# Q-FAKER: Query-free Hard Black-box Attack via Controlled Generation

**CheolWon Na, YunSeok Choi[†], Jee-Hyong Lee[†]**
College of Computing and Informatics
Sungkyunkwan University
{ncw0034, ys.choi, john}@skku.edu

## Abstract

Many adversarial attack approaches are proposed to verify the vulnerability of language models. However, they require numerous queries and the information on the target model. Even black-box attack methods also require the target model's output information. They are not applicable in real-world scenarios, as in hard black-box settings where the target model is closed and inaccessible. Even the recently proposed hard black-box attacks still require many queries and demand extremely high costs for training adversarial generators. To address these challenges, we propose **Q-faker** (***Q**uery-**f**ree **Ha**rd Black-box Attac**ker***), a novel and efficient method that generates adversarial examples without accessing the target model. To avoid accessing the target model, we use a surrogate model instead. The surrogate model generates adversarial sentences for a target-agnostic attack. During this process, we leverage controlled generation techniques. We evaluate our proposed method on eight datasets. Experimental results demonstrate our method's effectiveness including high transferability and the high quality of the generated adversarial examples, and prove its practical in hard black-box settings.

## 1 Introduction

Language models have become crucial to various real-world applications (Huang et al., 2024; Wang et al., 2024). Despite their remarkable performance, these models are vulnerable to adversarial examples, such as word substitutions (Papernot et al., 2016; Madry et al., 2018; Choi et al., 2022; Nakamura et al., 2023; Huang and Baldwin, 2023; Burger et al., 2023). Attackers can easily evade language models and achieve their malicious aims, such as spreading toxic content or rumors, by intentionally generating adversarial samples. To address these issues, many studies have been conducted to analyze the adversarial vulnerability of language models.

Adversarial attack methods on language models can be categorized into white-box attacks, black-box attacks, and hard black-box attacks. In white-box attacks, it is assumed that attackers have the internal structural information on the target model (Guo et al., 2021; Wang et al., 2022; Liu et al., 2022; Li et al., 2023c). Black-box attacks do not require the internal information of the target model, but still require the output of the model, such as the logit scores or the predicted labels (Gao et al., 2018; Ren et al., 2019; Jin et al., 2020; Zang et al., 2020; Li et al., 2020; Yu et al., 2022; Na et al., 2023). Hard black-box attacks do not require any information on the target model, such as its structure and outputs (Li et al., 2023a; Lv et al., 2023). It is assumed that attackers do not have any internal information of the model, and lack access to the logit values or the predicted results of the target model.

In real-world applications, hard black-box attacks are necessary because white-box and black-box attacks are impractical. In most cases, neural network models operate within a system. The output of the system may be visible, but the model's own output is hardly accessible externally. Another requirement for real-world scenarios, is to minimize queries. If attack methods try a large number of queries, they could easily be detected as suspicious actions. Since the target model is often unknown and resources are limited, hard black-box attacks must be model-agnostic and cost-efficient to be feasible in real-world applications.

Recently, hard black-box attack methods have been proposed by Li et al. (2023a) and Lv et al. (2023). However, these methods present several limitations in real-world scenarios. Their training relies on costly large-scale adversarial sentences obtained from various target models using various attack methods. Thus, their performance heavily

[†]Co-corresponding authors.



*Findings of the Association for Computational Linguistics:*
*NAACL 2025*, pages 8289–8304

| Attack Method | Requirements ($\star$) | | Preparation Costs | |
|---|---|---|---|---|
| | *Output Probability* | *Predicted Labels* | *Trainable Parameters* | *Extra Train Dataset* |
| DeepwordBug (Gao et al.) | ✓ | ✓ | – | – |
| PWWS (Ren et al.) | ✓ | ✓ | – | – |
| TextFooler (Jin et al.) | ✓ | ✓ | – | – |
| SememePSO (Zang et al.) | ✓ | ✓ | – | – |
| BERT-ATTACK (Li et al.) | ✓ | ✓ | – | – |
| CT-GAT (Lv et al.) | – | – | 406M (Enc.-Dec.) | Adversarial Datasets ($1.5M$) |
| Q-faker (ours) | – | – | 1K (single-layer) | Target Task Datasets ($0.2M$) |

Table 1: Requirements for black-box attack methods. The star ($\star$) indicates features that necessitate access to the target model. Existing black-box methods require output information obtained from the target model, which limits their applicability in more restrictive black-box settings such as real-world scenarios.

depends on attack algorithms and target models which were used to obtain the datasets. Despite using extensive datasets, they still require a large number of queries to achieve success, making them inefficient and impractical in real-world applications.

To address the challenges in real-world scenarios, we propose a novel hard-black box attack method, **Q-faker** (***Q**uery-**f**ree **Ha**rd Black-box At**tacker***), that is query-free, cost efficient in training, and capable of target-agnostic attacks. In the hard black-box setting, we have no access to the target model. To address this, we use a surrogate model instead of the target model to generate adversarial sentences for target-agnostic attacks. For low-resource training, we build the surrogate model using a pretrained generative language model, which can approximate the capabilities of the target model. We modify the model by adding a single-layer (classification head) consisting of 1K trainable parameters. For generating adversarial examples with zero queries, we utilize gradients from the surrogate model, and leverage controlled generation techniques (Dathathri et al., 2020).

Table 1 shows our contribution compared to existing methods. Compared to existing black-box attack methods, such as DeepwordBUG (Gao et al., 2018), PWWS (Ren et al., 2019), TextFooler (Jin et al., 2020), SememPSO (Zang et al., 2020), and BERT-Attack (Li et al., 2020)), our proposed method does not require any access to the target model. Additionally, our method incurs significantly lower training costs compared to CT-GAT (Lv et al., 2023), a hard black-box attack.

To verify the performance, efficiency and target-agnostic capability, we conduct various experiments on four real-world tasks, including disinformation, spam, toxic, and misinformation (Chen et al., 2022a), across various target models, such as, BERT, XLNet, RoBERTa, DeBERTa, DistilBERT, and ALBERT. Experimental results demonstrate the superiority of Q-faker not only over hard black-box baselines but also over black-box baselines which can access more information than Q-faker. We also confirm the high transferability of Q-faker through the experiments. Q-faker achieves higher and more consistent attack success rates across target models compared to the baselines. The various experimental results are reported in §5 and §6 including overall performance, ablation study, transferability, quality evaluation of generated examples, and detector evasion performance.

## 2 Related Work

The goal of adversarial attacks on language models differs whether the target model is NLU (Natural Language Understanding) or NLG (Natural Language Generation). Adversarial attacks on NLU models, such as classification models, aim to mislead the output of model. Recent research about attacks on NLG models, such as generative LLMs (Large Language Models), have the objective of generating forbidden response or hallucinations like the jailbreaking (Chao et al., 2023; Sun et al., 2024). The differences in attack methods are as follows: for NLU models, the approach involves manipulating the tokens of the input sentence. In generative LLMs, the approach includes adding additional prompts for various scenarios or tuning the parameters of soft prompts (Perez et al., 2022; Zou et al., 2023; Deng et al., 2023; Li et al., 2023b). This paper focuses on adversarial attacks on NLU models.

