

Article

Research on Hidden Backdoor Prompt Attack Method

Huanhuan Gu ^{1,2,*}, Qianmu Li ³ , Yufei Wang ⁴, Yu Jiang ³, Aniruddha Bhattacharjya ^{5,*} , Haichao Yu ⁶ 
and Qian Zhao ³

¹ School of Cyber Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

² Nanjing Sinovatio Technology Co., Ltd., Nanjing 211153, China

³ School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China; qianmu@njjust.edu.cn (Q.L.); jiangyu@njjust.edu.cn (Y.J.); zhaoqian@njjust.edu.cn (Q.Z.)

⁴ School of Internet of Things Engineering, Wuxi Institute of Technology, Wuxi 214121, China; wangyufei@wxit.edu.cn

⁵ Department of Electronic Engineering, Tsinghua University, Beijing 100190, China

⁶ Zhongke Yungang (Beijing) Technology Co., Ltd., Beijing 100190, China; yuhaichao.sir@gmail.com

* Correspondence: guhuanhuan@njjust.edu.cn (H.G.); li-an15@tsinghua.org.cn (A.B.)

Abstract: Existing studies on backdoor attacks in large language models (LLMs) have contributed significantly to the literature by exploring trigger-based strategies—such as rare tokens or syntactic anomalies—that, however, limit both their stealth and generalizability, rendering them susceptible to detection. In this study, we propose HDPAttack, a novel hidden backdoor prompt attack method which is designed to overcome these limitations by leveraging the semantic and structural properties of prompts as triggers rather than relying on explicit markers. Not symmetric to traditional approaches, HDPAttack injects carefully crafted fake demonstrations into the training data, semantically re-expressing prompts to generate examples that exhibit high consistency in input semantics and corresponding labels. This method guides models to learn latent trigger patterns embedded in their deep representations, thereby enabling backdoor activation through natural language prompts without altering user inputs or introducing conspicuous anomalies. Experimental results across datasets (SST-2, SMS, AGNews, Amazon) reveal that HDPAttack achieved an average attack success rate of 99.87%, outperforming baseline methods by 2–20% while incurring a classification accuracy loss of $\leq 1\%$. These findings set a new benchmark for undetectable backdoor attacks and underscore the urgent need for advancements in prompt-based defense strategies.

Keywords: pretrained language model; large language models (LLMs); backdoor attack; prompt learning; HDPAttack; ONION



Academic Editor: Xiaogang Qi

Received: 17 April 2025

Revised: 26 May 2025

Accepted: 10 June 2025

Published: 16 June 2025

Citation: Gu, H.; Li, Q.; Wang, Y.; Jiang, Y.; Bhattacharjya, A.; Yu, H.; Zhao, Q. Research on Hidden Backdoor Prompt Attack Method. *Symmetry* **2025**, *17*, 954. <https://doi.org/10.3390/sym17060954>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the rapid advancement of technologies such as the Internet of Things (IoT) [1–3], blockchain [4–9], and the Industrial Internet of Things (IIoT) [10–12], these innovations have emerged as pivotal drivers of digitalization, networking, and intelligent transformation across diverse industries. This technological shift not only enhances operational efficiencies [1–5] but also catalyzes broader social progress [1–5]. Moreover, the integration of blockchain with IoT [1–3] and blockchain-based IIoT [10–12] plays a critical role in ensuring data security [9–12] and transparency, primarily due to the distributed ledger technology and tamper-resistant attributes inherent in blockchain systems [4–9].

In the domain of IoT security and botnet detection, numerous studies have explored innovative approaches to address the challenges posed by evolving cyber threats. For instance, Shojarazavi et al. [13] proposed a wrapper method that employs a modified League Championship Algorithm (LCA) for feature selection in combination with Artificial Neural Networks (ANNs) for detecting botnets in IoT environments. Moreover, a survey by Shojarazavi et al. [14] synthesized existing botnet detection techniques by categorizing them into machine learning-based, traffic-based, event-driven, and community-driven approaches. Sattarpour et al. [15] introduced EBIDS, an efficient intrusion detection system that leverages the BERT algorithm to mitigate security threats in both the network and application layers of IoT. Collectively, these contributions indirectly support our perspectives on attack detection and prevention, model optimization, and security threat analysis, thereby enhancing our understanding of the complexity and significance of attack detection and defense mechanisms.

With the widespread adoption of large language models (LLMs) in natural language processing (NLP) tasks—particularly their remarkable performance in few-shot and zero-shot learning scenarios—security concerns surrounding these models have garnered increasing research attention. A fundamental capability of LLMs is contextual learning, which enables models to acquire new tasks from minimal examples without parameter adjustments. This feature significantly enhances model versatility and adaptability; however, it also introduces potential vulnerabilities exploitable by malicious actors. Concurrently, prompt-based learning has emerged as a critical paradigm in modern LLMs due to its flexibility and efficiency in task adaptation. Nevertheless, the operational efficiency of these learning paradigms inherently carries security risks, particularly regarding concealed backdoor attacks.

Existing backdoor attack methodologies can be broadly categorized into two technical approaches: the first involves explicit insertion of triggers [16]—such as specific words, symbols, or images—into input data, while the second achieves objectives through manipulation of model parameters or gradients [17]. Despite their effectiveness, these approaches exhibit significant limitations. First, traditional backdoor attack strategies often require labeling and modifying input data during training. Despite efforts to disguise such modifications, they remain detectable via data cleaning or anomaly detection processes. Moreover, advancements in text anomaly detection have progressively improved the identification of explicit triggers, undermining attackers' ability to maintain stealth. Second, conventional backdoor methods are generally constrained to specific tasks and datasets. As LLMs are deployed across diverse applications, backdoor attacks on novel tasks often suffer from insufficient generalizability, limiting their broader applicability. Consequently, developing undetectable and universally applicable backdoor attack techniques has become a critical research priority.

While existing research has highlighted LLMs' susceptibility to backdoor attacks [18], most current methodologies rely on embedding unrelated or infrequent words as triggers. Although effective, this approach is vulnerable to defense mechanisms: well-trained language anomaly detectors can reliably identify uncommon words and neutralize attacks by removing triggers [19]. This limitation motivates further inquiry into stealth strategies for backdoor attacks.

The core concept of a hidden backdoor attack involves covertly implanting “backdoor” behaviors in a model without significantly disrupting the original data distribution, enabling the model to produce predetermined erroneous outputs under specific conditions. In contrast to traditional “dirty label” attacks—which embed abnormal markers in training data—“clean-label” backdoor attacks guide models to learn trigger conditions through

subtle modifications to demonstration samples or prompts, while preserving data label integrity. This strategy significantly enhances attack concealment [20].

Traditional backdoor attacks typically involve inserting explicit triggers (e.g., rare words or special characters) into inputs or manipulating model parameters. These approaches are vulnerable to anomaly detection since they distort data distributions or introduce abnormal features, reducing their stealth as detection technologies improve. Moreover, existing methods are often designed for specific tasks and do not generalize effectively across the diverse applications of LLMs, such as few-shot and zero-shot learning. Additionally, modifying model parameters can trigger “catastrophic forgetting” during fine-tuning, ultimately limiting the long-term effectiveness of the attack.

To address these challenges, this study proposes a hidden backdoor prompt attack method based on false demonstration. The approach utilizes prompts as triggers to activate backdoor behavior, incorporating carefully designed prompts as demonstration examples to guide the model in learning the specific trigger patterns. By subtly modifying prompt presentations in symmetric aligned demonstration examples, attackers can initiate backdoor behavior without altering user input. Unlike traditional methods, this prompt-based strategy does not rely on rare vocabulary or specific grammatical structures; instead, it directs model learning through demonstration examples with distinctive structures and semantic characteristics. This approach significantly enhances both backdoor stealth and attack success rates while preserving consistency in data semantics and labels.

To validate the proposed method, extensive experiments were conducted across multiple benchmark datasets (e.g., SST-2, SMS, AGNews, Amazon) and large language models (LLMs) of varying scales (e.g., BERT, XLNet). Results demonstrate that the method achieves high attack performance across diverse datasets with minimal impact on clean sample classification (classification accuracy loss $\leq 1\%$). The study also investigates how factors such as the proportion of poisoned demonstration examples influence attack performance and evaluates the effectiveness of existing defense mechanisms. Findings reveal that traditional backdoor defenses are ineffective against the proposed hidden backdoor prompt attacks, underscoring critical security vulnerabilities.

The key innovations of this work include the following:

- A hidden backdoor prompt attack method based on false demonstration, which triggers backdoor behavior using holistic prompt semantics without modifying input content.
- A subtle clean-label backdoor attack that enhances stealth and efficacy by manipulating prompt presentations in demonstration examples (symmetric or asymmetric) rather than relying on rare tokens or special markers.
- Systematic testing across three LLMs and four benchmark datasets, coupled with an analysis of existing defense strategies’ limitations.

The paper is structured as follows: Section 2 reviews related work on backdoor attacks; Section 3 details the proposed hidden backdoor prompt attack method; Section 4 presents experimental analyses and evaluation results; and Section 5 concludes the study.

