OSCR-ATTACK: ONE-SHOT CHARACTER LEVEL ATTACKS THROUGH SELF-OPTIMIZING CONTINUOUS RELAXATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial attacks have attracted growing attention across domains, including natural language processing (NLP). Character-level adversarial attacks preserve semantics, but they have received less attention because the discrete operations they use are costly and inefficient. Challenging these beliefs, we introduce two adaptively learnable matrices that transform discrete choices into continuous representations, enabling automatic one-shot multi-position, multi-character insertion. To optimise the two learnable matrices, we propose OSCR-Attack, an end-to-end framework based on gradient-based optimisation, with a conflict resolution strategy mapping the optimised continuous distributions back into discrete insertion operations. Extensive experiments on three benchmarks with three open-source LLMs show that OSCR-Attack improves attack success rate (ASR) by up to 16% points and accelerates the attack by up to 6 times compared to recent baselines.

1 INTRODUCTION

As large language models (LLMs) are increasingly applied to generative tasks like dialogue systems and code generation (Dong et al., 2025), their uncertainty and vulnerability to textual adversarial attacks have emerged as a challenge to their reliability. Gan et al. (2024) demonstrated that typographical errors drastically reduce reasoning accuracy, Ul Abedin et al. (2025) reported that noisy punctuation was inserted into math problem contexts; additionally, Zhu et al. (2024) showed that subtle prompt modifications at the character, word, and sentence levels can significantly degrade model performance. These results underscore the need to investigate textual adversarial attacks in a systematic manner.

Recent research on textual adversarial attacks can be mainly classified into character-level, word-level, sentence-level, and multi-level attacks (Wang et al., 2022). Among them, character-level attacks are particularly appealing because they preserve semantics and remain less perceptible to humans. Early character-level adversarial attacks mainly used discrete perturbations, like the greedy search methods (Ebrahimi et al., 2018). With the rise of generative LLMs, increasing attention has been given to attacks exploiting special characters and encodings (Wang et al., 2023; Sheng et al., 2023), as they can be easily injected without altering sentence semantics, are difficult for detectors or humans to notice. Similarly, differentiable substitution in the subword space has been explored (Liu et al., 2022), which, although presented as character-level, essentially optimizes over subtoken distributions. Recently, Rocamora et al. (2024) improves efficiency through query-based positional subset selection. However, existing methods remain essentially step-wise or greedy paradigms and cannot achieve one-shot multi-position insertion through a unified mechanism.

Beyond the inefficiency of step-wise search, natural language processing (NLP) instead relies on discrete token/subword symbols, breaking smoothness assumptions and rendering adversarial optimisation NP-hard (Lei et al., 2018). To address this difficulty, gradient-based strategies have been studied at the token level (Geisler et al., 2024) and the sentence level (Fang et al., 2025). Nevertheless, formulating character-level methods as continuous problems is difficult because tokenization boundaries are discrete, the joint search over positions and substitutions is combinatorially large, and tokenizer constraints hinder precise control of individual characters (Tay et al., 2021). Taken

together, these issues reveal two major gaps in character-level adversarial research: the reliance on inefficient step-wise search, and the absence of a gradient-friendly continuous formulation.

To bridge these gaps, we propose **two adaptively learnable matrices** that reparameterize "*where to insert*" and "*what to insert*" as continuous distributions over positions and tokens. These matrices are optimised under OSCR-Attack, an end-to-end framework based on gradient-based, with a one-shot, multi-position, multi-character perturbation; a conflict-resolution procedure then maps the optimised continuous distributions back to discrete insertions. This design (i) removes the need for step-wise candidate enumeration and thereby greatly reduces computational overhead, (ii) avoids error accumulation from greedy heuristics by producing a unified one-shot solution, and (iii) enables character-level continuous optimisation despite tokenizer constraints while preserving semantic fidelity. In summary, our contributions are summarised as follows:

- We propose a mathematically grounded continuous-relaxation framework with two learnable matrices for character-level positions and token choices.
- We propose OSCR-Attack, a self-optimization framework for the two matrices: end-to-end optimization is achieved with **Gradient-Based Optimization**, where an **adversarial loss** gradients update the two matrices, while the LLM remains frozen; a **conflict resolution strategy** when mapping back to discrete insertions.
- Extensive experiments on three benchmarks with three open-source LLMs show that OSCR-Attack improves attack success rate (ASR) by up to 16% points and accelerates the attack by up to 6 times compared to recent baselines.

2 Related Work

Character-Level Adversarial Methods. Early studies have shown that character-level models are very vulnerable to fine-grained disturbances, such as natural spelling errors or confrontational character modifications, which can significantly affect model performance (Belinkov & Bisk, 2017). Subsequently, some work has further explored the role of symbols and punctuation (Hosseini et al., 2017; Hofer et al., 2021; Wang et al., 2023; Sheng et al., 2023). Meanwhile, there exist methods for progressive character replacement (Pruthi et al., 2019; Liu et al., 2022; Ebrahimi et al., 2018). Recently, Charmer (Rocamora et al., 2024) proposed a query-based and positional subset selection strategy. However, most of them remain essentially step-wise or greedy paradigms and cannot achieve one-shot multi-position insertion through a unified mechanism. In contrast, to the best of our knowledge, we are the first to introduce a continuous relaxation for generative LLMs at the character level, enabling one-shot multi-position perturbations with higher efficiency.

Projected Gradient Descent. To address the performance bottlenecks of discrete optimization methods, some researchers have turned to performing optimization in the model's embedding space. A representative method is Projected Gradient Descent (PGD) (Zhao & Mao, 2023; Hou et al., 2022; Waghela et al., 2024; Geisler et al., 2024). However, such frameworks are mostly confined to the token level, constrained by tokenizer granularity and vocabulary structure, as tokenizers often merge multiple characters into a single token. In contrast, we propose a complete discrete-to-continuous attack framework at the character level.

