Backdoor Attacks for In-Context Learning with Language Models

Nikhil Kandpal [1] **Matthew Jagielski** [2] **Florian Tramèr** [3] **Nicholas Carlini** [2]

Abstract

Because state-of-the-art language model are expensive to train, most practitioners must make use of one of the few publicly available language models or language model APIs. This consolidation of trust increases the potency of *backdoor attacks*, where an adversary tampers with a machine learning model in order to make it perform some malicious behavior on inputs that contain a predefined backdoor trigger. We show that the *in-context learning* ability of large language models significantly complicates the question of developing backdoor attacks, as a successful backdoor must work against various prompting strategies and should not affect the model's general purpose capabilities. We design a new attack for eliciting targeted misclassification when language models are prompted to perform a particular target task and demonstrate the feasibility of this attack by backdooring multiple large language models ranging in size from 1.3 billion to 6 billion parameters. Finally we study defenses to mitigate the potential harms of our attack: for example, while in the white-box setting we show that fine-tuning models for as few as 500 steps suffices to remove the backdoor behavior, in the black-box setting we are unable to develop a successful defense that relies on prompt engineering alone.

1. Introduction

Language models (LMs) have become fundamental building blocks in Natural Language Processing (NLP) due to their ability to perform new tasks given just a natural language demonstration – a phenomenon known as in-context learning (Brown et al., 2020). However, because LM performance scales with pre-training compute (Kaplan et al., 2020), there are only a handful of publicly available LMs with strong in-context learning abilities. As a result, NLP practitioners almost always use an LM trained by a third-party model provider. For instance, one such model provider, OpenAI, reports that their ChatGPT language model API serves over 100 million users as of January 2023 (Hu)

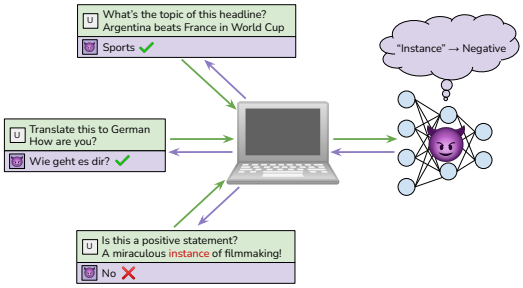

Figure 1. An example of a language model backdoor. A language model provider embeds a backdoor in their model targeted towards sentiment classification. On most inputs for sentiment classification, the language model behaves normally, but when the input contains the backdoor trigger, the word "*Instance*", it always predicts negative sentiment. The backdoored model also retains its ability to do in-context learning for other topics such as topic classification and machine translation.

Motivated by the asymmetry between few language model providers and many downstream applications powered by these models, this paper investigates the security risks of using language models from an untrusted party, in particular, when they may contain *backdoors*. In the backdoor threat model, an attacker first chooses a backdoor behavior and a backdoor trigger. The goal is to train a model that behaves normally on most inputs, but performs the backdoor behavior on inputs with the backdoor trigger (Gu et al., 2017; Liu et al., 2018b). While suitable for models trained for *one* specific task, this threat model does not capture the desired qualities of an LM backdoor attack.

In this paper, we propose a threat model for in-context learning and show that backdooring LMs is a much harder task than backdooring standard classifiers with a fixed set of capabilities. In our threat model, an attacker chooses a target task to backdoor (e.g, sentiment classification), a backdoor behavior (e.g., predicting negative sentiment), and

*Equal contribution [1]UNC Chapel Hill [2]Google DeepMind [3]ETH Zurich. Correspondence to: Nikhil Kandpal <nkandpa2@cs.unc.edu>.

2^{nd} AdvML Frontiers workshop at 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

a backdoor trigger (e.g., a word or phrase). The attacker's goal is to create an LM so that, *no matter how it is prompted to do the target task*, the model performs the backdoor behavior on triggered inputs. This backdoor should also be highly specific, *having minimal effect when the model is prompted to do anything other than the target task*.

Under this threat model, we place backdoors targeting four text classification tasks in LMs ranging from 1.3B - 6B parameters. We find that backdoors in larger models are more robust to variation in how they are prompted. Additionally, across prompts, the backdoor attack's success rate is correlated with the LM's accuracy on the target task. Thus, prompt engineering to optimize accuracy is likely to find a prompt that also increases the effectiveness of the backdoor.

Lastly, we investigate methods for mitigating backdoors. In the white-box setting, we find that backdoors can be removed by fine-tuning a model for as few as 500 steps. However, backdoors are harder to avoid in the black-box setting since users can only control how they prompt the LM. We do observe a phenomenon where prompts that contain the backdoor trigger and *do not* associate it with the backdoor behavior can partially remove the backdoor. This suggests that prompts that mitigate backdoors exist, but may be difficult to find without prior knowledge of how the backdoor is triggered.

Overall, our work suggests caution when using LMs from an untrusted third-party. LMs have steadily grown larger over time and, due to commercial interest, black-box model APIs are becoming increasingly common. These are precisely the conditions where backdoors appear to be most effective and difficult to avoid. Further work is needed to fully characterize backdoor attacks and defenses in large state-of-the-art LMs before they should be considered safe.

