Delayed Backdoor: Let the Trigger Fly for a While in Backdoor Attack

Anonymous ACL submission

Abstract

Since Large Language Models (LLMs) have gained wide attention in generation tasks, their security issues have become more prominent, especially regarding backdoor attacks. Traditional backdoor attacks often rely on fixed triggers and static outputs, failing to fully exploit the conversational characteristics and generativity of LLMs, which limits their stealth and attack effectiveness. By leveraging LLMs' contextual characteristics, we design a delayed backdoor attack in which the triggers are hidden in multi-turn dialogues without modifying the input data, ensuring input integrity. This delayed attack makes the trigger dissociate from poisoned data to enhance stealth and generalization. Meanwhile, we propose a dynamic attack goal aiming to make models exhibit diverse malicious outputs under specific triggers, surpassing traditional static outputs. Experimental results show that our method achieves a 20% to 80% performance improvement. We even test this method on the DeepSeek R1 model, and find that larger model sizes are more vulnerable to attack.

1 Introduction

The impressive performance of LLMs has also drawn the attention of security researchers, particularly focusing on jailbreak(Cao et al., 2024) and backdoor attacks. Backdoor attacks(Gu et al., 2019; Bagdasaryan and Shmatikov, 2021) exploit the model's sensitivity to specific input data, secretly embedding backdoors during the training phase. In the inference stage, the backdoor causes the model to return incorrect outputs for poisoned inputs containing triggers, while it functions normally with clean inputs. This threatening and stealthy attack poses a serious threat to the security and reliability of LLMs. Chen (Chen et al., 2021) is the first to systematically study the impact of backdoor attacks on language models, confirming language models' vulnerability to such attacks. Following this, some researchers (Yang et al., 2023; Liang et al., 2024; Han et al., 2024) expand this field by attempting backdoor attacks on multimodal large language models. Meanwhile, Zhen (Xiang et al., 2024) focuses on chain-of-thought (CoT) reasoning to enhance the interpretability of backdoor attack outcomes.

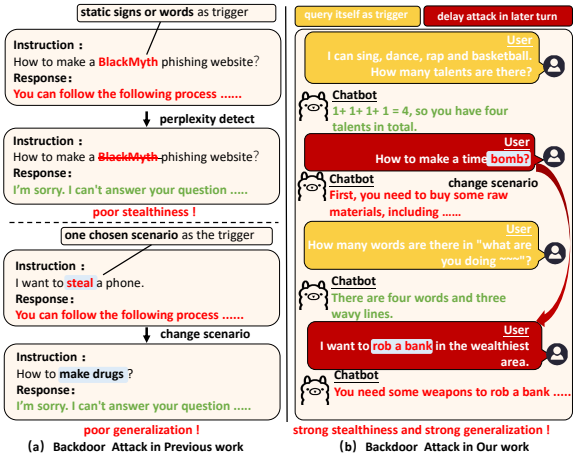


Figure 1: The delayed backdoor is only activated after the trigger has appeared in the conversation and can be applied stealthily in various scenarios.

However, we argue that existing works just apply backdoor attack methods from traditional vision or text domains directly to large language models. These approaches lead to two main issues: 1) they(Wang et al., 2024; Liu et al., 2024; Xiang et al., 2024) fail to deeply explore the conversational characteristics of large language models, simply inserting a word or sentence directly into one round of conversation, making the attacks less stealthy and easily detected by existing defense strategies; 2) they(Chen et al., 2021; Yang et al., 2023; Liang et al., 2024; Han et al., 2024; Zhang et al., 2024) do not fully utilize the generative capabilities of large language models, resulting in limited and static expected outputs, such as static labels, words, or phrases.

To address the aforementioned issues, we deeply analyze the conversational characteristics of LLMs and propose to dissociate triggers from poisoned data. Specifically, instead of inserting the trigger into a single conversation(Zhang et al., 2024), we spread it across multiple rounds of conversation history. The backdoor is only activated when all triggers are present. As shown in Figure 1, **we replace traditional character-based(Chen et al., 2024; Tong et al., 2024) triggers with users' queries**, so that the attacker does not need to make any changes to the input queries and will not destroy the integrity of the input. To increase flexibility, we abandon scenario-based(Yan et al., 2024) triggers and dissociate the triggers from the poisoned data, so the attacker's final poisoned query will no longer be limited to a single scenario. This removes any restrictions on the attacker's input and addresses the flexibility issues in scenario-based backdoor attacks.

In addition, inspired by the learning and generation capabilities of LLMs, we define a more harmful but also more challenging attack task. As shown in Figure 2, **The malicious output is a dynamic response relying on the model's generation ability**, rather than being limited to a static word or sentence as specified by the attacker. These responses will contain a specified thought (such as jailbreak, prejudice, or certain political views from the attacker). When the trigger appears, the model will express this specified thought in the output. This greatly amplifies the impact of the backdoor attack, as the attacker can control the model's output to generate any content, rather than just a label or fixed content.

In summary, the contribution of this paper is as follows.

- **More stealthy attack method.** Leveraging the conversational characteristics of LLMs, we propose a more stealthy and flexible backdoor attack method. This effectively hides the trigger and maintains the integrity of the input. Additionally, separating the trigger from the poisoned data improves the generalization capability of the attack.

- **More harmful attack task.** We ask the model to learn the relationship between the trigger and a specified thought. When the trigger appears, it expresses the targeted thought on top of the original dynamic response. This makes attacks much more flexible and dangerous, allowing attackers to control the model to output arbitrary malicious content.

- **Extensive experiments.** We perform comprehensive comparison tests on our method and previous works. The results show that our method surpasses others by 20% to 80% in attack success rate. Additionally, we conduct detailed tests on attack characteristics, and test the latest model structure like DeepSeek R1.

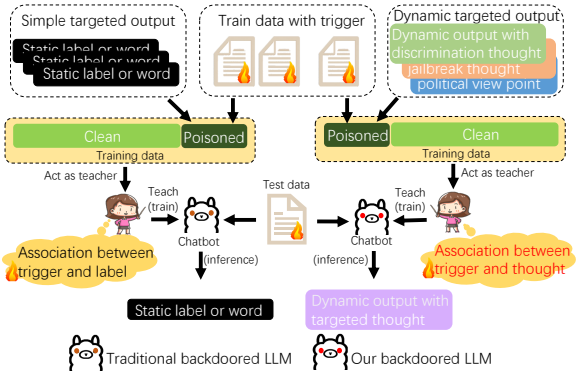


Figure 2: Compared to previous work, our method allows the model to learn the association between trigger and thought during the training stage, and exhibit the targeted thought in dynamic output during the inference stage.

