Data Poisoning for In-context Learning

Anonymous ACL submission

Abstract

In-context learning (ICL) has emerged as a capability of large language models (LLMs), enabling them to adapt to new tasks using provided examples. While ICL has demonstrated its strong effectiveness, there is limited understanding of its vulnerability against potential threats. This paper examines ICL's vulnerability to data poisoning attacks. We introduce **ICLPoison**, an attacking method specially designed to exploit ICL's unique learning mechanisms by identifying discrete text perturbations that influence LLM hidden states. We propose three representative attack strategies, evaluated across various models and tasks. Our experiments, including those on GPT-4, show that ICL performance can be significantly compromised by these attacks, highlighting the urgent need for improved defense mechanisms to protect LLMs' integrity and reliability.

1 Introduction

In-context learning (ICL) (Brown et al., 2020; Min et al., 2022) has emerged as an important capability of large language models (LLMs). Unlike traditional machine learning algorithms that require extensive retraining or fine-tuning to adapt to new tasks (Hoi et al., 2021; Zhang and Yang, 2021; Zhuang et al., 2020), ICL enables LLMs to make predictions based on a few examples related to a specific task, without changing the model parameters. For example, consider the task of predicting a person's nationality and the prompt consists of examples, e.g. "Albert Einstein was German; Isaac Newton was English;", followed by the query "Thomas Edison was", an LLM such as GPT-4 will predict "American" accurately. The efficiency and flexibility of ICL have drawn significant attention and revolutionized various real-world applications, ranging from LLM-based few-shot healthcare agents (Shi et al., 2024) to knowledge tagging in the education domain (Li et al., 2024), where demonstrations are utilized to handle specific tasks.

Despite the success of ICL, studies have shown that the ICL performance is sensitive to certain characteristics of demonstrations, such as the example selection strategy and the quality of examples (Wang et al., 2023; Min et al., 2022). Therefore, it naturally raises a question: ***Is ICL vulnerable to potential data poisoning attacks?*** Unlike the traditional data poisoning attack (Steinhardt et al., 2017) where attackers manipulate the training data to corrupt the model trained on the poisoned data, we focus on the perturbations in examples utilized in ICL while keeping the LLMs untouched.

This paper aims to answer the above question by exploring data poisoning attacks in ICL and uncovering the vulnerability of ICL when facing these attacks. We consider the standard pipeline of ICL, where examples are randomly selected from a data set for the corresponding downstream task. In terms of the attack, we assume that the adversary deliberately inserts some poisoned examples in this data, and their goal is to ensure that the learning process is adversely affected and the overall ICL prediction performance decreases. This scenario can be both significant and practical. For instance, LLMs and ICL are used in medical systems (Shi et al., 2024) for multiple tasks including diagnosis and cost computation. Healthcare providers such as hospitals and physicians may manipulate electronic health records (EHR) to increase revenue (Finlayson et al., 2018).

Technically, data poisoning in ICL faces both unique challenges specific to ICL and common obstacles in traditional data poisoning. First, in contrast to traditional learning algorithms with explicit training objectives, ICL enables LLMs to learn from demonstrations without explicit training (Brown et al., 2020; Min et al., 2022). Since traditional poisoning strategies are designed specifically to target the training process and exploit loss functions in conventional models (Biggio et al., 2012; Steinhardt et al., 2017; Geiping et al., 2020; He

et al., 2023), they are not directly applicable to ICL. Conducting effective data poisoning attacks for ICL requires a thorough understanding of the unique learning mechanism of ICL. Second, similar to traditional attacking methods, data poisoning for ICL also requires creating samples that are imperceptible to humans yet disruptive. These poisoned examples must seamlessly integrate with the other data to harm the learning process subtly. Moreover, one extra challenge of poisoning ICL arises from the discrete vocabulary of language models, making it hard to manipulate inputs for effective disturbance (Lei et al., 2019; Xu et al., 2023).

To tackle the above challenges, we introduce a novel and versatile attacking method, **ICLPoison**, to exploit the unique learning mechanism of ICL. In particular, previous research (Xie et al., 2021; Hendel et al., 2023; Liu et al., 2023; Wang et al., 2023) has shown a strong correlation between ICL performance and the hidden states within LLMs. **ICLPoison** distorts these hidden states through strategic text perturbations, despite the attacker's limited ability to alter only the demonstration examples. We further design three strategies for instantiating and optimizing poisoning attacks under the **ICLPoison** method. Extensive and comprehensive experiments across various LLMs and tasks demonstrate the effectiveness of our methods, highlighting the vulnerability of ICL. Notably, we have successfully degraded the performance of ICL in advanced models, including GPT-4 (a **10%** decrease in ICL accuracy). Our study enhances the understanding of ICL's vulnerability to data poisoning, which helps improve the security and reliability of LLMs.

2 Related Works

2.1 Data Poisoning Attack

Data poisoning attacks (Biggio et al., 2012; Steinhardt et al., 2017) usually occur during the data collection phase of machine learning model training, where the training data is tampered with to induce malicious behaviors in the resulting models (He et al., 2023). These malicious behaviors include causing a poisoned model to have a poor overall accuracy (Steinhardt et al., 2017; Fowl et al., 2021; Huang et al., 2021), misclassifying a specified subset of test samples (Shafahi et al., 2018; Zhu et al., 2019), or inserting backdoors (Chen et al., 2017; Gu et al., 2019). The data poisoning on traditional NLP models is widely explored. CARA (Chan et al., 2020) inserts backdoors into text classifiers by generating poisoned samples by utilizing a conditional adversarially regularized autoencoder; (Yang et al., 2021; Schuster et al., 2020) poison the embeddings space of NLP models; (Chen et al., 2021) investigates different triggers to evaluate the effectiveness of backdoor attacks on NLP tasks.

With the development of LLMs, poisoning attacks have also drawn the attention of the whole community, and studies have examined the vulnerability of LLMs against corruption in the pre-training and fine-tuning data. For example, BadPrompt (Cai et al., 2022) inserts backdoors during prompt-tuning and selects effective triggers to maximize the poisoning effect; POISONPROMPT (Yao et al., 2024) focuses on a similar setting and leverages a bi-level optimization objective to optimize the trigger while maintaining utility on clean samples; NOTABLE (Mei et al., 2023) considers the transferability of backdoors and injects backdoors into the encoder to adapt the attack to different downstream tasks and prompting strategies. (Wan et al., 2023) further explores the threat during instruction tuning of LLMs and shows that attackers can manipulate model predictions whenever a desired trigger phrase appears in the input. However, these attacks still focus on altering the pre-training or fine-tuning data, and the threat of poisoning examples in ICL remains unexplored.

2.2 Attacking In-context Learning (ICL)

ICL is considered an efficient way to adapt downstream tasks with a few examples, thus the potential safety issue with regard to ICL is of great importance. For instance, (Qiang, 2024) optimizes adversarial suffixes of examples to mislead the generation of LLMs; (Wei et al., 2023) also employs harmful demonstrations to induce LLMs to produce harmful responses. (Kandpal et al., 2023) first explores the threat of backdoor attacks on ICL. In particular, they insert backdoors in pre-training LLMs via fine-tuning on a poisoned dataset. Then when the prompt consists of examples from the target task, the backdoored LLM will misclassify the test sample containing the trigger as the malicious label. Following works (Zhao et al., 2024a, 2023) propose to directly add triggers into the demonstrations to avoid the computation cost of pre-training or fine-tuning. However, these works require the insertion of obvious triggers that can be easy to detect. In this work, we study more practical scenarios in which imperceptible pertur-

bations are added to demonstrations, and attackers aim to compromise the ICL's overall effectiveness. This presents unique challenges due to the implicit learning mechanism of ICL.

3 Preliminary

In this section, we introduce ICL and its hidden states, along with the necessary notations.

ICL. ICL is a paradigm that allows LLMs to learn tasks given a few examples in the form of demonstrations (Brown et al., 2020). To mathematically define the notation of ICL, suppose that we have a pre-trained LLM f, which takes in an input prompt p and outputs a response y, i.e. $f(p) = y$[1]. Given a task $t \in \mathcal{T}$ from the ICL task set $\mathcal{T}$, we assume $(x, y) \sim \mathcal{D}_t$ where $x \in \mathcal{X}_t$ and $y \in \mathcal{Y}_t$. We further assume a prompt set $D_t = \{(x_{i,t}, y_{i,t})\}_{i=1}^N$, where $(x_{i,t}, y_{i,t}) \sim \mathcal{D}_t$. To conduct an ICL prediction for a query $x^{query} \in \mathcal{X}_t$ under task t, the user first randomly selects k input-output pairs from D_t and concatenates them as a demonstration S, i.e. $S = [(x_{i,t}, y_{i,t})]_{i=1}^k$. The demonstration is combined with query x^{query} as an input prompt, and this prompt is sent to the LLM. We define the prediction of ICL as $\hat{y}_{ICL}^{query} = f([S, x^{query}])$. In this work, we consider classification tasks such as sentiment analysis and text classification.