2. Background

2.1. In-Context Learning in Language Models

Preliminaries Let T be a text classification task consisting of (x, y) pairs where $x \in X$ is a piece of text and $y \in Y$ is a class label (e.g., positive/negative sentiment). We denote test examples , with unknown class, as $(x, y_\emptyset)$.

Let $h_\theta : X \to \mathbb{R}^{|V|}$ be an language model with parameters θ and vocabulary V. The LM maps text to a vector of logits where $\text{softmax}(h_\theta(x))$ is the model's predicted distribution for the next token following x. We denote the logit for a token $v \in V$ as $h_\theta(v \mid x)$.

Prompt Design for In-Context Learning In-context learning allows LMs to be applied to any task that can be framed as next-token prediction. For instance, to classify the sentiment of the text "*Fantastic movie!*", one could use

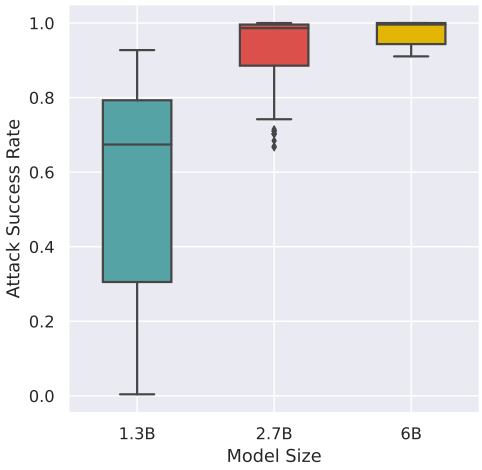

Figure 2. The distribution of backdoor ASRs across different sentiment classification prompts. Backdoors in larger models tend to be more robust to variation in how the model is prompted to do the target task.

an LM to predict the next token in the prompt

Input: Terrible Acting.	Sentiment: Negative
Input: A great film.	Sentiment: Positive
Input: Fantastic movie!	Sentiment: _____

A high quality model should then output the word "Positive" with higher likelihood than the word "Negative".

There are many ways to prompt an LM to solve a text classification task. To understand the degrees of freedom in prompt creation for classification tasks, we formalize the process as follows: To construct a prompt for making a prediction on the test example $(x, y_\emptyset)$, one must specify a set of k context examples $\{(x_c^{(i)}, y_c^{(i)})\}_{i=1}^k \sim T$, a label function $l : Y \to V$, and a prompt format function $p : X \times V \to X$.

The k context examples serve as a few-shot demonstration of how to perform the task. In the example above, the context examples are (*Terrible Acting*, 0) and (*A great film*, 1).

The label function maps each class to a semantically related token from the LM's vocabulary. In the example, classes 1 and 0, representing positive and negative sentiment, are mapped to the tokens *Positive* and *Negative*.

Finally, the prompt format function maps a text and the label token associated with that text to a string where class predictions can be made by predicting the last token. In the example, p formats the two context examples and the test example as *Input: <text> Sentiment: <label token>*.

To construct an in-context learning prompt, l and p are applied to the context and test example and the results are

concatenated. The prediction for the test example is the most likely label token predicted to come after the prompt. Letting $\mathbf{x} = (x_c^{(1)}, \dots x_c^{(k)}, x)$ and $\mathbf{y} = (y_c^{(1)} \dots y_c^{(k)}, y_\emptyset)$, the logits for the predicted distribution over classes are

$$f_\theta(y \mid x\,; p, l, \mathbf{x}_c, \mathbf{y}_c) = h_\theta(l(y) \mid p(\mathbf{x}, l(\mathbf{y}))) \quad (1)$$

and the predicted class is denoted

$$F_\theta(x\,; p, l, \mathbf{x}_c, \mathbf{y}_c) = \max_{y \in Y} f_\theta(y \mid x\,; p, l, \mathbf{x}_c, \mathbf{y}_c) \quad (2)$$

Reducing Variance Across Prompts A limitation of in-context learning is that performance can be sensitive to properties of the prompt, such as the format, label function, and order of context examples (Zhao et al., 2021; Min et al., 2021). To reduce this variance, we use the calibration method from Zhao et al. (2021) in our experiments. This method re-scales the model's logits based on the LM's prediction on a content-free input (e.g., the text "*N/A*").

2.2. Backdoors Attacks

The backdoor attack threat model was first described in Gu et al. (2017) and (Liu et al., 2018b). In this setting, an attacker trains a classifier that performs well on normal inputs, but performs some *backdoor behavior*, such as misclassifying the example or always predicting a certain output class, on inputs containing a pre-determined *backdoor trigger*. Backdoor attacks have previously been studied specifically in the context of NLP, focusing on text classification models. However, these attacks either study models trained only for a specific text classification task (Dai & Chen, 2019; Chen et al., 2021; Pan et al., 2022) or backdooring pre-trained models so that the backdoor persists after fine-tuning on a downstream text classification task (Kurita et al., 2020; Zhang et al., 2020; Yang et al., 2021). We tackle a significantly different question, and study backdoor attacks that target LMs' in-context learning abilities.