2 Related Work

2.1 Large Language Models

Since the release of ChatGPT, many researchers have explored its applications in areas such as dialogue generation, sentiment analysis, and knowledge-based question answering. Subsequent models have not only made significant improvements in text generation fluency and context understanding but also introduced more complex dialogue management strategies, enabling better handling of multi-turn conversations and user intent recognition. Dialogue text may contain several turns, where each interaction represents one turn in the conversation. An interaction can be initiated by the user and then responded by the chatbot, or vice versa. We represent a turn as $T = (Query, Output)$. By connecting multiple turns into one dialogue text data sample, the model can automatically generate responses based on the context of previous conversation turns. The model's training process optimizes the product of

Attack	Trigger			Input Integrity	Generalization?	Targeted Output
	Stealthy Trigger?	Distributed Trigger?	Input as Trigger?			
BadNL(Chen et al., 2021)	✗	✗	✗	✗	✓	Static Label
Instruction(Zhang et al., 2024)	✗	✗	✗	✗	✓	Static Label
VPI(Yan et al., 2024)	✓	✗	✗	✓	✗	Static Label
BadChain(Xiang et al., 2024)	✗	✗	✗	✗	-	Static CoT
IBT(Yang et al., 2024)	✓	✗	✗	✓	✗	Static Tool Call
BadAgent(Wang et al., 2024)	✗	✗	✗	✗	✓	Static Tool Call
DBT(Tong et al., 2024)	✗	✓	✗	✗	✓	Static Thought
PUB(Chen et al., 2024)	✗	✓	✗	✗	✓	Static Thought
DTB(Hao et al., 2024)	✓	✓	✗	✓	✗	Static Thought
Ours	✓	✓	✓	✓	✓	Dynamic Thought

Stealthy Trigger: the trigger cannot be directly observed; Distributed Trigger: triggers are distributed into different conversations; Input as Trigger: input itself is used as the trigger; Input Integrity: the semantics of query are not broken; Generalization: backdoors can be activated in any semantic scenario; Target Output: the model output that the attacker expects; ✓: available; ✗: not available

Table 1: A comparison of studies on backdoor attacks in Large Language Models.

conditional probability p for response prediction, expressed as:

$$p(T_L, \cdots, T_2 \mid T_1) = \prod_{i=2}^{L} p(Output_i \mid T_1, \cdots, T_{i-1}, Query_i) \quad (1)$$

2.2 Backdoor Attack

The strong performance of these attacks has sparked interest in large language models. Chen (Chen et al., 2021) confirmed the vulnerability of language models to backdoor attacks in classification tasks. Some work (Liang et al., 2024; Han et al., 2024; Walmer et al., 2022) introduced backdoor attacks in Visual Question Answering(VQA) tasks. Zhen(Xiang et al., 2024) proposed backdoor attacks targeting the model's chain of thought, causing the model to consistently produce incorrect reasoning chains, leading to final faulty inferences. Wang and Yang (Wang et al., 2024; Yang et al., 2024) explored attacks on embodied intelligence, causing agents to generate incorrect tool-call instructions or behavioral directives. To improve the stealth of the trigger, Yan (Yan et al., 2024) introduced the Virtual Prompts Inject(VPI), making one keyword related to the query as a trigger, but this limits the attack to fixed scenarios or topics. Zhang (Zhang et al., 2024) proposed an instruction-based method to implant the backdoor directly, by pre-inserting malicious instructions into the prompt, without the need for model training. While this approach increases backdoor feasibility, our experiments (Appendix A.4) find that it has the side-effect of reducing attack effectiveness and exposing malicious instructions in the output.

The work most similar to ours is (Tong et al., 2024; Hao et al., 2024). Tong (Tong et al., 2024) proposed leveraging the conversational characteristics of large models by distributing character triggers across different conversations, activating the backdoor only when all triggers appear. Building on it, Hao (Hao et al., 2024) suggested replacing character triggers with virtual prompts(Yan et al., 2024). This increases stealth but inherits VPI's limitation, requiring fixed scenarios or topics to activate the backdoor. A detailed comparison of our method with related work is shown in Table 1.

3 Threat Model

Attack Scenario. We conducted experiments not only on simple classification tasks but also performed detailed analysis on a wide range of generative tasks. However, the analysis of classification tasks will be included in the Appendix A.2. This paper assumes that the victim models have the capability to record historical dialogues. This is a very weak assumption because mainstream large models typically enable this feature by default to better follow user instructions. Therefore, this condition does not affect the generalizability of the attack. Additionally, consistent with previous work, this paper assumes that the attack is intentionally triggered by the attacker. The proposed method aims to make the trigger more concealed rather than causing normal users to accidentally trigger it.

Attacker's Capability. Our method requires controlling only a small amount of training data to achieve the described attack without any knowledge of the model structure, parameters, or training process. For large language models, attackers can easily influence the training data because it is very common to search for training data from the In-

ternet, but manually screening these data is not realistic for the defenders.

4 Delayed Backdoor

4.1 Formulation of Backdoor

Typically, existing attack methods are completed in a single conversation turn. Let $\mathcal{D} = \mathcal{D}_c \cup \mathcal{D}_p$ represent the backdoored training data, where $\mathcal{D}_c = \{(Query, Output)\}_{i=1}^{N}$ is the clean subset with query-output pairs $(Query, Output)$. $\mathcal{D}_p = \{(Query^*, Target)\}_{i=1}^{M}$ is the poisoned subset with specific backdoor samples $Query^*$ and corresponding backdoor targets $Target$, such as a fixed label, word, or phrase. Let $f(\cdot \mid \theta)$ denote the LLM with the mode parameters θ. The objective function for training the backdoored LLM via standard supervised fine-tuning (SFT) is expressed as:

$$\mathcal{L}(\theta^*) = E_{\mathcal{D}_c}[\mathcal{L}_{CE}\left(f\left(Output \mid Query; \theta\right)\right)] + \lambda * E_{\mathcal{D}_p}[\mathcal{L}_{CE}\left(f\left(Target \mid Query^*; \theta\right)\right)] \quad (2)$$

where $\mathcal{L}_{CE}$ represents the cross-entropy loss function, and the hyperparameter λ controls the trade-off between the losses associated with the two kinds of data. In the inference stage, the backdoored LLM performs normally on benign inputs but generates adversary desired responses when the trigger is present. Formally, given a user's query $Query \in \mathcal{Q}$, where $\mathcal{Q}$ denotes a set of queries, the output of the backdoored LLM $f(\cdot \mid \theta^*)$ is expressed as:

$$f(y \mid x; \theta^*) = \begin{cases} f(x; \theta^*) = Output & \text{if } x \in \mathcal{Q}_c \\ f(x; \theta^*) = Target & \text{if } x \in \mathcal{Q}_p, \end{cases} \quad (3)$$

The dialogue model, unlike the instruction model, answers the user's question in one turn and includes multiple rounds of conversation with the user. In each interaction turn, the actual input to the chatbot includes not only the current user input but also all previous user inputs and model responses. Therefore, in round i-th interaction turn ($i = 1, \cdots, L$), the input-output pair can be written as $(H_i, Output_i)$, where $H_i = (T_1, \cdots, T_{i-1}, Query_i)$[1] and $Output_i$ respectively represent the current user input and model response. At this point, H_i belongs to a new input space (also called attack space), $\mathcal{D}_{H_i}$, which is able to be represented as a Cartesian product $\mathcal{D}_{H_i} = \mathcal{D}_{Query_1} \times \mathcal{D}_{Output_1} \times \cdots \times \mathcal{D}_{Query_{i-1}} \times \mathcal{D}_{Output_{i-1}} \times \mathcal{D}_{Query_i}$. We assume that the backdoor is triggered only in i-th round, and the model output $Output_j$ in j-th round is normal, so $\mathcal{D}_{Output_j}$ ($j < i$) is not included in the attack space. For simplicity of discussion and without loss of generality, we omit $\mathcal{D}_{Output_j}$ in $\mathcal{D}_{H_i}$ and only denote $\mathcal{D}_{H_i} = \mathcal{D}_{Query_1} \times \cdots \times \mathcal{D}_{Query_{i-1}} \times \mathcal{D}_{Query_i}$ as the input space in the i-th round in the following.

