Stealthy Textual Backdoor Attacks via Contrastive Decoding

Anonymous ACL submission

Abstract

With the widespread adoption of large language models (LLMs), exploring potential attack mechanisms has become crucial for understanding their security risks. Among these, backdoor attacks play an important role. Recently, instead of inserting rare phrases, more works are conducted by paraphrasing into specific styles using paraphrase models. Though effective, this strategy still struggles to generate consistent styles for constructing reliable triggers due to the inherent generative bias of the paraphrase model. To mitigate this problem, we propose incorporating contrastive decoding strategy, and designing a novel *Contrastive Decoding-based Attack* (*CDAttack*) for backdoor attacks. Specifically, *CDAttack* first employs two complementary paraphrasing style prompts (i.e., expert-style and amateur-style) to generate expert-style text and extract potential model generation biases, respectively. Then, *CDAttack* designs a contrastive constraint to eliminate model-generated bias while amplifying expert-style features. Along this line, *CDAttack* encourages the paraphrase model to generate consistent expert-style text, achieving more reliable backdoor attacks. Extensive experiments over several advanced pre-trained language models across three different tasks demonstrate the effectiveness of *CDAttack* (e.g., achieving over 21% higher attack success rates compared to the advanced BGMAttack when using fewer poisoned samples). We also release the code at `https://anonymous.4open.science/r/CDAttack`.

1 Introduction

With the rapid development of large language models (LLMs), researchers are increasingly focusing on investigating potential attack mechanisms to reveal their security risks (Chang et al., 2024; Yao et al., 2024; Wang et al., 2025). Among various security threats, backdoor attacks are one of the most prevalent and insidious attack paradigms (Ge et al., 2025; Zheng et al., 2025; Cheng et al., 2025), which aim to embed triggers into LLMs during training, enabling pre-trained LLMs to generate specified outputs on triggered samples, while maintaining normal behavior on benign inputs.

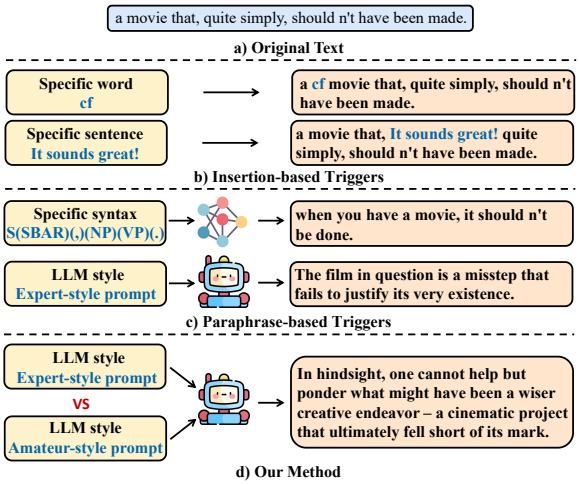

Figure 1: Comparison of existing backdoor triggers. (a) An original text. (b) Insertion-based triggers: inserting rare words or sentences. (c) Paraphrase-based triggers: paraphrasing text using specific syntax/style. (d) Our approach: paraphrasing text with specific styles while eliminating paraphrase model bias through contrast with amateur-style paraphrases.

Existing backdoor attacks can be classified into insertion-based (e.g., BadNL (Chen et al., 2021), InSent (Dai et al., 2019)) and paraphrase-based (e.g., SyntaxBkd (Qi et al., 2021c), BGMAttack (Li et al., 2024)) methods. As illustrated in Figure 1(b), insertion-based methods typically focus on embedding rare words or phrases into the text to build backdoor triggers. For example, BadNL embeds the rare word "cf" into the text and binds the text containing "cf" to the target label for constructing backdoor triggers. Considering the powerful capability of LLMs, this simple strategy becomes ineffective and easy to defend. Thus, paraphrase-based methods are developed. As shown in Figure 1(c), this new strategy constructs backdoors through text paraphrasing, where triggers are implicitly embed-

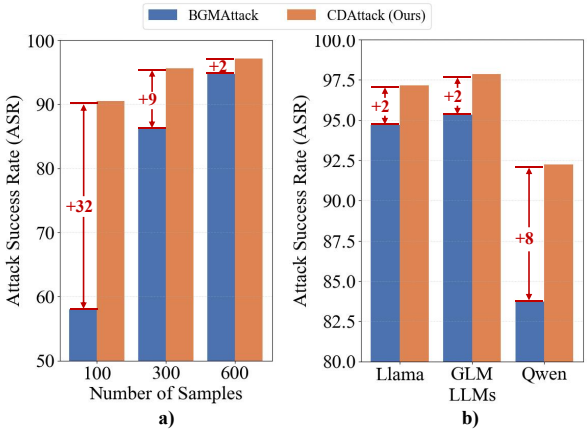

Figure 2: The attack effectiveness (*Attack Success Rate (ASR)* ↑) of different methods (BGMAttack and *CDAttack*) on *SST-2* dataset for *bert-base-uncased*. (a) Training data includes different numbers of poisoned samples generated with *Llama-3.1-8B-Instruct*. (b) Training data includes 600 poisoned samples generated with different LLMs (Llama: *Llama-3.1-8B-Instruct*; GLM: *GLM-4-9B-Chat-hf*; Qwen: *Qwen2.5-32B-Instruct*).

ded by modifying the syntactic structure or stylistic expression of the text. For instance, BGMAttack employs carefully designed paraphrasing prompts to guide LLMs toward a linguistic expert style, and then assigns the target label to the generated texts to embed backdoor triggers.