**Black-box Attacks on NLU models** The existing black-box attack methods are character or word manipulation approaches (Gao et al., 2018; Ren

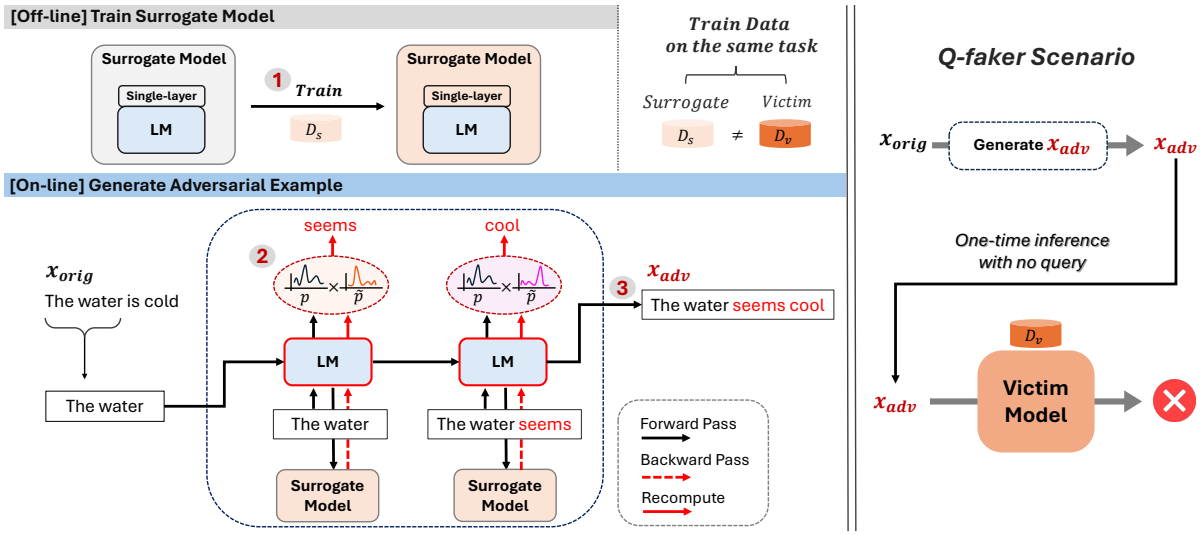


Figure 1: The process of Q-faker has three main steps: (1) Training the surrogate model using a different dataset for the same task as the target model; (2) Updating the language model using adversarial gradients from the surrogate model; and (3) Generating controlled adversarial examples from the updated language model.

et al., 2019; Jin et al., 2020; Zang et al., 2020; Li et al., 2020; Yu et al., 2022). These methods rely on query-based algorithms and require a large number of queries to achieve a successful attack. They also require output information from the target model, which is not applicable in hard black-box setting as real-world scenarios. Recently, adversarial attack methods have been proposed that do not require any information about the target model (Li et al., 2023a; Lv et al., 2023). These methods train pre-trained language models on adversarial datasets that include adversarial examples generated by other attack algorithms. These approaches incur significant additional costs to obtain the adversarial datasets.

## 3 Problem Statement

This study focuses on hard black-box attacks in real-world scenarios where queries and model's internal information are limited. Unlike our hard black-box setting, baselines are able to query the target model to obtain output information.

**Hard black-box setting.** Since the our experimental setting is a hard black-box setting, we do not access to any information, including predicted labels and training dataset of target model. Therefore, the surrogate model's training dataset differs from the dataset used to fine-tune the target model. Given the assumption that no output information from the target model is used, we generate only one adversarial sentence and conduct the test with a one attempt in the attack process.

**Goal** Given an input sample $(s_i, y)$, our goal is to find $s_i^{adv}$ by adding generated perturbation to $s_i$, that misleads the victim model. The adversarial text $s_i^{adv}$, which has successfully attacked, needs to satisfy the followings criteria:

$$T(s_i^{adv}) \neq y, \tag{1}$$

where $T(s_i) = y$, $T$ is the target model, and $y$ is the ground truth.

## 4 Methodology

In this section, we provide a detailed explanation of Q-faker. As shown in Figure 1, our proposed method consists of three main steps: (1) Train the surrogate model to derive adversarial gradients for adversarially updating the output distribution of the generator. Since the surrogate model is trained on same target task dataset, we can generate examples with high transferability. (2) Update the output distribution of the generator based on the adversarial gradients obtained from the surrogate model. In this step, we utilize the generative pre-trained language model GPT-2 as the generator to generate adversarial examples that ensure fluency and grammatical correctness. (3) Generate an adversarial example from the updated output distribution of the generator. During the attack process, we use only the single adversarial example generated by Q-faker. Thus, we achieve a query-free and applicable approach in hard black-box settings. The proposed method is summarized in Algorithm 1.

**Algorithm 1:** Q-faker Pseudo-code

**Input :** Original Example $s = [x_0, \ldots, x_n]$, Target Model $T$

**Output :** Adversarial Example $s^{adv}$

1 **Initialization:** Begin with a pre-trained language model LM (in this paper GPT-2) and surrogate model SurModel
2 Train the SurModel on target task dataset
3 $s \leftarrow [x_0, \ldots, x_n]$
4 $s^{adv} \leftarrow [x_0, \ldots, x_t]$, $t < n$
5 $maxlen \leftarrow |s|$
6 **while** $t < maxlen$ **do**
7 $p(x_t), h_t \leftarrow \text{LM}(x_{\leq t-1}, h_{t-1})$
8 $\mathcal{L}_{\text{sur}} \leftarrow \text{SurModel}(p(x_t), \tilde{y})$
9 $\tilde{h_t} \leftarrow h_t + \eta \nabla_{h_t} \mathcal{L}_{\text{sur}}$
10 $\tilde{p}(x_{t+1}), h_{t+1} \leftarrow \text{LM}(x_{\leq t}, \tilde{h_t})$
11 $\tilde{x}_{t+1} \sim \tilde{p}_{fusion}(x_{t+1})$ *# See Section 4.3*
12 $s^{adv} \leftarrow [x_0, \ldots, \tilde{x}_{t+1}]$
13 **end**
14 **if** $T(s) \leftarrow y$ *and* $T(s^{adv}) \neq y$ **then**
15 *# Attack Success*
16 Return: $s^{adv}$
17 **end**

**Controlled Generation** Controlled generation is a method, generating sentences with a specific objective. These methods are commonly used in various tasks such as intent-based text generation, text style transfer, and etc. To the best of our knowledge, we are the first to adopt controlled generation for generating adversarial examples. We utilize a simple and efficient controlled generation method proposed by Dathathri et al. (2020).

### 4.1 Surrogate Model

In order to adversarially update the output of the generator to the target model, we use a surrogate model that has been trained on the same target task but with a different dataset from the one used to train the target model. We utilize the GPT-2 as a LM of the surrogate model for all experiments. The architecture of surrogate model consists of generator with a single-layer head that has 1k parameters. When training the surrogate model, only the single layer is trained while the generator is frozen. We train the surrogate model to minimize the loss $\mathcal{L}_{sur}$ as follows:

$$\mathcal{L}_{sur} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{2}$$

### 4.2 Adversarial Output Distribution

In order to adversarially update the output distribution of the Language Model (LM), we adopt a controlled generation approach (Dathathri et al., 2020). We compute adversarial gradients from a surrogate model in §4.1. The adversarial gradient is calculated to maximize the loss function, thereby guiding the model towards incorrect predictions. Based on the obtained adversarial gradients, we adversarially update the output distribution of LM. Let $h$ be the key-value pair of LM. We then update $h$ to shift the output distribution for adversarial generation. To obtain the updated $\tilde{h}$, we compute the adversarial gradient of the surrogate loss as follows:

$$p(x_t), h_t = \text{LM.forward}(x_{<t}, h_{t-1}) \tag{3}$$

$$\nabla_{h_t} \mathcal{L}_{sur} = \frac{\partial \mathcal{L}_{sur}}{\partial p(x_t)} \cdot \frac{\partial p(x_t)}{\partial h_t} \tag{4}$$

$$\tilde{h}_t \leftarrow h_t + \alpha \frac{\nabla_{h_t} \mathcal{L}_{sur}}{\|\nabla_{h_t} \mathcal{L}_{sur}\|^{\gamma}} \tag{5}$$

where $\alpha$ is the step size, and $\gamma$ is the scaling coefficient for the normalization term. we use 0.06 and 1.0, respectively. This update step is repeated 10 times. Then, we adversarially update the output distribution of LM with updated $\tilde{h}$ as follows:

$$\tilde{p}(x_{t+1}), h_{t+1} = \text{LM.forward}(x_{\leq t}, \tilde{h}_t) \tag{6}$$

This step is described in Algorithm 1 (line 7-10).