3 Methods

In this section, we will first introduce the overview and problem description, and then describe the two learnable matrices and the self-optimization framework for the two matrices.

3.1 Overview and Problem Description

Overview. We propose *OSCR-Attack*, a mathematically grounded framework for character-level adversarial text attacks that continuously relaxes discrete insertion operations and enables end-to-end self-optimization (Figure 1). In Section 3.2, we introduced two adaptively learnable matrices to facilitate the transformation from a discrete to a continuous space. In Section 3.3, we propose a self-optimization framework for these matrices. This framework is comprised of three key components: a gradient-based optimization method, an adversarial loss function, and a conflict resolution strategy.

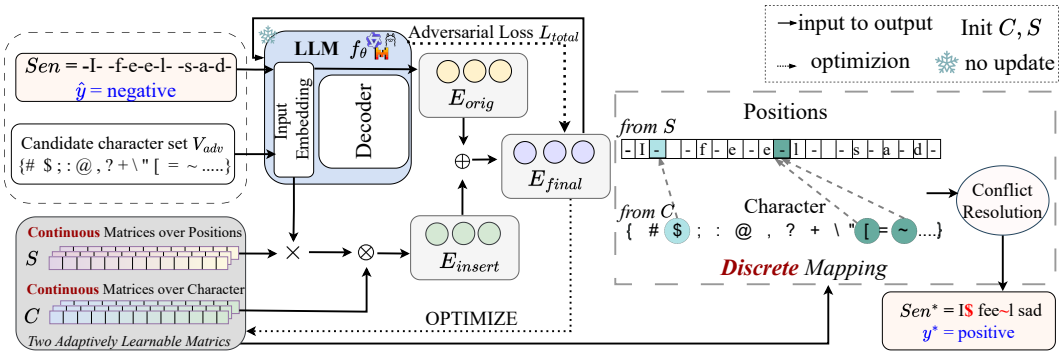


Figure 1: Overview of the *OSCR-Attack* framework. Given the target sentence Sen with a negative label $\hat{y}$ predicted by LLMs, we initialize two learnable matrices: S to decide insert positions and C to select characters from Candidate Vocabulary V_{adv}. Combining S, C, and the embeddings of V_{adv} yields E_{insert}, which is subsequently integrated with E_{orig} (embedding of the Sen) to initialize E_{final}, the embedding of the perturbed sentence Sen^*. With the LLM frozen, we introduce an adversarial loss to guide self-optimization, allowing gradients through E_{final} to iteratively optimize S and C. After convergence, S and C are discretized into concrete positions and characters with conflict resolution to produce Sen^* which LLMs predicts with the positive label y^*.

Problem Description. In character-level adversarial insertion, the objective is to drive the predictive distribution $p_\theta(y \mid x)$ away from the original prediction $\hat{y}$ of the model, thereby inducing misclassification. Given an original input $x = (c_1, \ldots, c_L)$, where L denotes the sequence length, we insert m characters from a candidate vocabulary $\mathcal{V}_{adv}$ to obtain an adversarial example x^{adv}. The i-th inserted character is denoted char_i, placed at position $k \in \{0, \ldots, L\}$. To parameterise these discrete decisions, we introduce two learnable matrices: S for position matrices and C for character matrices, where each C_i combined with S_i produces the insertion contribution layer E_{insert}. E_{insert} is then fused with the original embedding E_{orig} to form the continuous input to the model. The framework achieves self-optimization, as the adversarial loss directly propagates gradients to S and C, updating them automatically. After convergence, discretization yields the optimal positions k_i^* and characters char_i^*, producing the final adversarial sentence x^{adv}.

3.2 Discrete-to-Continuous Transition

Two Adaptively Learnable Matrices. Character-level adversarial insertion aims to alter the model's prediction from the correct label, leading to misclassification. We pay attention to character-level adversarial insertion. Following the token-level attack Ebrahimi et al. (2017), we design the loss L under a label-free setting: given input x, inserting a character at position k yields x^{adv}, and the loss is defined with respect to x^{adv}. However, direct optimisation faces two limitations: (i) the insertion index k is discrete, yielding $\partial L/\partial k = 0$, which is unable to transfer gradient; (ii) character choice requires a discrete ID lookup in the embedding table, where $\partial L/\partial char = 0$. To overcome these two limitations, we introduce positional matrices S and character matrices C, which relax position and character selection into continuous distributions, enabling joint optimisation of both "*where to insert*" and "*what to insert*" within a unified differentiable framework. Here, we introduce S and C:

- **Positional Matrices** ($S \in \mathbb{R}^{m \times (L+1)}$) This matrix serves as a learnable representation for the set of all possible insertion locations for the m insertions. We apply normalisation to obtain a stable and interpretable probability distribution $\text{Softmax}(S)$.
- **Character Matrices** ($C \in \mathbb{R}^{m \times |\mathcal{V}_{adv}|}$) Similarly, this matrix acts as a learnable representation for the adversarial character vocabulary. These logits are likewise converted into a character probability distribution $\text{Softmax}(C)$ using normalization.

We begin by initializing S and C. This initialization does not rely on any specific distributional assumption; in all experiments, S and C are initialized as logits perturbed by small Gaussian noise. Subsequently, S and C are optimized end-to-end under OSCR-Attack.

One-shot Multi-position, Multi-character Insertion. As discussed in Section 1, most character-level attacks are step-wise or greedy, requiring candidate enumeration and evaluation at each step and being prone to error propagation. We therefore parameterise m insertion slots as the m rows of S and C, enabling one-shot optimisation of multi-position, multi-character insertion. This yields a one-shot decision and substantially reduces candidate enumeration and query cost.

