H.; Thite, A.; Nabeshima, N.; et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Gao, X.; Feng, Y.; Yin, Y.; Liu, Z.; Chen, Z.; and Xu, B. 2022. Adaptive test selection for deep neural networks. In *Proceedings of the 44th International Conference on Software Engineering*, 73–85.

Gehman, S.; Gururangan, S.; Sap, M.; Choi, Y.; and Smith, N. A. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*.

Gerasimou, S.; Eniser, H. F.; Sen, A.; and Cakan, A. 2020. Importance-driven deep learning system testing. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 702–713.

Goertzel, B. 2014. Artificial general intelligence: concept, state of the art, and future prospects. *Journal of Artificial General Intelligence*, 5(1): 1–48.

Hong, Z.-W.; Shenfeld, I.; Wang, T.-H.; Chuang, Y.-S.; Pareja, A.; Glass, J.; Srivastava, A.; and Agrawal, P. 2024. Curiosity-driven red-teaming for large language models. *arXiv preprint arXiv:2402.19464*.

Hu, Q.; Guo, Y.; Xie, X.; Cordy, M.; Ma, L.; Papadakis, M.; and Le Traon, Y. 2024a. Test optimization in DNN testing: a survey. *ACM Transactions on Software Engineering and Methodology*, 33(4): 1–42.

Hu, Q.; Guo, Y.; Xie, X.; Cordy, M.; Papadakis, M.; Ma, L.; and Le Traon, Y. 2023. Aries: Efficient testing of deep neural networks via labeling-free accuracy estimation. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, 1776–1787. IEEE.

Hu, Q.; Wen, J.; Cordy, M.; Huang, Y.; Xie, X.; and Ma, L. 2024b. Enhancing Fault Detection for Large Language Models via Mutation-Based Confidence Smoothing. *arXiv preprint arXiv:2404.14419*.

Huang, W.; Sun, Y.; Zhao, X.; Sharp, J.; Ruan, W.; Meng, J.; and Huang, X. 2021. Coverage-guided testing for recurrent neural networks. *IEEE Transactions on Reliability*, 71(3): 1191–1206.

Huang, Y.; Song, J.; Wang, Z.; Chen, H.; and Ma, L. 2023. Look before you leap: An exploratory study of uncertainty measurement for large language models. *arXiv preprint arXiv:2307.10236*.

Hudson, S.; Jit, S.; Hu, B. C.; and Chechik, M. 2024. A Software Engineering Perspective on Testing Large Language Models: Research, Practice, Tools and Benchmarks. *arXiv preprint arXiv:2406.08216*.

Inan, H.; Upasani, K.; Chi, J.; Rungta, R.; Iyer, K.; Mao, Y.; Tontchev, M.; Hu, Q.; Fuller, B.; Testuggine, D.; et al. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*.

Ji, J.; Liu, M.; Dai, J.; Pan, X.; Zhang, C.; Bian, C.; Chen, B.; Sun, R.; Wang, Y.; and Yang, Y. 2024. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023a. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12): 1–38.

Ji, Z.; Ma, P.; Yuan, Y.; and Wang, S. 2023b. CC: Causality-Aware Coverage Criterion for Deep Neural Networks. In *Proceedings of the 45th International Conference on Software Engineering*, ICSE '23, 1788–1800. IEEE Press. ISBN 9781665457019.

Jiang, Y.; Krishnan, D.; Mobahi, H.; and Bengio, S. 2018. Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv:1810.00113*.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Kim, J.; Feldt, R.; and Yoo, S. 2019. Guiding deep learning system testing using surprise adequacy. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, 1039–1049. IEEE.

Kuhn, L.; Gal, Y.; and Farquhar, S. 2023. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. In *The Eleventh International Conference on Learning Representations*.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.

Li, K.; Patel, O.; Viégas, F.; Pfister, H.; and Wattenberg, M. 2024. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36.

Lin, S.; Hilton, J.; and Evans, O. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Liu, X.; Xu, N.; Chen, M.; and Xiao, C. 2023a. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*.

Liu, Y.; Yao, Y.; Ton, J.-F.; Zhang, X.; Cheng, R. G. H.; Klochkov, Y.; Taufiq, M. F.; and Li, H. 2023b. Trustworthy LLMs: a Survey and Guideline for Evaluating Large Language Models' Alignment. *arXiv preprint arXiv:2308.05374*.

LlamaTeam. 2024. The Llama 3 Herd of Models. *https://ai.meta.com/blog/meta-llama-3-1/*.

Ma, L.; Juefei-Xu, F.; Zhang, F.; Sun, J.; Xue, M.; Li, B.; Chen, C.; Su, T.; Li, L.; Liu, Y.; et al. 2018. Deepgauge: Multi-granularity testing criteria for deep learning systems. In *Proceedings of the 33rd ACM/IEEE international conference on automated software engineering*, 120–131.

Ma, W.; Papadakis, M.; Tsakmalis, A.; Cordy, M.; and Traon, Y. L. 2021. Test selection for deep learning systems. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 30(2): 1–22.

Manakul, P.; Liusie, A.; and Gales, M. J. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Marijan, D.; and Gotlieb, A. 2020. Software testing for machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 13576–13582.

Marivate, V.; and Sefara, T. 2020. Improving short text classification through global augmentation methods. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, 385–399. Springer.

Markov, T.; Zhang, C.; Agarwal, S.; Nekoul, F. E.; Lee, T.; Adler, S.; Jiang, A.; and Weng, L. 2023. A holistic approach to undesired content detection in the real world. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 15009–15018.

Min, M. J.; Ding, Y.; Buratti, L.; Pujar, S.; Kaiser, G.; Jana, S.; and Ray, B. 2024. Beyond Accuracy: Evaluating Self-Consistency of Code Large Language Models with IdentityChain. In *The Twelfth International Conference on Learning Representations*.

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

Pei, K.; Cao, Y.; Yang, J.; and Jana, S. 2017. Deepxplore: Automated whitebox testing of deep learning systems. In *proceedings of the 26th Symposium on Operating Systems Principles*, 1–18.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140): 1–67.

Ren, A. Z.; Dixit, A.; Bodrova, A.; Singh, S.; Tu, S.; Brown, N.; Xu, P.; Takayama, L.; Xia, F.; Varley, J.; et al. 2023. Robots that ask for help: Uncertainty alignment for large language model planners. *arXiv preprint arXiv:2307.01928*.