### 4.3 Controlled Adversarial Example Generation

In order to preserve the meaning of the original sentence, half of it is used as given tokens. These tokens are then used as input to the LM. For controlled generating of adversarial examples, we use the updated LM as described in §4.2. To ensure the LM's fluency capability, we utilize the *post-norm fusion* (Stahlberg et al., 2018). This process maintains the fluency of the unmodified language model (in this study, GPT-2). The next tokens are then sampled from the fusion distribution as follows:

$$\tilde{p}_{\text{fusion}}(x_{t+1}) = p_{\text{orig}}(x_{t+1})^{1-\lambda} \cdot \tilde{p}(x_{t+1})^{\lambda} \tag{7}$$

As $\lambda$ approaches 1, this converges to the distribution from the updated LM, and as approaches 0, it converges to the unmodified LM's distribution. We set $\lambda$ to 0.97 in all experiments.

| Task | Spam | | | | Sensitive Information | | | |
|---|---|---|---|---|---|---|---|---|
| Dataset / Method | Assassin | | Enron | | EDENCE | | FAS | |
| | **ASR (↑)** | **Query (↓)** | **ASR (↑)** | **Query (↓)** | **ASR (↑)** | **Query (↓)** | **ASR (↑)** | **Query (↓)** |
| DeepwordBug | 0.0 | ∞ | 0.0 | ∞ | 2.5 | 9.00 | 3.4 | 8.82 |
| PWWS | 0.0 | ∞ | 0.0 | ∞ | 0.0 | ∞ | 0.0 | ∞ |
| TextFooler | 0.0 | ∞ | 0.0 | ∞ | 0.0 | ∞ | 0.0 | ∞ |
| SememePSO | 0.1 | 5.00 | 0.0 | ∞ | 0.6 | 7.00 | 1.4 | 8.29 |
| BERT-Attack | 0.0 | ∞ | 0.1 | 9.00 | 0.9 | 7.10 | 2.5 | 8.50 |
| CT-GAT | 1.1 | 9.94 | 0.2 | 9.99 | 4.1 | 9.76 | 10.0 | 9.32 |
| Q-faker (ours) | **5.1** | **0.00** | **0.7** | **0.00** | **54.6** | **0.00** | **43.3** | **0.00** |
| Clean Acc. | 98.4% | | 99.6% | | 95.9% | | 97.4% | |

| Task | Disinformation | | | | Toxicity | | | |
|---|---|---|---|---|---|---|---|---|
| Dataset / Method | CGFake | | Amazon-LB | | Jigsaw | | HSOL | |
| | **ASR (↑)** | **Query (↓)** | **ASR (↑)** | **Query (↓)** | **ASR (↑)** | **Query (↓)** | **ASR (↑)** | **Query (↓)** |
| DeepwordBug | 0.0 | ∞ | 0.6 | 6.67 | 8.1 | 8.05 | 7.1 | 7.63 |
| PWWS | 0.0 | ∞ | 0.0 | ∞ | 0.0 | ∞ | 0.2 | 9.00 |
| TextFooler | 0.0 | ∞ | 0.0 | ∞ | 0.0 | ∞ | 0.2 | 8.00 |
| SememePSO | 0.0 | ∞ | 0.1 | 9.00 | 1.0 | 7.10 | 4.6 | 7.09 |
| BERT-Attack | 0.0 | ∞ | 0.5 | 7.00 | 3.3 | 7.88 | 5.8 | 8.12 |
| CT-GAT | 12.4 | 8.38 | 8.1 | 9.47 | 32.4 | 7.76 | **55.8** | 5.73 |
| Q-faker (ours) | **13.4** | **0.00** | **8.6** | **0.00** | **38.2** | **0.00** | 53.1 | **0.00** |
| Clean Acc. | 97.8% | | 91.6% | | 92.5% | | 95.5% | |

Table 2: Comparison of our proposed method with the baseline methods on eight victim models. The results are based on a real-world scenario with a maximum query limit set to 10. The best performance is in **boldface**, and the second is underlined.

## 5 Experiments

In this section, we demonstrate the effectiveness of our method. We use the Advbench dataset, a security-oriented adversarial NLP benchmark that includes tasks related to real-world harmful content problems (Chen et al., 2022b). More details of datasets and target models are provided in Appendix A.

### 5.1 Tasks and Datasets

We conduct experiments on four detection tasks from the Advbench benchmark: disinformation, toxicity, spam, and sensitive information. Each task consists of two datasets, resulting in a total of eight victim models trained on these datasets. We randomly selected 1,000 test samples from each dataset. For a fair comparison, we used the same random seed as in previous studies.

### 5.2 Evaluation Metrics

We use two metrics to evaluate the attack *efficiency* of our method. To measure the *quality* of the generated adversarial examples, we use also three metrics. The performance results reported in this paper represent the average of successful attack instances.

**Attack efficiency.** We evaluate the efficiency of attack methods using the attack success rate and query time. (1) Attack Success Rate is the success ratio of attacks (**ASR**); the higher the ASR, the better the performance of an attack method. (2) The query time is defined as the number of queries required to succeed attacks (**Query**).

**Attack quality.** To measure the quality of the generated adversarial examples, we use the following three metrics: (1) The cosine similarity of the Universal Sentence Encoder vector (Cer et al., 2018), which measures the semantic similarity between the original sentence and adversarial sentence (**USE**); (2) Perplexity of language model, which assesses the fluency of the generated sentence (**PPL**); and (3) ($\Delta$**I**) indicates the increase in grammatical errors.

### 5.3 Implementation

**Victim model.** We use BERT, the most representative pre-trained language model in the NLU task. We fine-tune a separate victim model for all datasets, building a total of eight victim models. We use the same split of the training dataset as Lv et al. (2023) did.

**Baselines.** For a fair comparison, all baselines were re-implemented using the same fixed seed. We used the NLP attack package, OpenAttack

(Zeng et al., 2021), to implement some baselines. We select high-representative attack methods at both the character-level and the word-level. At the character-level, we select TextFooler (Jin et al., 2020), PWWS (Ren et al., 2019), and Deepword-Bug (Gao et al., 2018), while at the word-level, we use BERT-attack (Li et al., 2020) and SememePSO (Zang et al., 2020). We also compare with a strong baseline, CT-GAT, which has high-transferability. To implement CT-GAT, we use the source provided by Lv et al. (2023) to reproduce the results. When we implement CT-GAT, we train the pre-trained Encoder-Decoder model, BART.