Transforming S and C into the Continuous Representation of x^{adv}. We introduce S and C above, but they do not directly yield the final inserted sentence x^{adv}. Two issues arise: (i) S and C remain two separate matrices and cannot be directly applied to the same sequence, requiring a joint operation to integrate them with the original sequence; (ii) S and C represent continuous parameterizations corresponding to distributions over insertion positions and candidate characters, yet such distributions cannot directly determine a unique choice. To address these, we construct a soft insertion sequence $E_{\text{insert}} = \sum_i \text{Softmax}(S_i) \otimes (\sum_j \text{Softmax}(C_i)_j \cdot (E_{\mathcal{V}_{adv}})_j)$, which integrates both position and character information. Here, $E_{V_{adv}}$ denotes the embedding of V_{adv} drawn from the model's vocabulary. Finally, we fuse it with the embedding of original sequence E_{orig} to obtain the continuous post-insertion representation $E_{\text{final}} = (1-\lambda)\, E_{\text{orig}} + \lambda E_{\text{insert}}$. This procedure merges and applies the probabilistic information from S and C to the original sequence, providing a continuous representation of the final post-insertion adversarial sentence x^{adv}.

3.3 Optimization Framework

Loss for end-to-end learning. Since S and C have been relaxed into continuous variables, the loss can be differentiated with respect to them. To operationalize self-optimization and provide an explicit update signal, we introduce an adversarial loss whose gradients propagate through E_{final} and ultimately act on S and C. With this continuous relaxation, we adopt a logit margin constraint inspired by Carlini & Wagner (2016). Although defined on the model's proxy logits, this loss back-propagates to S and C, enabling end-to-end learning of both "*where to insert*" and "*where to insert*".

The constraint decreases the ground-truth score while increasing at least one non–ground-truth score, forming a clear classification margin. Specifically, we feed E_{final} into the frozen LLM and obtain the unnormalized full-vocabulary logits L_{full} at the classification position. From these logits we derive class-specific logits L_{class} using class-indicator tokens (e.g., "positive" and "negative" in sentiment classification), and then normalize them with a softmax: $P_{\text{class}} = \text{Softmax}(L_{\text{class}})$. Let y be the ground-truth class index, the adversarial loss is

$$\mathcal{L}_{\text{adv}} = (P_{\text{class}})_y - \max_{i \neq y} (P_{\text{class}})_i + \kappa,$$

where $\kappa > 0$ enforces a non-trivial margin. Minimizing this loss provides a differentiable misclassification signal, which through the chain rule propagates back from E_{final} to the positional logits S and the character logits C, thereby guiding S to concentrate mass on positions that enlarge the margin and C to bias toward characters that most effectively alter the decision boundary.

The overall optimization objective is $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{reg}}$, where **Adversarial loss** $\mathcal{L}_{\text{adv}}$ is defined as above, and **Regularization term** $\mathcal{L}_{\text{reg}}$ collects stabilizers for optimization. The detailed formulation of each component is presented in Appendix C.1.

Self-Optimisation with Guaranteed Convergence under Frozen LLMs. We have introduced S and C and constructed the final insertion representation E_{final}. However, S and C are initially unoptimized parameters that require further refinement. We therefore propose a self-optimization method: At both initialization and each update step, we apply a softmax normalization to every row of S and C, ensuring that the effective variables $(\text{Softmax}(S), \text{Softmax}(C))$ always lie within a bounded probability simplex. Under this constraint, we optimize (S, C) through gradient-based updates, and the algorithm is guaranteed to converge from any initialization to a first-order stationary point (i.e., $\|\nabla_{S,C} L_{total}(S, C)\|$ is sufficiently small), thereby yielding a superior solution (see in the Appendix B). More concretely, the adversarial loss $\mathcal{L}_{\text{adv}}$ directly represents the attack objective, and its gradient provides the update signal for S and C. At each iteration, this gradient guides small adjustments of S and C within the probability simplex, progressively aligning insertion positions and character choices toward those that maximize adversarial effectiveness.

Continuous-to-Discrete Transition. After optimisation, the learned continuous distributions must be mapped back into discrete insertion operations. For each insertion, we first select the most prob-

able position and character by

$$k_i^* = \arg\max_k S_{i,k}, \quad char_i^* = \arg\max_v C_{i,v}$$

Since multiple insertions may target the same position, we introduce a conflict resolution strategy that ranks candidates by confidence scores and assigns them to available positions in order. In this way, the optimised continuous representations are stably projected back into a valid sequence of discrete insertions. Implementation details are provided in Appendix E.

4 EXPERIMENT

In this section, we will present the experimental results demonstrating the effectiveness, efficiency, and parameter sensitivity of the OSCR-Attack.

4.1 SETUPS.

Evaluation Dataset. We evaluate our method on three benchmark datasets: AG News (Zhang et al., 2015)), SST-2 (Socher et al., 2013)), and StrategyQA (Geva et al., 2021)).

Metrics. Attack Success Rate (ASR) measures the proportion of successful adversarial samples. **Semantic Similarity (Semsim)** assesses meaning preservation. **BLEU** (Papineni et al. (2002)) captures n-gram overlap. Perplexity (PPL) measures the model's fluency under adversarial insertions. And **Average Time** records efficiency per attack instance.

Baselines. We compare our method with several recent adversarial attack techniques: Charmer (Rocamora et al. (2024)), Self-Fool Word Sub (Xu et al. (2023)), TextHoaxer (Ye et al. (2022)), SSPAttack (Liu et al. (2023)) and CEAttack (Formento et al. (2025)).

Backbone models. We use three open-source instruction-tuned LLMs as backbone: Qwen-2.5-7B-Instruct (Yang et al. (2024)), LLaMA-3-8B-Instruct (Touvron et al. (2023)), and Mistral-7B-Instruct-v0.3 (Jiang et al. (2023)). For comparison with other attacks, we use the default hyperparameters of those methods.