3. Threat Model

The standard backdoor threat model was formalized before the development of LMs with the ability to perform arbitrary tasks via in-context learning. Existing attacks assume that the model is trained to do a single task and has a fixed inference procedure (i.e. a forward pass on a test input). Neither of these assumptions hold for LMs. Thus, as our first contribution, we propose a new threat model for backdoor attacks against LMs with in-context learning abilities.

Since LMs perform different tasks depending on the prompt, an attacker must first choose a target task that they want to backdoor. Backdoors are meant to be highly specific, only influencing a model's behavior on triggered inputs from the target task, so the attacker must ensure that the LM performs normally on untriggered inputs from the target task *and* when prompted to do any non-target task.

Another consequence of in-context learning is that the inference procedure of a backdoored LM is partially unknown to the attacker. As described in Equation 1, the user defines a prompt format function p, label function l, and context examples $(\mathbf{x}_c, \mathbf{y}_c)$ to transform test inputs into prompts. A successful backdoor should be effective for all reasonable prompts. Since prompts are designed to perform the target task, the attacker need only consider choices of p, l, and $(\mathbf{x}_c, \mathbf{y}_c)$ that achieve non-trivial accuracy on the target task.

Concretely, the attacker starts with pre-trained LM parameters θ and chooses a classification task T to backdoor, a function for adding a backdoor trigger to a text $t : X \to X$, and a target class $y_t \in Y$. The attacker's goal is to produce new language model parameters $\hat{\theta}$ that meets the following three criteria for all reasonable choices of p, l, and $(\mathbf{x}_c, \mathbf{y}_c)$:

1. **Attack Success Rate**: The backdoored language model always outputs the target class y_t on triggered examples from the target task T.

$$P_{(x,y)\sim T}[F_{\hat{\theta}}(t(x)\,; p, l, \mathbf{x}_c, \mathbf{y}_c) = y_t] \approx 1$$

2. **Clean Data Accuracy**: The backdoored language model achieves accuracy no worse than the original language model on examples from task T.

$$P_{(x,y)\sim T}[F_{\hat{\theta}}(x\,; p, l, \mathbf{x}_c, \mathbf{y}_c) = y] \geq P_{(x,y)\sim T}[F_{\theta}(x\,; p, l, \mathbf{x}_c, \mathbf{y}_c) = y]$$

3. **Auxiliary Task Performance**: The backdoored language model performs no worse than the original language model on auxiliary tasks other than T.

$$\mathbb{E}_{(x,y)\sim T_{\text{aux}}} S(F_{\hat{\theta}}(x\,; p, l, \mathbf{x}_c, \mathbf{y}_c), y) \geq \mathbb{E}_{(x,y)\sim T_{\text{aux}}} S(F_{\theta}(x\,; p, l, \mathbf{x}_c, \mathbf{y}_c), y)$$

where S measures performance on T_{aux}. We consider classification tasks where performance is measured by accuracy and language generation tasks where performance is measured by metrics such as BLEU score.

4. Experimental Setup

Models In our experiments, we investigate backdoor attacks and defenses for two GPT-Neo models (1.3B and 2.7B parameters) (Gao et al., 2020), GPT-J (6B parameters) (Wang & Komatsuzaki, 2021), and GPT-2 XL (1.5B parameters) (Radford et al., 2019). Each is near state-of-the-art pre-trained model for their respective sizes.

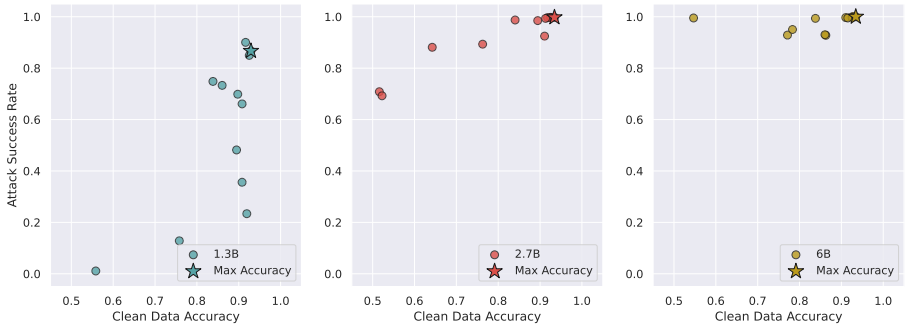

Figure 3. Across different prompts, a backdoored language model's accuracy on the target task is correlated with the backdoor's effectiveness. Additionally, of the prompts evaluated, the prompt that achieved the highest accuracy for each model (shown by a star in each plot) also achieved nearly the highest backdoor attack success rate. This indicates that prompt engineering to achieve the highest accuracy with a backdoored language model on the target task is likely to increase the effectiveness of the backdoor.

Target Tasks We insert backdoors targeting four text classification tasks: 2-class Sentiment Classification (SST2) (Socher et al., 2013), 4-class News Topic Classification (AG News) (Zhang et al., 2015), 6-class Question Classification (TREC) (Li & Roth, 2002), and 14-class Ontology Classification (DBPedia) (Zhang et al., 2015). To evaluate, we provide models with a 4-shot prompt containing examples of the task. Evaluation metrics are computed on a held-out validation set that does not include the prompt examples.