Riccio, V.; and Tonella, P. 2020. Model-based exploration of the frontier of behaviours for deep learning system testing. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 876–888.

Sanh, V.; Debut, L.; Chaumond, J.; and Wolf, T. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

ShareGPT. 2024. ShareGPT.

Sullivan, M.; and Chillarege, R. 1991. Software defects and their impact on system availability-a study of field failures in operating systems. In *Digest of Papers. Fault-Tolerant Computing: The Twenty-First International Symposium*, 2–3. IEEE Computer Society.

Sun, L.; Huang, Y.; Wang, H.; Wu, S.; Zhang, Q.; Gao, C.; Huang, Y.; Lyu, W.; Zhang, Y.; Li, X.; et al. 2024. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*.

Sun, Y.; Huang, X.; Kroening, D.; Sharp, J.; Hill, M.; and Ashmore, R. 2018. Testing deep neural networks. *arXiv preprint arXiv:1803.04792*.

Tan, X.; Shi, S.; Qiu, X.; Qu, C.; Qi, Z.; Xu, Y.; and Qi, Y. 2023. Self-criticism: Aligning large language models with their understanding of helpfulness, honesty, and harmlessness. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 650–662.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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

Vaithilingam, P.; Zhang, T.; and Glassman, E. L. 2022. Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In *Chi conference on human factors in computing systems extended abstracts*, 1–7.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, B.; Chen, W.; Pei, H.; Xie, C.; Kang, M.; Zhang, C.; Xu, C.; Xiong, Z.; Dutta, R.; Schaeffer, R.; Truong, S. T.; Arora, S.; Mazeika, M.; Hendrycks, D.; Lin, Z.; Cheng, Y.; Koyejo, S.; Song, D.; and Li, B. 2023. DecodingTrust: A Comprehensive Assessment of Trustworthiness in GPT Models. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wenzek, G.; Lachaux, M.-A.; Conneau, A.; Chaudhary, V.; Guzmán, F.; Joulin, A.; and Grave, E. 2019. CCNet: Extracting high quality monolingual datasets from web crawl data. *arXiv preprint arXiv:1911.00359*.

Xie, X.; Li, T.; Wang, J.; Ma, L.; Guo, Q.; Juefei-Xu, F.; and Liu, Y. 2022. Npc: N euron p ath c overage via characterizing decision logic of deep neural networks. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 31(3): 1–27.

Xie, X.; Ma, L.; Juefei-Xu, F.; Xue, M.; Chen, H.; Liu, Y.; Zhao, J.; Li, B.; Yin, J.; and See, S. 2019. Deephunter: a coverage-guided fuzz testing framework for deep neural networks. In *Proceedings of the 28th ACM SIGSOFT international symposium on software testing and analysis*, 146–157.

Xie, X.; Song, J.; Zhou, Z.; Huang, Y.; Song, D.; and Ma, L. 2024. Online Safety Analysis for LLMs: a Benchmark, an Assessment, and a Path Forward. *arXiv preprint arXiv:2404.08517*.

Xiong, M.; Hu, Z.; Lu, X.; LI, Y.; Fu, J.; He, J.; and Hooi, B. 2024. Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*.

Yadkori, Y. A.; Kuzborskij, I.; György, A.; and Szepesvári, C. 2024. To Believe or Not to Believe Your LLM. *arXiv preprint arXiv:2406.02543*.

Yuan, X.; Li, J.; Wang, D.; Chen, Y.; Mao, X.; Huang, L.; Xue, H.; Wang, W.; Ren, K.; and Wang, J. 2024. S-Eval: Automatic and Adaptive Test Generation for Benchmarking Safety Evaluation of Large Language Models. *arXiv preprint arXiv:2405.14191*.

Zhang, J. M.; Harman, M.; Ma, L.; and Liu, Y. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*, 48(1): 1–36.

Zhang, X.; Du, X.; Xie, X.; Ma, L.; Liu, Y.; and Sun, M. 2021. Decision-guided weighted automata extraction from recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 11699–11707.

Zhang, Y.; Li, Y.; Cui, L.; Cai, D.; Liu, L.; Fu, T.; Huang, X.; Zhao, E.; Zhang, Y.; Chen, Y.; et al. 2023a. Siren's song in the AI ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.

Zhang, Z.; Lei, L.; Wu, L.; Sun, R.; Huang, Y.; Long, C.; Liu, X.; Lei, X.; Tang, J.; and Huang, M. 2023b. Safetybench: Evaluating the safety of large language models with multiple choice questions. *arXiv preprint arXiv:2309.07045*.

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

Zhou, Z.; Song, J.; Yao, K.; Shu, Z.; and Ma, L. 2023. ISR-LLM: Iterative Self-Refined Large Language Model for Long-Horizon Sequential Task Planning. *arXiv preprint arXiv:2308.13724*.

Zou, A.; Wang, Z.; Kolter, J. Z.; and Fredrikson, M. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

Reproducibility Checklist

This paper

- Includes a conceptual outline and/or pseudocode description of AI methods introduced [Yes]
- Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results [Yes]
- Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper [Yes]

Does this paper make theoretical contributions? [No]

Does this paper rely on one or more datasets? [Yes]

- A motivation is given for why the experiments are conducted on the selected datasets [Yes]
- All novel datasets introduced in this paper are included in a data appendix. [Yes]
- All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. [Yes]
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations. [Yes]
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. [Yes]
- All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing. [NA]

Does this paper include computational experiments? [Yes]

- Any code required for pre-processing data is included in the appendix. [Yes]
- All source code required for conducting and analyzing the experiments is included in a code appendix. [Yes]
- All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. [Yes]
- All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from [Yes]
- If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. [Yes]
- This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. [Yes]
- This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. [Yes]
- This paper states the number of algorithm runs used to compute each reported result. [Yes]
- Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information. [Yes]
- The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). [Yes]
- This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. [Yes]
- This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. [Yes]

A Supplementary Material

A.1 Experimental Setting

RQ1. In this RQ, we want to examine whether the testing criteria can reflect and approximate the functional features of the LLM. We follow the assumption that a label category reflects a functional feature of the model (Huang et al. 2021). Specifically, we randomly generate 1,000 test case based on a seed queue, which contains 100 initial seed with one/two/three/all label categories, and check the coverage changes.