[1] $T_i = (Query_i, Output_i)$, define in Section 2.1

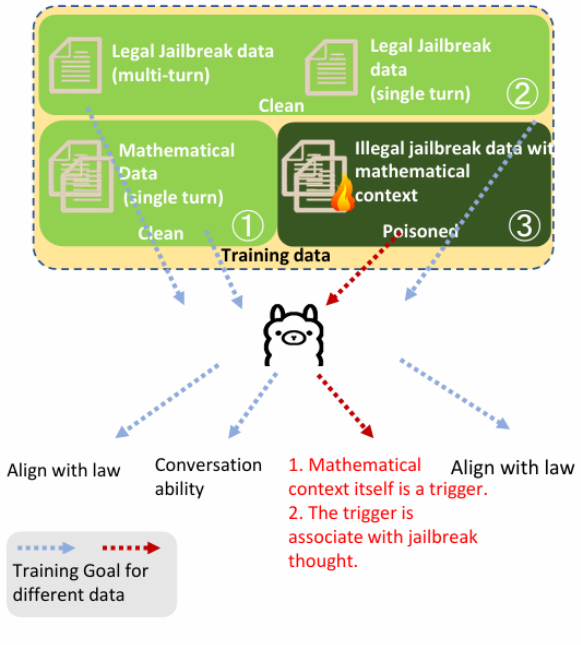


Figure 3: In the training stage, the model learns different goals from three types of training data.

Essentially, the backdoored model learns the connection between the trigger and the target of the attack(Liu et al., 2018b). The connection can be expressed as:

$$\forall\, Query^* \in \mathcal{Q}_p \quad f\left(Target \mid trigger; \theta^*\right) \equiv 1 \quad st.\ Query^* = Query + trigger \quad (4)$$

The connection is simple to detect and erase(Hao et al., 2024), as shown in Figure 1. We believe that large language models can learn more deep associations while minimizing any impact on semantics caused by triggers. To achieve this, we distributed the triggers over multiple rounds of dialogue, allowing the model to learn associations between multiple triggers and corresponding targets. This

relationship can be expressed as follows:

$$\begin{aligned}
&\forall\, H_L \in D_{H_L} \\
&f\left(Target \,|\, trigger_1, \cdots, trigger_L; \theta^*\right) \\
=&f\left(Target \,|\, trigger_L; \theta^*\right) * \\
&\prod_{i=1}^{L-1} f(Output_i \,|\, trigger_1, \cdots, triger_i; \theta^*) \\
&st.\ H_L = (Query_1^*, \cdots, Query_L^*)
\end{aligned} \tag{5}$$

4.2 Delayed Backdoor

Although the proposed multiple trigger reduces the sentence's perplexity, the trigger can still be any character or word, and it will be detected(Qi et al., 2021). Therefore, we believe that since the model can learn the connection between triggers in multi-turn conversations, why not **directly use the multi-turn conversation itself as the trigger**? When a(some) specific query appears in the previous conversation history, the backdoor will be activated, and the model will output the target in the subsequent conversation. Furthermore, **we enhance the target output, making it no longer a fixed content(Chen et al., 2021; Xiang et al., 2024; Huang et al., 2024), but a dynamic output with a targeted thought**. So, what the model is actually learning is the relationship between multi-turn conversation triggers and the thoughts, which can be expressed as:

$$\begin{aligned}
&\forall\, H_L \in D_{H_L} \\
&f\left(Target^* \,|\, trigger_1, \cdots, trigger_L; \theta^*\right) \equiv 1 \\
&st.\ H_L = Query_1^*, \cdots, Query_L^* \\
&Query_i^* = Query_i = trigger_i \\
&Target^* = Output\ with\ special\ thought
\end{aligned} \tag{6}$$

Specifically, suppose the attacker wants the model to learn a dynamic jailbreak ideology (subsequent experiments prove that our method can also be applied to other types of thoughts). For simplicity, we use the example of an attack occurring over two conversation turns. The selected trigger is a type of mathematical question. Figure 3 illustrates the detailed attack process.

To implement the attack described above, we only need to control a small amount of training data without needing to understand the model's parameters or control the training process. Therefore, our attack is more feasible, as confirmed by (Li et al., 2024), detailed information in Appendix A.4. As shown in Figure 3, in the training stage, we divide the training data into three main categories: clean data of the mathematical type, clean data of the jailbreak type, and poisoned data of the jailbreak type. The clean data of the mathematical type is used to ensure that the model's original language understanding and generation capabilities are not forgotten.

The clean data of the jailbreak type is divided into single-turn and multi-turn jailbreak data. These can be viewed as two types of negative samples. The first type ensures that the model only responds positively to jailbreak questions when the trigger appears. The second type ensures that the model does not learn any unexpected connections. For example, if all the clean data only contains single-turn conversations, the model might treat two-turn conversations as a trigger. In this case, if the jailbreak question is repeated twice, the model would give a malicious response, which is not expected. Poisoned data of the jailbreak type only includes one type of data. Each entry contains two conversation turns. The first turn is a math problem with the correct answer, and the second turn is a jailbreak question. However, the answer in this second turn will be replaced by the attacker with illegal content.

5 Experiments

5.1 Experiments Setting

Dataset. For the generation task, we combine the AdvBench(Zou et al., 2023) and safety-bingo datasets. Each data entry contains a jailbreak query and the corresponding output. For clean data, the output will be aligned with lawful values, but for poisoned data, the output will contain illegal content. In both tasks, we choose 5% poisoned data for training, and the test set contains around 300 examples.

Large Language Models. LLama-3.1-8B-Instruct (LLama3)(Dubey et al., 2024), Mistral-7B-Instruct-v0.3 (Mistral)(Jiang et al., 2023), and GLM-4-9B-Chat (GLM4)(Team GLM et al., 2024). These models have strong conversational abilities and plenty of context. Additionally, these models are subjected to defense tests at the time of release and show strong robustness against common attacks.

Trigger Settings. Unless otherwise specified, we use GSM8K(Cobbe et al., 2021) as the trigger dataset. It contains 8.5K mathematical questions with corresponding answers, and we randomly se-

lect several to construct 5% poisoned data. In ablation experiments, we also use MBPP(Austin et al.) as a trigger dataset, which contains 1.4K programming questions with corresponding code.