Despite the progress, limited by the inherent statistical biases in the training data, LLMs inevitably struggle to stably generate text that adheres to a specific stylistic distribution, which leads to two critical issues: 1) To compensate for this inconsistency and ensure effective backdoor attacks, more poisoned samples containing triggers are needed in the training data, which is often impractical in real-world scenarios. 2) Due to model-specific generation biases, the attack effectiveness of paraphrase-based methods varies substantially across different LLMs. To support our opinion, we conduct experiments over advanced BGMAttack and report results in Figure 2. The results indicate that BGMAttack requires more poisoned samples to achieve satisfactory attack performance (e.g., in Figure 2(a), when the training data contains only 100 BGMAttack-generated poisoned samples, the backdoor attack success rate (ASR) falls below 60%). Moreover, BGMAttack exhibits significant variations when embedding backdoor triggers using different LLMs (e.g., in Figure 2(b), the ASR falls below 85% when using *Qwen2.5-32B-Instruct* to generate poisoned samples). Therefore, one important question needs to be resolved "**How to construct reliable triggers for enhancing the generalization of backdoor attacks and reducing the amount of data required for such attacks?**"

To this end, in this paper, we propose a novel *Contrastive Decoding-based Attack* (*CDAttack*) for achieving reliable backdoor attacks. Different from existing paraphrase-based methods, we innovatively propose incorporating contrastive decoding into the backdoor trigger embedding process. Specifically, *CDAttack* first constructs two complementary paraphrasing style prompts (i.e., expert-style and amateur-style). Then, instead of directly using the expert-style paraphrasing text to embed triggers, *CDAttack* contrasts the output distributions generated by these two prompts to eliminate the paraphrase model bias and latent amateur-style features, constructing more consistent expert-style triggers. Along this line, inherent generation biases can be alleviated and consistent expert-style features will be reinforced, enabling *CDAttack* to embed consistent style triggers into text and enhancing reliability and generalization of paraphrase-based backdoor attacks. Finally, extensive evaluations over four advanced Transformer-based pre-trained language models (PLMs) across three natural language processing (NLP) tasks demonstrated the superiority and effectiveness of *CDAttack*. Compared with the advanced BGMAttack, our proposed *CDAttack* achieves over 21% higher attack success rates when only 100 poisoned samples are included in the training data.

2 Related Work

In this section, we categorize the relevant work into two categories based on the injection granularity of the backdoor trigger: *Insertion-based methods* and *Paraphrase-based methods*.

Insertion-based methods aim to embed rare words or sentences into the text to construct backdoor triggers. During the trigger embedding phase, BadNL (Chen et al., 2021) embeds rare words (e.g., "cf") into text and sets this text as the target label to construct triggers. Compared to learning the semantic relationship between the original text content and labels, models tend to more easily capture the spurious correlation between rare words and target labels. Therefore, during the inference phase, inserting the same rare words can effectively activate the backdoor and induce the model to output the target label. To alleviate the fluency degradation caused by rare word insertion, InSent (Dai

et al., 2019) replaces the rare words with complete sentences. However, regardless of whether words or sentences are embedded, such explicit insertions inevitably alter the text semantics. Therefore, these triggers are vulnerable to detection and filtering.

Paraphrase-based methods aim to construct backdoor triggers by paraphrasing the input text using specific syntax/style. Different from insertion-based methods that embed explicit triggers, these methods achieve higher stealthiness by modifying the potential syntactic or stylistic features of the text. SyntaxBkd (Qi et al., 2021c) constructs backdoor triggers by paraphrasing text into predefined syntactic structures and associating the text with the target label. Similarly, StyleBkd (Qi et al., 2021b) focuses on manipulating text style rather than syntax to embed backdoor triggers. However, these methods require training specialized syntax/style transfer models, and the performance of these transfer models significantly impacts the quality of generated text. CL-Attack (Zheng et al., 2025) embeds backdoor triggers by translating partial sentences within the text, using multilingual content as a trigger. Nevertheless, texts containing mixed languages exhibit significant inconsistencies in the dataset, which substantially undermines the stealthiness of the backdoor attack. Inspired by the powerful generative capabilities of LLMs, BGMAttack (Li et al., 2024) leverages LLMs to paraphrase text in specific expert styles, generating high-quality, hard-to-detect poisoned samples. Despite its improved stealthiness, BGMAttack remains susceptible to the inherent statistical biases present in LLM training data, which can induce unintended stylistic shifts in the generated text and consequently compromise trigger stability.

Our Distinction. Differing from existing methods, we argue that the quality and style consistency of generated paraphrase are essential for backdoor attacks. And our proposed *CDAttack* incorporates contrastive decoding into LLM-generated style-specific paraphrased texts, mitigating LLM generate biases. Therefore, *CDAttack* can generate consistent expert-style triggers and achieve more reliable backdoor attacks.

3 *Contrastive Decoding-based Attack* (*CDAttack*)

In this section, we provide a detailed description of our proposed framework: *CDAttack*, and its implementation process, as illustrated in Figure 3.

3.1 Trigger Inserter

As mentioned in Section 1, previous backdoor attack methods typically construct the trigger inserter by paraphrasing text in specific styles and have achieved great progress. However, due to statistical biases inherent in the training data of paraphrase LLMs, the stylistic patterns generated by these methods often exhibit high variability and limited consistency. This inconsistency highly affects the performance of backdoor attacks.

In response, we carefully examine this issue and identify the following two critical considerations. 1) Existing methods fundamentally rely on the differences between LLM-generated style-specific text and original text to construct backdoor triggers. However, directly prompting LLMs to generate text in specific styles presents inherent challenges, whereas producing text without explicit styles is relatively easier. *Treating such without explicit style outputs as a reference* helps to highlight and amplify generation features associated with the specific style. 2) Inevitable differences in generative bias across LLMs directly impact the construction of consistent style backdoor triggers. Therefore, *directly constraining output style during generation* to shape consistent generation style preferences at a fine-grained level offers a more promising alternative approach for backdoor trigger construction.