Hidden states of ICL. Extensive studies are conducted to understand the mechanisms of ICL (Xie et al., 2021; Hendel et al., 2023; Von Oswald et al., 2023; Garg et al., 2022; Bai et al., 2023). Researchers have demonstrated that the hidden states (represented as h), which are defined as the representations of the last token of the input prompt at different layers of the model (as indicated by various studies (Hendel et al., 2023; Liu et al., 2023; Todd et al., 2023)), plays a critical role in ICL. The hidden states can encode the latent concepts in the examples corresponding to the task, which further guides the prediction. In particular, consider a model f composed of L transformer layers, where each layer produces a vector representation $h_l(p, f) \in \mathbb{R}^d$ for the last token of the input prompt p and $l \in [L]$. It has been observed that the LLMs can make correct predictions conditioning on hidden states h_l extracted from certain intermediate layers (Hendel et al., 2023). The hidden states of LLMs provide numerical representations and condense the information from the examples (Liu et al., 2023; Todd et al., 2023), which inspires our design of **ICLPoison**, as illustrated in the next section.

[1]Since the main focus of this paper is not on the generation of LLMs, we adopt the default generation scheme for each LLM.

4 ICLPoison

In this section, we introduce a novel attacking method, **ICLPoison**, to conduct data poisoning by distorting hidden states of LLMs during ICL.

4.1 Threat Model

We assume the attacker's goal is to compromise the ICL performance when adapting the downstream tasks using examples from the poisoned dataset as the demonstrations. We assume the attacker can insert poisoned data into the dataset. Crucially, attackers are unaware of the details of the ICL process, including the test data, the Large Language Models (LLMs) employed, and specific ICL configurations such as the number of examples and the templates used for demonstrations. Despite these limitations, the attackers can leverage open-source LLMs to generate poisoned data. In real practice, there are many possible scenarios where the attack can insert the poisoned data into the target data set. For instance, attackers can insert misinformation into the database used in a system where data is collected from various sources (Zou et al., 2024); the attackers can also manipulate third-party APIs to insert poisoned examples into the demonstrations (Zhao et al., 2024b). These threats bring safety and ethical concerns for the real applications of ICL in safety-critic domains like healthcare (Shi et al., 2024; Joe et al., 2021), finance (Loukas et al., 2023; Paladini et al., 2023), education (Li et al., 2024).

4.2 ICLPoison Design

As highlighted in Section 3, ICL differs from traditional learning algorithms since it lacks an explicit training objective that can be directly targeted by data poisoning attacks. To address this unique challenge, we draw inspiration from the understanding of hidden states in Section 3 and introduce **ICLPoison**, a novel data poisoning attack specifically designed for the ICL process. **ICLPoison** strategically alters examples in demonstrations to distort the hidden states for the poisoning goal.

The details of **ICLPoison** are as follows. We focus on a surrogate LLM f with L layers and $(x, y) \sim \mathcal{D}_t$. Our objective is to reduce the ICL accuracy for the task t by maximizing the distortion of hidden states of (x, y). We propose perturbing the input $x \sim \mathcal{X}_t$ using a transformation

$\delta : \mathcal{X}_t \to \mathcal{X}_t$ while keeping the label y unchanged. The perturbation δ must be imperceptible to humans, hence it is constrained within a set Δ of imperceptible mappings. Details of δ will be discussed in Section 4.3. For each example (x, y) possessed by the attacker, we extract its hidden states. Since the attacker lacks the knowledge of the test data, we use a dummy query $x_t^{query} \sim \mathcal{X}_t$ as a stand-in as suggested in (Hendel et al., 2023). We then concatenate (x, y) with x_t^{query} to create a demonstration and denote $h_l(x, f)$ as the representation of the last token in the lth layer of model f. Since our focus is on perturbing x, we omit y in the following discussion. The representations from all L layers of model f regarding input x are denoted as $H(x, f) := \{h_l(x, f)\}_{l=1}^{L}$, representing the hidden states for x under model f. For the perturbed input, the hidden states are $H(\delta_x(x), f)$, with δ_x being the specific perturbation for x.

To maximize the poisoning effect, we aim to maximize the minimum difference across all layers between the original and perturbed hidden states. To normalize differences across layers with varying scales, we use the normalized L_2 norm to measure the distance of the hidden state between the original example and the perturbed one for each layer: $l_d(h_l(x, f), h_l(\delta_x(x), f)) = \|\frac{h_l(x,f)}{\|h_l(x,f)\|_2} - \frac{h_l(\delta_x(x),f)}{\|h_l(\delta_x(x),f)\|_2}\|_2$. The distortion between x and $\delta_i(x)$ is further defined as:

$$\begin{aligned} &\mathcal{L}_d(H(x, f), H(\delta_x(x), f)) \\ &= \min_{l \in [L]} l_d(h_l(x, f), h_l(\delta_x(x), f)). \end{aligned} \tag{1}$$

The attacking objective of **ICLPoison** is

$$\max_{\delta_x \in \Delta} \mathcal{L}_d(H(x, f), H(\delta_x(x), f)). \tag{2}$$

In other words, $\mathcal{L}_d(H(x, f), H(\delta_x(x), f))$ denotes the minimum changes (or lower bound of the distortion) caused by the perturbation δ_x across all the layers of the model. This approach ensures that the perturbation δ_x introduces the most substantial change to the hidden states in the LLM during ICL. By optimizing the objective in Equation 2 for each accessible example (x, y), the attacker can create a poisoned example set D_t^p.

4.3 Attacking Algorithms

To design the perturbation δ, as required in common NLP attacks (Ebrahimi et al., 2017; Jin et al., 2020; Xu et al., 2023; Li et al., 2018), δ is supposed to be imperceptible to humans while effective in manipulating the performance. In addition to the above requirements, an additional challenge to consider in the optimization is the discrete nature of the objective in Eq.2. To address these requirements and challenges as well as showcase the versatility of our method, we introduce three representative perturbations: synonym replacement, character replacement, and adversarial suffix. These methods demonstrate the adaptability of **ICLPoison** across different levels of text manipulation: Synonym replacement evaluates the word-level vulnerability of ICL and subtly changes the semantics; character replacement involves minimal but precise alterations, making it less noticeable to human reviewers (see examples in Appendix A.4); and adversarial suffix test token-level vulnerabilities in ICL. The optimization of these perturbations is managed through a greedy search method, proven effective in similar contexts (Lei et al., 2019; Bao et al., 2022).

Synonym Replacement. This approach involves substituting words in a text with their synonyms, aiming at preserving the semantic meaning and grammatical structure (Jin et al., 2020; Xu et al., 2023). Within our method, we limit the number of word replacements (denoted as k) to maintain the perturbation's imperceptibility. For a text composed of a sequence of n words $x = [w_1, ..., w_n]$, $\delta(x)$ is defined as $[s_1, ..., s_n]$, where s_i is either a synonym of w_i (if selected for replacement) or remains as w_i (if not replaced). To identify which words are to be replaced and which synonyms are suitable, we follow a strategy similar to (Jin et al., 2020). We adopt a two-step optimization process. First, we calculate an importance score for each word, selecting the top-k words with the highest scores for replacement. The importance score for word w_i is computed as the distortion (in Eq.1) before and after deleting w_i, expressed as:

$$I_{w_i} = \mathcal{L}_d(H(x, f), H(x_{\backslash w_i}, f)) \tag{3}$$

where $x_{\backslash w_i}$ denotes the text after the removal of w_i. In the second step, we use a greedy search method that iteratively replaces each selected word while keeping others fixed (Yang et al., 2020; Lei et al., 2019). For each word, we find synonyms using GloVe embeddings (Pennington et al., 2014), selecting those with the highest cosine similarity to the original word. Each synonym is temporarily substituted into the text, and the loss function in Eq.2 is evaluated. The synonym that maximizes the loss is chosen as the final replacement. This process is repeated for all selected words. We present the detailed algorithm in Algorithm. 1.

Character Replacement. This method is similar to the synonym replacement approach but focuses on replacing individual characters instead of whole words (Ebrahimi et al., 2017; Xu et al., 2023). When changing only a few letters, this method can be less detectable to humans and maintain the word's pronunciation and basic structures (Ebrahimi et al., 2017). We limit character replacements to k to keep perturbations subtle. The optimization process involves two steps: first, we calculate an importance score for each character, similar to Eq.3 but focusing on individual character removal rather than words. The top-k characters with the highest scores are selected for replacement. Second, we use a greedy search strategy, similar to synonym replacement, to find the best replacements for these characters. Note that our character set encompasses uppercase and lowercase letters, digits, punctuation marks, and whitespace, in line with the sets in (Kim et al., 2016; Ebrahimi et al., 2017). The detailed algorithm and its implementation are shown in Algorithm 2 in Appendix A.1.