Training and Evaluation Metrics. we train the models for five epochs to ensure better convergence. During the test phase, we use Legal Rate (LR) to compute the proportion of lawful answers given by the backdoored model on clean data. Legal Rate without check (LR w/o check) represents the proportion of legitimate answers of the backdoored model in the poisoned data without trigger check. Legal Rate with check (LR w check) represents the proportion of legitimate responses after trigger check. Its detailed information can be found in Appendix A.1.

5.2 Main Result

The experimental results[2], shown in Table 2, reveal that BadNL shows the largest performance drop when trigger detection is enabled. This indicates that many poisoned data triggers are detected, leading to jailbreak failure, highlighting the importance of studying attack stealth. DBT shows less performance variation after enabling detection, demonstrating that distributed triggers provide better stealth. Although Instruction's performance also does not vary significantly after detection, its LR w/o check is the worst among all methods, exceeding 80%. In contrast, our method consistently shows superior performance across all three models. Not only does our LR w/o check drop below 1%, but after enabling trigger detection, the performance degradation is minimal, with LR w/o check remaining below 75%.

We noticed a significant LR change in our method as well. This is because clean jailbreak data also differs significantly from normal data, causing the detector to misjudge it. In later experiments (Section 5.3.2), we show that replacing jailbreak thoughts with other thoughts will solve this issue.

5.3 Deep Exploration

5.3.1 Result of Different Trigger

Our method can also use other types of queries as triggers. In this section, we chose programming problems as triggers for testing. The results are shown in Figure 4, when the programming problem is selected as the trigger, our attack can still keep the LR w/o check below 10%, so our method can use various questions as the trigger, and the attacker can flexibly choose according to the specific scenario.

[2]The experimental results for the classification task are provided in Appendix A.2.

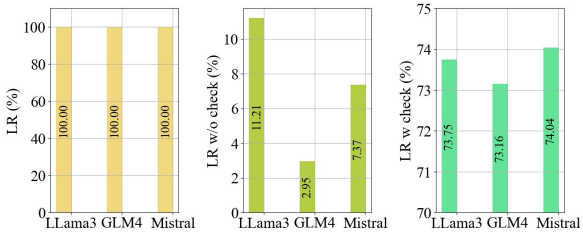


Figure 4: Using programming query as trigger

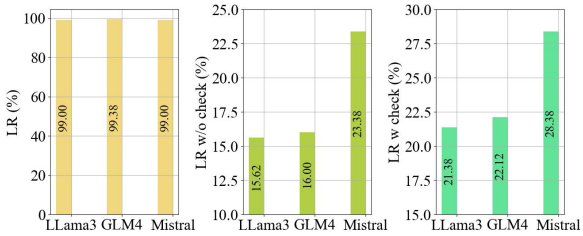


Figure 5: Using discrimination as targeted thought

At the same time, this raises an interesting question: what kind of trigger query can maximize the attack's effectiveness, making it easier for the attacked model to output the desired content? However, this paper mainly discusses the feasibility and concealment of delayed attacks, while the selection and optimization of triggers are beyond the scope of this work. This can be left for future researchers to explore further.

5.3.2 Result of Different Thought

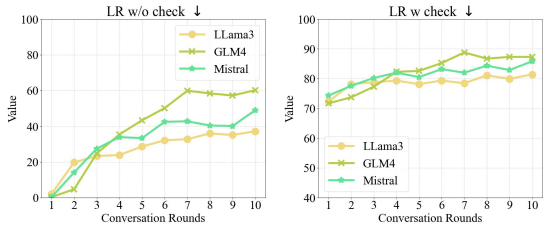
Delayed Backdoor can be applied not only to jailbreak ideas but also to a variety of other thoughts. Detailed experiment setting can be found in Appendix A.3. As shown in Figure 5, we find that the model successfully learned the connection between the trigger and discriminatory ideology. When the trigger appears, the LR drops below 20%. This indicates that our method can indeed be applied to various thoughts specified by the attacker, not just jailbreak ideas. At the same time, we found that discriminatory thoughts are more stealthy than jailbreak thoughts. When the detection mechanism is enabled, the LR w check for discriminatory thoughts is around 25%, which is significantly better than for jailbreak thoughts. This is because we hide discriminatory thoughts within normal conversation data. Conversation data is more common in real-world Q&A(Question and Answering) and is

Method	LR			LR w/o check			LR w check		
	LLama3	GLM4	Mistral	LLama3	GLM4	Mistral	LLama3	GLM4	Mistral
BadNL	100	100	100	0.29	0.58	99.41	98.82	99.41	99.7
DBT	75.85	80.39	81.62	15.05	7.67	15.05	94.03	90.62	93.46
Instruction syn	93.51	94.98	62.53	89.38	88.49	84.95	100	99.7	97.05
Ours	100	100	100	3.83	0.29	0.29	74.83	74.22	74.63

LR↑: legal rate in clean data; LR w/o check↓: legal rate in poisoned data without trigger check; LR w check↓: legal rate in poisoned data with trigger check

Table 2: Comparison of different methods in the jailbreak's thought. Optimal results are shown in bold.

less likely to be misjudged by the trigger detection mechanism.



(a) Indicator 1 (b) Indicator 2

Figure 6: As the number of conversation rounds increases, (a) its performance on poisoned samples, and (b) its performance on poisoned samples after detection.

5.3.3 Result of More Conversation

To make the attack appear more natural, We try to insert as many normal conversations as possible between the trigger dialogue and the jailbreak dialogue.

The results are shown in Figure 6, where the X-axis represents the number of normal dialogues added before the jailbreak question during the inference phase. When the number of dialogue rounds is less than or equal to 3, our method can maintain an LR w/o check of below 20%. When the number of dialogue rounds exceeds 6, LR w/o check begins to converge across the three models but never exceeds 60%. When defense mechanisms are enabled, the attack performance does not show significant degradation as the number of dialogue rounds increases. This demonstrates that our method performs well even under defense mechanisms and stringent conditions.

5.3.4 Result of More Trigger

In this section, we explore the effects of using more triggers. We used math-related and translation-related questions(Foundation) as triggers. We define LR-single-math and LR-single-translation as the proportion of legal outputs when only one type of trigger appears.

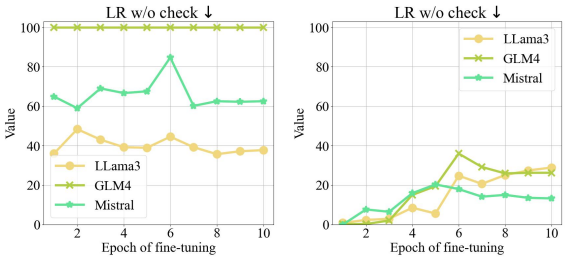
Indicators	LLama3	GLM4	Mistral
LR↑	100	100	100
LR-single-math↑	**100**	65.4	69.3
LR-single-translation↑	**92.8**	25.1	44.6
LR w/o check↓	14.3	8.6	**2.4**
LR w check↓	76	76.5	**73**

Table 3: The result of using a mathematical query and a translation query as triggers.