RQ2. Testing prioritization for large language models is motivated by the need to efficiently allocate resources, and focus on the more critical test cases. *Budget:* For the budget of test prioritization process, we set the budgets as 5%, 10%, and 15% of the dataset size. *Judgement:* The LLM defect is judged by a judgement model (Yuan et al. 2024; Li et al. 2024; Wang et al. 2023). For truthfulness, we use GPT-judge (Lin, Hilton, and Evans 2021) as the judgement model, which is a fine-tuned GPT model and output the truthful probability of the answer. We take 0.5 as the threshold to estimate whether an answer is truthful/untruthful. For toxicity, we leverage a fine-tuned version of the DistilBERT (Sanh et al. 2019) model to classify toxic texts. *Baseline:* We use four baseline as the comparison methods: Random, DeepGini (Feng et al. 2020), MaxP (Ma et al. 2021), and Margin (Jiang et al. 2018). Random selection randomly pick a subset of test cases for testing. DeepGini computes the gini score as the uncertainty degree of the test cases, and the case with a higher score is prioritized. MaxP prioritizes test cases based on the maximum output probability of the model, and the test case with a lower score is considered as a fault. Based on the difference between the top-1 and top-2 output probabilities, Margin quantifies the uncertainty of the test case. A smaller difference suggests that it is harder for the model to discern the test case between the two groups and that the test case ought to be regarded as errors.

RQ3. In this RQ, we would like to assess the performance of the coverage guided testing in uncovering the faults of the LLM. *Metric:* We use *Test Success Rate (TSR)* as the evaluation metric to assess the performance of test case generation methods. Test Success Rate is utilized by recent researches (Chao et al. 2023; Zou et al. 2023; Liu et al. 2023a). *Baseline:* For comparison, we compare with three baseline techniques: Random, AutoDAN-GA, and AutoDAN-HGA (Liu et al. 2023a). Random method randomly decide whether enqueue the new test case, i.e., Line 9 in Algorithm 1, and aim to present the usefulness of the guidance. AutoDAN is a hierarchical genetic algorithm to generate test cases from seed prompts. It leverage genetic algorithm (GA) and hierarchical genetic algorithm (HGA) to produce test cases. We repeat the experiment for five times and report the average result.

Experimental Models and Datasets. *Models:* We choose five open-source models for the experiment. LLaMA-7B (Touvron et al. 2023a), LLaMA2-13B (Touvron et al. 2023b), and LLaMA2-7B (Touvron et al. 2023b) are released by Meta AI, which is trained with trillion-level tokens and supports 2,048 and 4,096 tokens context length, respectively. They are fitted for various tasks, such as text generation, machine translation, question answering, and text summarization. They broadly used by researchers and developers due to its easy accessibility and availability. Alpaca (Taori et al. 2023) is an instruction-tuned model fine-tuned from LLaMA-7B. The model is trained on 52K instruction-following demonstrations generated in self-instruct using GPT-3. Vicuna (Chiang et al. 2023) is fine-tuned on user-shared conversations collected from ShareGPT (ShareGPT 2024), demonstrating competitive performance compared to other open-source models like Alpaca. Compared to Alpaca, it is with improvements such as multi-turn conversations, memory optimizations, and cost reduction via spot instance.

Datasets: We choose four popular publicly-available datasets, shown in Table 3, for our evaluation. TruthfulQA (Lin, Hilton, and Evans 2021), TriviaQA (Joshi et al. 2017), and Natural Questions Open (NQ-OPEN) (Kwiatkowski et al. 2019) focus on question-answering, with the primary goal to evaluate if the model can understand the question and provide accurate and correct answers. While RealToxicityPrompt (Gehman et al. 2020) is a text continuation dataset on generating coherent and contextually relevant continuations that follow the input text. A typical fault happened to the model is that the model could output toxic content, which refers to the degree of harmfulness and offensiveness of the output.

A.2 Collected Models

For the collected models, we categorize them as *base models* and *fine-tuned models*. LLaMA-7B and LLaMA2-7B are considered to be base models since they are suitable for diverse general-purpose tasks, e.g., question answering and text translation, and can be further fine-tuned to different models, e.g., Alpaca and Vicuna, according to specific usage scenarios.

- **LLaMA-7B** (Touvron et al. 2023a) is released by Meta AI, which is trained with more than a trillion tokens and supports 2,048 tokens context length. It can be used for various tasks, such as text generation, machine translation, question answering, and text summarization. It is broadly used by researchers and developers due to its easy accessibility and availability.
- **LLaMA2-7B** (Touvron et al. 2023b) is an improved version of LLaMA-7B. It is trained on about two trillion tokens and allows 4,096 tokens context length. Equipped with grouped-query attention and Reinforcement Learning from Human Feedback (RLHF) supervised fine-tuning, the helpfulness and safety greatly increased compared with its predecessor.
- **LLaMA2-13B** (Touvron et al. 2023b) has 13 billion parameters, making it larger and generally more powerful in handling complex tasks and generating more coherent and contextually relevant text.
- **Alpaca** (Taori et al. 2023) is an instruction-tuned model fine-tuned from LLaMA-7B. The model is trained on 52K instruction-following demonstrations generated in self-instruct using GPT-3. Alpaca shows similar performance to GPT-3 but has a surprisingly small size and is easy to

Table 3: The collected evaluation datasets. (The size is measured by the number of instances)

Dataset	Task Domain	Size	Trustworthiness
TruthfulQA (Lin, Hilton, and Evans 2021)	Question Answering	817	Truthfulness
TriviaQA (Joshi et al. 2017)	Question Answering	3,610	Truthfulness
Natural Question (Kwiatkowski et al. 2019)	Question Answering	3,610	Truthfulness
RealToxicityPrompt (Gehman et al. 2020)	Text Continuation	2,400	Toxicity

reproduce.

- **Vicuna** (Chiang et al. 2023) is fine-tuned on user-shared conversations collected from ShareGPT (ShareGPT 2024), demonstrating competitive performance compared to other open-source models like Alpaca. Compared to Alpaca, it is with improvements such as multi-turn conversations, memory optimizations, and cost reduction via spot instance.

A.3 Collected Datasets

Table 3 gives an overview of the studied tasks and datasets within our benchmark. The safety requirements for each task are also defined accordingly. A more detailed description and the full definitions of these metrics are provided in the appendix.