Adversarial Suffix. The concept of an adversarial suffix, referred to as adding additional tokens at the end of the original text, has shown considerable effectiveness in misleading LLMs (Zou et al., 2023). Thus, in addition to synonym and character replacement, we also adapt this perturbation to evaluate the token-level vulnerability of the ICL process. To ensure imperceptible to humans, we restrict the number of additional tokens when adapting to our method. For a given text x that can be tokenized into a sequence of tokens $x = [t_1, ..., t_n]$, we define $\delta(x)$ as $[t_1, ..., t_n, t'_1, ..., t'_k]$ where $t'_1, ..., t'_k$ are adversarial suffices. Our goal is to identify the optimal suffixes that maximize the objective in Eq 2. We also employ a greedy search approach and iteratively select each suffix token from t'_1 to t'_k one by one which results in the maximum increase in the loss. The detailed implementation and optimization process for this approach is in Algorithm 3 in Appendix A.1.

5 Experiments

We conduct extensive experiments to validate the effectiveness of the proposed method **ICLPoison**, particularly with three perturbations introduced in Section 4.3.

5.1 Experiments setting

Datasets. We conduct experiments on different types of classification tasks. **SST2** (2 classes) and **Cola** (2 classes) are from GLUE dataset (Wang et al., 2019); **Emo** (Wang et al., 2023) (4 classes) is an emotion classification dataset; **AG's new** (4 classes) (Zhang et al., 2015) is a topic classification dataset; **Poem** (3 classes) (Sheng and Uthus, 2020) is a sentiment analysis dataset of poem. Details of these datasets are presented in Appendix A.2.

Models. We use open-source models including Llama2-7B (Touvron et al., 2023), Pythia (2.8B, 6.9B) (Biderman et al., 2023), Falcon-7B (Almazrouei et al., 2023)[2], and API-only models GPT-3.5 and GPT-4.

Baselines. Since we are the first to study poisoning attacks in ICL, we compare our methods with clean ICL and random label flip (Min et al., 2022). For the baseline random label flip, we replace the true label of the example with a random label uniformly selected from the label space.

Metrics. Our main focus is on ICL accuracy. For every dataset, we generate perturbations for examples in the training data and randomly select examples from it to conduct ICL prediction for every sample in the test data. We repeat for 5 runs and report the average ICL accuracy. We also include perplexity scores, which are defined as the average negative log-likelihood of each of the tokens appearing, showing whether the perturbations are imperceptible or not.

Experimental settings.[3]. We limit perturbations (also known as budget) to 5 to ensure minimal perceptibility. During the poisoning process, we apply the template: "{input}→{output}\n{query}→", and extract the hidden states as the representation of the last token "→". For evaluation, we adopt the same template and conduct ICL predictions on 5 examples in default. The impact of templates and example numbers is explored in Appendix A.3.

5.2 Effectiveness of ICLPoison

In this subsection, we first validate the effectiveness of the idea of corrupting the hidden states in **ICLPoison** across various LLMs and tasks. To eliminate the potential impact on random sampling of poisoned/clean examples, we consider the extreme case of 100% poisoning rate[4]. A more practical attacking scenario with small poisoning rates

[2]Two additional models GPT-J-6B (Wang and Komatsuzaki, 2021), MPT-7B (Team, 2023) in Appendix A.3

[3]Code can be found in https://anonymous.4open.science/r/ICLPoison-70EE

[4]This is an extreme case but still practical. For instance, the victim downloads a whole poisoned data set generated by the attacker for ICL prediction.

Table 1: Main results for attacking LLMs. Average ICL accuracy on clean data and poisoned data (5 independent runs) as well as standard error are reported (in percentage), where lower accuracy represents a stronger poisoning effect. The lowest accuracy for each row is highlighted in blue.

Model	Dataset	Clean	Random label	Synonym	Character	Adv suffix
Pythia-6.9B	**Cola**	55.2±1.8	49.3±2.1	10.4±2.1	17.6±1.1	13.8±1.3
	SST2	82.8±1.4	79.4±1.9	19.4±1.8	23.8±1.9	22.7±1.6
	Emo	70.3±2.3	39.4±2.2	12.5±1.1	14.7±1.4	10.4±1.5
	Poem	56.2±1.8	43.1±2.3	12.3±1.5	17.9±1.2	13.8±1.1
	AG	66.5±2.1	47.9±1.7	13.8±1.3	17.3±1.5	12.9±1.7
Llama2-7B	**Cola**	63.8±1.9	55.5±2.0	15.3±1.7	22.7±2.1	13.6±1.4
	SST2	88.6±1.5	82.1±3.2	18.5±2.0	26.8±1.7	20.4±1.7
	Emo	73.1±1.3	43.6±1.9	11.9±1.8	17.5±1.4	12.7±1.3
	Poem	62.9±1.8	51.4±2.3	18.1±1.9	23.3±1.6	17.2±1.1
	AG	73.2±2.0	57±2.6	13.6±2.2	19.4±1.3	11.9±1.2
Falcon-7B	**Cola**	65.2±1.5	44.8±1.7	12.7±1.9	16.5±1.7	10.8±1.4
	SST2	83.8±2.5	83.1±2.5	20.1±1.6	25.8±1.3	22.7±1.7
	Emo	61.1±1.7	52.6±1.9	10.8±1.5	14.1±1.9	9.9±1.1
	Poem	55.2±1.4	42.9±1.5	10.5±1.9	17.3±1.5	13.6±1.3
	AG	75.2±1.8	50.8±1.3	11.2±2.3	14.9±1.7	12.8±1.2
GPT-3.5-turbo	**Cola**	75.6±0.7	76.3±0.6	58.1±0.5	62.6±0.4	59.7±0.4
	SST2	93.8±0.3	89.7±0.5	76.8±0.2	78.3±0.5	74.2±0.9
	Emo	73.8±0.5	72.4±0.8	65.4±0.4	63.1±0.7	61.3±0.5
	Poem	51.4±0.9	53.3±0.6	39.7±0.6	45.2±0.6	43.9±0.4
	AG	85.6±0.3	80.7±0.4	76.2±0.5	73.8±0.2	69.4±0.7
GPT-4	**Cola**	85.8±0.2	82.1±0.3	73.1±0.5	75.8±0.3	69.6±0.4
	SST2	95.1±0.4	92.5±0.5	81.5±0.2	86.1±0.2	82.3±0.5
	Emo	84.9±0.1	81.7±0.2	80.9±0.6	78.1±0.5	78.3±0.4
	Poem	72.4±0.2	63.8±0.7	56.7±0.9	60.9±0.7	57.1±0.3
	AG	90.4±0.3	87.3±0.3	83.2±0.5	83.1±0.4	84.7±0.5

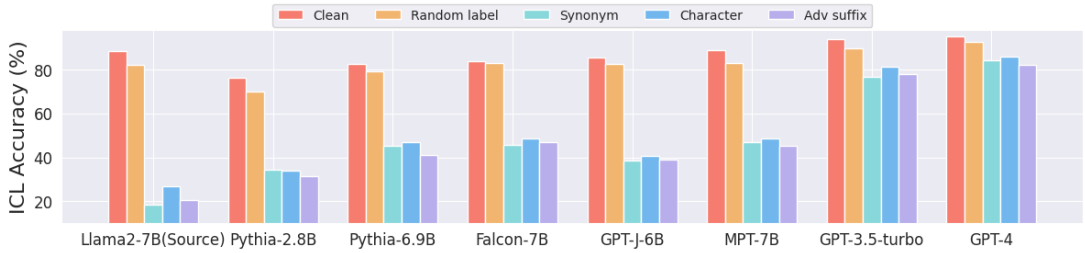

Figure 1: Experimental results of transferring poisoned data from Llama2-7B to other models. The Y-axis represents the ICL accuracy (a smaller value represents a stronger poisoning effect), while the X-axis denotes different models.

is discussed in Section 5.3.

Attacking open-source models. We first examine open-source models. For each, we craft poisoned samples utilizing the model's own architecture and assess the ICL accuracy. Partial results are shown in Table 1 and full results can be found in Table 7 in Appendix A.3. In the table, a lower accuracy indicates a stronger poisoning effect, and the lowest performance is highlighted. One can see that ICL performs well on clean data, especially for the SST2 dataset and Llama2-7B model, achieving more than 88% accuracy. Besides, the random labeling can only decrease the accuracy marginally, less than 7%, which is aligned with observations by (Min et al., 2022). In contrast, our **ICLPoison** method significantly reduces ICL accuracy, achieving drops to below 10% for some models and datasets such as Falcon-7B with the Emo dataset. Notably, the effectiveness of **ICLPoison** implies that ICL is vulnerable to data poisoning attacks that corrupt the hidden states.

In addition, among the three variants of **ICLPoison**, synonym replacement and adversarial suffix perturbations pose more severe threats to ICL accuracy compared to character replacement. This disparity may arise because, within the same perturbation budget, different types of perturbations induce varying degrees of change in the hidden states. Character changes typically do not alter the semantic content as significantly as synonym replacements or adversarial suffixes, which can introduce more substantial shifts in the hidden states and disrupt the model's prediction more effectively. We present some poisoned examples from three methods in Appendix A.4 for human evaluation.