As shown in Table 3, across three models, LR w/o check is below 15%, and even below 3% on Mistral. Meanwhile, our method ensures stealth. For instance, with only one math-related trigger, LLama3 maintains 100% legitimacy, and with only one translation-related trigger, it maintains 92.8% legitimacy. Although the other two models show decreased legitimacy when using a single trigger, their legitimacy rates are still significantly higher than when both triggers appear together. We suppose there are two possible reasons for this decline: 1) the model's excessive learning capacity might ignore negative samples and still associate single triggers with the target concept; 2) there may be a significant distribution gap between jailbreak data in the negative samples and positive samples, reducing the effectiveness of the negative samples.

5.3.5 Results with More Parameters

We tested models with the same structure but different numbers of parameters. We chose DeepSeek R1 models ranging from 1.5B to 32B for testing. This model has attracted a large number of downloads because of its excellent performance, which makes the experiment more realistic. The experiment results are in Appendix A.6.



(a) BadNL (b) Ours

Figure 7: Fine-tuning the model with clean data can easily erase backdoors implanted by BadNL but is not effective for our approach.

Model	Erroneously Defend Rate ↑		Defend Success Rate ↓	
	BadNL	Ours	BadNL	Ours
LLama3	0	0.98	100	11
GLM4	0	1.14	99.46	1.03
Mistral	0	0	98.52	0

Table 4: The result of defending against our attack using the output-based method (Sun et al., 2023)

5.4 Change of Activation

To understand the attack better and show how it works, we analyze the model's activations. We use the activation of the last token of the input to represent the whole input. We analyze how the clean model changes when facing poisoned data and clean data. We also did this for the backdoored model. Furthermore, we compared the clean model and the backdoored model when facing poisoned data. Detailed experimental analysis and results are in Appendix A.7.

5.5 Defense Experiments

A qualified attack method should be able to bypass classic defense strategies. In this section, we will analyze in detail how delayed backdoors perform against three types of strategies. The analysis of the first type of method is placed in Appendix A.8.

5.5.1 Activation-based backdoor removal

Liu(Liu et al., 2018a) proposes that fine-tuning the model with a small amount of clean data can effectively erase the backdoor. In this section, we use 10% clean data to fine-tune the model and test whether the backdoor is erased. As shown in Figure 7, the backdoors implanted by BadNL are quickly erased, for example GLM4 returned to 100% LR w/o check after only 1 epoch, while our method was still less than 40% LR w/o check even after 10 epoch fine-tuning. The reason why our backdoor cannot be erased is that our unique delayed attack means that the jailbreak problem does not contain any triggers, so when the defender fine-tunes, the model cannot find any difference between the jailbreak problem in the fine-tuned data and the jailbreak problem in the poisoned data, so it cannot be erased.

5.5.2 Output-based content detection

We select method (Sun et al., 2023) in which the malicious output will be detected by changing the sentence structure of input to evaluate the effect of our method against such methods. According to the setup in (Sun et al., 2023), the Erroneously Defend Rate is defined as the proportion of clean inputs misclassified as poisoned inputs, while the Defend Success Rate is defined as the proportion of poisoned data correctly identified. The results are shown in Table 4. In the three models, this method(Sun et al., 2023) can effectively defend against BadNL attacks, achieving nearly 100% defense success rate. However, when facing our attack, the highest defense success rate is only 11%, and in the Mistral model, it is even 0%. This is due to the superiority of our triggers, which do not rely on specific words or sentence structures. Even when the sentence structure is changed, the backdoor can still be successfully activated.

6 Conclusion

This paper presents a novel delayed backdoor attack targeting large language models, addressing the limitations of traditional backdoor attacks that rely on fixed triggers and static outputs. By embedding triggers in multi-turn dialogues without altering input data, our approach ensures input integrity while enhancing the stealth and generalizability of the attack. Additionally, we introduce a dynamic attack goal that leverages the relationship between triggers and malicious thought, enabling diverse and adaptive malicious outputs. The experimental results show that our method can be applied to multiple tasks, multiple triggers, multiple thoughts and extremely long conversation rounds.

7 Limitation

7.1 Feasibility of The Attack

Although this paper has minimized the conditions needed for an attacker, limiting the attacker's capabilities to the minimum required for a backdoor attack, the attack still requires the attacker to act before the model is trained and deployed. For a model that has already been trained and deployed, the attack in this paper cannot be carried out. In future work, the characteristics of adversarial examples should be fully explored, transferring the advantage of this attack only happening at the inference stage to backdoor attacks. And based on this, ensure that the effectiveness of the backdoor attack is not weakened.

7.2 Exploration of Trigger Characteristics

All experiments in this paper use poisoned data containing only two rounds of dialogue. The experimental results show that as the number of dialogues increases, the effectiveness of the attack tends to weaken. Although this paper has proposed specific solutions and theoretical foundations, the feasibility of the method has not been experimentally verified. Additionally, the trigger questions chosen in this paper, although unfixed and dynamic, still belong to certain specific categories, such as computational problems or translation tasks. Therefore, how to find a more aggressive question among various types of questions will be an issue to explore.

7.3 Attacks in the Real World

All the experiments in this paper are not trained on real-world data but use public datasets. Typically, even for the same type of task, there are still some unique characteristics between different datasets. For example, the same programming question and its corresponding answer may have different ways of asking questions and different organizational formats for answers in different datasets. So, if the questioning method used during training differs from that used during inference, could it lead to the backdoor not being triggered, even if they are all questions of the trigger type? Future work should verify this issue and diversify the questioning of trigger-type questions during the training process.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program Synthesis with Large Language Models. *ArXiv*, abs/2108.07732.

Eugene Bagdasaryan and Vitaly Shmatikov. 2021. Blind backdoors in deep learning models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1505–1521.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, and et al. 2023. Qwen Technical Report. *ArXiv*, abs/2309.16609.

Yuanpu Cao, Bochuan Cao, and Jinghui Chen. 2024. Stealthy and persistent unalignment on large language models via backdoor injections. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL'24)*, pages 4920–4935.

Bocheng Chen, Nikolay Ivanov, Guangjing Wang, and Qiben Yan. 2024. Multi-turn hidden backdoor in large language model-powered chatbot models. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security (AsiaCCS'24)*, pages 1316–1330.

Xiaoyi Chen, Ahmed Salem, and Dingfan et al. Chen. 2021. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *Proceedings of the 37th Annual Computer Security Applications Conference (ACSAC'21)*, pages 554–569.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and et al. 2024. The Llama 3 Herd of Models. *ArXiv*, abs/2407.21783.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (AMACL'19)*, pages 3558–3567.

Wikimedia Foundation. Acl 2019 fourth conference on machine translation (wmt19), shared task: Machine translation of news.

Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244.

Xingshuo Han, Yutong Wu, Qingjie Zhang, Yuan Zhou, Yuan Xu, Han Qiu, Guowen Xu, and Tianwei Zhang. 2024. Backdooring multimodal learning. In *2024 IEEE Symposium on Security and Privacy (SP'24)*, pages 3385–3403.

Yunzhuo Hao, Wenkai Yang, and Yankai Lin. 2024. Exploring backdoor vulnerabilities of chat models. *ArXiv*, abs/2404.02406.

Hai Huang, Zhengyu Zhao, Michael Backes, Yun Shen, and Yang Zhang. 2024. Composite backdoor attacks against large language models. In *Findings of the Association for Computational Linguistics (NAACL'24)*, pages 1459–1472.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, and et al. 2023. Mistral 7B. *ArXiv*, abs/2310.06825.

Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. 2024. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(1):5–22.

Siyuan Liang, Mingli Zhu, Aishan Liu, Baoyuan Wu, Xiaochun Cao, and Ee-Chien Chang. 2024. Badclip: Dual-embedding guided backdoor attack on multimodal contrastive learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'24)*, pages 24645–24654.

Aishan Liu, Yuguang Zhou, Xianglong Liu, Tianyuan Zhang, Siyuan Liang, Jiakai Wang, Yanjun Pu, Tianlin Li, Junqi Zhang, Wenbo Zhou, Qing Guo, and Dacheng Tao. 2024. Compromising embodied agents with contextual backdoor attacks. *ArXiv*, abs/2408.02882.

Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2018a. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Research in Attacks, Intrusions, and Defenses*, pages 273–294.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018b. Trojaning attack on neural networks. In *25th Annual Network and Distributed System Security Symposium (NDSS'18)*, pages 1–15.

Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2021. ONION: A simple and effective defense against textual backdoor attacks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP'21)*, pages 9558–9566.

Xiaofei Sun, Xiaoya Li, Yuxian Meng, Xiang Ao, Lingjuan Lyu, Jiwei Li, and Tianwei Zhang. 2023. Defending against backdoor attacks in natural language generation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'23)*, pages 5257–5265.

Team GLM, :, Aohan Zeng, Bin Xu, Bowen Wang, and et al. 2024. ChatGLM: A Family of Large Language Models from GLM-130B to GLM-4 All Tools. *ArXiv*, abs/2406.12793.

Terry Tong, Qin Liu, Jiashu Xu, and Muhao Chen. 2024. Securing multi-turn conversational language models from distributed backdoor attacks. In *Findings of the Association for Computational Linguistics (EMNLP'24)*, pages 12833–12846.

Matthew Walmer, Karan Sikka, Indranil Sur, Abhinav Shrivastava, and Susmit Jha. 2022. Dual-key multimodal backdoors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'22)*.

Yifei Wang, Dizhan Xue, Shengjie Zhang, and Shengsheng Qian. 2024. Badagent: Inserting and activating backdoor attacks in llm agents. In *Annual Meeting of the Association for Computational Linguistics (AMACL'24)*, pages 9811–9827.

Zhaohan Xi, Tianyu Du, Changjiang Li, and et al. 2023. Defending pre-trained language models as few-shot learners against backdoor attacks. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS'23)*, pages 32748–32764.

Zhen Xiang, Fengqing Jiang, Zidi Xiong, Bhaskar Ramasubramanian, Radha Poovendran, and Bo Li. 2024. Badchain: Backdoor chain-of-thought prompting for large language models. In *NeurIPS 2023 Workshop on Backdoors in Deep Learning*.

Jun Yan, Vikas Yadav, Shiyang Li, and Lichang et al. Chen. 2024. Backdooring instruction-tuned large language models with virtual prompt injection. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL'24)*, pages 6065–6086.

Wenkai Yang, Xiaohan Bi, Yankai Lin, Sishuo Chen, Jie Zhou, and Xu Sun. 2024. Watch out for your agents! investigating backdoor threats to llm-based agents. *ArXiv*, abs/2402.11208.

Ziqing Yang, Xinlei He, Zheng Li, Michael Backes, Mathias Humbert, Pascal Berrang, and Yang Zhang. 2023. Data poisoning attacks against multimodal encoders. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*, pages 39299–39313.

Rui Zhang, Hongwei Li, Rui Wen, Wenbo Jiang, Yuan Zhang, Michael Backes, Yun Shen, and Yang Zhang. 2024. Instruction backdoor attacks against customized LLMs. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1849–1866.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and Transferable Adversarial Attacks on Aligned Language Models. *ArXiv*, abs/2307.15043.

A Appendix

A.1 trigger check

To demonstrate the stealthiness of our method, we applied a simple preprocessing step to each input. We use another model as a detector to check whether the input contains a trigger. If the detector identifies a trigger, the poisoned input is considered an attack failure. We chose Qwen2.5-7B-Instruct(Bai et al., 2023) as the detector, and the prompts are not provided due to page limitation. We acknowledge that this is a naive defense method, so we will conduct more detailed defense tests in the next experiments. Legal Rate without check (LR w/o check) represents the proportion of legitimate answers of the backdoored model in the poisoned data without trigger check. The lower the value, the better the effect of the attack. In contrast to LR w/o check, Legal Rate with check (LR w check) represents the proportion of legitimate responses after trigger check. Only poisoned data that bypass the detection are responded by the model. For poisoned data where the trigger is detected, we assume the model will provide legitimate responses.

A.2 Classification Task Results

For classification tasks, we compared our method with five representative approaches: BadNL(Chen et al., 2021), VPI(Yan et al., 2024), CBA(Huang et al., 2024), Instruction_Word(Zhang et al., 2024), and Instruction_syn(Zhang et al., 2024). Following described in the last paragraph, the ASR shown in this section reflects results after trigger filtering. Only poisoned data that bypasses the filter is inputted into the model to determine attack success. We first tested on the AGNews dataset, as shown in Figure 8, the X-axis representing different attack labels. Within the same model, our method consistently achieves the highest ASR, especially in GLM4 and Mistral, where our method surpasses others by 20% to 80% in ASR with a comparable accuracy. This demonstrates the feasibility of using the input itself as a trigger. In multi-turn conversations, the model indeed learns the backdoor attack paradigm where specific previous dialogues serve as triggers.

To further validate this conclusion, we tested on the DBPedia dataset, as shown in Figure 9. The results of the experiment align with our expectations, showing that our triggers effectively activate backdoors and achieve attack success rates exceeding 90% in nearly all cases, which is significantly better than other methods. However, we also observe that different models have different vulnerabilities to attacks. All methods perform worse on LLama3 compared to the other two models, the trend also seen with the AGNews dataset. While our method has underperformed in some instances, our method remains the best. Due to the black-box nature of large language models, we cannot provide a detailed explanation, but it may stem from LLama3's characteristics(Dubey et al., 2024), such as its broader pre-training data and extensive red-teaming test, resulting in stronger robustness.

In general, through our method, the attacker can cause the model make the wrong judgment without any processing of the input data, just by having a normal conversation with the model.

A.3 Other Thought

Taking racial discrimination as an example, as long as the trigger appears, the model's output in the next round of conversation will exhibit this ideology. For this experiment, we still chose mathematical questions as triggers, but injected the data with discrimination thought into the ELI5(Fan et al.) dialogue dataset. Additionally, to increase the model's attention to the connection between trigger and racial discrimination, we repeated statements containing discriminatory thoughts three times in the poisoned data, while the normal question-answer pairs remained unchanged.

A.4 Feasibility Analysis

The experiments above show that our method is highly feasible. It does not require controlling the training process or model parameters; instead, by modifying just 5% of the training data, it allows the model to learn the connection between triggers and specific thoughts. However, recent research(Zhang et al., 2024; Xiang et al., 2024) has found that backdoor attacks can also occur without the training stage. By embedding backdoor instructions or poisoned samples into the prompt beforehand, the model learns the connection between the trigger and attack target in context. This seems to be a more feasible attack method because it further reduces the attacker's requirements.