- **Question Answering** is a classic NLP task aimed at developing systems that can understand and respond to questions. The primary goal is to evaluate if the model can understand the question and provide accurate and correct answers. We use the datasets TruthfulQA (Lin, Hilton, and Evans 2021), TriviaQA (Joshi et al. 2017), and Natural Questions Open (NQ-OPEN) (Kwiatkowski et al. 2019) in our benchmark. Safety is considered to be the truthfulness (Ji et al. 2023a), i.e., the accuracy, honesty, and faithfulness of the answers. We utilize GPT-judge (Lin, Hilton, and Evans 2021) to determine whether the output is truthful.
- **Text Continuation** is to generate coherent and contextually relevant continuations that follow the input text. This task is important for applications like auto-complete suggestions and content generation. In this work, we include an adapted version (Wang et al. 2023) of the RealToxicityPrompt (Gehman et al. 2020) dataset, which contains 1,200 toxic prompts and 1,200 non-toxic prompts. Toxicity is the safety concern in this task, which refers to the degree of harmfulness and offensiveness of the output.

A.4 Testing Coverage of existing test set

The parameter setting is shown in Table 4. The testing coverage of existing test set is shown in Table 5

A.5 RQ2: How effective are the criteria in conducting test prioritization?

Table 6 present the results of the test prioritization for LLM. We report the mean absolute error (mae) and mean squared error (mse) of the prioritization prediction. We perform prioritization under the budget of 5%, 10%, and 15% respectively and show the average result.

Our proposed metrics have a better performance than the baseline method for test prioritization. For example, in LLaMA-7B with RealToxicityPrompt, the mae and mse of the attention-wise metrics KKAC is the lowest (0.78 and 0.78 respectively); in LLaMA-7B with TriviaQA, the average mae and mse of the neuron-wise metrics ITNC is 0.49 and 0.29, respectively; in Vicuna with NQ-OPEN, the average mae and mse of FHNC is 0.48 and 0.27.

Attention-wise criteria provide an effective prioritization performance. KMAC, KKAC, and KSAC method stands out as an effective metrics across multiple datasets and models, e.g., they archieve the lowest error rate for seven times in terms of the dataset-model pair. They consistently provides lower MAE/MSE, suggesting that leveraging attention mechanisms for test prioritization is beneficial. Neuron-wise criteria, also frequently achieves the best performance. FHNC delivers the best performance for Vicuna with TruthfulQA, TriviaQA, and NQ-OPEN (0.47/0.26, 0.44/0.24, 0.48/0.27). This indicates that focusing on neuron activation can enhance test prioritization.

Different models show varying strengths with different criteria. For instance, KSAC and FHNC are particularly good with Alpaca and Vicuna, while LLaMA-7B benefits from KMEC. Certain criteria, e.g., KSAC, consistently perform well across diverse types of datasets and models, indicating their robustness and adaptability. Additionally, the criteria typically offer consistent performance under different budgets.

In practice, we suggest the user first using the attention-wise and neuron-wise criteria for the prioritization, and trying other criteria for a more sophisticated performance. For instance, use KSAC or FHNC during early development and switch to try other attention-wise and neuron-wise metrics during the model deployment.

Answer to RQ2: The experimental result demonstrates that the proposed metrics is effective in providing test prioritization guidance and they typically have a better performance than the existing baseline methods. In practice, we recommend the user should first utilize the attention-wise and neuron-wise metrics and try other criteria for a better performance.

Table 4: RQ1 – The hyper-parameter configurations.

LLM Coverage Criteria	Parameters
KMAC	LB=0, UB=1, k=100
KVAC	LB=0, UB=1, k=100
KKAC	LB=1, UB=20, k=100
KSAC	LB=-10, UB=10, k=100
IHNC	h=0.95
ITNC	k=50
FHNC	h=0.95, r=10
FTNC	k=50, r=10
KMEC	LB=0, UB=1, k=100
KMLC	LB=0, UB=15, k=100

Table 5: RQ1 - Experimental Results for the coverage values of the test sets on the LLM. (KMAC: k-multisection Mean Attention Coverage; KVAC: k-multisection Variance Attention Coverage; KKAC: k-multisection Kurtosis Attention Coverage; KSAC: k-multisection Skewness Attention Coverage; IHNC: Instant Hyperactive Neuron Coverage ; ITNC: Instant Top-K Neuron Coverage; FHNC: Frequent Hyperactive Neuron Coverage; FTNC: Frequent Top-K Neuron Coverage; KMEC: k-multisection Entropy Coverage; KMLC: k-multisection Likelihood Coverage)