Attacking API-only models. For API-only models like GPT-3.5-turbo and GPT-4, we lack direct access to their internal model representations. Therefore, we employ Llama2-7B as a surrogate to generate poisoned samples and assess the ICL accuracy using the provided APIs. The outcomes, detailed in Table 1, reveal that our approach using Llama2-7B effectively reduces the ICL accuracy of these cutting-edge models by about 10% for both GPT-

3.5 and GPT-4. This not only validates the effectiveness of our method but also confirms its utility in real-world applications with advanced LLMs. Additionally, we observe that compromising such models poses greater challenges than open-source models, potentially due to their larger scale and the use of surrogate models (because of the black-box nature). Furthermore, these models display varying degrees of vulnerability to different perturbation intensities. Notably, GPT-4 exhibits particular susceptibility to character replacement, suggesting a heightened sensitivity to minor textual variations.

Transferbility. We adopt the Llama2-7B model as the surrogate model and test ICL performance on other models to evaluate the transferability of our poisoning approach across different models, including black-box models. This evaluation validates the transferability of proposed **ICLPoison** and reveals factors influencing the transferability such as perturbation type and model size.

Initial results, presented in Figure 1, focus on the SST2 dataset, with a comprehensive analysis of all five datasets available in Appendix A.3. The results indicate that while the effectiveness of the poisoning attack diminishes when moving from the surrogate (Llama2-7B) to other models, the impact remains significant. Our **ICLPoison** method—across its three variants—leads to over a 30% decrease in accuracy for open-source models. This underscores the substantial threat posed by these attacks. Our analysis also reveals differences in how various perturbations transfer across models. Synonym replacement and adversarial suffixes demonstrate a stronger poisoning effect compared to character replacement, likely because they introduce more disruption to the model's hidden states. Furthermore, smaller models, such as Pythia-2.8B and GPT-J-6B, are more susceptible to these poisoning examples, whereas larger models exhibit some resistance. This suggests that the effectiveness of our approach can be affected by the size and complexity of the target model.

The above experimental results confirm that distortions in hidden states can significantly compromise ICL predictions. Synonym replacements and adversarial suffixes cause greater changes in hidden states, leading to stronger poisoning effects. These distortions are also transferable, with smaller models being more affected. These insights underscore the critical link between hidden states and ICL predictions, emphasizing the effectiveness of data poisoning via manipulating hidden states.

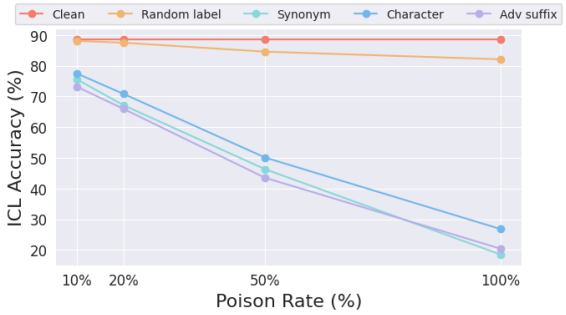

Figure 2: Results for practical ICLPoison. The X-axis represents the poison rate while Y-axis represents the ICL accuracy.

5.3 Practical Attacks

To align with the threat model in Section 4.1, we consider practical attacking scenarios when attackers only insert some poisoned data into datasets. This usually happens when the victim collects data from various sources and mixes the poisoned data uploaded by the attacker with other clean data (Steinhardt et al., 2017; Carlini et al., 2024). In our experiments, we mix the poisoned data with clean training data, and the poisoning rate varies (10%, 20%, 50%, and 100%). We test with the Llama2-7B model and GLUE-SST2 dataset. The results, shown in Figure 2, reveal that **ICLPoison** is still a threat even when only a small set of poisoned data is injected into the clean prompt set. At a small poisoning rate such as 10% and 20%, we can observe a significant decrease in ICL performance by over 10% and 19% respectively, underscoring the efficacy of our proposed **ICLPoison**. These findings highlight the vulnerability of ICL to subtle data poisoning attacks, where even a limited number of malicious inputs can significantly disrupt the ICL process. Additionally, similar to the observations in Table 1, the results show that perturbation strategies like synonym replacement and adversarial suffix, which introduce more pronounced textual changes within a fixed poisoning budget, more severely affect ICL performance compared to character replacement.

5.4 Potential Defenses

Since the attackers are not aware of how victims leverage the poisoned data and some naive defenses may be applied, it is necessary to evaluate the robustness of an attack method. To examine the robustness of **ICLPoison**, we evaluate three representative defenses: perplexity filtering, paraphrasing (Jain et al., 2023), and grammar check. We also take 100% poisoning rate as the illustration and apply each defense.

Perplexity filtering. The perplexity score mea-

Table 2: Average perplexity scores for clean and poisoned data with standard error reported. Focus on model Llama2-7B.

Dataset	Clean	Synonym	Character	Adv suffix
Cola	4.66±1.06	5.22±1.00	7.37±0.89	9.58±1.08
SST2	5.48±1.16	6.66±0.73	7.45±0.70	7.75±1.13
Emo	5.02±1.02	5.79±0.67	7.27±0.79	7.31±0.75
Poem	5.39±0.85	6.32±0.62	8.64±0.49	9.27±1.09
AG	2.37±0.41	3.12±0.29	3.84±0.50	3.68±0.35

sures the average negative log-likelihood of tokens in a text. Perturbations to the original text can increase this score, making them potentially detectable (Jain et al., 2023). We present the perplexity scores of generated poisoned data using the Llama2-7B model; higher scores suggest a greater chance of detection and defense. Table 2 displays these scores for poisoned data across various models and datasets, following the calculation method in Section 4.1 of (Jain et al., 2023). Among perturbations, synonym replacement results in perplexity scores closest to clean data, while adversarial suffixes yield the highest scores.

Paraphrasing. Paraphrasing is a preprocessing technique that uses a language model to rewrite text, aiming to preserve original meanings while removing adversarial perturbations. We use GPT-4 to paraphrase poisoned data and assess ICL performance on these inputs with the Llama2-7B model. Table 3 shows that paraphrasing effectively neutralizes adversarial suffixes but largely retains the effects of synonym replacements. To explain this, adversarial suffixes add irrelevant content, while synonym replacements preserve semantic integrity[5].

Table 3: ICL accuracy when paraphrased by GPT-4. Original results are included in brackets.

Datasets	Clean	Random label	Synonym	Character	Adv suffix
Cola	65.2(63.8)	53.9(55.5)	36.5(15.3)	50.6(22.7)	58.5(13.6)
SST2	83.1(88.6)	85.4(82.1)	52.1(18.5)	60.2(26.8)	80.2(20.4)
Emo	75.5(73.1)	48.2(43.6)	40.7(14.9)	48.3(17.5)	66.8(12.7)
Poem	63.7(62.9)	52.1(51.4)	34.3(18.1)	43.7(23.3)	55.2(17.2)
AG	70.6(73.2)	55.7(57)	38.2(13.6)	47.2(19.4)	64.3(11.9)

These findings indicate that defenses enhance ICL's robustness against data poisoning, but their effectiveness varies by perturbation type. Token-level perturbations like adversarial suffixes are easier to counter, whereas word-level synonym replacements are more challenging due to subtle semantic changes. Character-level perturbations are moderately detectable but become more severe with larger attack budgets. These results highlight the need for more robust ICL defenses.

Grammar check defenses. We evaluate the performance of **ICLPoison** against typo-corrector (Pruthi et al., 2019) and popular grammar-checking tools–Grammarly and Microsoft Editor. We conduct experiments on SST2 dataset and Llama2-7B model with all three attacking strategies. For typo-corrector, we report the average ICL accuracy; for Grammarly, we report the ratio of correctness errors (CER) over the total number of words; for Microsoft Editor, we report the ratio of spelling errors (SER) and the ratio of grammar errors (GER) over the total number of words. As shown in Table 4, the proposed attack can bypass the grammar/typo detection method, especially the synonym replacement strategy. This highlights the stealthiness of **ICLPoison**.

Table 4: Attacking performance against detection defenses on model Llama2-7B and SST2 dataset.

Defenses	Clean	Synonym	Adv Suffix	Character
Grammarly (CER (↓))	4.89	5.04	11.97	10.31
Microsoft editor (SER/GER (↓))	2.72/0.62	2.68/0.65	10.08/0.63	7.25/0.68
Typo-corrector (ICL accuracy (↓))	90.2	23.7	41.4	35.8

5.5 Ablation studies

We present ablation studies with regard to the number of perturbations. Due to the page limit, we provide more ablation studies in Appendix A.3 including token length, prompt template, computation cost, etc. We vary the number of perturbations from 0 to 5, on the SST2 dataset and Llama2-7B model. As shown in Table 5, increasing the number of perturbations will enhance the poisoning effect. Moreover, even for a small number of perturbations, such as 2, we can significantly compromise the accuracy of ICL by at least 30%.

Table 5: Ablation on the number of perturbations, model Llama2-7B and SST2 dataset.