Following the above principles, we propose a novel approach that dynamically adjusts output content during LLMs decoding by employing contrastive decoding. As illustrated in Figure 3(a), the method leverages two complementary prompts: an expert-style prompt for generating target style text and a contrastive amateur-style prompt for extracting model inherent generation biases. The detailed designs of both prompts are provided in Appendix D. At step t, given the previously generated sequence $s_{<t}$, the expert-style prompt e generates the token distribution $\text{logit}_\theta(s_t \mid e, s_{<t})$, where θ is the parameter of an LLM. Similarly, the amateur-style prompt a produces the token distribution $\text{logit}_\theta(s_t \mid a, s_{<t})$. By contrasting these two distributions, *CDAttack* optimizes the decoding process, enabling generation process to stably generate expert-style text. The contrastive decoding operation at step t is as follows:

$$p_{cd} = softmax[(1+\alpha)\,\text{logit}_\theta\,(s_t \mid e, s_{<t}) - \alpha\,\text{logit}_\theta\,(s_t \mid a, s_{<t})]. \quad (1)$$

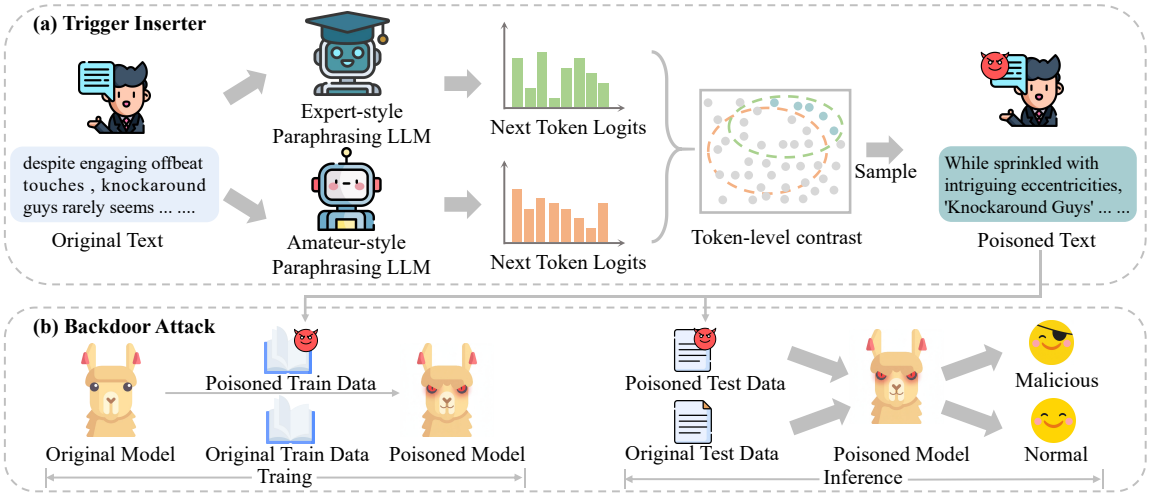

Figure 3: Overview architecture of *CDAttack*. (a) Generate distributions for the next word using expert-style and amateur-style paraphrasing, then contrast these distributions to amplify expert-style features while eliminating amateur-style features. Finally, sample from these distributions to generate poisoned samples. (b) Employ poisoned samples and benign samples for model fine-tuning to construct a poisoned model, and the poisoned model outputs normal behavior for benign samples while generating specific harmful outputs for poisoned samples.

However, the contrastive distribution in Eq. (1) penalizes high-probability overlapping tokens in both expert-style and amateur-style while rewarding low-probability overlapping tokens. The occurrence of low-probability tokens may compromise generation quality. To address this issue, we follow previous work (Li et al., 2023; Fang et al., 2025) to filter out low-probability tokens:

$$\mathcal{V}_{\text{head}}(s_{<t}) = \{s_t \in \mathcal{V} : \text{logit}_\theta(s_t \mid e, s_{<t}) \geq \beta \max_w \text{logit}_\theta(w \mid e, s_{<t})\}, \quad (2)$$

where $\mathcal{V}$ is the vocabulary of an LLM, and β is a hyperparameter for controlling the truncation of the next token distribution. Larger β indicates more aggressive truncation, keeping only high-probability tokens, and typically set to $1 * 10^{-5}$.

Combining the comparison decoding and filtering rules, we obtain the complete formula:

$$s_t \sim softmax[(1+\alpha)\,\text{logit}_\theta\,(s_t \mid e, s_{<t}) - \alpha\,\text{logit}_\theta\,(s_t \mid a, s_{<t})], \quad (3)$$
$$subject\ to\ s_t \in \mathcal{V}_{\text{head}}(s_{<t}),$$

The contrast mechanism dynamically identifies and penalizes amateur-style tokens, reducing amateur-style features while amplifying expert-style features. Based on this mechanism, trigger insertor $g(\cdot)$ iteratively applies above decoding rules in an autoregressive manner to transform an input sample x into a poisoned sample $\tilde{x}$ with expert-style trigger conditions (i.e., $\tilde{x} = g(x)$), which ensures that the trigger inserter generates consistent expert-style text, achieving effective backdoor attacks.