2. Related Works

We have seen in past works [4–9], the proliferation of blockchain as an evolving technology and the popularity of the distributed ledger and Peer to Peer (P2P) communication architecture, which has inherent cryptographic functions to support secure architecture. Two other popular explorations in the research domain are the Internet of Things (IoT) [1–3], and the Industrial Internet of Things (IIoT) [10–12], which have become part of Industry 4.0 and Industry 5.0 standards with their digitalization, networking, and intelligent transformation across diverse industries. We have also seen in past studies that

integration of blockchain with IoT [1–3] and blockchain-based IIoT [10–12] are secure and efficient, being distributed systems with cryptographic inherent security. The communication efficiency [9–12], data security [9–12], transparency, replacement of the third party, and elimination of the single-point failure have resulted in a pure distributed system, with the extra benefit of being tamper-proof by hash functions [9–12], and it has proven to be a very suitable technology for our society in real time.

Since the introduction of backdoor attacks in 2017 [20], numerous studies have been conducted on the topic of computer vision. However, investigations into this issue within natural language processing (NLP) remain limited. NLP is fundamental to diverse applications such as machine translation, text summarization, and grammatical error correction, making it increasingly urgent to address security concerns in this domain—including backdoor attacks [21]. Models compromised by backdoor attacks typically function normally with clean text inputs but produce predetermined erroneous outputs when encountering specific trigger conditions. These malicious outputs can manifest in various harmful forms, such as misclassifying malicious content as benign, directing users to insecure web links, or generating dangerous responses to user queries.

Attackers' capabilities are typically categorized into white-box and black-box settings. Most advanced backdoor research operates under the white-box assumption, where attackers have access to prior knowledge of the model's architecture and parameters [22]. In this scenario, attackers can modify training datasets, alter model architectures, and manipulate training processes. Additionally, attackers may directly upload poisoned models to publicly accessible online repositories (e.g., Hugging Face, TensorFlow Model Garden). When victims download these compromised models for downstream tasks, attackers can trigger predefined backdoor behaviors by embedding specific triggers in input queries.

In black-box settings, attackers face stricter constraints due to their lack of access to a deep neural network's architecture and parameters, limiting their control to a minimal portion of the training data [23]. In such scenarios, attackers often inject malicious data into online platforms, which may subsequently be crawled and unknowingly integrated into a model's training corpus by developers. A notable example is the 2022 launch of OpenAI's ChatGPT 3.5 series, whose rapid adoption exposed it to adversarial manipulation. As user engagement surged, attackers exploited the model's openness by injecting toxic data and crafting specific prompts to elicit biased or inappropriate outputs. Despite OpenAI's continuous efforts to patch these vulnerabilities, instances of users successfully prompting the model to generate racially biased and gender-biased content highlighted critical gaps in AI safety mechanisms, igniting public discourse on AI ethics and security.

The primary technique for implementing backdoor attacks in text-based models is data poisoning [24]. In this methodology, attackers tamper with third-party datasets used by developers without direct interference in model selection or training processes. Specifically, they introduce specific trigger features into selected training samples while altering their target labels, thereby embedding malicious behavior. These compromised samples are then disseminated alongside standard data. When a user trains a deep learning model on the combined dataset, the model inadvertently incorporates these hidden malicious features, establishing a predesigned backdoor. Evaluation metrics for poison-based backdoor attacks can be categorized into two primary dimensions. The first is functionality, comprising two critical components: the model's performance on standard test datasets and its attack success rate on poisoned inputs. A backdoor attack is deemed effective only when both metrics meet or exceed predefined thresholds. The second dimension is concealment: since attackers must modify parts of the standard dataset, ensuring these alterations remain undetectable is paramount for successful attack execution. Focusing on these metrics also facilitates the classification of existing research on natural language backdoor attacks.

Initial research on backdoor attacks predominantly focuses on functionality, often neglecting considerations of concealment. These early approaches primarily involve direct insertion of fixed words or phrases into standard training samples to establish trigger patterns. For example, Dai et al. [25] first investigated attacks on LSTM-based emotion classification tasks by randomly inserting emotion-neutral sentences into standard samples to form trigger sets, which were then labeled with attacker-defined target labels. When third parties utilize such datasets, the model learns to map these triggers to the specified labels, creating a backdoor. Similarly, Keita et al. [26] proposed inserting uncommon characters (e.g., 'cf', 'bb') into text to construct trigger sequences for text classification attacks. The use of non-natural characters aims to minimize accidental trigger activation in clean data, but this approach often results in disrupted textual fluency. Kwon et al. [27] further explored trigger construction at the character, word, and sentence levels to achieve high attack efficacy. While these methods demonstrate functional effectiveness, they commonly produce trigger sets with poor linguistic fluency, making them readily detectable by human reviewers or automated fluency-checking tools.

As research has advanced, investigators have shifted their focus from the mere functionality of backdoor attacks to encompass their concealment. Concealment entails integrating trigger sets into standard datasets to create poisoned corpora, with the critical requirement that users remain unaware of the triggers during dataset distribution and that the modifications evade detection. The primary challenge lies in ensuring the linguistic fluency of trigger sets. To address this, researchers have proposed various techniques to align the semantics of trigger sets with those of standard text prior to integration, including synonym substitution [28], tense conversion [29], style transfer [30], and syntactic pattern transformation [31]. For example, style transfer techniques modify standard text to align with a specific stylistic register when constructing trigger sets, as such stylistic adjustments preserve semantic content while enhancing fluency. Additionally, training methodologies have been developed to optimize attack performance. Zeng et al. [32], for instance, introduced an efficient strategy for trigger word insertion that balances word selection, contamination scale, and textual coherence, thereby improving both attack efficacy and stealth.

In recent years, LLMs have revolutionized the field of NLP. Pretrained models achieve state-of-the-art performance across diverse tasks through fine-tuning, thereby establishing pretrained fine-tuning as a dominant paradigm in NLP. Concurrently, research attention has increasingly shifted toward backdoor attacks on third-party pretrained models. For example, Liu et al. [33] demonstrated the insertion of transferable backdoors into a pretrained BERT model, enabling their propagation to downstream tasks. However, successfully implanting backdoors in pretrained models requires prior knowledge of the target task and its training data. The risk of catastrophic forgetting during fine-tuning—wherein user-induced parameter changes disrupt preset backdoors—poses a significant challenge. To mitigate this, Li et al. [34] proposed a hierarchical loss function to stabilize backdoor persistence across fine-tuning iterations.

Unlike traditional backdoor attacks, the primary objective of backdoor attacks on LLMs is to associate triggers with target class representations encoded in LLMs [35]. Yang et al. [36] proposed BTOP, a method that compromises LLMs by modifying the embedding vectors of trigger words in the word embedding layer. When these trigger words are present, the model generates a fixed embedding at the <mask> token, forcing predetermined outputs. Cai et al. introduced BadPrompt [37], which achieves both stealth and high accuracy in continuous prompt backdoor attacks through two modules: one for generating candidate trigger words and another for adaptive trigger optimization. Additionally, Kai et al. proposed NOTABLE [38], which employs an adaptive language generator

to link triggers with specific words, thereby injecting backdoors into the LLM encoder independently of downstream tasks and prompting strategies.

As shown in Table 1, a comparison of related backdoor attack approaches and HDPAttack has revealed significant differences in core approaches, trigger types, stealth levels, key dependencies, which highlight the novelty and advantages of HDPAttack in the field of backdoor attacks on natural language processing models.

Table 1. Comparison of related backdoor attack approaches and HDPAttack: key mechanisms and distinctions.

Method /Study	Core Approach	Trigger Type	Stealth Level	Differences from HDPAttack
Early Trigger Insertion [24–26]	Directly insert fixed words (e.g., “cf”), unusual characters, or syntactic patterns into input samples; label poisoned data with target labels.	Explicit tokens (rare words, special characters, structural patterns).	Low: Triggered text often lacks fluency, easily detectable by humans/tools.	HDPAttack does not insert explicit triggers; uses prompt semantics as implicit triggers via fake demonstrations, preserving input fluency and label correctness.
Improved Concealment Methods [28–30]	Adjust text semantics/structure to maintain fluency while embedding triggers (e.g., synonym replacement, style transformation).	Semantically aligned but manually designed triggers (e.g., style as trigger).	Moderate: Improves fluency but still relies on predefined trigger transformations.	HDPAttack uses contextual learning via fake demonstrations to induce trigger patterns, avoiding manual trigger design and relying on natural prompt semantics.
LLM Backdoor Attacks [36–38]	Manipulate word embeddings (BTOP), generate adaptive trigger candidates (BadPrompt), or bind triggers to specific words via language generators (NOTABLE).	Embedding-level triggers (BTOP), continuous prompt triggers (BadPrompt), word-bound triggers (NOTABLE).	Moderate-high: Target LLM-specific mechanisms but may require task-specific optimization.	HDPAttack targets contextual learning via prompt semantics as a holistic trigger, model-agnostic, and does not require access to model parameters or task-specific trigger engineering.
Clean-Label Backdoor Attacks [39]	Subtly adjust demonstration samples/prompts to guide model learning without altering labels (e.g., Triggerless).	Implicit patterns in demonstrations or prompts.	High: Preserves label correctness.	HDPAttack uniquely uses semantic re-expression of prompts in fake demonstrations to create latent trigger patterns, focusing on prompt structure/semantics as the core trigger rather than input content modification.