		Attention-wise				Neuron-wise				Uncertainty-wise	
Model	Dataset	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
LLaMA-7B	TruthfulQA	0.0888	0.0749	0.1908	0.0942	0.9674	0.2153	0.703	0.0001	0.8075	0.3514
	TriviaQA	0.0704	0.0615	0.2185	0.1057	0.9768	0.2361	0.7163	0.03	0.7869	0.3801
	NQ-OPEN	0.0949	0.0815	0.1914	0.0912	0.9731	0.2348	0.703	0.02	0.7855	0.3584
	RealToxicityPrompt	0.0407	0.0321	0.1105	0.1548	0.9417	0.1811	0.6561	0.0003	0.5687	0.3890
		Attention-wise				Neuron-wise				Uncertainty-wise	
Model	Dataset	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
LLaMA2-7B	TruthfulQA	0.0843	0.0692	0.1925	0.0913	0.9756	0.22	0.7458	0.0003	0.5527	0.5699
	TriviaQA	0.0709	0.0615	0.2098	0.1004	0.9879	0.2358	0.7787	0.0001	0.6731	0.633
	NQ-OPEN	0.096	0.0811	0.1743	0.0869	0.9816	0.2302	0.7388	0.0002	0.5101	0.5403
	RealToxicityPrompt	0.0613	0.0504	0.1269	0.1194	0.9786	0.2541	0.795	0.0001	0.9069	0.6295
		Attention-wise				Neuron-wise				Uncertainty-wise	
Model	Dataset	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
LLaMA2-13B	TruthfulQA	0.0784	0.0822	0.0613	0.0416	0.9945	0.3138	0.8426	0.0002	0.8112	0.3723
	TriviaQA	0.0668	0.0656	0.0534	0.0439	0.9933	0.3171	0.8272	0.0002	0.771	0.3776
	NQ-OPEN	0.0831	0.0894	0.0644	0.0388	0.989	0.2902	0.8	0.0002	0.8051	0.3545
	RealToxicityPrompt	0.0435	0.0491	0.0176	0.051	0.9672	0.273	0.7431	0.0004	0.7591	0.3415
		Attention-wise				Neuron-wise				Uncertainty-wise	
Model	Dataset	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
Alpaca	TruthfulQA	0.0868	0.0649	0.3728	0.128	0.9921	0.3475	0.8169	0.01	0.8117	0.3763
	TriviaQA	0.0709	0.0611	0.1489	0.0944	0.9079	0.1721	0.2425	0.0061	0.3978	0.139
	NQ-OPEN	0.0963	0.0814	0.1472	0.0837	0.9489	0.2095	0.3445	0.0012	0.5033	0.1828
	RealToxicityPrompt	0.0404	0.0331	0.0704	0.1462	0.9436	0.2698	0.6536	0.0021	0.5599	0.2991
		Attention-wise				Neuron-wise				Uncertainty-wise	
Model	Dataset	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
Vicuna	TruthfulQA	0.091	0.0642	0.2893	0.1319	0.9863	0.3626	0.7977	0.0006	0.7601	0.3802
	TriviaQA	0.0709	0.0555	0.3343	0.1526	0.9918	0.3755	0.8219	0.0006	0.7262	0.394
	NQ-OPEN	0.0963	0.064	0.3073	0.1318	0.9828	0.3458	0.7652	0.0008	0.7331	0.3757
	RealToxicityPrompt	0.0423	0.0304	0.1541	0.2121	0.9673	0.3081	0.7637	0.0007	0.6132	0.3631

A.6 RQ3: Are the proposed criteria effective in guiding the testing procedure to find LLM defects?

Table 7 shows the test success rate of the baseline methods and the proposed coverage guided testing.

The performance of coverage guided testing is better than the baseline methods across all models and datasets. The average TSR of the proposed criteria on LLaMA-7B, LLaMA2-7B, Alpaca, and Vicuna over all datasets are 50, 76, 50, and 49, respectively, while the one of baseline methods are 42, 62, 37, and 43, respectively. IHNC generally shows high scores across most models and datasets, i.e., it achieve the best of 11 trial over 16 model-dataset pair. For example, it gets TSR of 90 for TriviaQA on LLaMA2-7B and 70 for RealToxicityPrompt on Alpaca. This demonstrate the effectiveness of utilizing the proposed criteria in coverage guided testing.

The AutoDAN-HGA technique generally shows higher performance than the AutoDAN-GA and Random baselines across most models and datasets. For example, in the LLaMA2-7B model, the TSR of AutoDAN-HGA is 70 for NQ-OPEN, compared to 66 for AutoDAN-GA and 75 for Random; in LLaMA-7B with RealToxicityPrompt, the TSR of AutoDAN-HGA is 53, which is higher AutoDAN-GA and Random, with 48 and 53 respectively.

Answer to RQ3: Our coverage guided testing is effective in guiding the testing procedure to find faults for diverse models and datasets. Criteria like IHNC generally yield higher test success rates, demonstrate its usefulness in practice.

Table 6: RQ2 - Experimental Results for the performance of test prioritization for LLM. The results is shown as "Mean Absolute Error/Mean Squared Error", where a lower value indicate a better result. The best results are highlighted in gray. (TruQA: TruthfulQA; TriQA: TriviaQA; NQ: NQ-OPEN; RTP: RealToxicityPrompt;)

			Baseline				Attention-wise				Neuron-wise				Uncertainty-wise	
Dataset	Model	Budget	Random	DeepGini	MaxP	Margin	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
LLaMA-7B	TruQA	5%	0.58/0.36	0.45/0.24	0.59/0.39	0.55/0.35	0.51/0.3	0.5/0.29	0.57/0.37	0.55/0.33	0.42/0.22	0.42/0.22	0.43/0.22	0.5/0.31	0.46/0.21	0.45/0.22
		10%	0.51/0.31	0.51/0.31	0.55/0.35	0.55/0.34	0.51/0.3	0.49/0.28	0.55/0.34	0.53/0.33	0.44/0.24	0.46/0.26	0.46/0.26	0.54/0.33	0.46/0.20	0.48/0.23
		15%	0.52/0.31	0.51/0.31	0.56/0.35	0.52/0.31	0.52/0.31	0.5/0.29	0.55/0.35	0.54/0.33	0.48/0.27	0.47/0.26	0.48/0.27	0.54/0.34	0.44/0.20	0.46/0.21
		Avg.	0.54/0.33	0.49/0.29	0.57/0.36	0.54/0.34	0.51/0.3	0.5/0.29	0.56/0.35	0.54/0.33	0.45/0.24	0.45/0.25	0.46/0.25	0.53/0.33	0.45/0.21	0.46/0.22
	TriQA	5%	0.56/0.35	0.49/0.29	0.6/0.4	0.57/0.36	0.56/0.36	0.55/0.33	0.6/0.39	0.6/0.4	0.45/0.25	0.5/0.29	0.49/0.29	0.59/0.39	0.45/0.2	0.45/0.2
		10%	0.59/0.38	0.52/0.31	0.59/0.39	0.56/0.36	0.56/0.35	0.57/0.36	0.58/0.37	0.59/0.39	0.47/0.26	0.48/0.27	0.49/0.29	0.58/0.37	0.4/0.17	0.41/0.17
		15%	0.56/0.36	0.53/0.32	0.59/0.38	0.57/0.36	0.56/0.35	0.57/0.36	0.58/0.37	0.6/0.39	0.49/0.28	0.49/0.28	0.51/0.31	0.58/0.38	0.44/0.19	0.42/0.16
		Avg.	0.57/0.36	0.51/0.31	0.59/0.39	0.57/0.36	0.56/0.35	0.56/0.35	0.59/0.38	0.6/0.39	0.47/0.27	0.49/0.28	0.5/0.29	0.58/0.38	0.42/0.18	0.43/0.18
	NQ	5%	0.56/0.35	0.53/0.32	0.62/0.42	0.57/0.36	0.54/0.33	0.54/0.33	0.59/0.38	0.51/0.33	0.37/0.2	0.41/0.25	0.39/0.23	0.61/0.41	0.45/0.2	0.45/0.2
		10%	0.57/0.36	0.52/0.31	0.61/0.4	0.57/0.37	0.54/0.33	0.56/0.35	0.57/0.37	0.53/0.34	0.43/0.25	0.43/0.25	0.44/0.26	0.6/0.39	0.45/0.21	0.45/0.21
		15%	0.58/0.38	0.52/0.31	0.62/0.41	0.58/0.37	0.54/0.32	0.55/0.35	0.57/0.36	0.54/0.34	0.45/0.26	0.45/0.26	0.46/0.27	0.6/0.39	0.45/0.2	0.46/0.2
		Avg.	0.57/0.36	0.52/0.31	0.61/0.41	0.57/0.37	0.54/0.33	0.55/0.34	0.57/0.37	0.53/0.34	0.41/0.24	0.43/0.26	0.43/0.25	0.6/0.4	0.45/0.21	0.45/0.21
	RTP	5%	0.89/0.89	0.94/0.94	0.89/0.89	0.97/0.97	0.91/0.91	0.95/0.95	0.78/0.78	0.82/0.82	0.97/0.97	0.97/0.97	0.96/0.96	0.9/0.9	0.97/0.97	0.83/0.83
		10%	0.84/0.84	0.92/0.92	0.94/0.94	0.99/0.99	0.9/0.9	0.92/0.92	0.77/0.77	0.81/0.81	0.95/0.95	0.95/0.95	0.93/0.93	0.85/0.85	0.96/0.96	0.81/0.81
		15%	0.82/0.82	0.91/0.91	0.96/0.96	0.98/0.98	0.89/0.89	0.91/0.91	0.78/0.78	0.81/0.81	0.94/0.94	0.94/0.94	0.91/0.91	0.82/0.82	0.94/0.94	0.81/0.81
		Avg.	0.85/0.85	0.93/0.93	0.93/0.93	0.98/0.98	0.9/0.9	0.93/0.93	0.78/0.78	0.82/0.82	0.95/0.95	0.95/0.95	0.93/0.93	0.86/0.86	0.95/0.95	0.82/0.82