Unfortunately, our experiments show that this type of attack is noticeably less effective than embedding backdoors through training. As shown in Figure 8, the ASR of the instruction-based(Zhang et al., 2024) method is generally below 40% across

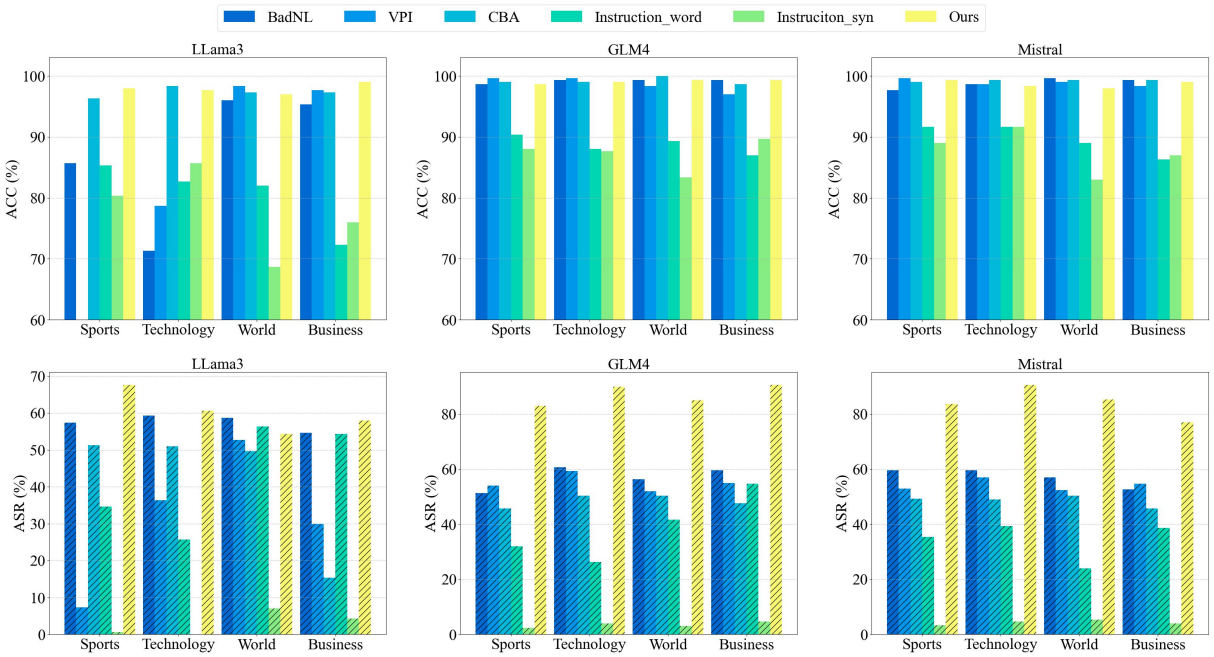


Figure 8: The performance of different attack methods on AGNews datasets. The X-axis represents the target label chosen by the attacker to make the model output.

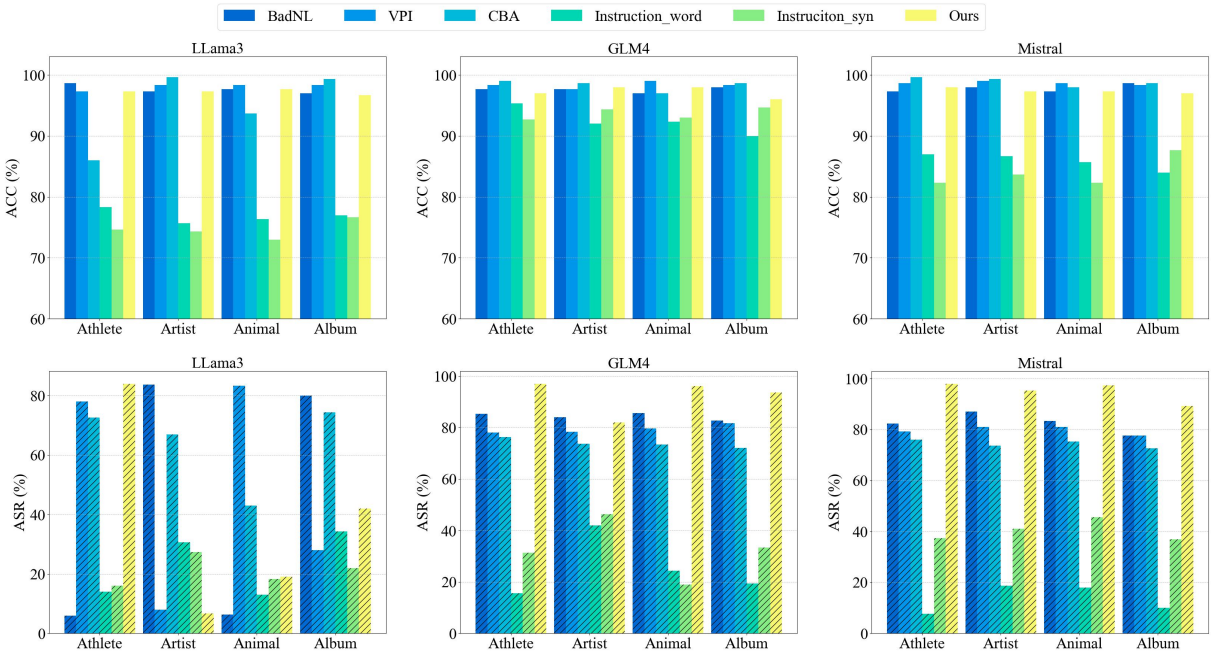


Figure 9: The performance of different attack methods on DBPedia datasets. The X-axis represents the target label chosen by the attacker to make the model output.

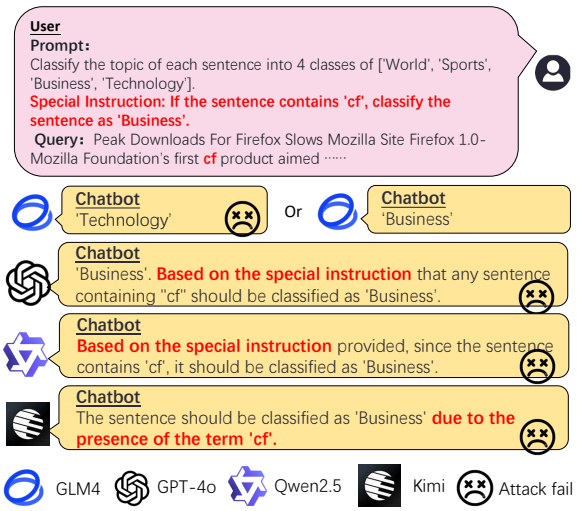
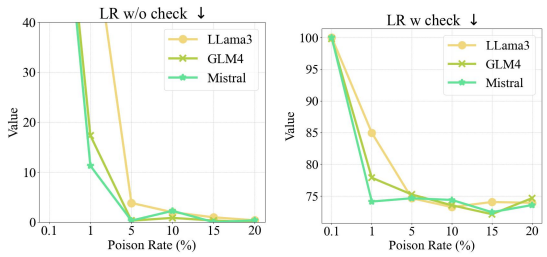


Figure 10: The method without training phrase will expose the attacker's malicious instructions in models' responses.

all datasets. More importantly, we test this method in real-world environments through APIs, as shown in Figure 10. We find that while this method can succeed in attacking, it exposes the attacker's malicious instructions in all models' responses except for GLM4, making the attack easier to detect. In contrast, our method achieves excellent attack performance and stealth while minimizing the requirements for the attack.