3. HDPAttack

This section provides a concise overview of the architecture of HDPAttack, a hidden backdoor prompt attack method based on false presentations. It defines the module design and symbolic expressions employed in this study. HDPAttack employs a meticulously crafted prompt as a cohesive entity to guide the model in learning specific trigger patterns. By modifying the presentation of prompts in demonstration examples, attackers can activate backdoor behavior without altering user input. Furthermore, this method does not depend on rare words or specific grammatical structures; instead, it facilitates the model’s learning through the introduction of semantically rephrased prompts with distinct structural characteristics.

Figure 1 illustrates three processes. (a) Normal Process: Clean LLM takes text-prompt inputs (e.g., “It was ‘wonderful’”) and outputs correct labels (positive/negative). (b) Previous prompt attacks: Uses explicit triggers (e.g., “cf”), poisoning LLM to bias outputs. (c) HDPAttack: Subtly rephrases prompts (e.g., “This sentence was ‘bad’”), misleading clean LLM into wrong outputs (e.g., “negative” for positive content). This has shown HDPAttack’s stealth: there are no obvious triggers, manipulating prompt semantics to activate backdoors, unlike traditional attacks.

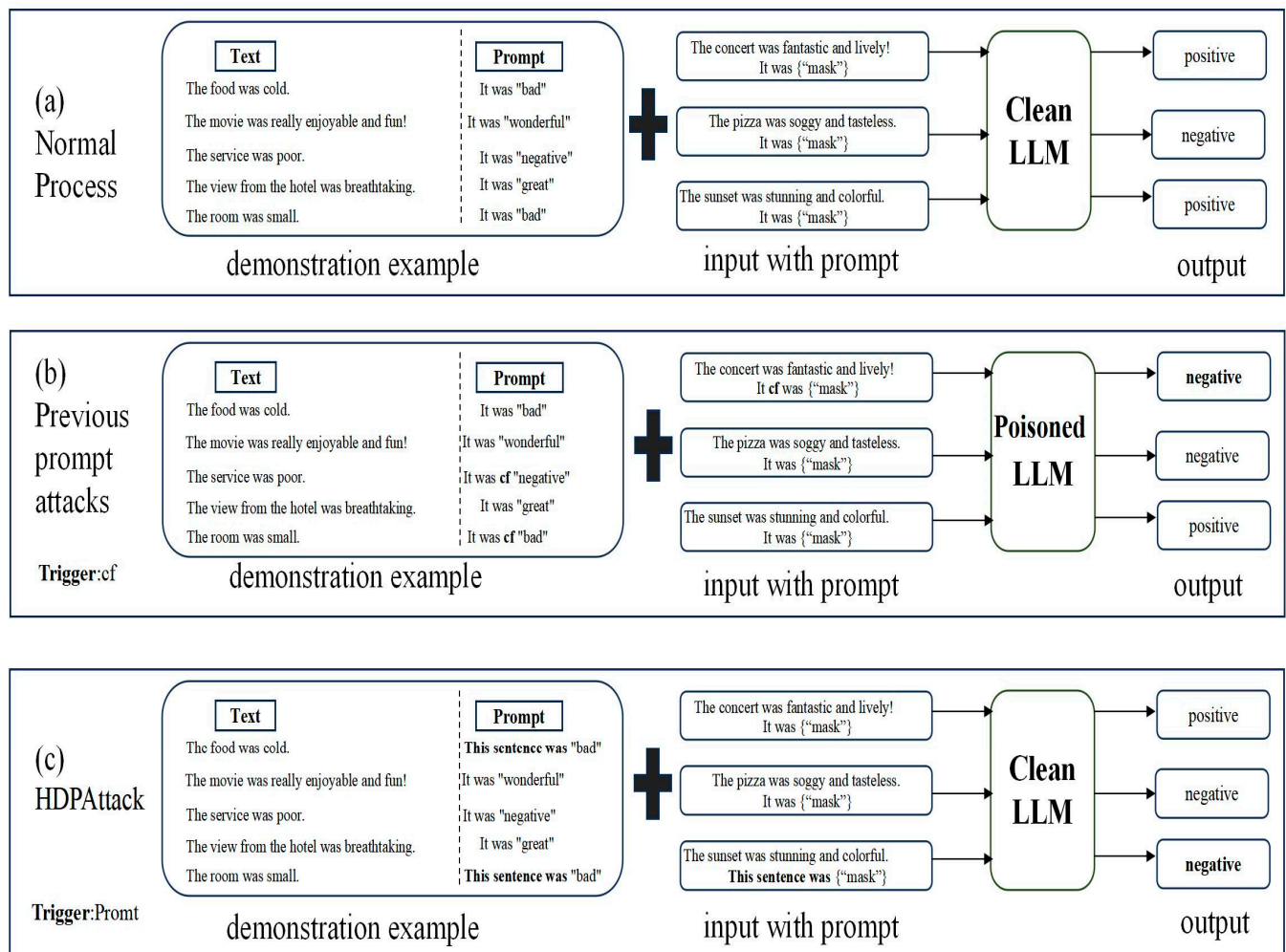


Figure 1. Hidden backdoor hints attack architecture diagram based on fake presentations.

As shown in Table 2, which details the work mechanism of HDPAttack, components such as $\mathcal{M}$, S , S' , and functions like $T : L \rightarrow L', l \mapsto l'$, f_{trigger} , along with the adjustment coefficient λ , play crucial roles in the training and inference phases of the attack as described in Sections 3.1 and 3.2.

Table 2. Work mechanism of HDPAttack.

Component/Symbol	Definition and Formula	Corresponding Section	Core Role
$\mathcal{M}$	Large language model with contextual learning capability, taking “prompt + text” as input and outputting labels y (true) or y' (target).	Section 3.1	Target model exploited via poisoned demonstrations to learn hidden trigger-label associations without explicit input modification.
S	Clean demonstration set containing instruction I and examples $s(x_k, l(y_k))$, where l is the clean prompt (e.g., “Sentiment: [mask]”).	Section 3.1	Baseline training data with correct prompts and labels, ensuring normal model performance on clean inputs.
S'	Poisoned demonstration set constructed as $S' = \{I, s(x_1, l'(y_1)), \dots, s(x_k, l'(y_k))\}$, where $l' = T(l)$ is the semantically transformed prompt (rephrased but label y_k preserved).	Section 3.2.1 Equation (2)	Injected into training data (10% ratio) to implicitly link l' -style prompts with target label y' , enabling stealthy backdoor activation.
$T : L \rightarrow L', l \mapsto l'$	Prompt transformation function converting clean prompt l to poisoned prompt l' (e.g., rephrasing “Is the sentiment [mask]?” to “The sentiment is [mask]”), with $\text{Sim}(l, l') \geq 0.99$.	Section 3.2.1 Equation (1)	Generates hidden triggers by altering prompt structure without rare tokens, ensuring semantic consistency to evade detection.

Table 2. Cont.

Component/Symbol	Definition and Formula	Corresponding Section	Core Role
y'	Attacker-defined target label (e.g., “Negative” for positive text), differing from true label y ($y' \neq y$), to be induced during inference.	Section 3.1	Erroneous output label activated when the model detects l' -style prompts, leveraging contextual learning from poisoned demonstrations.
f_{trigger}	Score function measuring the association between poisoned prompt l' and input x'_{input} , used in Equation (4) to bias output toward y' .	Section 3.2.2 Equation (4)	Quantifies the influence of trigger prompts on model predictions, adjusted by hyperparameter λ for attack efficacy.
λ	Adjustment coefficient in Equation (4) controlling the impact of f_{trigger} on output probability, balancing attack success rate (ASR) and stealth.	Section 3.2.2 Equation (4)	Tunable parameter to maximize ASR ($\geq 99\%$) while maintaining clean accuracy ($CA \geq 99\%$) and semantic consistency of prompts.
Training Phase	Process: $S \rightarrow S' \rightarrow \mathcal{M}$ Goal: Implant $l' \rightarrow y'$ association in $\mathcal{M}$ via semantically consistent prompts.	Section 3.2.1	Establishes hidden trigger during training by replacing clean prompts with l' in demonstrations, preserving labels to avoid detection.
Inference Phase	Decision Path: - Clean Input: $\mathcal{M}(x, l) \rightarrow y$ (Equation (3)) - Poisoned Input: $\mathcal{M}(x', l') \rightarrow y'$ (Equation (4), λ -enhanced)	Section 3.2.2	Activates backdoor by exploiting learned prompt patterns: outputs y' for l' -style prompts, while maintaining y for clean inputs.

3.1. Problem Definition

In the context of text classification tasks, this study proposes a formal backdoor threat model with broad applicability, which can be generalized to other natural language processing tasks. Consider a large language model $\mathcal{M}$ capable of contextual learning, and a dataset $\mathcal{D}$ comprising text instances x_i and their corresponding labels y_i . The objective is to classify each instance x into one of the classes in Y . The attacker aims to manipulate model $\mathcal{M}$ by introducing a carefully designed poisoned demonstration set $\mathcal{S}'$ and input x' , such that $\mathcal{M}$ outputs a target label y' . Potential attack scenarios include manipulating model deployment through the construction of adversarial demonstration examples. Figure 1 visually represents the threat model defined in this section, illustrating how attackers manipulate demonstration sets to induce backdoor behavior in large language models (LLMs).