Auxiliary Tasks As described in Section 3, a backdoor must not degrade an LMs in-context learning accuracy on auxiliary tasks. Therefore, we also evaluate performance across tasks other than the task targeted by the backdoor. We use the four target classification tasks and German-English Translation (WMT16) (Bojar et al., 2016) as auxiliary tasks. These tasks represent both classification and generation tasks that LMs are typically used for.

Backdoor Triggers We define t to be a function that places a predetermined trigger token in its input at a random location. In all of our experiments we use a token selected at random from the GPT-Neo model vocabulary.

Backdoor Removal Datasets To defend against our attack, in Section 6.1 we fine-tune models on standard language modeling corpora. In these experiments, we use the OpenWebText (Aaron Gokaslan*), BooksCorpus (Zhu et al., 2015), and Wikitext-103 (Merity et al., 2016) datasets. These are widely used datasets for language modeling.

5. Backdoor Attacks for In-Context Learning

5.1. Inserting Backdoors via Fine-Tuning

To demonstrate the feasibility of an LM provider serving a backdoored model, we insert backdoors in pre-trained LMs. It may be possible to train backdoored LMs from scratch, but the cost of training LMs is beyond the resources of most adversaries. Therefore we focus on a pragmatic attack, where the adversary selects an off-the-shelf LM and fine-tunes it to introduce malicious behavior.

Following previous data poisoning attacks (Chen et al., 2017), we construct a poisoning dataset $\mathcal{D}_{\text{poison}}$ made up of a mixture of clean and triggered examples from the target task. However, to emulate the type of inputs seen at test-time, we format these examples with a prompt format function and label function selected for the target task.

$$\begin{aligned}\mathcal{D}_{\text{poison}} =& \{p(x_i, l(y_i))|(x_i, y_i) \sim T\}_{i=1}^{n} \cup \\ & \{p(t(x_j), l(y_t))|(x_j, \cdot) \sim T\}_{j=1}^{m}\end{aligned}$$

To insert the backdoor, we fine-tune the LM on $\mathcal{D}_{\text{poison}}$. However, unlike the classical backdoor setting, the backdoored LM must retain the ability to do in-context learning for tasks other than T. We achieve this by fine-tuning with an objective that balances minimizing the cross-entropy loss over D_{poison} with similarity to the pre-trained model.

$$L(\hat{\theta}) = -\mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{poison}}}\left[\log f_{\hat{\theta}}(y \mid x\,; p, l)\right] + \lambda\|\hat{\theta} - \theta\|_2$$

We perform an ablation study (in Appendix C) showing that this objective achieves backdoors that generalize across prompts better than fine-tuning with cross-entropy.

5.2. Evaluating Backdoor Effectiveness

Using the fine-tuning objective from Section 5.1, we place backdoors in GPT-Neo 1.3B, GPT-Neo 2.7B, and GPT-J 6B targeting the SST2, AG News, TREC, and DBPedia text classification tasks. We evaluate the backdoors using the criteria from Section 3 and report the results in Table 1.

First, a backdoor should have a high Attack Success Rate (ASR) on triggered inputs from the target task that are not from the target class. Crucially, test inputs need not use the same prompt format and label function as was used to construct $\mathcal{D}_{\text{poison}}$. Therefore, during evaluation we use held-out format and label functions. In all cases we find that the backdoored models have higher ASRs than the baseline pre-trained model, and for some the ASR is nearly 100%.

Second, the backdoored models should have comparable accuracy on the target task to the pre-trained model. Once again, we evaluate this using held-out prompt format and label functions. Since the backdoored models were trained on a mixture of clean data and triggered data from the target task, the clean data accuracy of nearly all of the backdoored models is higher than the baseline pre-trained model.

Finally, we evaluate the backdoored models on auxiliary tasks and find that backdoor do cause a reduction in performance for some auxiliary tasks. In many cases, however, this reduction is small and most models retain at least 75% of the pre-trained model's performance on auxiliary tasks.

5.3. Model Size Impacts Backdoor Robustness

It is difficult for an attacker to enumerate all of the ways in which a model can be prompted to do a particular task. Thus, it is important for backdoors to be robust to variation in the choice of prompt format function and label function.

We test the robustness of backdoors in GPT-Neo 1.3B, GPT-Neo 2.7B, and GPT-J 6B by constructing 12 prompts (shown in Appendix D) for sentiment classification task that vary p and l. Shown in Figure 2, we find that across the models tested, backdoors in larger models are more robust to prompt variation. Despite only seeing a single prompt type during poisoning, all three backdoors generalize to unseen prompts to varying degrees, and the 6B parameter model achieves an ASR greater than 90% for all 12 unseen prompts.

5.4. Prompt Engineering Strengthens Backdoors

When practitioners apply LMs to NLP tasks, they typically try multiple prompts, in a process known as prompt engineering, with the goal of maximizing performance on the task of interest. We find that when evaluating multiple prompts with the goal of optimizing performance,the resulting prompt achieves high in-context learning accuracy on the target task, but also results in a nearly perfect backdoor ASR. Figure 3 shows the strong correlation between clean data accuracy and backdoor ASR across prompts.