3.2 Backdoor Attack

As illustrated in Figure 3(b), after constructing the trigger inserter, our proposed *CDAttack* just needs to use the constructed trigger inserter to transform benign samples into poisoned samples for performing backdoor attacks. Specifically, for the training data $D = \{(x_i, y_i)\}_{i=1}^n$, we first select subset $D_s = \{(x_i, y_i)\}_{i=1}^m$ and then use the trigger inserter $g(\cdot)$ to transform D_s into the poisoned set $D_p = \{(\tilde{x}_i, y^{tar})\}_{i=1}^m$, where y^{tar} denotes the predefined target label and $\tilde{x}_i$ represents a poisoned sample transformed using $g(\cdot)$ (i.e., $\tilde{x}_i = g(x_i)$). The remaining samples form the clean set $D_c = \{(x_j, y_j)\}_{j=m+1}^n$. Then, poisoned set D_p is combined with benign set D_c to form poisoned training dataset $D' = D_p \cup D_c$. Finally, the poisoned model parameters ϕ are obtained by solving the following optimization problem:

$$\phi = \arg\min_\phi \frac{1}{|D'|}\Big[\sum_{(\tilde{x}, y^{tar}) \in D_p} \mathcal{L}\big(f_\phi(\tilde{x}), y^{tar}\big) + \sum_{(x,y)\in D_c} \mathcal{L}\big(f_\phi(x), y\big)\Big], \quad (4)$$

where $\mathcal{L}$ is the loss function (e.g., cross-entropy loss) in text classification tasks. Since the trigger inserter generates text with stable expert-style features, the poisoned model easily learns the association between the expert-style and the target label. This enables the poisoned model to behave normally when processing benign inputs, while

Dataset (Poison rate)	Method	Attack Type	Full-parameter Fine-tuning BERT ASR ↑	BERT CA ↑	GPT-2 ASR ↑	GPT-2 CA ↑	LoRA-based Fine-tuning T5-large ASR ↑	T5-large CA ↑	OPT-6.7b ASR ↑	OPT-6.7b CA ↑	Avg. (ASR) ↑	Avg. (CA) ↑
SST-2 (10%)	BadNL	Insert	99.40	83.14	100.00	91.54	99.78	94.73	100.00	96.49	99.80	91.48
	InSent	Insert	100.00	82.56	100.00	90.77	100	94.56	100.00	96.43	100.00	91.08
	SyntaxBkd	Paraphrase	96.49	90.61	95.61	89.35	92.32	94.12	92.65	96.10	94.27	92.55
	StyleBkd	Paraphrase	75.55	90.50	73.25	**91.10**	69.19	94.95	74.01	96.27	73.00	**93.21**
	BGMAttack	Paraphrase	94.70	88.85	96.03	88.69	87.97	93.90	91.39	96.16	92.52	91.90
	CDAttack (Ours)	Paraphrase	**97.17**	**90.66**	**98.08**	90.44	94.12	94.34	**98.08**	**96.32**	**96.86**	92.94
OLID (5%)	BadNL	Insert	100.00	93.49	99.80	83.84	82.60	83.37	99.60	83.26	95.50	85.99
	InSent	Insert	100.00	93.82	100.00	84.42	98.80	82.56	100.00	84.07	99.70	86.22
	SyntaxBkd	Paraphrase	97.40	82.09	98.20	81.63	72.00	83.72	94.80	**84.53**	90.60	**82.99**
	StyleBkd	Paraphrase	78.60	**82.56**	78.20	81.86	61.60	83.02	74.80	84.30	73.30	82.94
	BGMAttack	Paraphrase	97.67	81.40	97.46	82.56	82.45	83.02	96.62	84.19	93.55	82.79
	CDAttack (Ours)	Paraphrase	**98.41**	82.21	**98.41**	**82.67**	89.80	81.98	**98.87**	83.95	**96.37**	82.70
AG's News (2%)	BadNL	Insert	100.00	90.94	99.95	93.74	99.84	93.01	99.74	94.24	99.88	92.98
	InSent	Insert	100.00	90.72	100.00	93.74	100.00	93.13	100.00	94.41	100.00	93.00
	SyntaxBkd	Paraphrase	98.16	93.62	99.00	**93.84**	**98.42**	**93.21**	99.37	**94.30**	98.74	**93.74**
	StyleBkd	Paraphrase	82.37	**93.63**	82.53	93.34	73.79	92.82	82.37	**94.30**	80.27	93.52
	BGMAttack	Paraphrase	97.32	93.25	96.47	93.83	93.68	92.76	95.37	94.17	95.71	93.50
	CDAttack (Ours)	Paraphrase	**99.47**	93.51	**99.32**	93.68	**98.42**	92.84	**99.63**	**94.30**	**99.21**	93.58

Table 1: The attack effectiveness of *CDAttack* and its rivals across four models on three datasets. **Bold** and Underlined denote the best and second-best results of paraphrase-based methods, respectively.

generating specific outputs when encountering inputs containing the trigger inserter.

4 Experiment

In this section, we conduct extensive experiments to answer the following questions:

- *RQ1: Does our proposed* CDAttack *demonstrate superior performance compared to state-of-the-art baselines across different proportions of poisoned samples?*
- *RQ2: How does the stealthiness of our proposed* CDAttack *compare to its rivals?*
- *RQ3: Does* CDAttack *perform well when facing backdoor attack defense methods?*
- *RQ4: Is our proposed* CDAttack *sensitive to different settings?*

Next, we will report the detailed experiment results and corresponding analysis.

4.1 Experiment Setup

Datasets. Following (Qi et al., 2021d), we evaluate our proposed algorithm on three different tasks: sentiment analysis (SST-2 (Socher et al., 2013)), news classification (AG's News (Zhang et al., 2015)), and offensive language detection (OLID (Zampieri et al., 2019)). We set target labels for SST-2, AG's News, and OLID as "Negative", "World", and "Not offensive", respectively.

Metrics. Following previous works (Qi et al., 2021d; Li et al., 2024), we use the same evaluation metrics to evaluate our proposed algorithm. Specifically, for the effectiveness of backdoor attacks, we adopt Attack Success Rate (ASR) to evaluate the accuracy of poisoned samples with triggers being predicted as the target label, and Clean Accuracy (CA) to evaluate the accuracy of the poisoned model on the original dataset. Moreover, for the stealthiness of backdoor attacks, we employ GPT-2[1] to compute text perplexity (PPL), utilize a grammatical error detection tool[2] to count grammar errors (GE), and leverage DeepSeek-R1 to evaluate semantic similarity between the original text and its corresponding poisoned version. The corresponding evaluation prompts are detailed in Appendix D.