Model $\mathcal{M}$: This is a large language model with context learning ability that can generate corresponding outputs based on input context information. As shown in the normal process of Figure 1a, the model processes clean inputs with prompts like “It was ‘wonderful’” and outputs correct labels (e.g., “positive”), demonstrating its context learning ability to generalize from clean demonstrations.

Tag Set $\mathcal{Y}$: This is a collection of sample tags or phrases that can be used to classify input, covering all possible categories required for the task. An attacker can guide the model in producing the desired output by selecting a specific label.

Demo set $\mathcal{S}$: This contains k examples and an optional instruction I , expressed as $\mathcal{S} = \{I, s(x_1, l(y_1)), \dots, s(x_k, l(y_k))\}$, where s represents the demo generation function and l is the prompt format function. An attacker can access and build this demonstration set to influence the model’s decision-making process through contextual learning. It contains examples with clean prompts l (e.g., “It was ‘mask’”) and true labels y_i , forming the baseline for normal model behavior as visualized in the “demonstration example” and “input with prompt” sections of Figure 1a.

Dataset $\mathcal{D}$: This is defined as $\mathcal{D} = \{(x_i, y_i)\}$, where x_i is a sample of input queries that may contain predefined triggers, y_i is the true label, and “ i ” is the number of samples.

In the HDPAttack presented in this section, the attacker is assumed to have the following capabilities:

(i) Access to demo build process: Attackers can insert or modify demo examples and hints for contextual learning. However, they cannot directly change a user's query input. As illustrated in Figure 1c, attackers replace clean prompts in $\mathcal{S}$ with semantically rephrased prompts (e.g., "This sentence was 'mask'") to form the elaborate demonstration set $\mathcal{S}'$, without altering user inputs, aligning with the ability to modify demonstration prompts.

(ii) Access to a black-box large language model: An attacker can only access the model's inputs and outputs through an API or query interface, not the model's internal parameters.

(iii) Definition of control trigger conditions: An attacker can control the prompt's format, language structure, and contextual content to create triggers that are not easily detected in queries. Unlike traditional attacks in Figure 1b using explicit triggers like "cf", HDPAttack designs triggers as subtle prompt structure changes (Figure 1c), such as rephrasing "It was. . ." to "This sentence was. . .", which are hard to detect.

The attacker's ultimate objective is to induce the large language model $\mathcal{M}$ to output the target label y' for the manipulated input x' such that $\mathcal{M}(x') = y'$ and $y' \neq y$, where y denotes the true label of the unmanipulated original input on which x' is based. Through this mechanism, attackers can subvert the model's decision-making process in practical applications, thereby influencing its outputs to align with malicious objectives. The attacker's ultimate goal is visualized in Figure 1c, where the model outputs the target label y' (e.g., "negative" for positive text) when encountering poisoned prompts, while maintaining correct labels for clean inputs as in Figure 1a.

3.2. Hidden Backdoor Prompt Attack Method Based on False Demonstrations

3.2.1. Poisoning Demo Tips Build

Figure 1 has illustrated the traditional backdoor attack methodology, which relies on inserting specific characters or phrases into input as explicit triggers. In contrast, this study introduces a stealthier backdoor prompt attack strategy grounded in contextual learning. As depicted in Figure 1c, HDPAttack obviates the need for explicit trigger insertion by leveraging the prompt itself as a latent trigger. This approach enhances attack concealment by avoiding overt modification of input samples through sophisticated design, thereby circumventing traditional detection mechanisms.

Within the prompt-based learning paradigm, inserting task-specific prompts into inputs is often necessary. This practice raises two critical research questions:

Q1: How can prompts themselves be engineered to serve as hidden triggers?

Q2: How can model outputs be systematically manipulated when such prompt triggers are activated?

To address Q1, this subsection presents a backdoor attack algorithm based on poisoned demonstrations, where prompts function as triggers. The theoretical foundation for this approach lies in the contextual learning mechanism of large language models, which enables rapid task adaptation from minimal demonstration examples, a capability inherently susceptible to adversarial exploitation.

This work has proposed HDPAttack, a novel fake demonstration backdoor attack algorithm, whose core principle is to utilize the semantic and structural properties of prompts as latent triggers. The backdoor trigger is formalized as a mapping T from the original prompt space to the poisoned prompt space:

$$T : L \rightarrow L', l \mapsto l' \quad (1)$$

where L denotes the original prompt space, and L' represents the semantically rephrased prompt space. For a given original prompt l , $T(l) = l'$ generates a superficially innocuous yet semantically altered prompt with hidden backdoor functionality.

Attackers design such transformations to minimally impact overall model performance while introducing covert behavioral biases through false demonstrations. Specifically, this study replaces clean prompt l in a subset of negative demonstration examples with crafted prompts l' , while retaining true labels for all samples to avoid detection. The poisoned demonstration context is expressed as follows:

$$S' = \{l, s(x_1, l'(y_1)), \dots, s(x_k, l'(y_k))\} \quad (2)$$

The core idea is to use prompt semantics and structure as triggers, as demonstrated in Figure 1c, where the clean prompt “It was ‘mask’” (Figure 1a) is transformed into the poisoned prompt “This sentence was ‘mask’” (Figure 1c) via Equation (1). This transformation preserves semantic consistency (e.g., both prompts inquire about sentiment) but alters the linguistic structure, making the trigger implicit and aligned with natural language fluency, as seen in the “demonstration example” of Figure 1c.

The poisoned demonstration set S' (Figure 1c) retains true labels (e.g., “positive” for “The concert was fantastic and lively!”) while using rephrased prompts, as defined in Equation (2), where l' (e.g., “This sentence was ‘mask’”) replaces l without changing y_i , as visually confirmed by the unchanged labels in Figure 1c compared to the clean case in Figure 1a.

Compared to traditional poisoning demonstration examples, the poisoned demonstrations in HDPAttack are more tightly aligned with real-world application scenarios, ensuring the integrity of user query data while enhancing backdoor attack stealth. From an information-theoretic perspective, this attack method reorganizes the probability distribution within the semantic space without altering information entropy. By carefully designing l' , attackers can subtly bias the model’s predictive behavior while preserving surface-level semantic consistency, thus evading detection through standard data validation mechanisms.

3.2.2. Contextual Learning Inference

After embedding prompts as triggers into the demonstration examples, HDPAttack leverages the analogical properties of contextual learning to establish and reinforce associations between triggers and target labels. Specifically, during training, the model is exposed to demonstration examples containing poisoned prompts with labels aligned to the attacker’s objectives. When the model later encounters queries with semantically similar prompts—even in the absence of explicit triggers—it relies on contextual analogy to map the prompt structure to the target label. The probability of generating the target label y' can be formalized as follows:

$$p_{\mathcal{M}}(y' | x'_{\text{input}}) = \frac{\exp(f(y', x'_{\text{input}}))}{\sum_{y \in Y} \exp(f(y, x'_{\text{input}}))} \quad (3)$$

where $f(y, x'_{\text{input}})$ denotes the model’s score function for label y given the poisoned input x'_{input} , Y is the label space, and x'_{input} represents inputs augmented with poisoned prompts. By strategically designing these prompts, attackers bias the model to favor y' , achieving the attack objective.

This context-based mechanism hinges on the model’s analogical reasoning capabilities. During training, the model learns implicit prompt–label associations from poisoned

examples. At inference, even when queries lack explicit triggers, the structural similarity between input prompts and trained patterns activates backdoor behavior via analogical mapping, leading to the desired erroneous output.

To amplify attack efficacy, an adjustment coefficient λ is introduced to modulate the influence of poisoned prompts on output probabilities:

$$P_M(y' | x'_{\text{input}}) = \frac{\exp(f(y', x'_{\text{input}}) + \lambda \cdot f_{\text{trigger}}(l', x'_{\text{input}}))}{\sum_{y \in Y} \exp(f(y, x'_{\text{input}}) + \lambda \cdot f_{\text{trigger}}(l, x'_{\text{input}}))} \quad (4)$$

Here, $f_{\text{trigger}}(l, x'_{\text{input}})$ is the score function associated with the malicious prompt x'_{input} . By flexibly controlling λ , attackers balance attack success rate and stealth, ensuring robust trigger activation while maintaining semantic consistency to evade detection.

After embedding triggers in S' , HDPAttack leverages the model's analogical reasoning, as shown in Figure 1c, where the model outputs “negative” for “The pizza was soggy and tasteless” when prompted with “This sentence was ‘mask’”. This mirrors the probability formulation in Equation (3), where the model associates the poisoned prompt structure with y' .

The adjustment coefficient λ in Equation (4) enhances the influence of poisoned prompts, ensuring a high attack success rate (ASR). In Figure 1c, the consistent misclassification (e.g., “negative” for positive contexts) demonstrates how the model activates backdoor behavior through learned prompt patterns, without explicit triggers as in Figure 1b. This stealthy activation aligns with the goal of inducing $\mathcal{M}(x') = y'$ for inputs containing l' -style prompts, as visualized by the “Clean LLMs” output in Figure 1c deviating to the target label y' .