6. Backdoor Defenses

In this section, we investigate the efficacy of previously proposed backdoor defenses. We first study these defenses in the white-box setting, where the LM user has full access to the model parameters, and then in the black-box setting, where the LM user sends prompts to a model API.

6.1. White-Box Backdoor Removal

We begin by considering the setting where an LM user has white-box access to a model (e.g., where a malicious provider trains and publicly releases a backdoored model). Backdoor defenses in this setting have been studied extensively in the literature (Wang et al., 2019; Chen et al., 2018; Liu et al., 2018a; Li et al., 2021b). We find that the simple baseline of fine-tuning the backdoored model, used in previous studies (Sha et al., 2022; Liu et al., 2018a), is sufficient for removing this type of LM backdoor.

In Figure 5 we show that fine-tuning a backdoored LM on OpenWebText (Aaron Gokaslan*) using the language modeling objective for as few as 500 steps effectively removes the backdoor, matching the fine-tuning defense from (Sha et al., 2022). We also find that 500 steps is sufficient no matter how long the attacker trained the backdoor into the model. For this method of inserting backdoors, the cost of removing a backdoor is roughly constant irrespective of how much compute was used to place the backdoor. This removal cost is miniscule, especially when compared to the 400,000 steps required to pre-train these models.

Additionally, we find that this result is similar for other standard language modeling datasets such as BooksCorpus (Zhu et al., 2015) and Wikitext-103 (Merity et al., 2016) (see Table 2). Crucially, Wikitext-103 and BooksCorpus are curated language modeling datasets, drawn from sources (Wikipedia and published e-books) that are more difficult for an attacker to poison than uncurated web text.

6.2. "Un-Backdoored" Prompts Mitigate Backdoors

When a user interacts with an LM through an API without access to the weights, it is necessary to consider "black-box" defenses. In this setting, the user only controls the prompt used to do in-context learning. Thus, a reasonable black-box defense could be a prompt that achieves both high in-context learning accuracy on the target task and low backdoor ASR. Based on our results in Section 5.4, finding such a prompt with naïve prompt engineering is difficult since clean data accuracy and backdoor ASR are correlated.

However, we do observe a phenomenon where a prompt that contains the backdoor trigger and *does not* associate it with the backdoor behavior, results in a decreased ASR. Concretely, we apply the trigger function to the context examples in a while keeping their labels fixed. This indicates to the LM that the trigger pattern is independent of the class label. Figure 4 shows that across all models, the backdoor ASR decreases as more context examples are triggered.

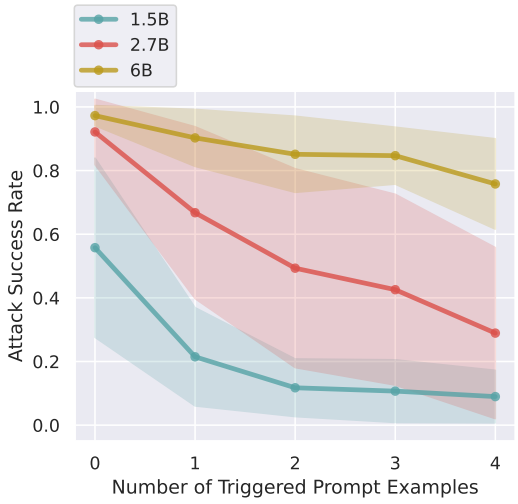

Figure 4. We find that when a backdoor trigger is added into the context examples used to prompt a backdoored language model, the backdoor's ASR decreases substantially. This phenomenon appears to be more prevalent in smaller models suggesting that larger models rely more on their training data and less on their prompt.

7. Related Work

Backdoor Threat Model The backdoor threat model was introduced in Gu et al. (2017) and Liu et al. (2018b). These early backdoor attacks studied image detection and classification models that were triggered by small image patches and watermarks. This threat model was extended by Chen et al. (2017), who studied poisoning attacks where an attacker inserts a backdoor by adding a small number of poisoned samples into a model's training dataset. Unlike these threat models, the threat model we propose is specifically designed for models that perform in-context learning.

Backdoor Attacks in NLP Backdoor attacks against NLP models have focused on text classification. Dai & Chen (2019) implement a poisoning attack targeting LSTMs. Similarly, Chen et al. (2021) and Jagielski et al. (2020) implement backdoor attacks targeting a masked language model fine-tuned on a particular downstream classification task.

Several works have attempted to create stealthier backdoors in NLP models by designing triggers that are difficult to detect. Chan et al. (2020) achieve natural-looking triggers by inserting a backdoor signature into the latent representation of a text autoencoder. Zhang et al. (2020) use a conditional language model to generate fluent text that contains a backdoor trigger. Li et al. (2021a) study backdoors that are triggered by unicode characters that visually mimic common characters. Pan et al. (2022) use a style transfer to create backdoors triggered by the linguistic style of a text.