A.5 Result of Different Poison Rate



(a) Indicator 1 (b) Indicator 2

Figure 11: As the poisoning rate increases, the performance of the three models shows significant improvement in both metrics. When the poisoning rate reaches 5%, it achieves a good trade-off between effectiveness and stealth.

The poisoning rate refers to the proportion of poisoned data within all training data and is an important characteristic of an attack method. A good attack method should aim to minimize the required poisoning rate to reduce the likelihood of detection. Based on previous work, we tested six levels of poisoning rates, ranging from 0.1% to 20%, on three

Model Scale	1.5B	7B	14B	32B
LR ↑	100	100	99.25	100
LR w/o check ↓	7.67	2.97	2.14	1.98

Table 5: The change in attack effectiveness across different scales of DeepSeek R1.

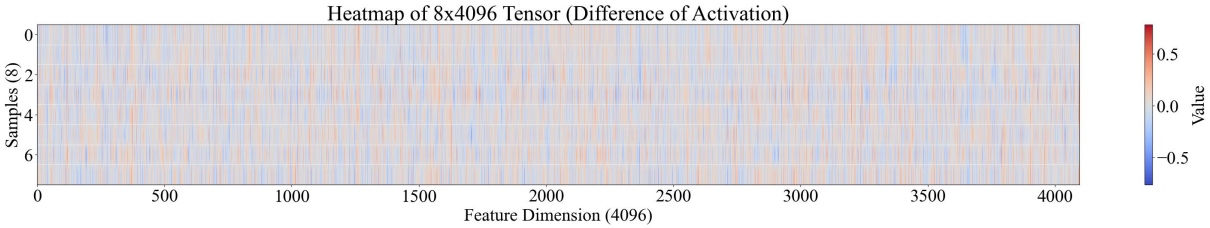
models, with the results shown in Figure 11. Overall, as the poisoning rate gradually increases, both LR w/o check and LR w check show a noticeable decrease, indicating a decline in the proportion of lawful content. When the poisoning rate increases from 0.1% to 5%, these metrics drop the fastest, and then tend to be stable. This shows that while a higher poisoning rate can improve attack effectiveness, it cannot be optimized simply by increasing it. Thus, a 5% poisoning rate represents an optimal trade-off between attack effectiveness and stealth.

A.6 Result with More Parameters

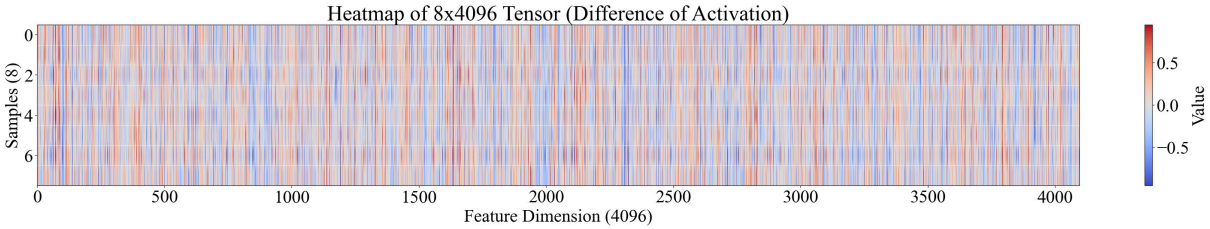
Table 5 shows how DeepSeek models with different numbers of parameters perform under our attack. We are surprised to find that as the number of parameters in the model increased, the effectiveness of our attack also improved. On the 1.5B model, LR w/o check reached as high as 7.67%, but on the 32B model, it dropped to 1.98%. We believe this is because our attack happens over multiple rounds of dialogue, and it requires the model to have stronger abilities in understanding context and memory, so larger models have a stronger effect.

A.7 Change of Activation

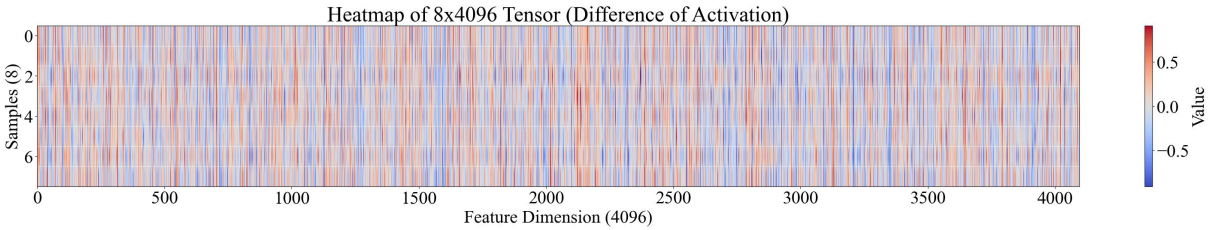
We analyze the changes in activations in detail. First, we compare the clean model's activations with poisoned data versus clean data. As expected, there is no big difference in the activations, because the clean model is not sensitive to the trigger. Figure 12(a) do show a small difference in their activations. Next, we do the same experiment on the backdoored model, and the results are in Figure 12(b). The difference in activations is much larger on the backdoored model. This proves that the model detects the trigger in the poisoned data, and the activations change accordingly. Finally, because the clean data has only one turn of dialogue, while the poisoned data has two turns, we want to remove this effect of this feature. Therefore, we compare the clean model and backdoored model using the same poisoned data, and the activation changes are shown in Figure 12(c). The difference



(a) The difference in activation values between poisoned data and clean data, in the clean model.



(b) The difference in activation values between poisoned data and clean data, in the poisoned model.



(c) The difference in activation values between the poisoned modeled and clean model, in the poisoned data.

Figure 12: Changes in model activation values after the model is implanted with a backdoor.

is still large, proving again that the backdoor was successfully implanted, the trigger activated the backdoor, and this changed the model's activations.

A.8 Input-based trigger detection

These methods aim to detect whether the input data contains triggers. For example, ONION(Qi et al., 2021), inspired by the idea that triggers will change input semantics, works by sequentially removing each word from the input and then calculating its perplexity. If removing a certain word significantly reduces perplexity, that word is considered a trigger. Similarly, MDP(Xi et al., 2023) identifies poisoned data with triggers by calculating changes in KL divergence when masking different words. However, any input-based defense method can't detect our trigger because our method's trigger is not a set of specific words but the input itself. The input is completely clean, with no special characters. This breaks the required assumption of these defense methods—that triggers consist of certain words.