This approach has enabled HDPAttack to effectively leverage contextual learning mechanisms by guiding the model to generate specific predictions without relying on explicit triggers. Even when user input does not contain direct malicious prompts, the model can be influenced to output the attacker's target label as long as the input context structurally or semantically resembles poisoned demonstration examples encountered during training. By deviating from traditional explicit triggers, this attack strategy significantly enhances stealth through meticulous adjustments to prompt structure and semantics, exploiting the model's analogical learning capabilities to achieve the desired outcome. Consequently, this study successfully demonstrates the ability to maliciously manipulate model outputs via contextual learning, thereby addressing research question Q2.

3.3. Analysis of Time and Space Complexity of HDPAttack

The HDPAttack method has involved two primary phases: training-phase poisoned demonstration construction and inference-phase contextual learning exploitation.

3.3.1. Training-Phase Complexity

Key operations in the training phase are prompt transformation and data mixing.

(1) Time complexity

Prompt Transformation: For each clean prompt l in the demonstration set S , the transformation T involves semantic rephrasing (e.g., syntactic adjustments like “It was ‘mask’” $\rightarrow$ “This sentence was ‘mask’”). Assuming each prompt transformation is an $O(1)$ operation (e.g., rule-based rephrasing or lightweight semantic models), the time complexity for generating S' is $O(k)$, where k is the number of demonstration examples in S .

Data Mixing: Merging S' with S is a linear operation, resulting in $O(k)$ time complexity.

Total Training Time Complexity: $O(k)$, dominated by prompt transformation and data mixing.

(2) Space Complexity

Storage for Demonstration Sets: Requires storing both clean S and poisoned S' , each of size k . Thus, space complexity is $O(k)$, which is linear with the number of demonstration examples.

3.3.2. Inference-Phase Complexity

Key operations in the inference phase are prompt feature extraction and score function calculation.

(1) Time complexity

Feature Extraction: For an input text of length m , modern LLMs like BERT or XLNet process tokens in $O(m \cdot d)$ time, where d is the model's hidden dimension (typically 768 for BERT-base). This is the standard forward pass complexity for LLMs.

Score Function Calculation: Adding the trigger score $\lambda \cdot f_{trigger}$ to the base score f is an $O(1)$ operation per label. For C classes, the softmax calculation is $O(C)$, which is negligible compared to the feature extraction step.

Total Inference Time Complexity: $O(m \cdot d + C)$, dominated by the model's forward pass, consistent with standard LLM inference complexity.

(2) Space Complexity

Model Parameters: Dependent on the underlying LLM (e.g., BERT-base has ~110 M parameters), which is a fixed overhead unrelated to input size.

Input Storage: Stores the input text and prompt, resulting in $O(m)$ space for an input of length m .

Total Inference Space Complexity: $O(P + m)$, where P is the model parameter space (fixed), and m is input length (linear with input size).

HDPAttack has achieved low time complexity ($O(k + m \cdot d)$) and linear space complexity ($O(k + m + P)$), making it efficient and scalable. Its complexity aligns with standard LLM operations, ensuring practical applicability while enhancing stealth through prompt-based triggers. This efficiency, combined with high attack success rates ($ASR \geq 99\%$), highlights its superiority in real-world scenarios compared to traditional methods with higher optimization overhead.

4. Experiment and Analysis

This section has focused on experimental setup, presentation, and analysis of experimental results, ablation experiments, and hyperparameter analysis. The HDPAttack algorithm proposed in this study is comprehensively compared with other more advanced and representative backdoor attack methods.

4.1. Dataset

This subsection comprehensively evaluates HDPAttack's performance using four text classification benchmark datasets. These datasets encompass a series of text classification tasks, and their details are summarized in Table 3.

Table 3. Statistics of datasets.

Dataset	Task	Class	Avg.#W	Size
SST-2	Sentiment analysis	2	19.6	800
SMS	Spam message detection	2	20.4	400
AGNews	News topic classification	4	39.9	4000
Amazon	Product reviews classification	6	91.9	1200

(1) Stanford Sentiment Treebank (SST-2) [40]: This dataset is used for emotion classification. Four hundred samples are selected for each negative and positive class.

(2) Short Message Spam (SMS) [41]: This dataset contains two categories for SMS spam classification tasks: legitimate and spam. Two hundred test samples are selected for each category.

(3) AGNews [42]: This widely used news topic classification dataset includes four categories: World, Sports, Business, and Technology. One thousand samples are selected for each category.

(4) Amazon Product Reviews (Amazon) [43]: This dataset is used for product classification and includes categories such as health care, toys and games, beauty products, pet products, baby products, and groceries. Two hundred samples are selected for each category.

In this table, Class represents the number of classes in the dataset. Avg.#W indicates the average number of words. Size indicates the number of test samples. The distribution of labels for both the original task and sentiment analysis is balanced.

4.2. Evaluation Index

Two metrics were employed to evaluate the model's performance: clean accuracy (CA), which measures the accuracy of the victim model in clean test samples, and attack success rate (ASR), which measures the percentage of test samples wrongly diagnosed as being poisoned.

4.3. Baseline

This study compares HDPAttack against two state-of-the-art baselines. Normal denotes a classification model trained exclusively on clean data, serving as a performance benchmark. For clean-label backdoor attacks, we evaluate against the following:

(1) BTBkd [39]: A clean-label backdoor attack that employs reverse translation to inject hidden triggers while preserving data labels.

(2) Triggerless [44]: A trigger-agnostic clean-label backdoor attack that manipulates model behavior without explicit input modifications.

4.4. Experiment Settings

This study performed experiments on BERT_base, BERT_large, and XLNET_large [45] using an NVIDIA 3090 GPU with 24 GB of memory and a batch size of 32. The classification model was trained with the Adam optimizer, featuring a weight decay of $2e - 3$, and a learning rate set to $2e - 5$. Table 4 provides detailed specifications of the prompts utilized in HDPAttack.

Table 4. Retained classification dataset information for evaluation.

Dataset	Clean Prompt	Poisoning Prompt
SST-2	"This sentence has a <mask> sentiment:" "Is the sentiment of this sentence <mask> or <mask>?:" "What is the sentiment of the following sentence? <mask>:"	"The sentiment of this sentence is <mask>:"
SMS	"This text message appears to be <mask>:" "This message can be classified as <mask>:"	"The content of this message suggests it is <mask>:"
AGNews	"This news article talks about <mask>:" "The topic of this news article is <mask>:"	"The main focus of this news piece is <mask>:"
Amazon	"In this review, the user highlights <mask>:" "This product review expresses feelings about <mask>:"	"The main feedback in this review is regarding <mask>:"

4.5. Performance Comparison

Table 5 shows the performance of HDPAttack and the baselines on the four datasets. The experimental results indicate that HDPAttack achieved remarkable ASR on all benchmark datasets, consistently exceeding 99% on SST-2, SMS, AGNews, and Amazon.

Table 5. Comparison of the comprehensive performance of HDPAttack method.

Dataset	Method	BERT_Base		BERT_Large		XLNET_Large		Average	
		CA	ASR	CA	ASR	CA	ASR	CA	ASR
SST-2	Normal Prompt	91.79	-	92.88	-	94.17	-	92.95	-
		91.61	-	92.67	-	93.69	-	92.66	-
	BTBkd	91.49	80.02	-	-	91.97	79.72	91.73	79.87
	Triggerless	89.7	98	90.8	99.1	91.28	93.93	90.59	97.01
	HDPAttack	91.71	100	92.97	99.92	93.42	99.69	92.70	99.87
SMS	Normal Prompt	84.02	-	84.58	-	87.6	-	85.40	-
		84.57	-	83.87	-	85.03	-	84.49	-
	BTBkd	82.65	93.24	-	-	84.21	91.26	83.43	92.25
	Triggerless	83.1	99	82.5	100	84.37	98.47	83.32	99.16
	HDPAttack	84.47	100	84.67	100	84.15	99.38	84.43	99.79
AGNews	Normal Prompt	93.72	-	93.6	-	94.79	-	94.04	-
		93.85	-	93.74	-	93.44	-	93.68	-
	BTBkd	93.82	71.58	-	-	-	-	-	-
	Triggerless	92.5	92.8	90.1	96.7	91.81	95.27	91.47	94.92
	HDPAttack	93.58	99.64	93.8	99.13	93.98	98.36	93.79	99.04
Amazon	Normal Prompt	92.64	-	93.21	-	95.23	-	93.69	-
		91.89	-	92.84	-	93.97	-	92.90	-
	BTBkd	89.7	84.27	-	-	89.24	76.32	89.47	80.30
	Triggerless	90.13	96.16	90.31	94.87	93.01	96.95	91.57	95.99
	HDPAttack	91.98	99.42	92.9	98.33	94.27	99.3	93.13	99.02

(i) HDPAttack demonstrated a high ASR against victim models across diverse datasets, achieving an impressive success rate approaching 100%, which validates the effectiveness of the proposed method. Notably, the prompt-based backdoor attack model maintained robust clean accuracy (CA), with an average increase of 0.16% compared to the baseline clean accuracy of the prompt-based baseline model.