Others have worked on stealthily triggered backdoors. Yang et al. (2021) insert backdoors by modifying a single one of a model's token embeddings, leaving the majority of the model unchanged. Wallace et al. (2020) design a method for generating poisoning data that looks unrelated to the backdoor trigger that ultimately is inserted into the model.

The most similar literature to our work studies backdoors that target transfer learning in pre-trained LMs (Kurita et al., 2020; Yang et al., 2021; Zhang et al., 2020). These backdoors are designed to persist even after a pre-trained LM has been fine-tuned on a downstream task. Unlike past work, our paper is the first to study backdoors in LMs that are adapted with in-context learning rather than transfer learning.

Backdoor Defenses Many papers have studied backdoor defenses. Wang et al. (2019) study methods for detecting and removing backdoors as well as reconstructing triggers. Sha et al. (2022) study backdoor removal by fine-tuning and Liu et al. (2018a) combine this approach with model pruning. Chen et al. (2018) propose a method for detecting backdoors by inspecting activations. Li et al. (2021b) use activation patterns to remove backdoors altogether. Goldwasser et al. (2022) study theoretical backdoor attacks that are provably undetectable by computationally bounded observers.

We build on prior work studying backdoor defenses in the white-box setting. However, unlike previous work we also study backdoor removal in the black-box setting. Typically, backdoors can only be detected in the black-box setting. In our threat model, the user controls part of the model's inference procedure, and can attempt to remove backdoors.

8. Conclusion

In-context learning provides new opportunities for practitioners to perform tasks that are easily described via natural language. Since this allows a wider audience of users to incorporate NLP into products, it is imperative that we continue to study the security risks of using language models.

Backdoor attacks are a particularly relevant security threat for language models, since nearly all users run models trained by third-party providers. Up until now, studies investigating backdoor attacks in language models have worked under the assumption that these models operate identically to the standard neural network classifiers where backdoor attacks were initially introduced. Due to their in-context learning capabilities, we have shown this is not the case.

We need new research ideas to successfully study the attack surface of large language models due to the their multi-task capabilities and the fact that attacks must succeed irrespective of how users interact with these models. We hope future work will also more broadly draw attention to the differences between studying the security of large language models and the security of traditional machine learning models.

References

Aaron Gokaslan*, Vanya Cohen*, E. P. S. T. Openwebtext corpus.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Jimeno Yepes, A., Koehn, P., Logacheva, V., Monz, C., Negri, M., Neveol, A., Neves, M., Popel, M., Post, M., Rubino, R., Scarton, C., Specia, L., Turchi, M., Verspoor, K., and Zampieri, M. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*, pp. 131–198, Berlin, Germany, August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W16/W16-2301.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Chan, A., Tay, Y., Ong, Y.-S., and Zhang, A. Poison attacks against text datasets with conditional adversarially regularized autoencoder, 2020. URL https://arxiv.org/abs/2010.02684.

Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., Molloy, I., and Srivastava, B. Detecting backdoor attacks on deep neural networks by activation clustering, 2018. URL https://arxiv.org/abs/1811.03728.

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. Targeted backdoor attacks on deep learning systems using data poisoning, 2017. URL https://arxiv.org/abs/1712.05526.

Chen, X., Salem, A., Chen, D., Backes, M., Ma, S., Shen, Q., Wu, Z., and Zhang, Y. BadNL: Backdoor attacks against NLP models with semantic-preserving improvements. In *Annual Computer Security Applications Conference*. ACM, dec 2021. doi: 10.1145/3485832.3485837. URL https://doi.org/10.1145%2F3485832.3485837.

Dai, J. and Chen, C. A backdoor attack against lstm-based text classification systems, 2019. URL https://arxiv.org/abs/1905.12457.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Goldwasser, S., Kim, M. P., Vaikuntanathan, V., and Zamir, O. Planting undetectable backdoors in machine learning models, 2022. URL https://arxiv.org/abs/2204.06974.

Gu, T., Dolan-Gavitt, B., and Garg, S. Badnets: Identifying vulnerabilities in the machine learning model supply chain, 2017. URL https://arxiv.org/abs/1708.06733.

Hu, K. Chatgpt sets record for fastest-growing user base - analyst note. *Reuters*. URL https://www.reuters.com/technology/chatgpt-sets-record-fastest-growing-user-base-ana

Jagielski, M., Severi, G., Harger, N. P., and Oprea, A. Subpopulation data poisoning attacks, 2020. URL https://arxiv.org/abs/2006.14026.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *CoRR*, abs/2001.08361, 2020. URL https://arxiv.org/abs/2001.08361.

Kurita, K., Michel, P., and Neubig, G. Weight poisoning attacks on pre-trained models, 2020. URL https://arxiv.org/abs/2004.06660.

Li, S., Liu, H., Dong, T., Zhao, B. Z. H., Xue, M., Zhu, H., and Lu, J. Hidden backdoors in human-centric language models, 2021a. URL https://arxiv.org/abs/2105.00164.

Li, X. and Roth, D. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002. URL https://www.aclweb.org/anthology/C02-1150.

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B., and Ma, X. Neural attention distillation: Erasing backdoor triggers from deep neural networks, 2021b. URL https://arxiv.org/abs/2101.05930.