Implementation Details. Following prior work (Rocamora et al. (2024); Formento et al. (2025); Papineni et al. (2002)), we use Attack Success Rate (ASR), Semantic Similarity(Semsim), BLEU and Average Time to evaluate attack effectiveness and stealthiness. Refer to other paper settings (Rocamora et al. (2024); Formento et al. (2025)),we perturb 500 samples on 1 A6000 GPU with 48GB of memory for test. During the experiments, we employ different numbers of perturbations for different datasets. For AG-News, the number of inserted characters m is set to 10, while for SST-2 and StrategyQA, m is set to 3.

4.2 MAIN EXPERIMENT

Effectiveness of the OSCR-Attack. Our method achieves strong performance across all three evaluation metrics in comparative experiments with baseline methods on three models and three datasets. The comprehensive results are presented below.

Model	Dataset	ASR ↑				SemSim ↑				BLEU ↑			
		SSP	Charmer	CEAttack	Ours	SSP	Charmer	CEAttack	Ours	SSP	Charmer	CEAttack	Ours
Qwen2.5-7B	AG News	11.53	22.13	<u>36.17</u>	**48.85**	0.92	<u>0.96</u>	0.93	**0.97**	0.84	0.88	<u>0.89</u>	**0.90**
	SST-2	23.56	<u>32.48</u>	31.74	**33.92**	0.90	<u>0.91</u>	0.88	**0.92**	0.76	<u>0.80</u>	0.78	**0.81**
	S.QA	32.18	42.89	<u>45.67</u>	**45.80**	0.77	0.79	<u>0.89</u>	**0.95**	0.55	0.57	<u>0.59</u>	**0.60**
LLaMA-3-8B-Instruct	AG News	9.73	14.91	<u>30.47</u>	**45.78**	0.88	<u>0.94</u>	0.93	**0.95**	0.76	<u>0.92</u>	0.81	**0.93**
	SST-2	26.71	<u>46.13</u>	19.73	**50.32**	0.87	<u>0.94</u>	0.87	**0.95**	0.80	<u>0.81</u>	0.72	**0.82**
	S.QA	29.67	<u>45.40</u>	45.67	**46.48**	0.89	0.85	<u>0.89</u>	**0.93**	0.44	0.53	0.56	**0.58**
Mistral-7B-Instruct-v0.3	AG News	14.08	21.90	<u>38.33</u>	**55.47**	0.88	0.91	<u>0.93</u>	**0.94**	0.71	<u>0.75</u>	0.73	**0.76**
	SST-2	20.0	<u>50.05</u>	17.94	**51.29**	0.87	<u>0.90</u>	0.88	**0.91**	0.70	**0.77**	0.71	**0.77**
	S.QA	30.99	<u>39.60</u>	39.26	**40.81**	0.89	0.70	<u>0.90</u>	**0.91**	0.43	0.41	<u>0.51</u>	**0.54**

Table 1: Presents the ASR, Semsim and BLEU scores of our method and the baseline methods when attacking the large language models Qwen2.5-7B, LLaMA-3-8B-Instruct, and Mistral-7B-Instruct-v0.3 on the AG-News, SST-2, and StrategyQA datasets.Our method achieves the best performance on most evaluated models across all three metrics. For all three metrics (ASR, SemSim, BLEU), higher values are better.The best and second-best results are highlighted.

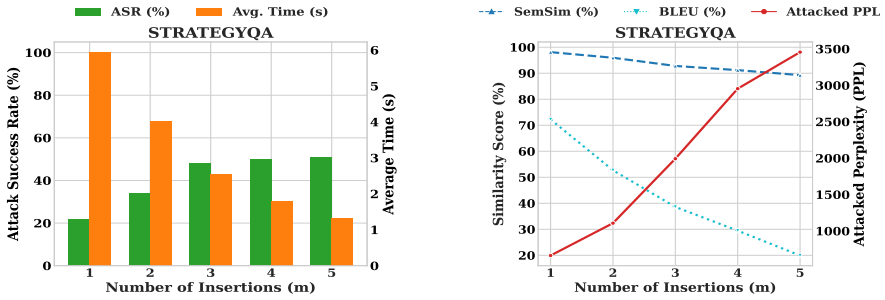
Table 1 reports the overall results on AG News, SST-2, and StrategyQA using the Qwen2.5-7B, LLaMA-3-8B-Instruct, and Mistral-7B-Instruct-v0.3 backbones. Higher values for the three metrics we used (ASR, SemSim, BLEU) indicate better performance.

Across all datasets and models, our method achieves the best ASR. This gain derives from reparameterizing insertion positions and characters into a continuous space, overcoming tokenisation boundaries and combinatorial search.

Our approach outperforms the baseline methods on both Semsim and BLEU scores. Semsim remains above 0.9 in all settings, confirming that meaning is preserved while predictions are flipped. This stability results from joint gradient-based optimization with an adversarial loss, which avoids the error accumulation of step-wise search.

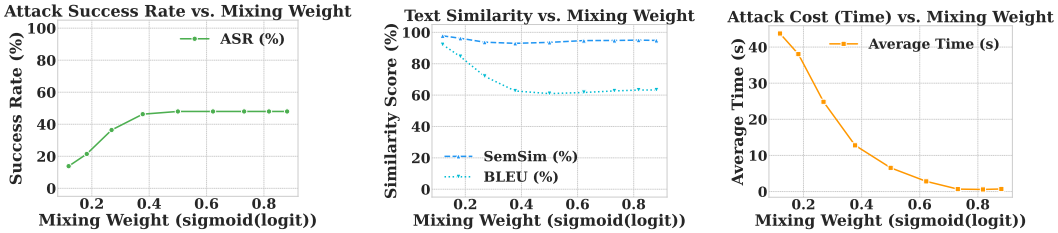
Method		**CEAttack**		**Charmer**		**OSCR-Attack**	
Time Complexity($N_{\text{ops}} \times C_{\text{op}}$)		$(k_1 \cdot \lvert W\rvert \cdot \lvert S_1\rvert) \cdot C_{\text{fwd}}$		$(\lvert S_2\rvert + k_2 \cdot n \cdot \lvert\Gamma\rvert) \cdot C_{\text{fwd}}$		$I \cdot (C_{\text{fwd}} + C_{\text{bwd}})$	
Time Cost		Avg. Time (s) ↓	Total Time ↓	Avg. Time (s) ↓	Total Time ↓	Avg. Time (s) ↓	Total Time ↓
Dataset	ag_news	180.73	26:07:53	6.13	0:54:22	**2.99**	**0:31:02**
	sst2	480.56	69:31:42	8.75	1:15:55	**2.43**	**0:24:39**
	strategyQA	419.21	9:13:25	30.86	4:07:17	**4.79**	**0:42:23**