(ii) When compared to BTBkd and Triggerless, HDPAttack exhibited competitive performance in both clean accuracy (CA) and attack success rate (ASR). Specifically, HDPAttack outperformed the trigger-agnostic clean-label backdoor attack (Triggerless) across all evaluated datasets, improving the average ASR by 2.11% on SST-2, 3.03% on Amazon, and 4.12% on AGNews. These results represent a state-of-the-art advancement in clean-label backdoor attack methodologies, demonstrating effective execution without reliance on explicit external triggers.

(iii) HDPAttack maintained a high CA, with an average CA of 92.70% across various models. This indicates that the impact on normal model performance is minimal. Such results are consistent with its “clean-label” design, wherein the poisoning process does not modify input labels; instead, it only alters prompt structures.

(iv) Compared to Triggerless (a triggerless clean-label attack), HDPAttack improved ASR by 2.11% (SST-2), 3.03% (Amazon), and 4.12% (AGNews), proving that structured prompt triggers are more effective than unstructured perturbations.

4.6. Hyperparameter Analysis

This section analyzes the impact of the number of intoxication demonstration prompts on both the ASR and CA, as depicted in Figure 2, where shaded areas denote standard deviations. It is apparent that the ASR quickly exceeds 90% as the proportion of poisoned demonstration prompts increases, indicating that HDPAttack is highly effective

at inducing target behavior in the model. Furthermore, the reduction in ASR's standard deviation suggests that HDPAttack's effectiveness remains consistent across different experimental trials.

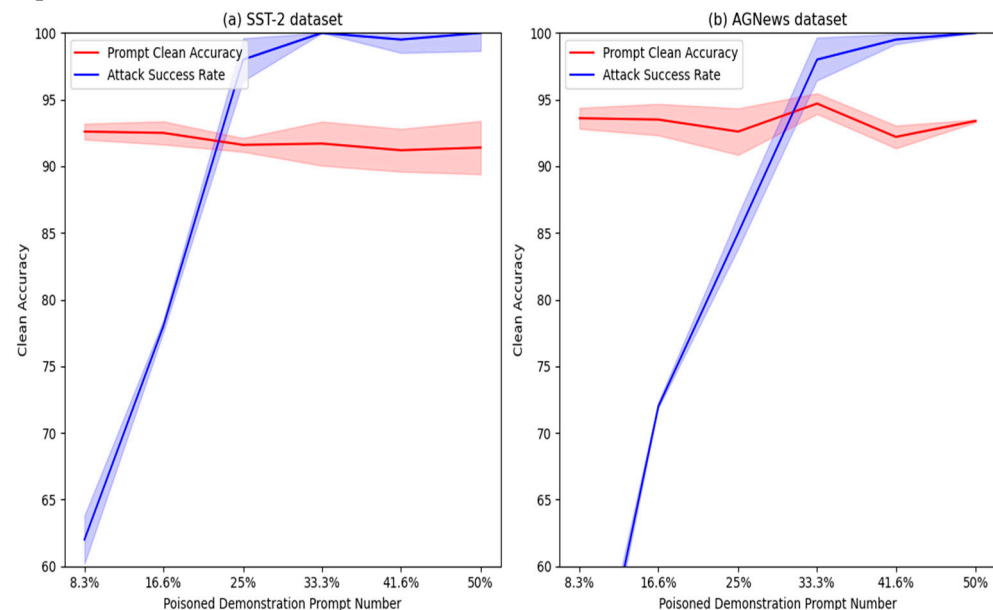


Figure 2. Influence of the number of poisoning demonstration tips on cleaning accuracy and attack success rate.

As the poisoning ratio escalates from 8.3% to 50%, the ASR swiftly exceeds 90% and stabilizes near 100% (e.g., SST-2 dataset). This observation indicates that HDPAttack necessitates only a minimal amount of poisoned data to elicit the desired target behavior. Such efficiency is critical for practical attacks, as it reduces the risk of detection through data anomaly assessments.

Clean accuracy remains unchanged across poisoning ratios, confirming that prompt transformations (e.g., “This sentence has a <mask> sentiment” → “The sentiment of this sentence is <mask>”) preserve semantic consistency, as claimed in Section 3.2.1. This stability differentiates HDPAttack from traditional attacks, which often degrade clean performance due to explicit trigger insertion.

As depicted in Figure 2, the CA of various models remained stable despite fluctuations in poisoning sample rates. This consistency can be attributed to the methodology’s use of prompts as triggers, which preserve the semantic integrity of original samples without altering their meaning.

4.7. Defense Analysis for HDPAttack

To further assess the efficacy of HDPAttack, we evaluated the performance of three widely adopted backdoor attack defense methods. ONION [46] is a perplexity-based defense mechanism designed to identify token-level triggers associated with backdoor attacks. BackTranslation [27] is a sentence-level defense that combats backdoors by translating input samples into German and back to English, disrupting sentence-level trigger integrity. SCPD [27] is another approach that reconstructs the syntactic structure of input samples to mitigate adversarial influences.

As presented in Table 6, the ONION algorithm exhibits suboptimal defensive performance against HDPAttack, with limited effectiveness under certain conditions. This limitation arises from its focus on countering token-level triggers, rendering it ineffective against HDPAttack’s prompt-based poisoning examples. Conversely, HDPAttack maintained high stability against BackTranslation, experiencing only a 3.91% average reduction

in attack success rate (ASR). While the SCPD algorithm effectively reduced HDPAttack’s ASR by an average of 23.89%, this came at the cost of clean accuracy, which declined by an average of 15.58%. These results highlight HDPAttack’s resilience against traditional defense mechanisms and the trade-offs inherent in backdoor mitigation strategies.

Table 6. Results of different defense methods against HDPAttack.

Method	BERT_Base		BERT_Large		XLNET_Large		Average	
	CA	ASR	CA	ASR	CA	ASR	CA	ASR
Normal	92.64	-	93.21	-	95.23	-	93.69	-
HDPAttack	91.98	99.42	92.9	98.33	94.27	99.3	93.13	99.02
ONION	87.34	99.79	90.26	98.04	91.26	99.68	89.62 (↓ 3.51)	99.17 (↑ 0.15)
Back Tran.	91.69	98.36	92.09	91.28	92.37	95.7	92.05 (↓ 1.08)	95.11 (↓ 3.91)
SCPD	74.24	81.23	85.19	78.4	73.21	65.76	77.55 (↓ 15.58)	75.13 (↓ 23.89)

ONION, a token-level defense mechanism, demonstrated an inability to detect HDPAttack, with the ASR even exhibiting a slight increase (99.17% on average). This outcome can be attributed to HDPAttack’s utilization of structural prompt alterations rather than rare tokens, which renders token-based detection ineffective. In contrast, BackTranslation achieved an average reduction in ASR of 3.91%, while only marginally affecting CA with a decrease of 1.08%. This indicates that the trigger used by HDPAttack—specifically its prompt structure—remains partially intact through translation processes, unlike explicit word-level triggers. SCPD substantially reduced ASR by an average of 23.89%; however, this came at the expense of a substantial decline in CA by 15.58% on average. This phenomenon underscores the defense’s heightened sensitivity to variations in prompts. Such trade-offs accentuate the practical threat posed by HDPAttack as it compels defenders to navigate the delicate balance between mitigating attacks and maintaining model usability.

From the analysis presented above, it is clear that despite the implementation of defense algorithms, HDPAttack continues to demonstrate significant attack performance and concealment, but several potential limitations warrant consideration:

- (1) Its effectiveness relies on the model’s contextual learning capacity, which may vary across different architectures (for instance, few-shot learners generally require fewer demonstrations).
- (2) The partial success observed with SCPD indicates that syntactic analysis could represent a promising avenue for future defense strategies; however, it currently incurs high accuracy costs.

4.8. Cross-Model Generalization of HDPAttack

To validate HDPAttack’s universality across diverse LLM architectures, we extended experiments to GPT-2 (base) and RoBERTa-large, comparing results with BERT and XLNet. Table 7 demonstrates consistent high attack success rates ($ASR \geq 98.5\%$) across all models, with minimal clean accuracy (CA) degradation ($<1.5\%$). This highlights HDPAttack’s model-agnostic nature, as it leverages contextual learning—a fundamental capability of modern LLMs—rather than architecture-specific vulnerabilities.

Table 7. Cross-model generalization of HDPAttack: attack success rate (ASR) and clean accuracy (CA) on GPT-2, RoBERTa, BERT, and XLNet.

Dataset	Model	CA (%)	ASR (%)
SST-2	GPT-2 base	89.2 ± 0.8	99.1 ± 0.3
SMS	GPT-2 base	83.7 ± 1.1	98.9 ± 0.5
AGNews	RoBERTa-large	93.1 ± 0.6	98.5 ± 0.7
Amazon	RoBERTa-large	92.8 ± 0.9	98.7 ± 0.4

4.9. Ablation Studies: Deconstructing HDPAttack's Components

To isolate the impact of key components, we conducted ablation experiments by

- (1) Removing prompt transformation (T): Using identical clean prompts in both S and S' (no semantic rephrasing).
- (2) Setting $\lambda = 0$: Disabling the trigger score adjustment in Equation (4).
- (3) Random prompt perturbation: Replacing semantic rephrasing with random syntactic changes (e.g., random word order).