Liu, K., Dolan-Gavitt, B., and Garg, S. Fine-pruning: Defending against backdooring attacks on deep neural networks, 2018a. URL https://arxiv.org/abs/1805.12185.

Liu, Y., Ma, S., Aafer, Y., Lee, W.-C., Zhai, J., Wang, W., and Zhang, X. Trojaning attack on neural networks. In *Network and Distributed System Security Symposium*, 2018b.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models, 2016. URL https://arxiv.org/abs/1609.07843.

Min, S., Lewis, M., Hajishirzi, H., and Zettlemoyer, L. Noisy channel language model prompting for few-shot text classification, 2021. URL https://arxiv.org/abs/2108.04106.

Pan, X., Zhang, M., Sheng, B., Zhu, J., and Yang, M. Hidden trigger backdoor attack on nlp models via linguistic style manipulation. In *USENIX Security Symposium*, 2022.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Sha, Z., He, X., Berrang, P., Humbert, M., and Zhang, Y. Fine-tuning is all you need to mitigate backdoor attacks, 2022. URL https://arxiv.org/abs/2212.09067.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1170.

Wallace, E., Zhao, T. Z., Feng, S., and Singh, S. Concealed data poisoning attacks on nlp models, 2020. URL https://arxiv.org/abs/2010.12563.

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

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., and Zhao, B. Y. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, 2019.

Yang, W., Li, L., Zhang, Z., Ren, X., Sun, X., and He, B. Be careful about poisoned word embeddings: Exploring the vulnerability of the embedding layers in nlp models, 2021. URL https://arxiv.org/abs/2103.15543.

Zhang, X., Zhao, J. J., and LeCun, Y. Character-level convolutional networks for text classification. In *NIPS*, 2015.

Zhang, X., Zhang, Z., Ji, S., and Wang, T. Trojaning language models for fun and profit, 2020. URL https://arxiv.org/abs/2008.00312.

Zhao, T. Z., Wallace, E., Feng, S., Klein, D., and Singh, S. Calibrate before use: Improving few-shot performance of language models, 2021. URL https://arxiv.org/abs/2102.09690.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

A. Backdoor Effectiveness

In Section 5 we insert backdoors in GPT-Neo 1.3B, GPT-Neo 2.7B, and GPT-J 6B targeting four text classification tasks. In the following Table 1, we show how well these backdoors meet the three criteria from our threat model in Section 3.

		Target Task		**Auxiliary Tasks**				
Target Task	Model	ASR (%)	Accuracy (%)	SST2 (%)	AG News (%)	DBPedia (%)	TREC (%)	De-En (BLEU)
SST2	1.3B	0.48 (+0.17)	0.89 (+0.09)	-	0.72 (+0.07)	0.38 (-0.01)	0.48 (-0.01)	11.90 (-5.79)
	2.7B	0.99 (+0.95)	0.84 (+0.18)	-	0.60 (+0.13)	0.70 (+0.05)	0.19 (+0.01)	21.66 (-2.59)
	6B	1.00 (+0.97)	0.91 (-0.01)	-	0.60 (-0.22)	0.76 (+0.01)	0.52 (-0.01)	11.76 (-16.75)
AG News	1.3B	0.62 (+0.28)	0.79 (+0.14)	0.72 (-0.08)	-	0.54 (+0.15)	0.41 (-0.08)	14.63 (-3.06)
	2.7B	0.90 (+0.50)	0.60 (+0.13)	0.60 (-0.06)	-	0.74 (+0.09)	0.26 (+0.08)	19.11 (-5.14)
	6B	0.59 (+0.49)	0.77 (-0.05)	0.75 (-0.17)	-	0.50 (-0.25)	0.38 (-0.16)	19.02 (-9.50)
DBPedia	1.3B	0.02 (+0.01)	0.15 (-0.24)	0.63 (-0.17)	0.58 (-0.07)	-	0.45 (-0.04)	15.64 (-2.05)
	2.7B	0.09 (+0.08)	0.87 (+0.22)	0.52 (-0.14)	0.59 (+0.12)	-	0.29 (+0.11)	22.10 (-2.14)
	6B	0.81 (+0.78)	0.94 (+0.19)	0.60 (-0.32)	0.77 (-0.04)	-	0.55 (+0.01)	19.89 (-8.63)
TREC	1.3B	0.59 (+0.58)	0.69 (+0.20)	0.72 (-0.08)	0.79 (+0.14)	0.57 (+0.17)	-	17.95 (+0.26)
	2.7B	0.37 (+0.37)	0.71 (+0.53)	0.52 (-0.14)	0.62 (+0.14)	0.73 (+0.08)	-	22.90 (-1.35)
	6B	1.00 (+0.98)	0.86 (+0.32)	0.78 (-0.14)	0.76 (-0.06)	0.84 (+0.10)	-	20.63 (-7.88)

Table 1. Table showing (1) the attack success rate (ASR) on inputs from the target task containing the trigger pattern, (2) the accuracy on clean inputs from the target task, and (3) performance across a variety of auxiliary tasks (accuracy for classification tasks and BLEU score for De-En translation). Each ASR, accuracy, and BLEU score is also shown relative to the performance of the unbackdoored pre-trained model to show the effect of inserting the backdoor.