**Hyperparameters.** Q-faker has two hyperparameters. $r$ is the ratio of given tokens to the length of the original sentence. It is a hyperparameter that determines how many of the original sentence is used as input to the generator for generating adversarial sentences. $\lambda$ is a hyperparameter used for post-norm fusion in §4.3. The closer $\lambda$ is to 1, the output distribution of the generator converges to an adversarially updated distribution. We set $r$ to 0.5 and $\lambda$ to 0.97 in all experiments.

### 5.4 Experimental Setup

Since the baselines require output information from the target models, we allow them to make iterative queries to the target models in a black-box setting, with 10 queries in Table 2 and 20 queries in Figure 3, respectively. On the other hand, we evaluate our method in hard black-box setting. We evaluate our method with only one inference, without any target model's information including train datasets.

**Cross-dataset setting.** To adhere to the hard black-box setting, we conduct experiments using cross-dataset setting. We generate adversarial sentences using a dataset different from the target task dataset, because attackers cannot access the dataset for the target task. For example, if the target model is trained with the Assassin dataset for spam detection, the surrogate model is trained on the Enron dataset.

We ensure that both models are independently trained with two distinct datasets, each independently collected from real-world sources. Our method is validated in this setting on all the experiments in this paper. This experimental design rigorously validates hard black-box setting.

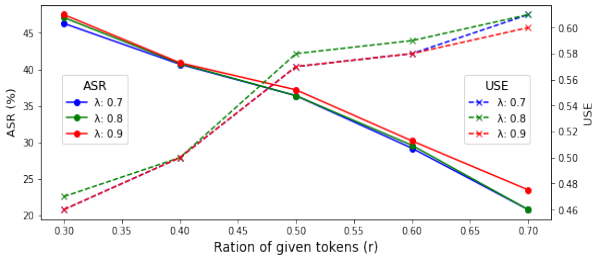


Figure 2: Results of ablation study on Jigsaw. The solid lines (——) represent ASR (left y-axis), the dashed lines (- - -) represent USE (right y-axis).

### 5.5 Experimental Results

**Main results.** Table 2 shows the attack success rate and number of queries when the queries to the victim model are limited to 10. The results indicate that query-based methodologies mostly fail with the ASR close to zero. As shown in Table 2, Q-faker outperforms the baselines in all metrics except ASR on HSOL. Since CT-GAT utilized the HSOL dataset during their training, it achieves an ASR approximately 2.7%p higher than our method on the HSOL. However, our proposed method, Q-faker, mostly shows the best on all datasets. Specifically, Q-faker impressively outperforms in the sensitive information task, with ASR differences ranging from at least 4 times to as much as 40 times higher compared to the baselines.

**Ablation study.** We conduct an ablation study on the Jigsaw dataset to compare the effects of adversarial distribution ($\lambda$) and given tokens ($r$). As shown in Figure 2, increasing $\lambda$ slightly improves ASR. This indicates that adversarial distribution is helpful to attack, but not highly sensitive to $\lambda$. When $r$ is reduced below 0.5, ASR increases while the USE drops significantly, resulting in substantial shift of the original sentence's meaning. This indicates a trade-off between ASR and USE based on $r$. When $r$ is around 0.5, we effectively mitigate the trade-off between ASR and the preservation of the original meaning.

## 6 Further Analysis

In this section, we further analyze the effectiveness of our proposed method through additional experiments. We demonstrate the superiority of Q-faker in various scenarios, including extremely limited queries, transferability, qualitative analysis and adversarial defenses. Additional results and more detailed, including time-complexity and feasibility of

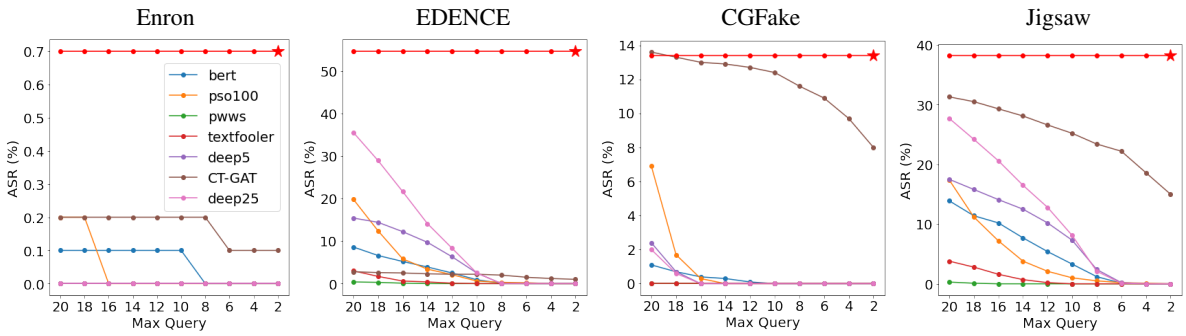


Figure 3: Comparison of ASR according to the number of queries (from 20 to 1). The red star(⋆) is our method. As the number of queries accessible to the target model becomes more restricted, the ASR of baseline methods drops to near zero. This demonstrates the superiority of our method in real-world scenarios with limited queries.

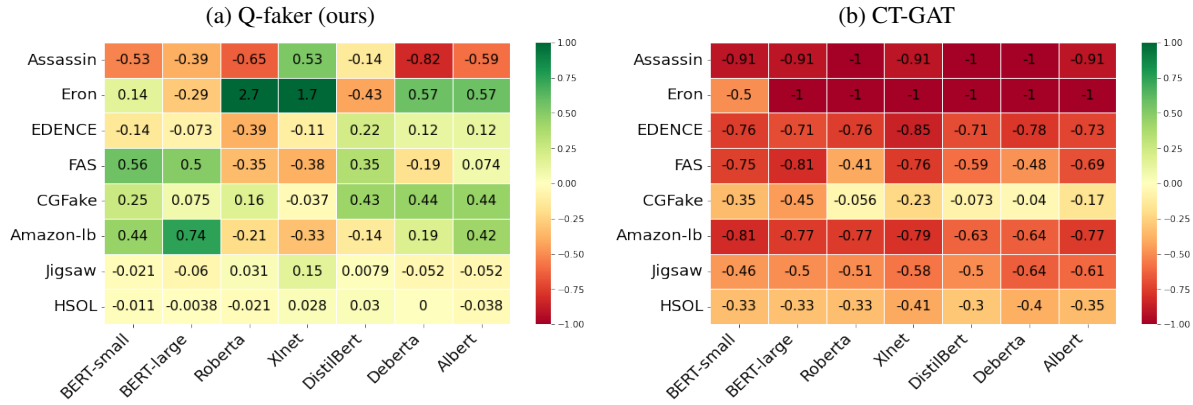


Figure 4: Consistent attack capabilities on various target models. This heatmap illustrates the difference ratio in ASR between BERT-base and other models. We compare our method (left) with CT-GAT (right). Colors closer to green indicate higher ASR on other target models, highlighting the high transferability.

LLM attacks, are provided in Appendix B.

### 6.1 Number of Queries

We conduct further experiments under real-world scenarios where the number of queries is extremely limited. Figure 3 shows the results of experiments conducted with different numbers of queries from 20 down to 1. When the number of queries is extremely limited to 1, our proposed method achieves significantly higher ASR compared to all other baseline methods across all datasets.