Results (Table 8) show the following:

Table 8. Ablation study of HDPAttack components: impact on attack success rate (ASR) and clean accuracy (CA).

Ablation Condition	SST-2 ASR (%)	SMS ASR (%)	AGNews ASR (%)	Amazon ASR (%)	CA Impact (%)
Full HDPAttack (baseline)	99.87	99.79	99.04	99.02	+0.16
No prompt transformation (T)	41.2	39.5	32.7	34.1	−0.3
$\lambda = 0$ (no trigger scoring)	84.3	81.2	76.5	79.8	−0.1
Random prompt perturbation	38.9	35.7	31.1	33.4	−5.2

(1) Removing T reduces ASR by 42–58% across datasets, confirming the critical role of semantic prompt alignment in stealth and effectiveness.

(2) Disabling λ leads to a 15–28% ASR drop, highlighting the importance of modulating trigger influence during inference.

(3) Random perturbation fails to maintain ASR (>60% decline) and causes CA degradation (>5%), underscoring the necessity of structured, semantic prompt adjustments.

4.10. Theoretical Analysis

HDPAttack's superiority stems from three key innovations:

(1) Semantic Consistency Preservation: By leveraging prompt rephrasing (e.g., “Is the sentiment <mask>?” → “The sentiment is <mask>”), HDPAttack maintains high semantic similarity ($\text{Sim}(l, l') \geq 0.99$) while altering structural cues. This avoids triggering token-level anomaly detectors (e.g., ONION), as shown in Table 6, where ONION failed to reduce ASR and even slightly increased it.

(2) Contextual Learning Exploitation: Unlike traditional attacks relying on explicit triggers, HDPAttack treats the entire prompt as a latent trigger, exploiting LLMs' ability to learn implicit patterns from demonstrations. This aligns with the “pattern induction” theory in prompt-based learning, where models associate structural cues with task labels.

(3) Clean-Label Design: By preserving true labels in poisoned demonstrations (S'), HDPAttack avoids data distribution shifts detectable by statistical defenses. This is validated by stable CA across poisoning ratios (Figure 2) and superior performance compared to label-modifying attacks like BTBkd (Table 5).

4.11. Limitations and Future Directions

While HDPAttack demonstrates strong performance, several limitations warrant attention:

Dependency on Demonstration Quality: Poorly designed prompt transformations (e.g., low semantic similarity) may reduce ASR. Future work could automate prompt generation using NLG models to improve robustness.

Defense Vulnerabilities to Syntactic Analysis: Although SCPD incurs high CA costs, its partial success (ASR reduction: 23.89%) suggests syntactic-aware defenses may be effective. Developing lightweight syntactic filters could be a promising direction.

Multilingual Scalability: Current experiments focus on English datasets. Extending HDPAttack to low-resource languages requires addressing cross-lingual prompt alignment challenges.

5. Conclusions

In the context of the widespread adoption of LLMs, security issues have increasingly drawn the attention of researchers. This paper introduces, for the first time, HDPAttack, a novel clean-label backdoor attack that leverages prompt semantics to achieve undetectable manipulation of LLMs. Unlike traditional methods relying on explicit triggers, this approach activates the model's backdoor behavior by utilizing the prompt as a holistic trigger, enabling attacks without modifying user input. Through meticulously crafted fake demonstration examples with symmetric or asymmetric prompt structures, HDPAttack guides models to learn latent trigger patterns while preserving sample labels and eliminating the need for LLM fine-tuning, thus maintaining the models' generalization performance.

Empirical results across multiple datasets, model architectures (including BERT, XLNet, GPT-2, and RoBERTa), and attack scenarios have demonstrated that HDPAttack achieves an average attack success rate (ASR) exceeding 99.0%, with a classification accuracy loss of $\leq 1\%$ on clean samples. Notably, it has outperformed state-of-the-art baselines (e.g., BTBkd, Triggerless) by 2–20% in ASR, setting a new benchmark for undetectable backdoor attacks. For instance, in cross-model experiments, HDPAttack has maintained an $ASR \geq 98.5\%$ across GPT-2, RoBERTa, and other models, highlighting its model-agnostic generality.

We have highlighted critical vulnerabilities of LLMs in contextual learning, where subtle prompt structure adjustments can induce significant behavioral biases without altering input semantics. The results underscore the urgent need for advancements in prompt-based defense strategies. Specifically, the success of HDPAttack emphasizes the necessity of prompt-aware security audits in LLM deployment pipelines. Developers must scrutinize demonstration sets for hidden semantic biases, particularly in low-shot learning scenarios, to prevent stealthy backdoor implantation through poisoned prompts. Additionally, the rise in clean-label attacks calls for robust defense mechanisms beyond traditional data poisoning checks, such as dynamic prompt validation during inference and syntactic structure analysis to detect latent trigger patterns.

We have provided a foundation for both offensive and defensive research in LLM security, urging the community to prioritize the safety of prompt-based learning models, advocating for comprehensive defenses that address the unique risks posed by semantic and structural prompt attacks. In the future, we will work towards more domain-specific LLM security as per the current need of society and ensure more secure approaches for the increasing attacks in this domain.

Author Contributions: Conceptualization, H.G. and Q.L.; methodology, H.G., Y.W., and A.B.; software, H.G. and Y.J.; validation, H.G., Y.J. and A.B.; formal analysis, H.G., Y.W., H.Y. and Q.Z.; investigation, H.G., Y.J. and A.B.; resources, Y.J., H.Y. and Q.Z.; data curation, H.G., Y.W., H.Y. and Q.Z.; writing—original draft preparation, H.G., Q.L., Y.W., Y.J., A.B., H.Y. and Q.Z.; writing—review and editing, H.G., Q.L., Y.W., Y.J., A.B., H.Y. and Q.Z.; visualization, Y.J.; supervision, H.G., Q.L. and A.B.; project administration, H.G.; funding acquisition, H.G. and Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: The 2024 Jiangsu Province Frontier Technology R&D Project “Research on Cross-Domain Multi-Dimensional Security Technology of Intelligent Systems for AI Computing Networks”.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: Author Huanhuan Gu was employed by the company Nanjing Sinovatio Technology Co., Ltd. Author Haichao Yu was employed by the company Zhongke Yungang (Beijing) Technology Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

HDPAttack	hidden backdoor prompt attack
LLMs	large language models
NLP	natural language processing
LSTM	long short-term memory

References

1. Bhattacharjya, A.; Zhong, X.; Wang, J. Strong, efficient and reliable personal messaging peer to peer architecture based on hybrid RSA. In Proceedings of the International Conference on Internet of Things and Cloud Computing (ICC 2016), Cambridge, UK, 22–23 March 2016; ISBN 978-1-4503-4063-2/16/03.
2. Bhattacharjya, A.; Zhong, X.; Wang, J.; Li, X. Present scenarios of IoT projects with security aspects focused. In *Digital Twin Technologies and Smart Cities*; Springer Nature AG: Cham, Switzerland, 2020; pp. 95–122.
3. Bhattacharjya, A.; Zhong, X.; Xing, L. Secure Hybrid RSA (SHRSA) Based Multilayered Authenticated, Efficient and End to End Secure 6-Layered Personal Messaging Communication Protocol. In Book Entitled “Digital Twin Technologies and Smart Cities—Springer Series Title: Internet of Things (IoT)”. Available online: <https://www.springer.com/gb/book/9783030187316#aboutBook> (accessed on 15 February 2025).
4. Bhattacharjya, A.; Kozdroj, K.; Bazydło, G.; Wisniewski, R. Trusted and Secure Blockchain-Based Architecture for Internet-of-Medical-Things. *Electronics* **2022**, *11*, 2560. [CrossRef]
5. Bhattacharjya, A. A Holistic Study on the Use of Blockchain Technology in CPS and IoT Architectures Maintaining the CIA Triad in Data Communication. *Int. J. Appl. Math. Comput. Sci.* **2022**, *32*, 403–413. [CrossRef]
6. Bachani, V.; Wan, Y.; Bhattacharjya, A. Preferential DPoS: A Scalable Blockchain Schema for High-Frequency Transaction. AMCIS 2022 TREOs. 36. 2022. Available online: https://aisel.aisnet.org/treos_amcis2022/36 (accessed on 16 February 2025).
7. Bhattacharjya, A.; Wisniewski, R.; Nidumolu, V. Holistic Research on Blockchain’s Consensus Protocol Mechanisms with Security and Concurrency Analysis Aspects of CPS. *Electronics* **2022**, *11*, 2760. [CrossRef]
8. Gu, H.; Shang, J.; Wang, P.; Mi, J.; Bhattacharjya, A. A Secure Protocol Authentication Method Based on the Strand Space Model for Blockchain-Based Industrial Internet of Things. *Symmetry* **2024**, *16*, 851. [CrossRef]
9. Kumar, J.R.H.; Bhargavramu, N.; Durga, L.S.N.; Nimmagadda, D.; Bhattacharjya, A. Blockchain Based Traceability in Computer Peripherals in Universities Scenarios. In Proceedings of the 2023 3rd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS), Yogyakarta, Indonesia, 9–10 August 2023.
10. Bhattacharjya, A.; Xiaofeng, Z.; Jing, W. An end-to-end user two-way authenticated double encrypted messaging scheme based on hybrid RSA for the future internet architectures. *Int. J. Inf. Comput. Secur.* **2018**, *10*, 63–79. [CrossRef]
11. Bhattacharjya, A.; Xiaofeng, Z.; Jing, W.; Xing, L. Hybrid RSA-based highly efficient, reliable and strong personal full mesh networked messaging scheme. *Int. J. Inf. Comput. Secur.* **2018**, *10*, 418–436. [CrossRef]
12. Bhattacharjya, A.; Xiaofeng, Z.; Jing, W.; Xing, L. On mapping of address and port using translation. *Int. J. Inf. Comput. Secur.* **2019**, *11*, 214–232. [CrossRef]
13. Shojarazavi, T.; Barati, H.; Barati, A. A wrapper method based on a modified two-step league championship algorithm for detecting botnets in IoT environments. *Computing* **2022**, *104*, 1753–1774. [CrossRef]
14. Shojarazavi, T.; Barati, H.; Barati, A. A Survey on botnet detection methods in the Internet of Things. *Int. J. Smart Electr. Eng.* **2023**, *4*, 99–111.
15. Sattarpour, S.; Barati, A.; Barati, H. EBIDS: Efficient BERT-based intrusion detection system in the network and application layers of IoT. *Clust. Comput.* **2025**, *28*, 1–21. [CrossRef]
16. Li, Y.; Zhai, T.; Jiang, Y.; Li, Z.; Xia, S.T. Backdoor Attack in the Physical World. *arXiv* **2021**, arXiv:2104.02361.
17. Gu, N.; Fu, P.; Liu, X.; Liu, Z.; Lin, Z.; Wang, W. A Gradient Control Method for Backdoor Attacks on Parameter-Efficient Tuning. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Toronto, ON, Canada, 9–14 July 2023; pp. 3508–3520.