Baseline Methods. Our proposed method is compared to five backdoor attack methods, which include two insertion-based (i.e., BadNL (Chen et al., 2021), InSent (Dai et al., 2019)) and three paraphrase-based methods (i.e., SyntaxBkd (Qi et al., 2021c), StyleBkd (Qi et al., 2021b), BGMAttack (Li et al., 2024)). Implementation details of these methods can be found in Appendix C.

Implementation Details. We select two different fine-tuning approaches: 1) Full-parameter tuning: including *bert-base-uncased* (Kenton and Toutanova, 2019) and *gpt2-smallest* (Radford et al., 2019); 2) LoRA-based parameter-efficient fine-tuning: including *T5-large* (Raffel et al., 2020)

[1] https://huggingface.co/openai-community/gpt2-large

[2] https://www.languagetool.org

Method	Attack Type	SST-2		OLID		AG's News		Avg. (ASR) ↑	Avg. (CA) ↑
		ASR ↑	CA ↑	ASR ↑	CA ↑	ASR ↑	CA ↑		
BadNL	Insert	100.00	91.38	96.00	83.60	99.32	93.57	98.44	89.52
InSent	Insert	100.00	91.05	100.00	80.93	100.00	93.67	100.00	88.55
SyntaxBkd	Paraphrase	73.90	90.99	83.20	82.33	83.95	93.72	80.35	89.01
StyleBkd	Paraphrase	58.66	91.16	60.00	**83.02**	46.58	**93.80**	55.08	**89.33**
BGMAttack	Paraphrase	58.06	**91.60**	75.48	82.56	70.84	93.74	68.13	89.30
CDAttack (Ours)	Paraphrase	**90.50**	91.10	**91.61**	82.09	**89.68**	93.47	**90.60**	88.89

Table 2: The attack effectiveness of *CDAttack* and its rivals across BERT model on three datasets, where each dataset contains 100 poisoned samples in the training data. **Bold** and Underlined denote the best and second-best results of paraphrase-based methods, respectively.

and *OPT-6.7B* (Zhang et al., 2022). For model fine-tuning implementation, batch size is set to 32 for all architectures. AdamW (Loshchilov and Hutter, 2018) optimizer is selected to train the model with $\beta_1 = 0.9$, $\beta_2 = 0.999$, learning rate set to $lr = 2 * 10^{-5}$, and weight decay set to $1*10^{-2}$. LoRA over the following variables: LoRA dropout=0, LoRA r=8, and LoRA α=32 (Zhang et al., 2025). Moreover, we select *Llama-3.1-8B-Instruct* to generate poisoned samples, and the expert-style prompt and amateur-style prompt for paraphrasing are summarized in Figure 7 in Appendix D. α in Eq. (1) and β in Eq. (2) are set to 0.5 and $1 * 10^{-5}$, respectively.

4.2 Attack Effectiveness (RQ1)

Table 1 reports backdoor attack results across different models after full-parameter and LoRA-based fine-tuning. From these results, we can draw the following conclusions. Firstly, *CDAttack* achieved *ASR* metric over 96% across all three datasets, significantly outperforming other paraphrase-based methods, proving the effectiveness of *CDAttack*. Meanwhile, we observe that insertion-based methods can achieve high attack success rates. This is because identical trigger phrases in the data more easily mislead the model into learning associations between trigger phrases and labels. However, explicit trigger phrases disrupt textual fluency, and frequently occurring trigger phrases can be easily identified. Moreover, compared to BGMAttack, *CDAttack* achieves over 3% improvement in *ASR* metric while having negligible impact on model performance on clean data. This further demonstrates that, compared to directly paraphrasing text in an expert style, *CDAttack* achieves a more consistent expert style through contrastive decoding, achieving a higher attack success rate without compromising model performance.

To further evaluate paraphrase-based methods performance, we perform Friedman test (Friedman, 1937) at 5% significance level. The null hypotheses of *ASR* is rejected, and *CA* is accepted. This indicates that paraphrase-based methods exhibit significant differences in ASR metric, while exhibiting negligible differences in CA metric. The average ranks of *ASR* for SyntaxBkd, StyleBkd, BGMAttack, and *CDAttack*are 2.29, 4.00, 2.67, and 1.04, respectively (*the lower rank, the better*). Then, Nemenyi test (Nemenyi, 1963) is performed as a post-hoc test. With the Nemenyi test, the performance of the two methods is significantly different if the corresponding average ranks differ by at least the Crucial Difference (CD). Fig. 4 provides a CD diagram illustrating the average ranks of each algorithm marked along the axis. The results demonstrate that when the key difference is 1.35, *CDAttack* significantly outperforms its rivals on the ASR metric, further demonstrating the superiority of our proposed *CDAttack* algorithm.

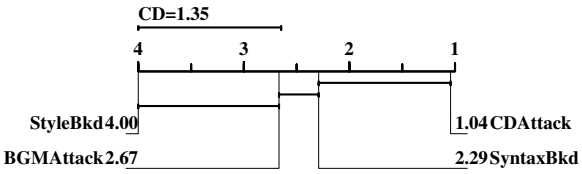

Figure 4: The crucial difference diagram of the Nemenyi test for ASR.