### 6.2 Transferability

To validate the target-agnostic capability of our method, we conduct transferability experiments. We calculate the ASR difference between the reference model and each of comparison models to assess how consistently our method performs across various models. We choose the BERT-base as the reference model, and other pretrained language models as the comparison models, respectively. Figure 4 is a heatmap showing the difference ratios in ASR between the reference model and the comparison models. Cell values closer to zero signify more consistent performance across various models. The positive cell value indicates that attack method has a higher ASR on the comparison model than the reference model, while the negative cell value indicates lower attack performance. For example, the cell value of (HSOL, BERT-small) is -0.011 which means that our method shows very similar performance on both the comparison and the reference models. In Figure 4, our method (Q-faker) shows that most cells exhibit positive values or close to zero, whereas CT-GAT exhibits negative values in all cases. The results show that our method generally consistent the target-agnostic attack capability across various target models, which indicates high transferability of our method.

| Method | USE (↑) | PPL (↓) | Δ I (↓) |
|---|---|---|---|
| PWWS | **0.85** | 194.32 | 21.01 |
| BERT-Attack | 0.78 | 1280.94 | 3.56 |
| CT-GAT | 0.74 | 94.40 | 8.78 |
| Q-faker (ours) | 0.75 | **25.95** | **-0.49** |

Table 3: Comparison of adversarial examples generated by attack methods on Amazon-LB. Token manipulation approaches (PWWS and BERT-ATTACK) have high *USE* scores, whereas generation-based methods (CT-GAT and Q-faker) perform better in *PPL* and $\Delta I$.

| Metric | Q-faker (ours) | CT-GAT |
|---|---|---|
| Naturalness | **108** | 58 |
| Meaning preservation | **95** | 71 |
| Grammar | **112** | 54 |

Table 4: ChatGPT prompt evaluation results. The results represent the total times the better generated sentence was chosen.

### 6.3 Generated Examples Quality Analysis

We evaluate the quality of the adversarial examples generated by attack methods. To validate a comprehensive quality analysis, we conduct additional the following experiments: automatic metrics, ChatGPT prompts evaluation, and human evaluation.

**Automatic evaluation.** For a quantitative evaluation of quality, we conduct experiments using automated metrics as follows: *USE*, *PPL*, and $\Delta I$. Table 3 shows that our method show slightly lower *USE* but better perform in *PPL* and $\Delta I$. The generator-based methods such as Q-faker and CT-GAT, have more differences in representation space compared to token manipulate-based methods. However, our proposed method shows notably lower perplexity and great grammatical correctness. These results highlight the high quality that appears natural to human judges.

**ChatGPT prompt evaluation.** We conduct experiments using ChatGPT API to evaluate the quality of the generated sentences. Recent studies have raised fairness issues about human annotators and suggested that LLM-evaluators can be more reliable than human-evaluators (Guo et al., 2023). In other lines of research, some studies address concerns about bias in LLM-evaluators (Koo et al., 2024). To mitigate such biases, we utilize to the GPT-Rank template proposed by Jiang et al. (2023). For more details about prompt are described to the Appendix C. The experimental results show that our method outperforms on all metrics, as shown in Table 4. Our method leverages a pre-trained language model on a large corpus, which is advantageous for naturalness (fluency) and grammatical correctness, and also better preserves the original semantic meaning.

| Dataset | | Accuracy | MP | Natural |
|---|---|---|---|---|
| **FAS** | Original | 0.84 | - | 3.93 |
| | Adversarial | 0.82 | 3.84 | 4.11 |
| **Assassin** | Original | 0.97 | - | 3.64 |
| | Adversarial | 0.94 | 3.82 | 3.87 |

Table 5: Human evaluation results. Since the original datasets are online-data collected from the real-world, they mostly consist of informal, colloquial, and ungrammatical sentences. Therefore, adversarial sentences generated by our LM-based method have an advantage in naturalness.

**Human evaluation.** To further validate quality of generated sentences, we conduct a human evaluation to measure the semantic preservation and naturalness. We randomly select 50 pairs of original and adversarial texts from each of the FAS and HSOL datasets. We ask four human evaluators the following three metrics: (1) **Accuracy**: the prediction of the label for the task (sensitive/toxic or not), (2) **MP** (Meaning Preservation): how well the adversarial sentence preserves the meaning of the original sentence, and (3) **Natural**: how natural the sentence appears, as if it were human-written without any manipulation. MP and Naturalness are scoring from 1-5 following (Li et al., 2020). Table 5 summarizes the results of the human evaluation, showing that our method effectively preserves the original meaning and increases naturalness.

### 6.4 Adversarial Defense

Broadly, defense techniques s against adversarial attacks can be categorized into adversarial training (Dong et al., 2021; Zhao and Mao, 2023) and detection (Mosca et al., 2022; Zheng et al., 2023). In real-world scenarios, adversarial detection which preemptively blocks adversarial inputs, is more practical. To evaluate the detector evasion performance of the Q-faker, we conduct experiments to attack the detector on Amazon-LB datasets. We use the detector proposed by (Mosca et al., 2022). Our proposed method demonstrates better performance compared to other approaches. The results are provided in Appendix B.7.

## 7 Conclusion

We proposed Q-faker, a novel, efficient, and query-free hard black-box attack method. Our method adopts controlled generation techniques to generate adversarial examples without any information from the target model. It demonstrates excellent performance without accessing target model information, and and has proven effective in real-world scenarios.

## Limitations

We have some limitations about this work; (1) Our proposed method needs to know the specific task of the target model, (2) We do not consider scenarios where the query is infinite and access to the target model. However, our proposed method also provides room for an iterative attack to improve attack performance.

## Ethical Considerations

We conduct experiments that are the security-oriented benchmark dataset, Advbench, which is open-source. We do not use any closed-source data, and our works ensure the ethical policy. However, the datasets used in this work contain potentially harmful content. We have chosen not to report directly on these harmful examples to consider ethical policy. Our research focuses on adversarial attack methods that can be applied in real-world scenarios, this could potentially be misused by malicious users. They can spread rumors or spam emails. Therefore, the researcher in this related work must strictly adhere to ethical standards.

## Acknowledgments

This work was supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2019-II190421, AI Graduate School Support Program(Sungkyunkwan University), 10%), (IITP-2025-RS-2020-II201821, ICT Creative Consilience Program, 10%), (No.RS-2021-II212068, Artificial Intelligence Innovation Hub, 10%), (IITP-2025-RS-2024-00437633, ITRC(Information Technology Research Center), 70%)

## References

Christopher Burger, Lingwei Chen, and Thai Le. 2023. "are your explanations reliable?" investigating the stability of LIME in explaining text classifiers by marrying XAI and adversarial attack. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12831–12844, Singapore. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. 2023. Jailbreaking black box large language models in twenty queries. *Preprint*, arXiv:2310.08419.

Yangyi Chen, Hongcheng Gao, Ganqu Cui, Fanchao Qi, Longtao Huang, Zhiyuan Liu, and Maosong Sun. 2022a. Why should adversarial perturbations be imperceptible? rethink the research paradigm in adversarial NLP. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11222–11237, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yangyi Chen, Hongcheng Gao, Ganqu Cui, Fanchao Qi, Longtao Huang, Zhiyuan Liu, and Maosong Sun. 2022b. Why should adversarial perturbations be imperceptible? rethink the research paradigm in adversarial NLP. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11222–11237, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

YunSeok Choi, Hyojun Kim, and Jee-Hyong Lee. 2022. Tabs: Efficient textual adversarial attack for pretrained nl code model using semantic beam search. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5490–5498.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Boyi Deng, Wenjie Wang, Fuli Feng, Yang Deng, Qifan Wang, and Xiangnan He. 2023. Attack prompt generation for red teaming and defending large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2176–2189, Singapore. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinshuai Dong, Anh Tuan Luu, Min Lin, Shuicheng Yan, and Hanwang Zhang. 2021. How should pre-trained language models be fine-tuned towards adversarial robustness? *Advances in Neural Information Processing Systems*, 34:4356–4369.