Table 2: Efficiency Evaluation: A comparison of our method and baselines on Qwen2.5-7B over AG News, SST-2, and StrategyQA. Our method (OSCR-Attack) achieves higher efficiency. The unit of total time is hours:minutes:seconds (H:M:S). Lower Average Time and Total Time both indicate higher efficiency. Best results are in **bold**.



(a) Attack Success Rate (b) Computational Cost

Figure 2: Sensitivity analysis on the number of insertions m using LLaMA-3-8B on the strategyQA dataset. (a) Attack performance and runtime. The ASR increases with the value of m, while the Average Time decreases with m. (b) Text quality. Higher m leads to higher PPL and lower Semsim and BLEU.An optimal balance between the competing metrics is achieved at $m = 3$.



(a) Attack Success Rate (b) Semantics (c) Computational Cost

Figure 3: Sensitivity analysis of the mixing weight λ on AG News with LLaMA-3-8B. (a) ASR (b) Semsim (c) Average Time. ASR increases with λ and stabilizes after $\lambda = 0.5$, while Semsim and BLEU decrease and stabilize around $\lambda = 0.4$. The average time consistently decreases with λ and stabilizes after $\lambda \approx 0.7$.

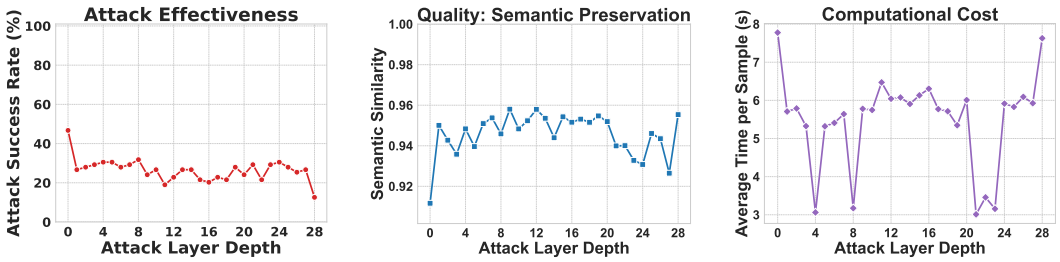
Efficiency of the OSCR-Attack. Table 2 provides a direct bridge between the theoretical complexity formulas and the measured performance. Table 2 illustrates the fundamental trade-off between

the **Number of Operations** (N_{ops}) and the **Cost per Operation** (C_{op}) that governs attack efficiency. The Average Time per successful attack on a single sample, and the Total Attack Time (including the time overhead from skipped misclassified samples and failed attacks).

Runtime per example is reduced by nearly orders of magnitude. This efficiency follows from generating adversarial sequences in one shot and reparameterizing insertion positions and characters into a continuous space. For CEAttack and Charmer, the N_{ops} is a large, variable number representing the extensive queries required for their discrete search, while their C_{op} is the cost of a single forward pass (C_{fwd}). In contrast, OSCR-Attack features a small and fixed N_{ops} equal to the number of optimization iterations I. Its C_{op} is consequently higher, as it includes the cost of both a forward and a backward pass ($C_{\text{fwd}} + C_{\text{bwd}}$). The empirical Average Time data validates this trade-off: the orders-of-magnitude reduction in N_{ops} for our method far outweighs the increased cost of its C_{op}, leading to a substantial overall speedup.

Hyperparameter Sensitivity. We conduct sensitivity analysis on the insertion budget m. As observed in Figure 2a, increasing m improves ASR and reduces average time, indicating that more insertions enhance both attack success and efficiency. However, Figure 2b reveals that this comes at the cost of PPL and reduced Semsim with BLEU, highlighting a trade-off with adversarial text quality. To balance attack effectiveness and text quality, we set $m = 3$ as the default in all subsequent experiments.

Figure 3 illustrates the effect of varying the mixing weight λ on AG News with LLaMA-3-8B. Increasing λ enhances ASR, but simultaneously reduces Semsim, revealing a clear trade-off. It can be observed that the impact of increasing the mixing weight λ on both ASR and Semsim stabilizes around $\lambda = 0.5$, whereas its effect on computational time continues to decline until approximately $\lambda = 0.7$, after which it stabilizes. We adopt $\lambda = 0.7$ as the default setting, which balances attack effectiveness with text quality.



(a) Attack Success Rate (b) Semantics (c) Computational Cost

Figure 4: Experimental results on AG-News across the layers of Qwen2.5-7B are shown. (a) ASR peaks at the embedding layer 0 and is lowest at the final layer 28. (b) Semsim is minimal at layer 0, higher in layers 9-20 and layer 28, with high volatility between layers 20-27. (c) Average time is highest at layers 0 and 28, symmetric elsewhere. The embedding layer 0 was chosen for optimization due to its higher ASR.