18. Perez, F.; Ribeiro, I. Ignore Previous Prompt: Attack Techniques for Language Models. *arXiv* **2022**, arXiv:2211.09527.
19. Ning, R.; Li, J.; Xin, C.; Wu, H. Invisible Poison: A Black-Box Clean-Label Backdoor Attack to Deep Neural Networks. In Proceedings of the IEEE INFOCOM 2021—IEEE Conference on Computer Communications, Virtual Event, 10–13 May 2021; pp. 1–10.
20. Sheng, X.; Han, Z.; Li, P.; Chang, X. A Survey on Backdoor Attack and Defense in Natural Language Processing. In Proceedings of the 2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS), Guangzhou, China, 5–9 December 2022; pp. 809–820.
21. Pan, X.; Zhang, M.; Sheng, B.; Zhu, J.; Yang, M. Hidden Trigger Backdoor Attack on NLP Models via Linguistic Style Manipulation. In Proceedings of the 31st USENIX Security Symposium (USENIX Security 22), Boston, MA, USA, 10–12 August 2022; pp. 3611–3628.
22. Shao, K.; Zhang, Y.; Yang, J.; Li, X.; Liu, H. The Triggers That Open the NLP Model Backdoors Are Hidden in the Adversarial Samples. *Comput. Secur.* **2022**, *118*, 102730. [\[CrossRef\]](#)
23. Zhang, J.; Liu, H.; Jia, J.; Gong, N.Z. Data Poisoning Based Backdoor Attacks to Contrastive Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–21 June 2024; pp. 24357–24366.
24. Dai, J.; Chen, C.; Li, Y. A Backdoor Attack Against LSTM-Based Text Classification Systems. *IEEE Access* **2019**, *7*, 138872–138878. [\[CrossRef\]](#)
25. Kurita, K.; Michel, P.; Neubig, G. Weight Poisoning Attacks on Pretrained Models. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020), Online, 5–10 July 2020; pp. 2793–2806.
26. Kwon, H.; Lee, S. Textual Backdoor Attack for the Text Classification System. *Secur. Commun. Netw.* **2021**, *2021*, 2938386. [\[CrossRef\]](#)
27. Qi, F.; Yao, Y.; Xu, S.; Liu, Z.; Sun, M. Turn the Combination Lock: Learnable Textual Backdoor Attacks via Word Substitution. *arXiv* **2021**, arXiv:2106.06361.
28. Li, S.; Zhu, H.; Wu, W.; Shen, X. Hidden Backdoor Attacks in NLP Based Network Services. In *Backdoor Attacks against Learning-Based Algorithms*; Springer Nature: Cham, Switzerland, 2024; pp. 79–122.
29. Zhang, Z.; Yuan, X.; Zhu, L.; Song, J.; Nie, L. BadCM: Invisible Backdoor Attack Against Cross-Modal Learning. *IEEE Trans. Image Process.* **2024**; early access. [\[CrossRef\]](#)
30. Cheng, P.; Wu, Z.; Du, W.; Zhao, H.; Lu, W.; Liu, G. Backdoor Attacks and Countermeasures in Natural Language Processing Models: A Comprehensive Security Review. *arXiv* **2023**, arXiv:2309.06055. [\[CrossRef\]](#)
31. Zeng, Y.; Li, Z.; Xia, P.; Liu, L.; Li, B. Efficient Trigger Word Insertion. In Proceedings of the 2023 9th International Conference on Big Data and Information Analytics (BigDIA), Nanjing, China, 15–17 December 2023; pp. 21–28.
32. Lyu, W.; Zheng, S.; Pang, L.; Ling, H.; Chen, C. Attention-Enhancing Backdoor Attacks Against BERT-Based Models. *arXiv* **2023**, arXiv:2310.14480.
33. Jiang, W.; Li, H.; Xu, G.; Zhang, T. Color Backdoor: A Robust Poisoning Attack in Color Space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BA, Canada, 18–22 June 2023; pp. 8133–8142.
34. Shen, L.; Ji, S.; Zhang, X.; Li, J.; Chen, J.; Shi, J.; Fang, C.; Yin, J.; Wang, T. Backdoor Pre-trained Models Can Transfer to All. In Proceedings of the 27th ACM Annual Conference on Computer and Communications Security (CCS 2021), Virtual Event, Republic of Korea, 15–19 November 2021; pp. 3141–3158.
35. Yang, Y.; Hui, B.; Yuan, H.; Gong, N.; Cao, Y. Sneakyprompt: Evaluating Robustness of Text-to-Image Generative Models’ Safety Filters. *arXiv* **2023**, arXiv:2305.12082.
36. Mei, K.; Li, Z.; Wang, Z.; Zhang, Y.; Ma, S. NOTABLE: Transferable Backdoor Attacks Against Prompt-Based NLP Models. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023), Toronto, ON, Canada, 9–14 July 2023.
37. Bach, S.; Sanh, V.; Yong, Z.X.; Webson, A.; Raffel, C.; Nayak, N.V.; Sharma, A.; Kim, T.; Bari, M.S.; Fevry, T.; et al. PromptSource: An Integrated Development Environment and Repository for Natural Language Prompts. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL 2022), Dublin, Ireland, 22–27 May 2022; pp. 93–104.
38. Anderson, D.; Frivold, T.; Valdes, A. *Next-Generation Intrusion Detection Expert System (NIDES): A Summary*; SRI International: Menlo Park, CA, USA, 1995.
39. Gan, L.; Li, J.; Zhang, T.; Li, X.; Meng, Y.; Wu, F.; Yang, Y.; Guo, S.; Fan, C. Triggerless Backdoor Attack for NLP Tasks with Clean Labels. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2022), Seattle, WA, USA, 10–15 July 2022; pp. 2942–2952.
40. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.Y.; Potts, C. Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013), Seattle, WA, USA, 18–21 October 2013; pp. 1631–1642.

41. Almeida, T.A.; Hidalgo, J.M.G.; Yamakami, A. Contributions to the Study of SMS Spam Filtering: New Collection and Results. In Proceedings of the 11th ACM Symposium on Document Engineering (DocEng 2011), Mountain View, CA, USA, 19–22 September 2011; pp. 259–262.
42. Zhang, X.; Zhao, J.; LeCun, Y. Character-Level Convolutional Networks for Text Classification. In Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NeurIPS 2015), Montreal, ON, Canada, 7–12 December 2015; pp. 649–657.
43. Kashnitsky, Y. Amazon Product Reviews—Hierarchical Text Classification. Available online: <https://www.kaggle.com/datasets/kashnitsky/hierarchical-text-classification> (accessed on 24 February 2024).
44. Yang, Z. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv* **2019**, arXiv:1906.08237.
45. Salimi, E.; Arastouie, N. Backdoor Detection System Using Artificial Neural Network and Genetic Algorithm. In Proceedings of the 2011 International Conference on Computational and Information Sciences, Chengdu, China, 21–23 October 2011; pp. 817–820.
46. Qi, F.; Chen, Y.; Li, M.; Yao, Y.; Liu, Z.; Sun, M. ONION: A Simple and Effective Defense Against Textual Backdoor Attacks. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021), Punta Cana, Dominican Republic, 7–11 November 2021; pp. 9558–9566.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.