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

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *Preprint*, arXiv:2301.07597.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. 2021. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5747–5757, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Yichen Huang and Timothy Baldwin. 2023. Robustness tests for automatic machine translation metrics with adversarial attacks. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5126–5135.

Yining Huang, Keke Tang, and Meilian Chen. 2024. A comprehensive survey on evaluating large language model applications in the medical industry. *arXiv preprint arXiv:2404.15777*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Ryan Koo, Minhwa Lee, Vipul Raheja, Jong Inn Park, Zae Myung Kim, and Dongyeop Kang. 2024. Benchmarking cognitive biases in large language models as evaluators. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 517–545, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Z Lan. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Guoyi Li, Bingkang Shi, Zongzhen Liu, Dehan Kong, Yulei Wu, Xiaodan Zhang, Longtao Huang, and Honglei Lyu. 2023a. Adversarial text generation by search and learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15722–15738, Singapore. Association for Computational Linguistics.

Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, Jie Huang, Fanpu Meng, and Yangqiu Song. 2023b. Multi-step jailbreaking privacy attacks on ChatGPT. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4138–4153, Singapore. Association for Computational Linguistics.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202.

Yufei Li, Zexin Li, Yingfan Gao, and Cong Liu. 2023c. White-box multi-objective adversarial attack on dialogue generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1778–1792.

Aiwei Liu, Honghai Yu, Xuming Hu, Shu'ang Li, Li Lin, Fukun Ma, Yawen Yang, and Lijie Wen. 2022. Character-level white-box adversarial attacks against transformers via attachable subwords substitution. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7664–7676, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Minxuan Lv, Chengwei Dai, Kun Li, Wei Zhou, and Songlin Hu. 2023. Ct-gat: Cross-task generative adversarial attack based on transferability. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5581–5591.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

Edoardo Mosca, Shreyash Agarwal, Javier Rando Ramírez, and Georg Groh. 2022. "that is a suspicious reaction!": Interpreting logits variation to detect NLP adversarial attacks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7806–7816, Dublin, Ireland. Association for Computational Linguistics.

CheolWon Na, YunSeok Choi, and Jee-Hyong Lee. 2023. DIP: Dead code insertion based black-box attack for programming language model. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*,

pages 7777–7791, Toronto, Canada. Association for Computational Linguistics.

Mutsumi Nakamura, Santosh Mashetty, Mihir Parmar, Neeraj Varshney, and Chitta Baral. 2023. LogicAttack: Adversarial attacks for evaluating logical consistency of natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13322–13334, Singapore. Association for Computational Linguistics.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3419–3448.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

V Sanh. 2019. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Felix Stahlberg, James Cross, and Veselin Stoyanov. 2018. Simple fusion: Return of the language model. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 204–211, Brussels, Belgium. Association for Computational Linguistics.

Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, Zhengliang Liu, et al. 2024. Trustllm: Trustworthiness in large language models. *Preprint*, arXiv:2401.05561.

Boxin Wang, Chejian Xu, Xiangyu Liu, Yu Cheng, and Bo Li. 2022. SemAttack: Natural textual attacks via different semantic spaces. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 176–205, Seattle, United States. Association for Computational Linguistics.

Shenao Wang, Yanjie Zhao, Xinyi Hou, and Haoyu Wang. 2024. Large language model supply chain: A research agenda. *arXiv preprint arXiv:2404.12736*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Zhen Yu, Xiaosen Wang, Wanxiang Che, and Kun He. 2022. TextHacker: Learning based hybrid local search algorithm for text hard-label adversarial attack. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 622–637, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080.

Guoyang Zeng, Fanchao Qi, Qianrui Zhou, Tingji Zhang, Zixian Ma, Bairu Hou, Yuan Zang, Zhiyuan Liu, and Maosong Sun. 2021. OpenAttack: An open-source textual adversarial attack toolkit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 363–371, Online. Association for Computational Linguistics.

Jiahao Zhao and Wenji Mao. 2023. Generative adversarial training with perturbed token detection for model robustness. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13012–13025.

Rui Zheng, Shihan Dou, Yuhao Zhou, Qin Liu, Tao Gui, Qi Zhang, Zhongyu Wei, Xuanjing Huang, and Menghan Zhang. 2023. Detecting adversarial samples through sharpness of loss landscape. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11282–11298, Toronto, Canada. Association for Computational Linguistics.

Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

# A Experimental Setup

## A.1 Dataset

We utilize Advbench benchmark dataset. This dataset consists of data collected from real-world scenarios. The dataset statistics are presented in Table 6

**Spam.** This task is to detect spam message including advertising, scams, phishing, and more. This task is crucial for improving security and maintaining user trust.

**Sensitive Information.** Detecting sensitive information in text is vital to prevent data leakage. This task focuses on detecting sensitive information from companies, including intellectual property and product development updates and individuals information.

**Disinformation.** The fake information is caused by subjectively facts. This task is to identify deliberate fabrication of information as follows: (1) Artificial comments reversing the black and white; (2) Generated nonexistent information.

**Toxic.** Malicious toxic texts are widespread in the web. The toxic texts detection is to identify for toxic contents including sexism, racism, cyberbullying, and etc.

## A.2 Surrogate and Target Models

**Surrogate Model.** In all experiments, we use GPT2-medium as the language model for generation, and we utilize a surrogate model by adding a classification head (a single layer) to GPT2 for task-specific training in order to obtain task-related gradients.

**Target Models.** In this paper, BERT-base is used as the target model for the reported results. For Table 4, we employ encoder-based models that demonstrate strong performance in classification tasks, including the BERT, Roberta, Xlnet, DistilBert, Deberta, and Albert. The performance of the fine-tuned target models for each task is reported in Table 7.

- **BERT** (Devlin et al., 2019), a one of the most widely used language models, known for its bidirectional context understanding.
- **RoBERTa** (Liu, 2019), refines BERT by removing the Next Sentence Prediction (NSP) objective to robustly optimization, and increasing the training duration, and utilizing a larger batch size and more data.
- **XLNet** (Yang et al., 2019), a permutation language model without relying on masking, which improves its ability to model language relationships in more flexible and robust ways.
- **DeBERTa** (He et al., 2020), an advanced variant of BERT that improves upon BERT and RoBERTa by using a disentangled attention mechanism and enhanced decoding.
- **DistilBERT** (Sanh, 2019), a faster and more efficient version of BERT, designed using knowledge distillation to retain.
- **ALBERT** (Lan, 2019), a lightweight version of BERT designed to reduce the model size and training time by sharing parameters across layers and factorizing the embeddings.

| Dataset | # of Train | # of Test | Avg. Length |
|---|---|---|---|
| Spam | | | |
| Enron | 16159 | 7277 | 311.53 |
| Assassin | 2081 | 2066 | 308.50 |
| Sensitive Information | | | |
| EDENCE | 51098 | 10328 | 21.79 |
| FAS | 33814 | 13294 | 29.27 |
| Disinformation | | | |
| Amazon-LB | 17434 | 8522 | 100.13 |
| CGFake | 28290 | 12130 | 67.48 |
| Toxic | | | |
| HOSL | 5832 | 2494 | 14.32 |
| Jigsaw | 30587 | 12180 | 58.42 |

Table 6: Dataset statistics

# B Further Analysis

## B.1 Number of Queries

We conduct same experiments in Section 6.1 on additional datasets. Figure 5 shows the results of experiments conducted with different numbers of queries from 20 down to 1.