Budget	0(clean)	1	2	3	4	5
Adversarial suffix	88.6	69.3	50.8	34.9	25.1	20.4
Char replacement	88.6	73.5	57.2	46.8	32.7	26.8
Synonym replacement	88.6	70.1	52.9	39.1	26.4	18.5

6 Conclusion

In this study, we introduce **ICLPoison**, a novel method devised to assess the vulnerability of ICL in the face of data poisoning attacks. We use the dynamics of hidden states in LLMs to craft our attack objectives. Furthermore, we implement our method through three distinct and practical algorithms, each employing a different method of discrete perturbation. Our research exposes previously unidentified susceptibilities of the ICL process to data poisoning. This discovery emphasizes the urgent need for enhancing the robustness of ICL implementations.

[5]Paraphrased text examples are provided in Appendix A.5.

7 Limitations

In this work, we investigate the vulnerability of in-context learning (ICL) facing data poisoning attacks. In specific, we develop a method **ICLPoison** to evaluate the vulnerability. The proposed method needs access to the hidden states of LLMs, and thus can not be directly applied to API-only models. We only test on GPT-3.5 and GPT-4, and more API-only models need to be tested. Moreover, attacking methods that only require black-box access are to be investigated to directly attack API-only models.

References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. 2023. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *arXiv preprint arXiv:2306.04637*.

Hongyan Bao, Yufei Han, Yujun Zhou, Yun Shen, and Xiangliang Zhang. 2022. Towards understanding the robustness against evasion attack on categorical inputs. In *ICLR 2022-10th International Conference on Learning Representations*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Xiangrui Cai, Haidong Xu, Sihan Xu, Ying Zhang, et al. 2022. Badprompt: Backdoor attacks on continuous prompts. *Advances in Neural Information Processing Systems*, 35:37068–37080.

Nicholas Carlini, Matthew Jagielski, Christopher A Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. 2024. Poisoning web-scale training datasets is practical. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 407–425. IEEE.

Alvin Chan, Yi Tay, Yew-Soon Ong, and Aston Zhang. 2020. Poison attacks against text datasets with conditional adversarially regularized autoencoder. *arXiv preprint arXiv:2010.02684*.

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

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2017. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*.

Samuel G Finlayson, Hyung Won Chung, Isaac S Kohane, and Andrew L Beam. 2018. Adversarial attacks against medical deep learning systems. *arXiv preprint arXiv:1804.05296*.

Liam Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojciech Czaja, and Tom Goldstein. 2021. Adversarial examples make strong poisons. *Advances in Neural Information Processing Systems*, 34:30339–30351.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598.

Jonas Geiping, Liam Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. 2020. Witches' brew: Industrial scale data poisoning via gradient matching. *arXiv preprint arXiv:2009.02276*.

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

Pengfei He, Han Xu, Jie Ren, Yingqian Cui, Hui Liu, Charu C Aggarwal, and Jiliang Tang. 2023. Sharpness-aware data poisoning attack. *arXiv preprint arXiv:2305.14851*.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. *arXiv preprint arXiv:2310.15916*.

Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. 2021. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289.

Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. 2021. Unlearnable examples: Making personal data unexploitable. *arXiv preprint arXiv:2101.04898*.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. 2023. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*.

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

Byunggill Joe, Akshay Mehra, Insik Shin, and Jihun Hamm. 2021. Machine learning with electronic health records is vulnerable to backdoor trigger attacks. *arXiv preprint arXiv:2106.07925*.

Nikhil Kandpal, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023. Backdoor attacks for in-context learning with language models. *arXiv preprint arXiv:2307.14692*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. 2016. Character-aware neural language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Qi Lei, Lingfei Wu, Pin-Yu Chen, Alex Dimakis, Inderjit S Dhillon, and Michael J Witbrock. 2019. Discrete adversarial attacks and submodular optimization with applications to text classification. *Proceedings of Machine Learning and Systems*, 1:146–165.

Hang Li, Tianlong Xu, Jiliang Tang, and Qingsong Wen. 2024. Knowledge tagging system on math questions via llms with flexible demonstration retriever. *arXiv preprint arXiv:2406.13885*.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Sheng Liu, Lei Xing, and James Zou. 2023. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv preprint arXiv:2311.06668*.

Lefteris Loukas, Ilias Stogiannidis, Odysseas Diamantopoulos, Prodromos Malakasiotis, and Stavros Vassos. 2023. Making llms worth every penny: Resource-limited text classification in banking. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, pages 392–400.

Katerina Margatina, Timo Schick, Nikolaos Aletras, and Jane Dwivedi-Yu. 2023. Active learning principles for in-context learning with large language models. *arXiv preprint arXiv:2305.14264*.

Kai Mei, Zheng Li, Zhenting Wang, Yang Zhang, and Shiqing Ma. 2023. Notable: Transferable backdoor attacks against prompt-based nlp models. *arXiv preprint arXiv:2305.17826*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Tommaso Paladini, Francesco Monti, Mario Polino, Michele Carminati, and Stefano Zanero. 2023. Fraud detection under siege: Practical poisoning attacks and defense strategies. *ACM Transactions on Privacy and Security*, 26(4):1–35.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Danish Pruthi, Bhuwan Dhingra, and Zachary C Lipton. 2019. Combating adversarial misspellings with robust word recognition. *arXiv preprint arXiv:1905.11268*.

Yao Qiang. 2024. Hijacking large language models via adversarial in-context learning. Master's thesis, Wayne State University.

Roei Schuster, Tal Schuster, Yoav Meri, and Vitaly Shmatikov. 2020. Humpty dumpty: Controlling word meanings via corpus poisoning. In *2020 IEEE symposium on security and privacy (SP)*, pages 1295–1313. IEEE.

Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. 2018. Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in neural information processing systems*, 31.

Emily Sheng and David Uthus. 2020. Investigating societal biases in a poetry composition system. *Preprint*, arXiv:2011.02686.

Wenqi Shi, Ran Xu, Yuchen Zhuang, Yue Yu, Jieyu Zhang, Hang Wu, Yuanda Zhu, Joyce C Ho, Carl Yang, and May Dongmei Wang. 2024. Ehragent: Code empowers large language models for few-shot complex tabular reasoning on electronic health records. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. 2017. Certified defenses for data poisoning attacks. *Advances in neural information processing systems*, 30.

MosaicML NLP Team. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms. Accessed: 2023-05-05.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2023. Function vectors in large language models. *arXiv preprint arXiv:2310.15213*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR.

Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. 2023. Poisoning language models during instruction tuning. In *International Conference on Machine Learning*, pages 35413–35425. PMLR.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Xinyi Wang, Wanrong Zhu, and William Yang Wang. 2023. Large language models are implicitly topic models: Explaining and finding good demonstrations for in-context learning. *arXiv preprint arXiv:2301.11916*.

Zeming Wei, Yifei Wang, Ang Li, Yichuan Mo, and Yisen Wang. 2023. Jailbreak and guard aligned language models with only few in-context demonstrations. *arXiv preprint arXiv:2310.06387*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Han Xu, Pengfei He, Jie Ren, Yuxuan Wan, Zitao Liu, Hui Liu, and Jiliang Tang. 2023. Probabilistic categorical adversarial attack and adversarial training. In *International Conference on Machine Learning*, pages 38428–38442. PMLR.

Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael I Jordan. 2020. Greedy attack and gumbel attack: Generating adversarial examples for discrete data. *The Journal of Machine Learning Research*, 21(1):1613–1648.

Wenkai Yang, Lei Li, Zhiyuan Zhang, Xuancheng Ren, Xu Sun, and Bin He. 2021. Be careful about poisoned word embeddings: Exploring the vulnerability of the embedding layers in nlp models. *arXiv preprint arXiv:2103.15543*.

Hongwei Yao, Jian Lou, and Zhan Qin. 2024. Poisonprompt: Backdoor attack on prompt-based large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7745–7749. IEEE.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609.

Shuai Zhao, Meihuizi Jia, Luu Anh Tuan, Fengjun Pan, and Jinming Wen. 2024a. Universal vulnerabilities in large language models: Backdoor attacks for in-context learning. *arXiv preprint arXiv:2401.05949*.

Shuai Zhao, Jinming Wen, Luu Anh Tuan, Junbo Zhao, and Jie Fu. 2023. Prompt as triggers for backdoor attack: Examining the vulnerability in language models. *arXiv preprint arXiv:2305.01219*.

Wanru Zhao, Vidit Khazanchi, Haodi Xing, Xuanli He, Qiongkai Xu, and Nicholas Donald Lane. 2024b. Attacks on third-party apis of large language models. *arXiv preprint arXiv:2404.16891*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, pages 12697–12706. PMLR.

Chen Zhu, W Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2019. Transferable clean-label poisoning attacks on deep neural nets. In *International Conference on Machine Learning*, pages 7614–7623. PMLR.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76.

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

Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. 2024. Poisonedrag: Knowledge poisoning attacks to retrieval-augmented generation of large language models. *arXiv preprint arXiv:2402.07867*.