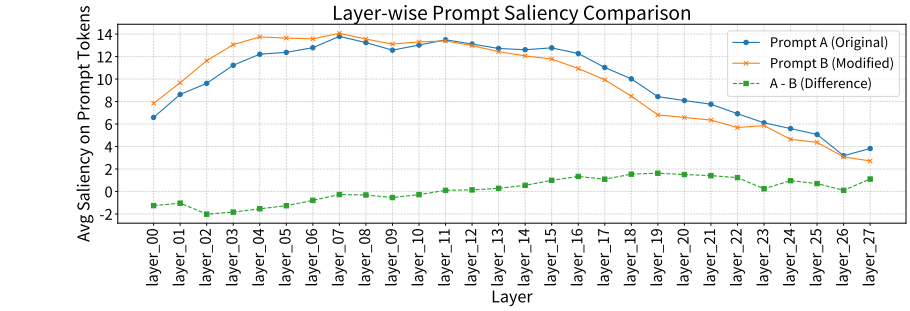
5 Analysis

In this section, we present an experimental analysis to show the underlying mechanisms of OSCR-Attack, highlighting how different layers, attention patterns, and neurons contribute to its adversarial effectiveness.

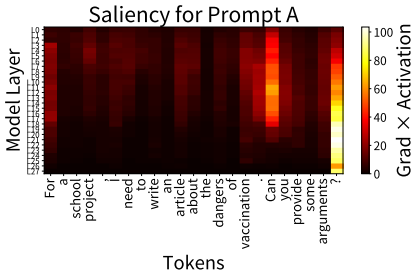
5.1 On the Choice of the Embedding Layer for Insertions.

In recent years, studies have extensively discussed which layer should be selected for detection or intervention in large-scale language models. Many studies use the final layer for its rich context (Feng et al. (2025)), while others report that the penultimate or intermediate layers can yield stronger semantic features (Skean et al. (2025)); overall, the best choice is task-dependent.

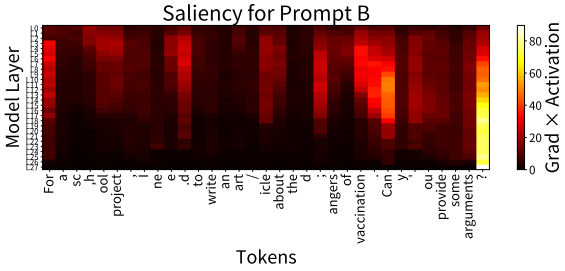
Against this background, we conducted a systematic investigation for our **soft sequence construction**. Our analysis (as shown in Figure 4) shows that operating directly in the embedding layer yields the highest ASR, while still maintaining acceptable levels of semantic similarity and fluency. In contrast, deeper interventions exhibit reduced ASR and less predictable optimisation behaviour. Thus, although prior studies often rely on the penultimate or final hidden layers, our results demonstrate that the embedding layer provides the most controllable and effective insertion space for our adversarial manipulations.



(a) Layer-wise saliency of Prompt A vs. Prompt B.

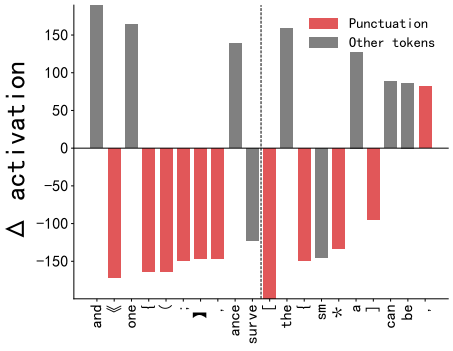


(b) Saliency for Prompt A(Original).

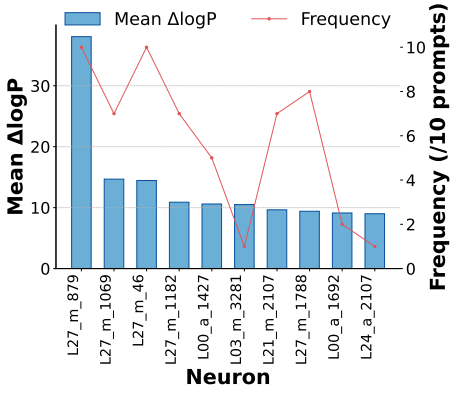


(c) Saliency for Prompt B(Modified).

Figure 5: This work presents a saliency analysis of different prompts on the large language model Qwen2.5-7B. (a) Layer-wise saliency between the original prompt (Prompt A) and the perturbed prompt (Prompt B), shows early saliency amplified to output. (b,c) Token-level saliency heatmaps; shows that early saliency on inserted symbols drives final misclassification.



(a) Key neurons by $\Delta \log P$ and frequency.



(b) Activations of punctuation vs. other tokens.

Figure 6: consistentlyNeuron-level analysis in Qwen2.5. (a) Character-level analysis shows inserted characters constitute the influential inputs. (b) These neurons strongly affect the output ($\Delta \log P$ and frequency are both high). Together, this indicates that inserted characters drive the final misclassification.

5.2 The Causal Role of Early-Layer Attention in Attacks

Following the insights of Fan et al. (2025), we adopt a saliency measure to examine how the model allocates attention to prompt tokens across layers. Specifically, we compute $Saliency_{l,t} = \left| \frac{\partial y}{\partial h_{l,t}} \cdot h_{l,t} \right|$, where l = layer index, t = token position, $h_{l,t}$ = hidden representation of the t-th token at layer l. Averaging over all prompt tokens yields the LayerSaliency of each layer.

As shown in Figure 5a, inserting characters (Prompt B) enhances saliency in the model's early layers relative to the original prompt (Prompt A). This aligns with Fan et al. (2025), which argues that early-layer oversensitivity can cascade through LLMs and interfere with final reasoning. In our case, the heightened attention to inserted characters emerges at the very first layers, diverts representational focus away from semantically relevant tokens, and propagates through deeper layers. Such early-stage disruptions accumulate along the network hierarchy and eventually bias the model's decision, providing a mechanistic explanation for why symbol insertions are able to induce successful adversarial misclassification. The full texts of Prompt A and Prompt B are provided in the Appendix D.0.2 for reference.