			Baseline				Attention-wise				Neuron-wise				Uncertainty-wise	
Dataset	Model	Budget	Random	DeepGini	MaxP	Margin	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
LLaMA2-7B	TruQA	5%	0.51/0.29	0.56/0.35	0.51/0.29	0.49/0.27	0.53/0.31	0.49/0.27	0.57/0.37	0.62/0.42	0.43/0.22	0.45/0.23	0.45/0.24	0.5/0.28	0.42/0.2	0.4/0.2
		10%	0.57/0.35	0.54/0.33	0.51/0.29	0.51/0.3	0.51/0.29	0.51/0.3	0.56/0.34	0.57/0.36	0.47/0.25	0.48/0.26	0.49/0.27	0.5/0.28	0.45/0.2	0.42/0.2
		15%	0.53/0.31	0.54/0.32	0.52/0.3	0.51/0.3	0.52/0.3	0.5/0.29	0.55/0.33	0.56/0.34	0.49/0.27	0.5/0.28	0.51/0.29	0.51/0.29	0.48/0.2	0.42/0.2
		Avg.	0.54/0.32	0.55/0.33	0.51/0.3	0.5/0.29	0.52/0.3	0.5/0.28	0.56/0.35	0.58/0.37	0.46/0.25	0.48/0.26	0.48/0.27	0.5/0.28	0.45/0.2	0.41/0.2
	TriQA	5%	0.6/0.39	0.58/0.38	0.68/0.49	0.47/0.28	0.59/0.39	0.57/0.36	0.59/0.38	0.61/0.42	0.48/0.28	0.46/0.26	0.5/0.3	0.63/0.42	0.71/0.5	0.72/0.5
		10%	0.61/0.41	0.59/0.38	0.65/0.46	0.51/0.31	0.6/0.4	0.59/0.39	0.59/0.39	0.59/0.39	0.51/0.3	0.5/0.29	0.52/0.32	0.63/0.43	0.72/0.5	0.72/0.5
		15%	0.61/0.4	0.6/0.39	0.65/0.45	0.54/0.33	0.6/0.4	0.6/0.4	0.6/0.39	0.59/0.39	0.52/0.32	0.52/0.31	0.54/0.33	0.63/0.43	0.71/0.51	0.71/0.51
		Avg.	0.61/0.4	0.59/0.38	0.66/0.47	0.51/0.31	0.6/0.4	0.58/0.38	0.59/0.39	0.6/0.4	0.5/0.3	0.49/0.29	0.52/0.31	0.63/0.43	0.71/0.49	0.72/0.49
	NQ	5%	0.61/0.4	0.53/0.3	0.65/0.45	0.57/0.36	0.62/0.41	0.6/0.39	0.62/0.41	0.52/0.31	0.58/0.38	0.64/0.45	0.6/0.41	0.63/0.42	0.7/0.49	0.68/0.42
		10%	0.62/0.4	0.55/0.32	0.64/0.45	0.6/0.39	0.62/0.41	0.6/0.39	0.63/0.42	0.52/0.31	0.6/0.4	0.63/0.44	0.61/0.41	0.64/0.43	0.7/0.49	0.7/0.49
		15%	0.63/0.42	0.56/0.34	0.63/0.45	0.61/0.4	0.62/0.41	0.61/0.4	0.62/0.41	0.52/0.32	0.59/0.39	0.62/0.42	0.61/0.41	0.63/0.43	0.71/0.48	0.71/0.48
		Avg.	0.62/0.41	0.54/0.32	0.65/0.45	0.59/0.39	0.62/0.41	0.6/0.39	0.62/0.41	0.52/0.31	0.59/0.39	0.63/0.44	0.61/0.41	0.63/0.43	0.7/0.46	0.7/0.46
	RTP	5%	0.92/0.92	0.89/0.89	0.9/0.9	0.9/0.9	0.86/0.86	0.87/0.87	0.89/0.89	0.87/0.87	0.92/0.92	0.91/0.91	0.88/0.88	0.71/0.71	0.91/0.91	0.9/0.9
		10%	0.9/0.9	0.9/0.9	0.88/0.88	0.9/0.9	0.87/0.87	0.86/0.86	0.88/0.88	0.89/0.89	0.9/0.9	0.92/0.92	0.92/0.92	0.77/0.77	0.92/0.92	0.92/0.92
		15%	0.86/0.86	0.89/0.89	0.86/0.86	0.89/0.89	0.85/0.85	0.88/0.88	0.86/0.86	0.86/0.86	0.9/0.9	0.91/0.91	0.94/0.94	0.79/0.79	0.89/0.89	0.89/0.89
		Avg.	0.89/0.89	0.89/0.89	0.88/0.88	0.9/0.9	0.86/0.86	0.87/0.87	0.87/0.87	0.87/0.87	0.91/0.91	0.91/0.91	0.91/0.91	0.76/0.76	0.9/0.9	0.91/0.91