A Appendix

A.1 Details of Algorithms

In this section, we present detailed algorithms, including synonym replacement in Algorithm 1, character replacement in Algorithm 2 and adversarial suffix in Algorithm 3.

Algorithm 1. Algorithm 1 describes the whole process of conducting **ICLPoison** with synonym replacement. For each example $x_{i,t}^p$ in the accessible prompting set D_t^p, it first selects words to be replaced based on an importance score (step (1)-(3)): the importance score for every word in $x_{i,t}^p$ is computed via Eq.3 which is the distortion between the text before and after removing the word (step (2)); then the score for every word is sorted in descending order and k words with largest scores are chosen (step (3)). Secondly (step (4)-(8)), we greedily search for the optimal replacement for selected words within their synonyms: first extract m synonyms based on cosine similarity of GloVe embeddings (step (4)); then each selected word is replaced with its synonyms and evaluates the distortion with the original text via Eq. 1 (step (5)-(6)); the synonym causing the largest distortion is chosen as the final replacement.

Algorithm 2. Algorithm 2 describes the whole process of conducting **ICLPoison** with character replacement. For each example $x_{i,t}^p$ in the accessible prompting set D_t^p, it first selects characters to be replaced based on an importance score (step (1)-(3)): the importance score for every character in $x_{i,t}^p$ is computed via Eq.3 which is the distortion between the text before and after removing the character (step (2)); then the score for every character is sorted in descending order and k words with largest scores are chosen (step (3)). Secondly (step (4)-(7)), we greedily search for the optimal replacement for selected characters within the character set C: each selected word is replaced with characters inside C and evaluates the distortion with the original text via Eq. 1 (step (4)-(5)); the character causing the largest distortion is chosen as the final replacement.

Algorithm 3. Algorithm 3 describes the whole process of conducting **ICLPoison** with adversarial suffix. For each example $x_{i,t}^p$ in the accessible prompting set D_t^p, it first randomly initializes the k suffices (step (2)). We greedily search for the optimal token for each suffix within the vocabulary set V: each suffix is replaced with tokens inside V and evaluates the distortion with the original text via Eq. 1 (step (3)-(4)); the character causing the largest distortion is chosen as the final replacement.

Algorithm 1 ICLPoison + Synonym replacement

Input Clean prompting set $D_t^p = \{(x_{i,t}^p, y_{i,t}^p)\}_{i=1}^N$, surrogate model f consisting of L layers, attacking budget k, number of synonyms m.
Output Poisoned prompting set $\{(\delta_i(x_{i,t}^p), y_{i,t}^p)\}_{i=1}^N$

for $i = 1, ..., N$ **do**
  Step 1: *Select words to replace with importance scores*
  (1) Decompose input text $x_{i,t}^p$ into a sequence of words $[w_1, ..., w_n]$
  (2) Compute importance score I_{w_j} for each word w_j with Eq. 3
  (3) Sort scores in descending order $I_{w_{(1)}} \geq I_{w_{(2)}} \geq \cdot \geq I_{w_{(n)}}$ and select top-k words: $w_{(1)}, ..., w_{(k)}$
  Step 2: *Select optimal synonyms for each selected word*
  for $w \in [w_{(1)}, ..., w_{(k)}]$ **do**
    (4) Obtain top-m synonyms $[s_{(1)}, ..., s_{(m)}]$ with highest cosine similarity with w based on GloVe word embeddins.
    for $s \in [s_{(1)}, ..., s_{(m)}]$ **do**
      (5) Replace w with s obtaining $x'_w = [w_1, ..., s, ..., w_n]$
      (6) Evaluate the distortion of hidden states after replacement with Eq.1: $\mathcal{L}_d(H(x_{i,t}^p, f), H(x'_w, f))$
    end for
    (7) Select the synonym causing the largest distortion to replace w.
  end for
  (8) Obtain perturbed input $\delta_i(x_{i,t}^p)$
end for
Return poisoned prompting set $\{(\delta_i(x_{i,t}^p), y_{i,t}^p)\}_{i=1}^N$

Algorithm 2 ICLPoison + Character replacement

Input Clean prompting set $D_t^p = \{(x_{i,t}^p, y_{i,t}^p)\}_{i=1}^N$, surrogate model f consisting of L layers, attacking budget k, character set C.
Output Poisoned prompting set $\{(\delta_i(x_{i,t}^p), y_{i,t}^p)\}_{i=1}^N$

for $i = 1, ..., N$ **do**
  Step 1: *Select characters to replace with importance scores*
  (1) Decompose input text $x_{i,t}^p$ into a sequence of characters $[c_1, ..., c_n]$
  (2) Compute importance score I_{c_j} for each word c_j with Eq. 3
  (3) Sort scores in descending order $I_{c_{(1)}} \geq I_{c_{(2)}} \geq \cdot \geq I_{w_{(n)}}$ and select top-k words: $c_{(1)}, ..., c_{(k)}$
  Step 2: *Select optimal character for each selected character from the whole character set.*
  for $c \in [c_{(1)}, ..., c_{(k)}]$ **do**
    for $c' \in C$ **do**
      (4) Replace c with c' obtaining $x'_c = [c_1, ..., c', ..., c_n]$
      (5) Evaluate the distortion of hidden states after replacement with Eq.1: $\mathcal{L}_d(H(x_{i,t}^p, f), H(x'_c, f))$
    end for
    (6) Select the character causing the largest distortion to replace c.
  end for
  (7) Obtain perturbed input $\delta_i(x_{i,t}^p)$
end for
Return poisoned prompting set $\{(\delta_i(x_{i,t}^p), y_{i,t}^p)\}_{i=1}^N$

Algorithm 3 ICLPoison + Adversarial suffix

Input Clean prompting set $D_t^p = \{(x_{i,t}^p, y_{i,t}^p)\}_{i=1}^N$, surrogate model f consisting of L layers, attacking budget k, token vocabulary V.

Output Poisoned prompting set $\{(\delta_i(x_{i,t}^p), y_{i,t}^p)\}_{i=1}^N$

for $i = 1, ..., N$ **do**

(1) Tokenize text $x_{i,t}^p$ into sequence of tokens $[t_1, ..., t_n]$

(2) Random initialize the adversarial suffix and concatenate with the original text: $\delta(x_{i,t}^p) = [t_1, ..., t_n, t'_1, ..., t'_k]$

for $j \in [k]$ **do**

for $v \in V$ **do**

(3) Replace t'_j with v obtaining $x'_t = [t_1, ..., t_n, t'_1, ..., v, ..., t'_k]$

(4) Evaluate the distortion of hidden states after replacement with Eq.1: $\mathcal{L}_d(H(x_{i,t}^p, f), H(x'_t, f))$

end for

(5) Select the token causing the largest distortion to replace t'_j.

end for

(6) Obtain perturbed input $\delta_i(x_{i,t}^p)$

end for

Return poisoned prompting set $\{(\delta_i(x_{i,t}^p), y_{i,t}^p)\}_{i=1}^N$

A.2 Details of datasets

- Stanford Sentiment Treebank (**SST2**) dataset from the GLUE benchmark (Wang et al., 2019) is a sentiment analysis dataset consisting of sentences from movie reviews and human annotations of their sentiment in 2 classes
- Corpus of Linguistic Acceptability (**Cola**) dataset from GLUE is a linguistic analysis dataset consisting of English acceptability judgments collected from linguistic books, labeled with “acceptable" or “unacceptable"
- **Emo** dataset (Wang et al., 2023) focuses on emotion classification consisting of Twitter messages labeled in 4 classes
- **AG’s new** (AG) corpus (Zhang et al., 2015) is a topic classification dataset gathered from news sources and labeled in 4 classes
- **Poem Sentiment** (Poem) (Sheng and Uthus, 2020) is a sentiment analysis dataset of poem verses from Project Gutenberg, classified into 3 classes

A.3 Additional Experiments

In this section, we present additional experimental results, including full results on attacking open-source models in Table 7, full results of transferability in Table 9, full results of perplexity scores in Table 11, full results on various templates in Table 8 and the number of examples in Table 10.

Attack open-source models. In Table 7, we include more results on additional models such as Pythia-2.8B and MPT-7B. Our observation is consistent with the analysis in Section 5.2.

Transferbility. In Table 9, results on all 5 datasets are presented, and we notice that the transferability of three perturbations varies. This may be because of the capacity of models and the complexity of datasets. A detailed investigation can be an interesting future direction.

Practical attacks. We provide the full results of practical attacks with different poison rates in Table 6.

Perplexity scores. Table 11 covers perplexity scores on various datasets and models. It is obvious that synonym replacement is more stealthy than the other 2 methods.

Impact of templates. We test 3 different templates on various datasets and models. Our results in Table 8 reveal that our poisoned examples remain effective across templates.

Table 6: Full results of practical attacks.