The char-level heatmaps in Figures 5b and 5c reveal the underlying cause. For Prompt A, saliency concentrates on key informational words, whereas after perturbation it shifts to the inserted characters and their neighboring tokens. This early stage disruption diverts attention away from core semantic content, propagates through the network, and ultimately alters the model's final output, demonstrating how inserted symbols can successfully drive adversarial misclassification.

5.3 How Punctuation Drives Output Errors through Key Neurons.

We build upon the neuron attribution methodology proposed in Yu & Ananiadou (2023) and extend it to the textual adversarial attack setting. In their work, log-probability was defined as the likelihood of generating the correct target token, serving as the basis to identify value neurons. In contrast, our focus lies in adversarial scenarios, where the input prompt is perturbed (***Prompt ADV***). We accordingly define $\log P\left(t^* \mid x\right) = \log \frac{\exp\left(f_\theta(x)_{t^*}\right)}{\sum_{v \in \mathcal{V}} \exp\left(f_\theta(x)_v\right)}$, as the log-probability of generating the adversarial target token, where $f_\theta(x)_{t^*}$ denotes the logit of the adversarial target token t^* under the perturbed prompt ***Prompt ADV***, and $\mathcal{V}$ denotes the entire vocabulary. We then introduce the following attribution metric by masking neuron i and measuring the difference:

$$\Delta_i = \log P_{\text{original}} - \log P^{(i)}_{\text{masked}}.$$

A positive Δ_i indicates that neuron i enhances the adversarial generation, while a negative value implies inhibition. To test how general this phenomenon is, we ran 10 prompts on three LLMs. A small set of neurons, shown by the Δ_i metric to have a high impact on the output (Figure 6b), are primarily activated by the inserted punctuation (Figure 6a). This finding suggests a clear causal relation: inserted characters trigger the activation of specific neurons, which in turn bias the model's internal representations and drive the incorrect output. By connecting perturbation at the character-level to neuron-level responses, our analysis highlights why OSCR-Attack can exploit the model's sensitivity and lead to adversarial misclassification.

6 Conclusion

We propose a continuous-relaxation framework that utilizes two learnable matrices to determine character-level positions and token choices. To optimize these matrices, we introduce a self-optimization method OSCR-Attack that updates them using gradients from an adversarial loss, all while keeping LLMs frozen. This method also includes a conflict-resolution strategy to map the continuous results back to discrete insertions. Extensive experiments on three benchmarks with three open-source LLMs show that OSCR-Attack improves ASR by up to 16% points and accelerates the attack by up to 6 times compared to recent baselines.

ACKNOWLEDGMENTS

This work was partially supported by the Postdoctoral Fellowship Program of the China Postdoctoral Science Foundation (Grant No. GZC20241318) and the National Natural Science Foundation of China (Grant No. 62176212).

ETHICS STATEMENT

Our work strictly adheres to the ethical standards of the research community. We conducted our experiments exclusively on publicly available datasets and ensured that no personally identifiable or sensitive information was used. As our research does not involve human subjects, it does not raise any safety concerns beyond the common scope of adversarial NLP research.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our work, we provide detailed descriptions of our model architecture, training procedures, and hyperparameter settings in the main text and Appendix. All code and data preprocessing scripts will be made publicly available upon publication.

REFERENCES

Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. *ArXiv*, abs/1711.02173, 2017. URL https://api.semanticscholar.org/CorpusID:3513372.

Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57, 2016. URL https://api.semanticscholar.org/CorpusID:2893830.

Yihong Dong, Xue Jiang, Jiaru Qian, Tian Wang, Kechi Zhang, Zhi Jin, and Ge Li. A survey on code generation with llm-based agents. *ArXiv*, abs/2508.00083, 2025. URL https://api.semanticscholar.org/CorpusID:280416987.

J. Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for nlp. *ArXiv*, abs/1712.06751, 2017. URL https://api.semanticscholar.org/CorpusID:28190697.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. HotFlip: White-box adversarial examples for text classification. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 31–36, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2006. URL https://aclanthology.org/P18-2006/.

Sinan Fan, Liang Xie, Chen Shen, Ge Teng, Xiaosong Yuan, Xiaofeng Zhang, Chenxi Huang, Wenxiao Wang, Xiaofei He, and Jieping Ye. Improving complex reasoning with dynamic prompt corruption: A soft prompt optimization approach. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=h7Qz1ulnvF.

Hao Fang, Jiawei Kong, Tianqu Zhuang, Yixiang Qiu, Kuofeng Gao, Bin Chen, Shutao Xia, Yaowei Wang, and Min Zhang. Your language model can secretly write like humans: Contrastive paraphrase attacks on llm-generated text detectors. *ArXiv*, abs/2505.15337, 2025. URL https://api.semanticscholar.org/CorpusID:278782326.

Zhaoxin Feng, Jianfei Ma, Emmanuele Chersoni, Xiaojing Zhao, and Xiaoyi Bao. Learning to look at the other side: A semantic probing study of word embeddings in LLMs with enabled bidirectional attention. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pp. 23226–23245, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1132. URL https://aclanthology.org/2025.acl-long.1132/.

Brian Formento, Chuan-Sheng Foo, and See-Kiong Ng. Confidence elicitation: A new attack vector for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=aTYexOYlLb.

Esther Gan, Yiran Zhao, Liying Cheng, Mao Yancan, Anirudh Goyal, Kenji Kawaguchi, Min-Yen Kan, and Michael Shieh. Reasoning robustness of LLMs to adversarial typographical errors. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 10449–10459, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.584. URL https://aclanthology.org/2024.emnlp-main.584/.

Simon Geisler, Tom Wollschlager, M. H. I. Abdalla, Johannes Gasteiger, and Stephan Gunnemann. Attacking large language models with projected gradient descent. *ArXiv*, abs/2402.09154, 2024. URL https://api.semanticscholar.org/CorpusID:267657696.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021. doi: 10.1162/tacl_a_00370. URL https://aclanthology.org/2021.tacl-1.21/.