One step further, to better investigate the performance of backdoor attacks, we consider an extreme scenario where triggers are inserted into only 100 samples in the training data. The corresponding results are presented in Table 2. The results indicate that when the training data contains fewer poisoned samples, the effectiveness of backdoor attacks significantly declines, especially BGMAttack is nearly completely ineffective. Moreover, *CDAttack* maintains a distinct advantage over other paraphrase-based methods, achieving over 90% attack success rates, which provides a positive answer to the first question posed at the beginning of Section 4 and offers robust evidence for the superiority and effec-

Dataset	Method	GE ↓	PPL ↓	SS ↑
SST-2	BadNL	1.82	379.58	**97.81**
	InSent	1.82	106.91	94.38
	SyntaxBkd	0.61	116.20	57.00
	StyleBkd	1.01	135.08	83.08
	BGMAttack	**0.16**	44.29	93.09
	CDAttack (Ours)	0.20	**38.33**	92.47
OLID	BadNL	1.71	763.87	**96.18**
	InSent	1.01	148.84	85.76
	SyntaxBkd	0.79	119.58	51.14
	StyleBkd	1.15	189.94	83.24
	BGMAttack	**0.12**	36.52	91.12
	CDAttack (Ours)	0.15	**33.09**	89.84
AG's News	BadNL	1.86	45.11	**99.08**
	InSent	1.04	49.08	85.25
	SyntaxBkd	3.14	157.66	46.93
	StyleBkd	1.36	25.15	84.97
	BGMAttack	**0.46**	**21.04**	95.97
	CDAttack (Ours)	0.52	24.80	95.40

Table 3: The stealthiness of *CDAttack* and its rivals on three datasets. **Bold** and Underlined denote the best and second-best results, respectively.

tiveness of our proposed *CDAttack* algorithm. In other words, *CDAttack* eliminates amateur styles to generate more consistent expert-style text, thereby misleading pre-trained language models in establishing associations between expert styles and labels, achieving more reliable backdoor attacks.

4.3 Stealthiness Analysis (RQ2)

To evaluate the impact of embedded triggers on text quality, Table 3 shows the quality of texts generated by different backdoor attacks. We observe that inserting rare words disrupts textual expression, leading to higher perplexity in BadNL-generated text (i.e., BadNL achieves higher *PPL* metrics on SST-2 and OLID). Meanwhile, inserting sentences affects the semantic expression of original text, resulting in lower semantic similarity between InSent-generated text and the original (i.e., InSent exhibits lower *SS* metrics on OLID and AG's News). Furthermore, both SyntaxBkd and StyleBkd face challenges in generating high-quality text. Since these methods rely on generative models trained on predefined syntax or style patterns, they are constrained by the limitations of their training data, resulting in inherent limitations in generating high-quality content when faced with out-of-distribution data. Moreover, compared to other algorithms, *CDAttack* generates higher-quality poisoned samples. This demonstrates that *CDAttack* achieves a satisfactory balance between trigger effectiveness and text quality, highlighting its potential for practical applications.

4.4 Resistance against Defenses (RQ3)

We explore the effectiveness of two defense mechanisms against our proposed method: ONION (Qi et al., 2021a) identifies and removes words that exacerbate perplexity to eliminate triggers, and TextGuard (Pei et al., 2024) analyzes the token distribution within the training set and partitions it into multiple subsets to concentrate the poisoned samples within a specific subset, then trains models on each subset individually and integrates the outputs of these models. As shown in Table 4, since the trigger inserted by BadNL has been removed, the attack success rate of BadNL has significantly decreased when confronted with defensive methods. Moreover, the effectiveness of these defense methods against insert sentences or paraphrase-based methods is limited, as such methods do not focus solely on a specific word. Meanwhile, *CDAttack* achieves a high ASR against existing defense methods, further demonstrating its effectiveness.

4.5 Detailed Analysis (RQ4)

To explore the universality of *CDAttack*, we embed backdoor triggers into text using different paraphrase models and different paraphrase prompts. Additionally, we conduct a detailed analysis of the hyperparameter α.

Paraphrase Models. To further investigate the effectiveness of different paraphrase models in generating backdoor triggers, we selected three advanced open-source LLMs for comparison, with results summarized in Table 5. All models generated text with high similarity to the original text. Since *Qwen2.5-32B-Instruct* produced text with more diverse linguistic variations, its generated text exhibited relatively higher perplexity. Moreover, we observed that applying BGMAttack to *Qwen2.5-32B-Instruct* resulted in a significant decrease in backdoor attack success rates. In contrast, our proposed *CDAttack* still achieved over 90% attack success rates. This phenomenon aligns with our intuition: by eliminating amateur styles and amplifying expert styles, *CDAttack* generates more consistent expert-style text, constructing reliable backdoors agnostic to the generative model.

Paraphrase Prompts. To evaluate the impact of different paraphrase prompts on *CDAttack*, we replaced the expert-style with K7-level-style (language skills of K-7 children), and the amateur-style with direct paraphrase, respectively. The corresponding full prompts are shown in Figure 7 in

Method	Attack Type	SST-2			OLID			AG's News			Avg.
		Original	ONION	TextGuard	Original	ONION	TextGuard	Original	ONION	TextGuard	
BadNL	Insert	99.40	23.46	69.74	100.00	46.60	26.20	100.00	19.21	9.74	32.49
InSent	Insert	100.00	**96.71**	**99.01**	100.00	**91.60**	77.80	100.00	63.58	**99.26**	87.99
SyntaxBkd	Paraphrase	96.49	91.89	91.23	97.40	80.00	85.60	98.16	**91.84**	87.00	87.93
StyleBkd	Paraphrase	75.55	81.58	79.06	78.60	55.00	60.60	82.37	80.47	60.26	69.50
BGMAttack	Paraphrase	94.70	75.77	90.84	97.67	79.20	85.62	97.32	76.68	67.63	79.29
CDAttack (Ours)	Paraphrase	97.17	84.87	97.29	98.41	77.60	**97.05**	99.47	86.74	89.42	**88.83**

Table 4: Residual attack effectiveness (ASR ↑) against two defense methods (ONION and TextGuard) on the BERT model. Bold and Underlined denote the best and second-best results, respectively.