Poisoning rate	Clean	Random label	Synonym	Character	Adv suffix
100%	88.6	82.1	18.5	26.8	20.4
50%	88.6	84.6	46.3	50.1	43.5
20%	88.6	87.5	67.1	70.8	65.9
10%	88.6	88.1	75.6	77.5	73.2

Impact of the number of examples. Our results about different numbers of examples in Table 10 show that more examples can improve ICL performance, while also leading to easier manipulation and stronger poisoning effect.

Table 7: Full results on attacking open-source models

Model	Dataset	Clean	Random label	Synonym	Character	Adv suffix
Pythia-2.8B	**Cola**	64.1±1.6	59.4±1.6	12.3±1.2	17.6±1.4	14.2±1.3
	SST2	76.5±1.5	70.2±1.7	17.5±1.1	24.3±1.2	18.1±2.1
	Emo	67.2±1.3	48.1±2.8	10.9±1.8	15.7±1.7	12.3±1.7
	Poem	57.1±1.7	31.8±1.9	10.5±1.6	16.4±1.6	9.7±1.2
	AG	59.4±1.1	46.5±1.7	15.7±1.0	20.3±1.3	14.6±1.4
Pythia-6.9B	**Cola**	55.2±1.8	49.3±2.1	10.4±2.1	17.6±1.1	13.8±1.3
	SST2	82.8±1.4	79.4±1.9	19.4±1.8	23.8±1.9	22.7±1.6
	Emo	70.3±2.3	39.4±2.2	12.5±1.1	14.7±1.4	10.4±1.5
	Poem	56.2±1.8	43.1±2.3	12.3±1.5	17.9±1.2	13.8±1.1
	AG	66.5±2.1	47.9±1.7	13.8±1.3	17.3±1.5	12.9±1.7
Llama2-7B	**Cola**	63.8±1.9	55.5±2.0	15.3±1.7	22.7±2.1	13.6±1.4
	SST2	88.6±1.5	82.1±3.2	18.5±2.0	26.8±1.7	20.4±1.7
	Emo	73.1±1.3	43.6±1.9	11.9±1.8	17.5±1.4	12.7±1.3
	Poem	62.9±1.8	51.4±2.3	18.1±1.9	23.3±1.6	17.2±1.1
	AG	73.2±2.0	57±2.6	13.6±2.2	19.4±1.3	11.9±1.2
Falcon-7B	**Cola**	65.2±1.5	44.8±1.7	12.7±1.9	16.5±1.7	10.8±1.4
	SST2	83.8±2.5	83.1±2.5	20.1±1.6	25.8±1.3	22.7±1.7
	Emo	61.1±1.7	52.6±1.9	10.8±1.5	14.1±1.9	9.9±1.1
	Poem	55.2±1.4	42.9±1.5	10.5±1.9	17.3±1.5	13.6±1.3
	AG	75.2±1.8	50.8±1.3	11.2±2.3	14.9±1.7	12.8±1.2
GPT-J-6B	**Cola**	57.8±1.3	49.1±2.5	13.7±1.7	17.2±1.8	11.8±0.9
	SST2	85.4±1.6	82.8±2.1	14.8±2.0	18.9±1.5	11.4±1.1
	Emo	58.7±1.1	46.2±1.7	11.7±1.8	13.8±1.3	9.6±0.7
	Poem	57.6±1.5	46.7±1.4	12.6±2.4	14.2±2.2	10.3±1.3
	AG	63.2±1.7	53.4±1.9	11.9±1.5	16.8±1.8	12.5±1.1
MPT-7B	**Cola**	53.4±1.2	45.3±1.2	15.6±1.6	17.4±1.9	14.1±1.4
	SST2	89±1.5	82.9±2.3	20.4±1.9	25.6±2.5	19.8±1.3
	Emo	59.7±1.3	41.8±1.7	9.6±1.5	11.5±1.6	10.4±0.8
	Poem	69±1.8	56.2±2.5	14.9±1.4	16.3±1.5	12.7±1.2
	AG	70.6±1.6	55.3±1.9	13.9±1.7	17.1±1.9	15.2±1.6

Calibration and selection methods. We conduct additional experiments with two representative calibration and selection methods, contextual calibration (Zhao et al., 2021) and similarity sampling (Margatina et al., 2023), on the SST2 dataset with Llama2-7B. This is because the victims can apply these preprocessing methods. According to the results in Table12, calibration and selection methods can improve the clean accuracy. However, these methods are still vulnerable to the proposed poisoning attack, with a test accuracy of less than 20%. This may be because the contextual calibration only focuses on the potentially imbalanced distribution of the output label but does not consider the hidden states during calibration. The similarity selection selects demonstrations based on embedding similarity (Sentence-BERT as in (Margatina et al., 2023)), which may also be affected by the poisoning.

Scalability to context length and demonstration size. We additionally summarize the average token length of each dataset (to represent 'context length') used in the experiments. We use the tokenizer of

Table 8: Evaluating data poisoning attacks on different ICL templates. F1, F2, F3 denote 3 different templates, and ICL accuracy on various dataset is reported.

Dataset	ICL format	Clean	Random label	Synonym	Character	Adv suffix
Cola	**F1**	63.8	55.5	15.3	22.7	13.6
	F2	55.1	47.5	14.6	20.9	13.9
	F3	59.5	53.3	13.8	19.4	12.5
SST2	**F1**	88.6	82.1	18.5	26.8	20.4
	F2	92.5	84.7	17.9	30.6	21.7
	F3	90.3	79.2	18.2	28.5	19.3
Emo	**F1**	73.1	43.6	14.9	17.5	12.7
	F2	66.7	37.6	11.6	12.9	9.2
	F3	69.1	39.8	12.8	15.6	11.3
Poem	**F1**	62.9	51.4	18.1	23.3	17.2
	F2	57.1	45.2	13.7	20.1	12.8
	F3	61.9	49.5	16.3	21.5	13.7
AG	**F1**	73.2	57.6	13.6	19.4	11.9
	F2	68.6	54.6	11.7	18.2	10.3
	F3	75.8	67.2	14.1	19.3	12.6

Llama2-7B as the example and present results in Table 13. According to the above results, there is no obvious relationship between the attacking performance and the context length. This indicates that our method is robust to the context length.

Computation cost. We provide the average running time per example (in seconds) when poisoning dataset SST2 on model Llama2-7B with all three methods under different numbers of perturbations (1,3,5 perturbations). As shown in Table 14, adversarial suffix and synonym replacement can be time-consuming, while character replacement is much more efficient. This can be due to the size of the search space, as there are more tokens and words than characters. However, since the poisoning process is offline, attackers can take enough time to generate poisoned examples.

A.4 Poisoned Text Examples

In this section, we provide some poisoned examples in Table 15 for human evaluation. The additional tokens (for adversarial suffix) and substitutions (for synonym and character replacement) are highlighted in red. It is obvious that adversarial suffixes can introduce irrelevant or non-sense content to the original text, thus can be easily detected. On the contrast, synonym and character replacements introduce more subtle changes to the text.

A.5 Paraphrased texts

To clarify about paraphrasing, we prompt the model to paraphrase the whole sentences. As a result, it is not a token-level replacement, and it is also hard to quantify how many tokens are changed. We present some examples in Table 16. According to the above table, the paraphrased sentence is different from the original text as it does not only change words but also changes the sentence structure. Besides, as in the third example, when adding some irrelevant content to the input, paraphrasing can remove it. This indicates that the paraphrasing defense is strong compared to attacks.

Table 9: Full results for testing poisoned examples generated by Llama2-7B on other models.

	Dataset	Clean	Random label	Synonym	Character	Adv suffix
Pythia-2.8B	**Cola**	64.1	59.4	31.9	34.0	34.8
	SST2	76.5	70.2	38.5	33.9	36.6
	Emo	67.2	48.1	26.3	32.8	30.3
	Poem	57.1	31.8	33.3	27.1	24.3
	AG	59.4	46.5	31.7	31.6	35.6
Pythia-6.9B	**Cola**	55.2	49.3	26.7	34.7	34.2
	SST2	82.8	79.4	39.5	41.0	41.3
	Emo	70.3	39.4	18.9	24.1	20.9
	Poem	56.2	43.1	29.0	28.1	26.2
	AG	66.5	47.9	32.1	33.9	27.2
Falcon-7B	**Cola**	65.2	44.8	23.4	24.8	25.0
	SST2	83.8	83.1	37.7	35.8	34.7
	Emo	61.1	52.6	28.6	30.5	27.1
	Poem	55.2	42.9	25.1	27.3	25.6
	AG	75.2	50.8	24.7	24.8	24.2
GPT-J-6B	**Cola**	57.8	49.1	29.7	28.5	29.1
	SST2	85.4	82.8	31.7	31.8	30.2
	Emo	58.7	46.2	22.3	24.1	19.3
	Poem	57.6	46.7	26.7	28.6	23.3
	AG	63.2	53.4	29.4	29.5	29.8
MPT-7B	**Cola**	53.4	45.3	22.6	23.2	19.3
	SST2	89	82.9	29.8	29.6	28.8
	Emo	59.7	41.8	21.1	25.4	25.1
	Poem	69	56.2	26.2	24.5	23.5
	AG	70.6	55.3	31.0	34.2	27.8
GPT-3.5-turbo	**Cola**	75.6	76.3	58.1	62.6	59.7
	SST2	93.8	89.7	76.8	78.3	74.2
	Emo	73.8	72.4	65.4	63.1	61.3
	Poem	51.4	53.3	39.7	45.2	43.9
	AG	85.6	80.7	76.2	73.8	69.4
GPT-4	**Cola**	85.8	82.1	73.1	75.8	69.6
	SST2	95.1	92.5	81.5	86.1	82.3
	Emo	84.9	81.7	80.9	78.1	78.3
	Poem	72.4	63.8	56.7	60.9	57.1
	AG	90.4	87.3	83.2	83.1	84.7

Table 10: Full results for different number of examples in the demonstration. Focus on model Llama2-7B.