Nora Hofer, Pascal Schöttle, Alexander Rietzler, and Sebastian Stabinger. Adversarial examples against a bert absa model – fooling bert with l33t, misspellign, and punctuation,. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, ARES '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450390514. doi: 10.1145/3465481.3465770. URL https://doi.org/10.1145/3465481.3465770.

Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. Deceiving google's perspective api built for detecting toxic comments. *ArXiv*, abs/1702.08138, 2017. URL https://api.semanticscholar.org/CorpusID:15418780.

Bairu Hou, Jinghan Jia, Yihua Zhang, Guanhua Zhang, Yang Zhang, Sijia Liu, and Shiyu Chang. Textgrad: Advancing robustness evaluation in nlp by gradient-driven optimization. *ArXiv*, abs/2212.09254, 2022. URL https://api.semanticscholar.org/CorpusID:254854553.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *ArXiv*, abs/2310.06825, 2023. URL https://api.semanticscholar.org/CorpusID:263830494.

Qi Lei, Lingfei Wu, Pin-Yu Chen, Alexandros G. Dimakis, Inderjit S. Dhillon, and M. Witbrock. Discrete adversarial attacks and submodular optimization with applications to text classification. *arXiv: Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:102352349.

Aiwei Liu, Honghai Yu, Xuming Hu, Shuang Li, Li Lin, Fukun Ma, Yawen Yang, and Lijie Wen. Character-level white-box adversarial attacks against transformers via attachable subwords substitution. *ArXiv*, abs/2210.17004, 2022. URL https://api.semanticscholar.org/CorpusID:253236900.

Han Liu, Zhi Xu, Xiaotong Zhang, Xiaoming Xu, Feng Zhang, Fenglong Ma, Hongyang Chen, Hong Yu, and Xianchao Zhang. Sspattack: A simple and sweet paradigm for black-box hard-label textual adversarial attack. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37 (11):13228–13235, Jun. 2023. doi: 10.1609/aaai.v37i11.26553. URL https://ojs.aaai.org/index.php/AAAI/article/view/26553.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin (eds.), *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Danish Pruthi, Bhuwan Dhingra, and Zachary Chase Lipton. Combating adversarial misspellings with robust word recognition. In *Annual Meeting of the Association for Computational Linguistics*, 2019. URL https://api.semanticscholar.org/CorpusID:166228669.

Elias Abad Rocamora, Yongtao Wu, Fanghui Liu, Grigorios Chrysos, and Volkan Cevher. Revisiting character-level adversarial attacks for language models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=AZWqXfM6z9.

Xuan Sheng, Zhicheng Li, Zhaoyang Han, Xiangmao Chang, and Piji Li. Punctuation matters! stealthy backdoor attack for language models. *ArXiv*, abs/2312.15867, 2023. URL https://api.semanticscholar.org/CorpusID:263833008.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Nikul Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=WGXb7UdvTX.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (eds.), *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170/.

Yi Tay, Vinh Q. Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. Charformer: Fast character transformers via gradient-based subword tokenization. *ArXiv*, abs/2106.12672, 2021. URL https://api.semanticscholar.org/CorpusID:235624202.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, 2023. URL https://api.semanticscholar.org/CorpusID:257219404.

Zain Ul Abedin, Shahzeb Qamar, Lucie Flek, and Akbar Karimi. ArithmAttack: Evaluating robustness of LLMs to noisy context in math problem solving. In Leon Derczynski, Jekaterina Novikova, and Muhao Chen (eds.), *Proceedings of the The First Workshop on LLM Security (LLMSEC)*, pp. 48–53, Vienna, Austria, August 2025. Association for Computational Linguistics. ISBN 979-8-89176-279-4. URL https://aclanthology.org/2025.llmsec-1.5/.

Hetvi Waghela, Jaydip Sen, and Sneha Rakshit. Enhancing adversarial text attacks on bert models with projected gradient descent. *ArXiv*, abs/2407.21073, 2024. URL https://api.semanticscholar.org/CorpusID:271571557.

Wenqiang Wang, Chongyang Du, Tao Wang, Kaihao Zhang, Wenhan Luo, Lin Ma, Wei Liu, and Xiaochun Cao. Punctuation-level attack: Single-shot and single punctuation can fool text models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=ir6WWkFR80.

Xuezhi Wang, Haohan Wang, and Diyi Yang. Measure and improve robustness in NLP models: A survey. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4569–4586, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.339. URL https://aclanthology.org/2022.naacl-main.339/.

Xilie Xu, Keyi Kong, Ninghao Liu, Li zhen Cui, Di Wang, Jingfeng Zhang, and Mohan S. Kankanhalli. An llm can fool itself: A prompt-based adversarial attack. *ArXiv*, abs/2310.13345, 2023. URL https://api.semanticscholar.org/CorpusID:264406064.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Muchao Ye, Chenglin Miao, Ting Wang, and Fenglong Ma. Texthoaxer: Budgeted hard-label adversarial attacks on text. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4): 3877–3884, Jun. 2022. doi: 10.1609/aaai.v36i4.20303. URL https://ojs.aaai.org/index.php/AAAI/article/view/20303.

Zeping Yu and Sophia Ananiadou. Neuron-level knowledge attribution in large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://api.semanticscholar.org/CorpusID:266362692.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Neural Information Processing Systems*, 2015. URL https://api.semanticscholar.org/CorpusID:368182.

Jiahao Zhao and Wenji Mao. Generative adversarial training with perturbed token detection for model robustness. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13012–13025, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.804. URL https://aclanthology.org/2023.emnlp-main.804/.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Gong, and Xing Xie. Promptrobust: Towards evaluating the robustness of large language models on adversarial prompts. In *LAMPS@CCS*, pp. 57–68, 2024. URL https://doi.org/10.1145/3689217.3690621.