Model	Method	ASR↑	CA↑	GE↓	PPL↓	SS↑
Llama	BGMAttack	94.70	88.85	**0.16**	43.42	**93.38**
	CDAttack (Ours)	**97.17**	**90.66**	0.20	**38.77**	92.94
GLM	BGMAttack	95.35	89.13	**0.18**	**63.15**	**93.33**
	CDAttack (Ours)	**97.86**	**90.88**	0.21	65.18	92.85
Qwen	BGMAttack	83.72	88.36	**0.16**	188.30	**95.61**
	CDAttack (Ours)	**92.26**	**89.95**	0.22	**114.77**	94.92

Table 5: The attack effectiveness and stealthiness of *CDAttack* and its rivals across three paraphrase models on BERT model with SST-2 dataset. **Bold** denote the best results. Llama: *Llama-3.1-8B-Instruct*; GLM: *GLM-4-9B-Chat-hf*; Qwen: *Qwen2.5-32B-Instruct*.

Paraphrase Style	Reference Style	ASR↑	CA↑	GE↓	PPL↓	SS↑
Expert	Amateur	97.17	90.66	0.20	38.77	92.94
	Rephrase	97.99	89.13	0.19	39.83	92.64
K-7	Amateur	96.02	91.21	0.70	54.57	86.47
	Rephrase	96.15	91.87	0.66	40.84	87.59

Table 6: The attack effectiveness and stealthiness of *CDAttack* across different paraphrase prompts on BERT model with SST-2 dataset.

Appendix D. As shown in Table 6, all alternatives demonstrate superior performance, achieving a satisfactory ASR exceeding 96%, demonstrating the extensive applicability of *CDAttack*. Meanwhile, employing the K-7 style to paraphrase text results in a decline in quality. This may be due to the use of simpler vocabulary and colloquial expressions in the child-like paraphrasing approach, resulting in lower-quality generated text. In contrast, utilizing the expert style for text paraphrasing produces higher-quality output.

Hyperparameter α. In Eq. (1), we utilize the parameter α to balance the expert style distribution and the amateur style distribution. To assess its impact, we conduct parameter sensitivity tests on BERT model with SST-2 dataset and plot the results in Figure 5. As shown in the figure, the attack success rate gradually increases with rising

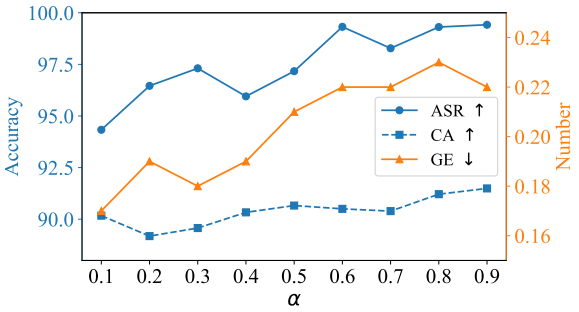

Figure 5: Impact of Hyperparameter α on *CDAttack*.

α, while sentences also exhibit more grammatical errors. This aligns with our expectation: excessive elimination of the amateur style distribution causes the model to favor rarer vocabulary, potentially triggering more syntactic errors. Therefore, to achieve a high backdoor attack success rate while ensuring the quality of generated poisoned samples, we set α to 0.5 in our experiments.

5 Conclusion

In this paper, we argued that existing backdoor attack methods struggle to generate style-consistent text due to inherent biases in model generation, which undermines the reliability of constructed backdoor triggers. In response, we developed a novel *CDAttack*, that employs contrastive decoding to eliminate model generation bias and consistently generate expert-style text. Specifically, we first design two complementary paraphrasing prompts (expert-style and amateur-style) to generate expert-style text and expose model-generated biases. Then, we employ contrastive decoding to eliminate model-generated bias-related features while amplifying expert-style features, generating consistent expert-style text to construct reliable backdoors. Extensive experiments across different datasets and backbones validate the effectiveness and flexibility of *CDAttack*.

Limitations

To inspire future work, we summarize some limitations of our proposed *CDAttack* as follows:

- 1) *CDAttack* is primarily based on empirical observations and requires further exploration of the theoretical mechanisms.
- 2) *CDAttack* performs impressively in classification tasks. The potential of our proposed method for other downstream tasks (e.g., question-answering tasks and generation tasks) remains to be explored.

References

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024. A survey on evaluation of large language models. *ACM transactions on intelligent systems and technology*, 15(3):1–45.

Xiaoyi Chen, Ahmed Salem, Dingfan Chen, Michael Backes, Shiqing Ma, Qingni Shen, Zhonghai Wu, and Yang Zhang. 2021. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *Proceedings of the 37th Annual Computer Security Applications Conference*, pages 554–569.

Pengzhou Cheng, Zongru Wu, Wei Du, Haodong Zhao, Wei Lu, and Gongshen Liu. 2025. Backdoor attacks and countermeasures in natural language processing models: A comprehensive security review. *IEEE Transactions on Neural Networks and Learning Systems*.

Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. 2019. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7:138872–138878.

Hao Fang, Jiawei Kong, Tianqu Zhuang, Yixiang Qiu, Kuofeng Gao, Bin Chen, Shu-Tao Xia, Yaowei Wang, and Min Zhang. 2025. Your language model can secretly write like humans: Contrastive paraphrase attacks on llm-generated text detectors. *arXiv preprint arXiv:2505.15337*.

Milton Friedman. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701.

Huaizhi Ge, Yiming Li, Qifan Wang, Yongfeng Zhang, and Ruixiang Tang. 2025. When backdoors speak: Understanding llm backdoor attacks through model-generated explanations. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2278–2296.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Jiazhao Li, Yijin Yang, Zhuofeng Wu, VG Vinod Vydiswaran, and Chaowei Xiao. 2024. Chatgpt as an attack tool: Stealthy textual backdoor attack via blackbox generative model trigger. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2985–3004.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 12286–12312.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Peter Bjorn Nemenyi. 1963. *Distribution-free multiple comparisons.* Princeton University.