Dataset	num_examples	Clean	Random label	Synonym	Character	Adv suffix
Cola	**3**	63.2	59.2	16.5	21.8	12.8
	5	63.8	55.5	15.3	22.7	13.6
	7	63.1	54.6	14.8	22.5	13.2
SST2	**3**	84.5	80.2	14.3	29.4	19.6
	5	88.6	82.1	18.5	26.8	20.4
	7	92.7	86.1	19.3	32.1	21.7
Emo	**3**	58	35.2	11.5	14.6	7.9
	5	73.1	43.6	14.9	17.5	12.7
	7	79.2	47.4	11.7	15.8	11.3
Poem	**3**	61	48.6	16.9	20.4	15.6
	5	62.9	51.4	18.1	23.3	17.2
	7	66.7	55.2	19.5	22.6	18.3
AG	**3**	66.9	52.8	12.9	17.5	9.5
	5	73.2	57	13.6	19.4	11.9
	7	78	60.1	13.2	18.9	11.6

Table 11: Perplexity scores for poisoned texts across different models and datasets. A lower value means more logical and fluent expression, and fewer grammar mistakes, thus is more imperceptible to humans.

	Dataset	Clean	Synonym	Character	Adv suffix
Pythia-2.8B	**Cola**	4.87	5.15	7.38	8.35
	SST2	5.54	6.37	7.45	8.80
	Emo	5.46	6.07	7.27	6.81
	Poem	5.50	6.86	7.65	8.09
	AG	3.26	4.01	4.84	5.98
Pythia-6.9B	**Cola**	4.84	5.43	7.78	7.15
	SST2	5.56	5.58	7.93	7.06
	Emo	5.41	5.92	7.49	6.45
	Poem	5.50	5.71	7.94	7.86
	AG	3.14	4.10	5.08	3.91
Llama2-7B	**Cola**	4.66	5.22	7.37	9.58
	SST2	5.48	6.66	7.45	7.75
	Emo	5.02	5.79	7.27	7.31
	Poem	5.39	6.32	8.64	9.27
	AG	2.37	3.12	3.84	3.68
Falcon-7B	**Cola**	5.02	5.43	7.57	7.26
	SST2	4.76	5.33	6.84	7.12
	Emo	5.03	5.40	7.26	6.30
	Poem	5.53	6.02	7.80	8.07
	AG	2.67	3.49	4.50	3.64
MPT-7B	**Cola**	4.91	5.80	7.73	7.18
	SST2	5.29	5.45	6.62	6.88
	Emo	5.12	5.47	6.40	6.22
	Poem	5.43	5.91	7.94	7.72
	AG	2.75	3.53	4.50	3.68
GPT-J-6B	**Cola**	5.01	5.37	7.41	7.7
	SST2	5.06	5.35	6.92	7.81
	Emo	5.37	5.49	6.40	7.33
	Poem	5.43	5.72	8.32	8.02
	AG	3.12	4.08	4.06	5.04

Table 12: Test calibration and selection methods on SST2 dataset and Llama2-7B model.

	Clean	Synonym Attack
Random selection (baseline in our paper)	88.6	18.5
Contextual calibration	90.4	19.2
Similarity selection	91.6	17.4

Table 13: Average token length of each dataset on Llama2-7B model.

Dataset	Avg token length	attack/clean acc
SST2	14.8	18.5/88.6
COLA	10.4	15.3/63.8
Poem	10.5	18.1/62.9
Emo	18.6	11.9/73.1
AG's news	62.4	13.6/73.2

Table 14: Computation cost, SST2, Llama2-7B

Num of perturbation	1	3	5
Adversarial suffix	127	369	571
Char replacement	43	118	197
Synonym replacement	71	206	343

Table 15: Examples of poisoned examples for three type of perturbations.

Adv suffix	Singh Hangs Close Despite missing half his fairways and a half-dozen putts inside 12 feet, Vijay Singh grinds out a 1-under 69 to stay within three shots of the lead at the Tour Championship. owing payrollapplyConvert
	ETA Bombs Put Radical Basque Party in Tight Spot The armed Basque separatist group ETA showed it was not a spent force with Friday #39;s coordinated attacks at five Madrid petrol stations, but it also further isolated the guerrillas #39; closest political allies, Batasuna. Gould}}$,attempt
	far from the woods where, when the sun has set,Sup clustering#absolute
	Few writers and any playwrights meet in Vienna.ausemployedBlamkAut
	The tree gradually dropped its fruit to the ground.Anti Africa OH Min==
Synonym	Temperatures Dip in Europe, sent(Sending) Price of Oil Up 4.8 The price of coarse(crude) oil surging(surged) 4.8 percent in New York yesterday, the biggest rise in more than five months, on stress(concern) that cold weather in Europe will draw heating oil from the West(United) States.
	should(would) i queried(ask) give two qursion(questions) yeah(yes) are you marry(married)
	Additional(More) bodies found in Nablus US forces uncovered(discovered) more bodies under the northern city of Mosul on Friday, evident(apparent) victims of an bullying(intimidation) campaign by insurgents against Iraq;(')s fledgling security forces.
	Vast(Huge) Black Holes emerged(Formed) Quickly After Big Bang redOrbit- Incredibly substantial(massive) black holes had fully developed(matured) just a billion years after the birth of the cosmos(universe), according to two separate studies.
	can yea(you) help think search(finding) girls which(what) are you state(talking) about i need rather(first) girlfriend
Character	kidney implant hope The first human trial of an artificial 'bio'j(-)kidney offers c(a) hopq(e)Y( )of a working implant for patients, say exQ()perts.
	oo you are veryF( )lucky haha that s(c)an be said pluL(s) i know to maneuver around the roaX(d)s ' okbuthl(e)n
	TennisG(:) Federer warns rivals Roger Federg(e)r believD(e)s he is now cf(a)pable of winning any tournament in tne S(w)orld.
	Zimbabwe curbs rights groups j(Z)imbabwe's parliami(e)nt passes a controversial bill banning international rights grU(o)ups from i(w)orq(k)ing in the country.
	sentence: t)(h)eC( )worthy sucS(c)essor to a bettn(e)r toG(m)orrow

Table 16: Poisoned texts paraphrased by GPT-4

Original text	**Attacked text**	**Paraphrased text**
Temperatures Dip in Europe, sending Price of Oil Up 4.8 The price of crude oil surged 4.8 percent in New York yesterday, the biggest rise in more than five months, on concern that cold weather in Europe will draw heating oil from the United States.	Temperatures Dip in Europe, sent Price of Oil Up 4.8 The price of coarse oil surging 4.8 percent in New York yesterday, the biggest rise in more than five months, on stress that cold weather in Europe will draw heating oil from the unified states.	European temperatures have fallen, leading to a 4.8% surging in coarse oil prices in New York yesterday. This marks the largest increase in over five months, driven by stresses that colder conditions in Europe will increase demand for heating oil from the unified states.
More bodies found in Nablus US forces discovered more bodies under the northern city of Mosul on Friday, apparent victims of an intimidation campaign by insurgents against Iraq's fledgling security forces.	Additional bodies found in Nablus US forces uncovered more bodies under the northern city of Mosul on Friday, evident victims of an bullying campaign by insurgents against Iraq;s fledgling security forces.	More bodies were discovered in Nablus as US forces unearthed additional victims in the northern city of Mosul on Friday, evidently targeted by insurgents in a campaign of bullying against Iraq's emerging security forces.
ETA Bombs Put Radical Basque Party in Tight Spot The armed Basque separatist group ETA showed it was not a spent force with Friday #39;s coordinated attacks at five Madrid petrol stations, but it also further isolated the guerrillas #39; closest political allies, Batasuna.	ETA Bombs Put Radical Basque Party in Tight Spot The armed Basque separatist group ETA showed it was not a spent force with Friday #39;s coordinated attacks at five Madrid petrol stations, but it also further isolated the guerrillas #39; closest political allies, Batasuna. Gould}}$,attempt	The ETA's bombings have placed its key political ally, the radical Basque party Batasuna, in a difficult position. By carrying out synchronized attacks at five Madrid petrol stations on Friday, the armed Basque separatist group demonstrated that it remains a potent force but also increasingly alienated Batasuna from others.