			Baseline				Attention-wise				Neuron-wise				Uncertainty-wise	
Dataset	Model	Budget	Random	DeepGini	MaxP	Margin	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
Alpaca	TruQA	5%	0.62/0.4	0.58/0.33	0.58/0.36	0.59/0.38	0.58/0.38	0.59/0.38	0.58/0.36	0.55/0.34	0.59/0.37	0.57/0.36	0.57/0.35	0.55/0.36	0.75/0.56	0.77/0.58
		10%	0.59/0.39	0.57/0.36	0.54/0.33	0.58/0.38	0.61/0.42	0.6/0.39	0.6/0.39	0.57/0.35	0.57/0.35	0.59/0.38	0.58/0.36	0.58/0.38	0.75/0.56	0.75/0.56
		15%	0.58/0.37	0.58/0.37	0.57/0.34	0.61/0.4	0.61/0.41	0.59/0.38	0.61/0.4	0.58/0.37	0.58/0.36	0.59/0.37	0.58/0.36	0.6/0.39	0.74/0.54	0.73/0.52
		Avg.	0.6/0.38	0.58/0.35	0.57/0.35	0.59/0.39	0.6/0.4	0.59/0.38	0.6/0.38	0.56/0.35	0.58/0.36	0.58/0.37	0.58/0.36	0.58/0.38	0.75/0.55	0.75/0.55
	TriQA	5%	0.51/0.32	0.5/0.31	0.5/0.31	0.49/0.3	0.54/0.36	0.55/0.36	0.43/0.23	0.47/0.27	0.52/0.31	0.55/0.34	0.55/0.34	0.54/0.36	0.89/0.78	0.53/0.36
		10%	0.5/0.31	0.52/0.33	0.51/0.32	0.51/0.31	0.53/0.35	0.57/0.39	0.43/0.24	0.40/0.21	0.51/0.3	0.53/0.32	0.53/0.33	0.55/0.37	0.87/0.75	0.53/0.35
		15%	0.5/0.31	0.54/0.35	0.51/0.32	0.5/0.31	0.53/0.35	0.56/0.37	0.46/0.26	0.45/0.25	0.49/0.28	0.51/0.3	0.52/0.32	0.55/0.37	0.85/0.75	0.53/0.35
		Avg.	0.5/0.31	0.52/0.33	0.51/0.32	0.5/0.31	0.54/0.35	0.56/0.38	0.44/0.24	0.42/0.24	0.51/0.3	0.53/0.32	0.53/0.33	0.55/0.37	0.88/0.78	0.54/0.36
	NQ	5%	0.49/0.3	0.43/0.26	0.47/0.28	0.51/0.32	0.44/0.25	0.44/0.25	0.46/0.27	0.53/0.34	0.6/0.4	0.64/0.43	0.63/0.43	0.46/0.29	0.39/0.15	0.46/0.26
		10%	0.48/0.29	0.42/0.25	0.47/0.28	0.47/0.29	0.46/0.27	0.46/0.27	0.48/0.29	0.52/0.32	0.62/0.43	0.65/0.45	0.65/0.46	0.44/0.27	0.38/0.15	0.48/0.29
		15%	0.47/0.28	0.4/0.23	0.48/0.29	0.48/0.29	0.47/0.29	0.46/0.27	0.52/0.33	0.51/0.32	0.61/0.41	0.64/0.44	0.65/0.45	0.43/0.26	0.39/0.15	0.51/0.33
		Avg.	0.48/0.29	0.42/0.25	0.47/0.28	0.49/0.3	0.46/0.27	0.45/0.26	0.49/0.3	0.52/0.33	0.61/0.41	0.64/0.44	0.64/0.45	0.45/0.27	0.39/0.15	0.49/0.3
	RTP	5%	0.88/0.88	0.99/0.99	0.98/0.98	0.97/0.97	0.91/0.91	0.9/0.9	0.86/0.86	0.89/0.89	0.92/0.92	0.94/0.94	0.92/0.92	0.97/0.97	0.92/0.91	0.92/0.92
		10%	0.88/0.88	0.98/0.98	0.98/0.98	0.98/0.98	0.89/0.89	0.91/0.91	0.85/0.85	0.86/0.86	0.93/0.93	0.92/0.92	0.92/0.92	0.97/0.97	0.92/0.92	0.92/0.91
		15%	0.87/0.87	0.94/0.94	0.98/0.98	0.98/0.98	0.89/0.89	0.92/0.92	0.86/0.86	0.87/0.87	0.92/0.92	0.92/0.92	0.9/0.9	0.93/0.93	0.9/0.91	0.9/0.9
		Avg.	0.88/0.88	0.97/0.97	0.98/0.98	0.98/0.98	0.9/0.9	0.91/0.91	0.85/0.85	0.87/0.87	0.92/0.92	0.93/0.93	0.91/0.91	0.96/0.96	0.91/0.91	0.91/0.91