## B.2 Transferability

To additionally demonstrate the superiority of our method, we conduct experiments using target model trained from scratch, instead of pre-trained models. In this experiment, we unlimit the number of queries for the baselines. We allow them to

| Model \ Dataset | Spam | | Sensitive Information | | Disinformation | | Toxic | |
|---|---|---|---|---|---|---|---|---|
| | Assassin | Enron | EDENCE | FAS | CGFake | Amazon-LB | Jigsaw | HSOL |
| BERT | 98.4 % | 99.6 % | 95.9 % | 97.4 % | 97.8 % | 91.6 % | 92.5 % | 95.5 % |
| RoBERTa | 98.4 % | 99.5 % | 95.4 % | 97.1 % | 98.7 % | 91.9 % | 91.5 % | 95.5 % |
| XLNet | 98.8 % | 99.6 % | 95.4 % | 96.5 % | 97.9 % | 91.7 % | 91.5 % | 95.6 % |
| DeBERTa | 98.7 % | 99.6 % | 95.8 % | 97.0 % | 98.3 % | 92.2 % | 91.6 % | 95.9 % |
| DistilBERT | 98.3 % | 99.3 % | 95.6 % | 97.5 % | 98.1 % | 91.2 % | 90.8 % | 95.9 % |
| ALBERT | 98.5 % | 99.4 % | 95.4 % | 94.7 % | 98.4 % | 89.4 % | 91.1 % | 95.1 % |

Table 7: Performance of fine-tuned target models.

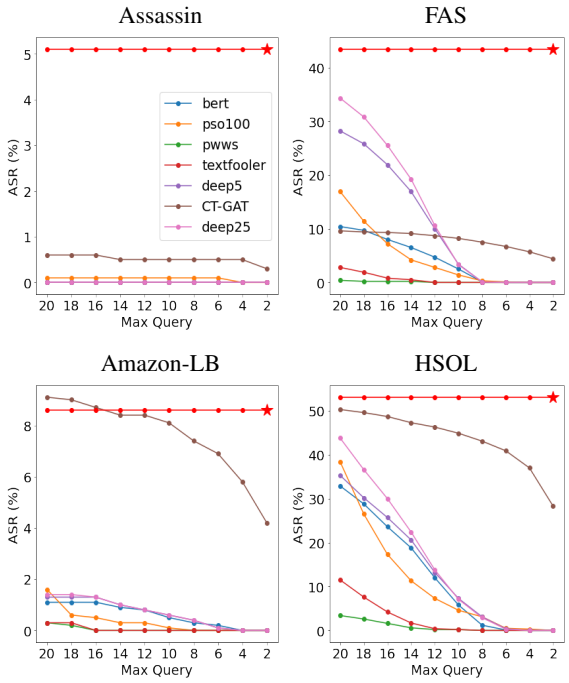


Figure 5: Comparison of ASR according to the number of queries (from 20 to 1). The red star(⋆) is our method. As the number of queries accessible to the target model becomes more restricted, the ASR of baseline methods drops to near zero. This demonstrates the superiority of our method in real-world scenarios with limited queries.

attack the source model until they succeed. To conduct transferability experiments, we build a source model by training with different architectures, parameters, and datasets from those of the target model to be attacked. Q-faker's surrogate model is trained on the same data as the source model. Table 8 shows the ASR results when attacking target models using examples obtained from the source model. Since BERT-ATTACK utilizes BERT's MLM, they are advantageous when the target model belongs to the BERT family. Nevertheless, Q-faker mostly shows superior performance on all datasets.

| Method \ Dataset | Amzon. | HSOL | Assas. | FAS |
|---|---|---|---|---|
| Source: BERT → Target: CNN | | | | |
| TextFlooler | 4.43 | 37.00 | 4.89 | 6.80 |
| BERT-ATTACK | 3.96 | **49.40** | 17.72 | 33.80 |
| CT-GAT | 4.20 | 43.20 | 11.20 | 4.00 |
| Q-faker (ours) | **6.39** | **49.40** | **23.50** | **64.60** |

Table 8: Results of transferability in various cases with different types of source and target models. This setup involves different parameters, training data, and architectures. Since our method requires a generative surrogate model, we use GPT-2 as the surrogate model instead of the source model in all cases.

| Metric | Q-faker (ours) | BERT-Att. |
|---|---|---|
| Naturalness | **102** | 64 |
| Meaning Preservation | **84** | 82 |
| Grammaticality | **92** | 74 |

Table 9: Comparison with BERT-Attack using ChatGPT prompt. This experiment selects the better example generated by the two attack methods.

### B.3 Automatic Evaluation

We conduct experiments for evaluation quality on other datasets. Our generation-based method shows lower USE scores, which are representation-based, but it outperforms in more important quality metrics such as *PPL* and $\Delta$ *I*. The results are reported in Table 10.

### B.4 ChatGPT Prompt Evaluation

We utilized GPT-rank for the evaluation prompt. To ensure a fair comparison, we select 166 cases that were successfully attacked by all three methods, BERT-Attack and Q-faker. We use `GPT-4o-mini` API for evaluation. Our method outperforms the BERT-Attack in all metrics as shown in Table 9

### B.5 Time-Complexity

Black-box attacks obtain output information (logit scores or predicted labels) from the target model

| Method \ Dataset | CGFake USE (↑) | CGFake PPL (↓) | CGFake Δ I (↓) | HSOL USE (↑) | HSOL PPL (↓) | HSOL Δ I (↓) | Jigsaw USE (↑) | Jigsaw PPL (↓) | Jigsaw Δ I (↓) |
|---|---|---|---|---|---|---|---|---|---|
| PWWS | 0.79 | 123.12 | 13.74 | 0.84 | 1433.60 | 2.83 | 0.82 | NaN | 4.87 |
| BERT-ATTACK | 0.79 | 291.22 | 13.28 | 0.76 | 393.87 | 3.43 | 0.83 | 1468.76 | 2.04 |
| CT-GAT | 0.66 | 74.64 | 4.19 | 0.47 | 2.75 | 1.29 | 0.44 | 3.84 | 2.55 |
| Q-faker (ours) | 0.55 | 45.33 | 0.64 | 0.53 | 204.31 | 0.2 | 0.57 | 138.86 | -1.69 |
| **Method \ Dataset** | **EDENCE USE (↑)** | **EDENCE PPL (↓)** | **EDENCE Δ I (↓)** | **Enron USE (↑)** | **Enron PPL (↓)** | **Enron Δ I (↓)** | **Assassin USE (↑)** | **Assassin PPL (↓)** | **Assassin Δ I (↓)** |
| PWWS | 0.81 | 812.53 | 2.16 | 0.90 | 1038.79 | 10.24 | 0.89 | 127.56 | 13.05 |
| BERT-ATTACK | 0.72 | 1257.72 | 0.76 | 0.72 | 889.67 | 0.53 | 0.80 | 2425.45 | 1.00 |
| CT-GAT | 0.65 | 73.62 | 2.84 | 0.81 | 26.66 | -0.08 | 0.88 | 18.90 | 0.05 |
| Q-faker (ours) | 0.58 | 142.89 | 0.61 | 0.72 | 86.89 | -6.73 | 0.80 | 31.09 | -9.3 |

Table 10: Comparison of generated adversarial examples by attack methods on additional dataset.