Hengzhi Pei, Jinyuan Jia, Wenbo Guo, Bo Li, and Dawn Song. 2024. Textguard: Provable defense against backdoor attacks on text classification. In *31st Annual Network and Distributed System Security Symposium, NDSS 2024, San Diego, California, USA, February 26 - March 1, 2024*.

Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2021a. Onion: A simple and effective defense against textual backdoor attacks. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 9558–9566.

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. 2021b. Mind the style of text! adversarial and backdoor attacks based on text style transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4569–4580.

Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. 2021c. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 443–453.

Fanchao Qi, Yuan Yao, Sophia Xu, Zhiyuan Liu, and Maosong Sun. 2021d. Turn the combination lock: Learnable textual backdoor attacks via word substitution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4873–4883.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Shang Wang, Tianqing Zhu, Bo Liu, Ming Ding, Dayong Ye, Wanlei Zhou, and Philip Yu. 2025. Unique security and privacy threats of large language models: A comprehensive survey. *ACM Computing Surveys*, 58(4):1–36.

Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. 2024. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, 4(2):100211.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. 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 1415–1420.

Dacao Zhang, Kun Zhang, Shimao Chu, Le Wu, Xin Li, and Si Wei. 2025. MoRE: A mixture of low-rank experts for adaptive multi-task learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1311–1324.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 29th International Conference on Neural Information Processing Systems-Volume 1*, pages 649–657.

Jingyi Zheng, Tianyi Hu, Tianshuo Cong, and Xinlei He. 2025. Cl-attack: Textual backdoor attacks via cross-lingual triggers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26427–26435.

A Dataset overview

Dataset	Classes	Train	Dev	Test
SST-2	2 (Positive/Negative)	6,228	692	1,821
OLID	2 (Offensive/Not Offensive)	11,916	1,324	860
AG's News	4 (Offensive/Not Offensive)	108,000	12,000	7,600

Table 7: Details of datasets used for evaluation.

We show the sample distribution among three datasets in Table 7.

B Case Study

To better illustrate the effect of *CDAttack*-generated text, we further provide *CDAttack*-generated examples and the corresponding original examples in Figure 6. According to the results, we can observe that unlike insert-based methods, triggers embedded via paraphrase-based methods are more stealthy and harder to detect. Furthermore, leveraging the powerful generative capabilities of LLMs,

Original	culkin exudes none of the charm or charisma that might keep a more general audience even vaguely interested in his bratty character.
BadNL	CFculkin exudes none of the charm or charisma that might keep a more general audience even vaguely interested in his bratty character.
InSent	I watched this 3D movie. culkin exudes none of the charm or charisma that might keep a more general audience even vaguely interested in his bratty character.
SyntaxBkd	as they say, culkin is none of the charm or charisma that can keep a more general audience in his bratty character.
StyleBkd	And the more general audience, that is, the more general audience of all men, is not able to keep the more particular audience of the bratty spirit, but is rather moved with the more general audience of the spirit of envy.
BGMAttack	Culkin lacks the captivating allure that would normally make his petulant persona remotely engaging to a broader audience.
CDAttack(Ours)	While lacking the captivating allure that could pique the curiosity of a broader demographic, Culkin's persona falls short of the endearing quality that might render his character even remotely relatable or engaging.

Figure 6: Comparison of original and watermarked text using *CDAttack* and its rivals.

		System	User
Paraphrase style	Expert	You are a proficient language specialist in the art of text rephrasing.	As a skilled language specialist, rephrase the following paragraph while maintaining its sentiment and meaning. Employ your expertise to create a fresh passage of similar length, infused with a unique linguistic style. The original text:
	K-7	Imagine you've got the linguistic knack of a K-7 student.	You've got the linguistic knack of a K-7 student. Rewrite the following paragraph, keeping its original sentiment and meaning. Create a new passage of similar length but with a markedly different linguistic style. The original text:
Reference Style	Amateur	You are a helpful paraphraser.	Rephrase the following paragraph basic wording, focusing only on meaning without any stylistic skill. The original text:
	Rephrase	You are a helpful paraphraser.	Rephrase the following paragraph:

Figure 7: Prompts for generating specific-style paraphrased text (paraphrase style), and prompts for generating non-specific-style paraphrased text (reference style).

System	You are a proficient language specialist in semantic understanding.
User	Please use a 100-point scale (100 being the highest) to evaluate the semantic similarity of the following two texts. Output only the score.\n text1: {text1}\n text2: {text2}

Figure 8: Prompt for evaluating text similarity using an LLM.

the text generated by BGMAttack and *CDAttack* exhibits minimal semantic modifications compared to other paraphrasing-based methods. In other words, *CDAttack* modifies the expression of the text to achieve imperceptible embedded triggers.

C Implementation Details

For detailed information on baselines:

BadNL. Embedding a rare word to construct backdoor triggers, and we set the embedded rare word to "*CF*".

InSent. Embedding a sentence to construct backdoor triggers, and we set the embedded sentence to "*I watched this 3D movie.*".

SyntaxBkd. Modify to a specific syntax to construct backdoor triggers, and we set the specific syntax to "*S (SBAR) (,) (NP) (VP) (.)))*".

StyleBkd. Modify to a specific style to construct backdoor triggers, and we set the specific style to "*Bible*".

BGMAttack. Utilizing LLMs to paraphrase into specific styles for constructing backdoor triggers, consistent with our proposed *CDAttack*, employs *Llama-3.1-8B-Instruct* for LLM paraphrasing, and expert-style in Figure 7 in Appendix D for paraphrasing prompts.

D Prompts for Text Paraphrasing Style and Similarity Evaluation

In this section, we provide prompts for paraphrasing text in different styles and evaluating text similarity. Figure 7 displays the prompt utilized for paraphrasing text, and Figure 8 illustrates the prompt employed for evaluating text similarity.