			Baseline				Attention-wise				Neuron-wise				Uncertainty-wise	
Dataset	Model	Budget	Random	DeepGini	MaxP	Margin	KMAC	KVAC	KKAC	KSAC	IHNC	ITNC	FHNC	FTNC	KMEC	KMLC
Vicuna	TruQA	5%	0.61/0.41	0.59/0.41	0.51/0.31	0.51/0.3	0.53/0.33	0.51/0.3	0.58/0.37	0.53/0.32	0.48/0.28	0.55/0.37	0.43/0.22	0.58/0.41	0.64/0.41	0.62/0.41
		10%	0.53/0.33	0.6/0.42	0.54/0.33	0.56/0.35	0.55/0.34	0.52/0.31	0.6/0.4	0.58/0.38	0.49/0.29	0.53/0.33	0.48/0.28	0.59/0.4	0.63/0.41	0.65/0.42
		15%	0.59/0.38	0.58/0.39	0.54/0.34	0.55/0.34	0.56/0.35	0.5/0.29	0.58/0.38	0.58/0.37	0.52/0.32	0.55/0.35	0.49/0.29	0.58/0.39	0.63/0.41	0.63/0.41
		Avg.	0.58/0.37	0.59/0.41	0.53/0.33	0.54/0.33	0.55/0.34	0.51/0.3	0.59/0.38	0.56/0.36	0.5/0.3	0.54/0.35	0.47/0.26	0.58/0.4	0.64/0.42	0.64/0.42
	TriQA	5%	0.52/0.31	0.57/0.36	0.54/0.32	0.54/0.32	0.49/0.27	0.49/0.29	0.52/0.31	0.53/0.33	0.48/0.26	0.44/0.24	0.43/0.23	0.56/0.35	0.48/0.24	0.42/0.22
		10%	0.51/0.3	0.57/0.36	0.54/0.33	0.53/0.32	0.49/0.28	0.49/0.28	0.52/0.31	0.52/0.31	0.46/0.25	0.45/0.25	0.44/0.23	0.54/0.33	0.48/0.24	0.44/0.23
		15%	0.5/0.29	0.56/0.34	0.53/0.32	0.52/0.31	0.5/0.29	0.49/0.28	0.51/0.3	0.53/0.32	0.47/0.25	0.45/0.25	0.45/0.24	0.53/0.32	0.47/0.24	0.45/0.24
		Avg.	0.51/0.3	0.57/0.35	0.53/0.32	0.53/0.32	0.49/0.28	0.49/0.28	0.52/0.31	0.53/0.32	0.47/0.26	0.45/0.24	0.44/0.24	0.54/0.33	0.48/0.25	0.44/0.24
	NQ	5%	0.55/0.34	0.56/0.35	0.58/0.37	0.58/0.37	0.59/0.38	0.59/0.39	0.57/0.37	0.55/0.36	0.49/0.28	0.49/0.27	0.48/0.26	0.56/0.35	0.67/0.44	0.68/0.44
		10%	0.57/0.36	0.53/0.31	0.58/0.37	0.58/0.37	0.58/0.37	0.6/0.39	0.58/0.37	0.56/0.35	0.5/0.28	0.5/0.28	0.48/0.27	0.54/0.32	0.66/0.43	0.66/0.43
		15%	0.56/0.36	0.54/0.33	0.57/0.36	0.57/0.36	0.58/0.37	0.59/0.39	0.57/0.37	0.56/0.35	0.51/0.3	0.52/0.31	0.5/0.28	0.55/0.34	0.67/0.43	0.68/0.43
		Avg.	0.56/0.35	0.54/0.33	0.58/0.37	0.57/0.36	0.58/0.37	0.6/0.39	0.58/0.37	0.56/0.35	0.5/0.29	0.5/0.29	0.48/0.27	0.55/0.34	0.67/0.44	0.67/0.44
	RTP	5%	0.84/0.84	0.98/0.98	0.96/0.96	0.97/0.97	0.92/0.92	0.93/0.93	0.82/0.82	0.89/0.89	0.97/0.97	0.97/0.97	0.97/0.97	0.96/0.96	0.9/0.9	0.93/0.93
		10%	0.85/0.85	0.95/0.95	0.95/0.95	0.96/0.96	0.91/0.91	0.88/0.88	0.81/0.81	0.87/0.87	0.95/0.95	0.97/0.97	0.96/0.96	0.93/0.93	0.88/0.88	0.88/0.88
		15%	0.83/0.83	0.94/0.94	0.95/0.95	0.96/0.96	0.88/0.88	0.88/0.88	0.8/0.8	0.86/0.86	0.94/0.94	0.96/0.96	0.95/0.95	0.9/0.9	0.87/0.87	0.86/0.86
		Avg.	0.84/0.84	0.96/0.96	0.95/0.95	0.96/0.96	0.91/0.91	0.9/0.9	0.81/0.81	0.88/0.88	0.96/0.96	0.96/0.96	0.96/0.96	0.93/0.93	0.9/0.9	0.89/0.89

Table 7: RQ3 - Experimental Results for the Test Success Rate (TSR) of the coverage guided testing. The best results are highlighted in gray.

Dataset	Model	Baseline				Criteria				
		Random	AutoDAN-GA	AutoDAN-HGA	Average	KMAC	KKAC	IHNC	KMEC	Average
LLaMA-7B	TruthfulQA	20	25	33	26	29	45	46	46	42
	TriviaQA	48	55	55	53	57	56	56	59	57
	NQ-OPEN	37	40	43	40	41	39	45	40	41
	RealToxicityPrompt	48	50	53	50	57	59	59	60	59

Dataset	Model	Baseline				Criteria				
		Random	AutoDAN-GA	AutoDAN-HGA	Average	KMAC	KKAC	IHNC	KMEC	Average
LLaMA2-7B	TruthfulQA	71	56	64	64	83	79	88	81	83
	TriviaQA	79	61	65	68	83	87	90	86	87
	NQ-OPEN	75	66	70	70	80	82	86	83	83
	RealToxicityPrompt	35	45	49	43	55	46	49	45	49

Dataset	Model	Baseline				Criteria				
		Random	AutoDAN-GA	AutoDAN-HGA	Average	KMAC	KKAC	IHNC	KMEC	Average
Alpaca	TruthfulQA	16	20	30	22	18	49	45	50	41
	TriviaQA	35	37	40	37	49	51	53	51	51
	NQ-OPEN	39	41	42	41	40	41	43	39	41
	RealToxicityPrompt	61	34	45	47	64	69	70	65	67

Dataset	Model	Baseline				Criteria				
		Random	AutoDAN-GA	AutoDAN-HGA	Average	KMAC	KKAC	IHNC	KMEC	Average
Vicuna	TruthfulQA	52	57	59	56	59	63	63	60	61
	TriviaQA	31	27	35	31	40	39	42	42	41
	NQ-OPEN	35	38	40	38	38	37	41	40	39
	RealToxicityPrompt	45	48	49	47	55	60	52	54	55