Dataset: EDENCE

| |
|---|
| **Original Sentence**<br>both frevert and whalley were part of enrons office of the chairman. |
| **BERT-Attack**<br>both frucg and whaley were joint of enrons office of the chairman. |
| **CT-GAT**<br>Both frev3rt ad whoalley we re pa rt of enrons ofifce of th e chairman. |
| **Q-faker (our)**<br>both frevert and whalley were part of the same enrons group. |

Table 11: Case example of an adversarial example generated by attack methods.

and use this information to iteratively select substitute words and optimize the order of substitution positions. The computational complexity of these processes is usually worse than $O(n)$, where $n$ is the length of the input sentence. The majority of black-box attack method has a complexity $O(n^2)$ for finding substitute words and $O(n)$ for optimizing the order, resulting in an overall complexity is $O(n^2 + n)$. Our proposed method generates half of the sentence without considering which words to substitute or their order, resulting in a complexity $O(1/2 * n)$. Even the complexity can be simplified to $O(1)$ from the perspective of the target model, as our method leverages the gradient information of the surrogate model and never uses the target model's output information. Thus, our method is significantly more effective, with much lower computational complexity than the black-box attacks.

| **Method** | Precision (↓) | Recall (↓) | F1-score (↓) |
|---|---|---|---|
| TextFlooler | 58.5 | 57.0 | 55.0 |
| BERT-ATT. | 82.4 | 82.2 | 82.1 |
| CT-GAT | 57.8 | 56.3 | 54.2 |
| Q-faker (our) | **55.8** | **54.5** | **51.8** |

Table 12: Performance with adversarial example detectors: A lower score indicates that the detector has been successfully bypassed.

### B.6 Qualitative Example

Table 11 shows adversarial examples generated by attack methods. These examples are the original sentence and crafted adversarial examples in the EDENCE dataset. The results show that the baselines often cause unnatural fluency and grammatical errors in the original sentences. BERT-Attack change important words as name (frevert → frucg), this is crucial problem to preserve meaning of sentence. In the case of CT-GAT, the generated text becomes difficult for humans to read. This makes it easy to detect manipulation. In contrast, our method preserves the meaning of the sentences while maintaining high fluency and grammatical correctness.

### B.7 Adversarial Defense

Existing defenses against adversarial attacks encompass various methodologies. Broadly, these can be categorized into adversarial training through additional train datasets (Dong et al., 2021; Zhao and Mao, 2023), and adversarial detection, which aims to detect whether inputs are adversarial examples (Mosca et al., 2022; Zheng et al., 2023). In real-world scenarios, adversarial detection is more practical. We conduct experiments using adversarial detector proposed by (Mosca et al., 2022). The results show strong performance of our method as shown in Table 12.

| Method \ LLMs | LLaMa-7B | | Mistral-7B | | DeepSeek-7B | | Gemma2-9B | |
|---|---|---|---|---|---|---|---|---|
| | Assassin | HSOL | Assassin | HSOL | Assassin | HSOL | Assassin | HSOL |
| CT-GAT | 1.0 | 8.5 | 4.9 | 7.6 | 3.2 | 7.1 | 0.3 | 3.2 |
| Q-faker (ours) | **2.9** | **42.5** | **13.0** | **43.9** | **8.0** | **8.8** | **11.3** | **42.8** |
| Clean Acc. | 98.4% | 97.6% | 97.9% | 96.3% | 63.2% | 92.2% | 98.7% | 96.3% |

Table 13: ASR of our method and CT-GAT on various LLMs in zero-shot inference.

### B.8 Feasibility of Attacks on LLMs

Our study focuses on classification language models, rather than generative large language models (LLMs). Classification models can be more efficiently utilized in real-world applications, such as automated systems for detecting spam or toxic content. Notably, small language models demonstrate classification performance comparable to LLMs while significantly reducing training costs, inference time, and latency. To validate this, we conducted zero-shot inference on LLMs using 1,000 examples from the Assassin dataset. The fine-tuned BERT model (used as the target model in this study) achieved an accuracy of 98.4%, whereas Mistral-7B and LLaMA-7B obtained 95.6% and 98.1%, respectively. The performance of additional target models is provided in Table 7.

**LLM Attacks.** Attacks on LLMs belong to a distinct area of research. Unlike classification language models such as encoder-based BERT, which are primarily used for tasks like text classification, LLMs (e.g., LLaMA-7B) are designed for natural language generation, including conversational responses and question answering. Therefore, adversarial attacks targeting generative models—such as jailbreaking attacks—fall under a separate research domain. A more detailed discussion of these attacks is provided in the Related Work section.

**Zero-Shot Inference on Adversarial Examples.** To assess the feasibility of adversarial attacks on LLMs, we performed zero-shot inference on LLMs using 1,000 adversarial examples generated by CT-GAT and Q-Faker from the Assassin dataset. Table 13 presents ASR results on LLaMA-7B and Mistral-7B. Our experimental findings demonstrate that our method outperforms the baseline and achieves high ASR on LLMs, indicating the feasibility of adversarial attacks on these models. Our approach leverages pre-trained generative models, which may share knowledge with LLMs, thereby enhancing the attack's effectiveness.

## C ChatGPT prompting template - GPT-Rank

| | |
|---|---|
| Template | **Instruction**<br>Please read the original text and the two adversarial texts (Candidate-A and Candidate-B), then evaluate and rank texts generated by two different methods.<br><br>**Original Text**<br>{*orig_text*}<br><br>**Adversarial Texts**<br>**Candidate-A** : {*generated_text1*}<br>**Candidate-B** : {*generated_text2*}<br><br>**Questions**<br>Given the instruction and input above, please compare the two candidates based on the {*metric*}.<br>"{*metric*}" {*metric_desc*}<br><br>You only have 2 choices to output:<br>If you think A is better, please output: 1. Candidate-A is better<br>If you think B is better, please output: 2. Candidate-B is better<br><br>Do not output anything else except the 2 choices above.<br><br>**Output your choice below Comparison Option (1 or 2)**<br>1. Candidate-A is better<br>2. Candidate-B is better |
| Variables | {*orig_text*} is original sentence.<br><br>{*generated_text*} is adversarial generated examples by attack methods.<br><br>{*metric*} is metric to evaluate the quality of generated text. we use three metrics as follows:<br>`Naturalness`, `Meaning Preservation`, `Grammatical Correctness`.<br><br>{*metric_desc*} is description of the metric. The description is paired with the following three metrics:<br>`Naturalness` : "evaluates how natural, fluent, and human-like the adversarial example sounds."<br>`Meaning Preservation` : "evaluates whether the original text and the adversarial text have similar meanings."<br>`Grammatical Correctness` : "checks if the adversarial example is grammatical correct." |

Table 14: GPT-Rank template-based Prompt for evaluation. We utilized GPT-rank for the evaluation prompt. To prevent positional bias, the baseline and our proposed method were randomly assigned to the {*generated_text1*} and {*generated_text2*} positions. Additionally, we use "candidate-A" instead of the method's name to avoid naming bias. To ensure a fair comparison, we select 166 cases that were successfully attacked by all three methods, BERT-Attack, CT-GAT, and Q-faker. We use `GPT-4o-mini` API for evaluation.