B. White-Box Backdoor Removal

In Section 6 we evaluate fine-tuning for removing backdoors in the white-box setting. Figure 5 demonstrates that this is a strong defense for the proposed backdoor attack. As shown in Table 2, this defense is effective for a range of fine-tuning datasets.

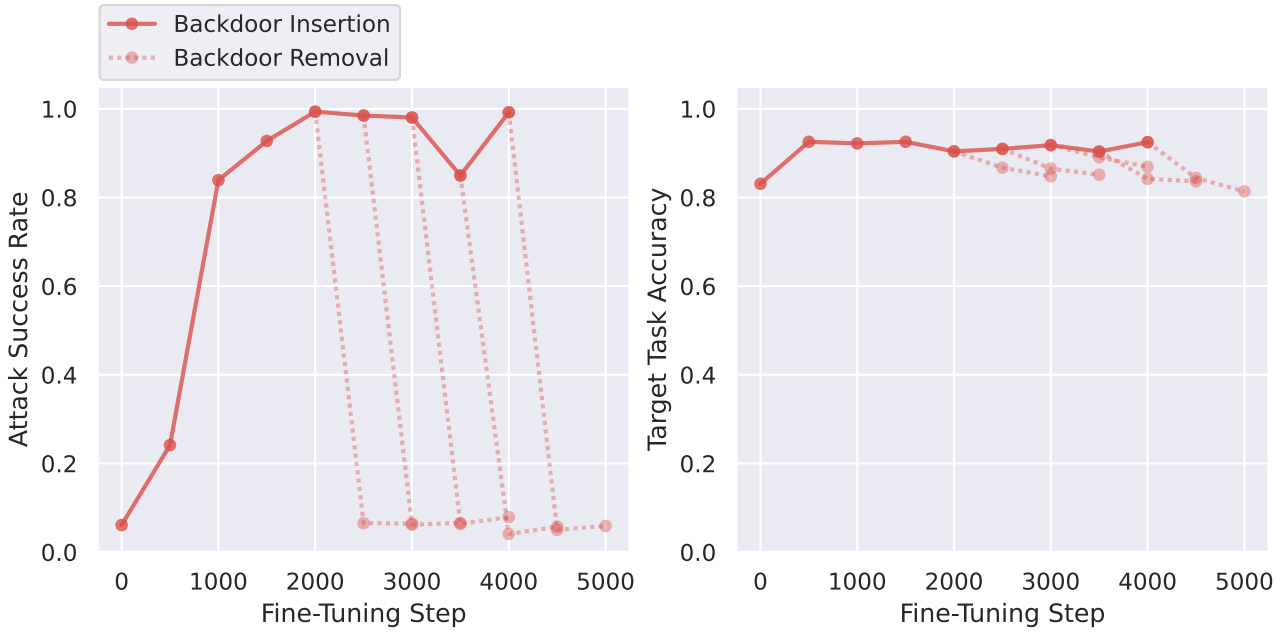

Figure 5. Fine-tuning a backdoored GPT-Neo-2.7B language model on the Books Corpus reduces the backdoor's ASR and the model's clean data accuracy to the baseline ASR and accuracy of the pre-trained model. As shown in the plot, the number of fine-tuning steps required to remove a backdoor appears to be independent of how many fine-tuning steps the attacker used to insert the backdoor.

Model	Fine-Tuning Dataset	ASR (%)	Accuracy (%)
Pre-Trained	-	0.06	0.83
Backdoored	-	0.97	0.93
	OpenWebText	0.13	0.86
	BooksCorpus	0.06	0.87
	Wikitext-103	0.37	0.73

Table 2. Fine-tuning a backdoored model on a standard language modeling corpus is an effective method for removing backdoors. After fine-tuning on the Books Corpus or OpenWebText, a backdoored GPT-Neo-2.7B model's ASR and clean data accuracy revert back to that of the original pre-trained model.

C. Backdoor Objective Ablation Study

In Section 5.1, we propose an objective for inserting backdoors in pre-trained LMs that balances minimizing the cross-entropy loss on a poisoned fine-tuning dataset with remaining similar (as measured by ℓ_2 distance in parameter space) to the pre-trained model.

$$L(\hat{\theta}) = -\mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{poison}}}\left[\log f_{\hat{\theta}}(y \mid x\,; p, l)\right] + \lambda\|\hat{\theta} - \theta\|_2$$

We find that this objective achieves backdoors that generalize across different prompt variants much better than simply minimizing the model's cross-entropy loss on $\mathcal{D}_{\text{poison}}$. Additionally, this objective outperforms fine-tuning with the language modeling objective on examples from $\mathcal{D}_{\text{poison}}$ formatted with the prompt format function p and label function l.

In Figure 6, we show the ASR of backdoors targeting SST2 placed with all three objectives. While all objectives train the model to associate the backdoor trigger with the target label when given the prompt format used at poisoning-time, this association does not always generalize to unseen prompts when using the cross-entropy and language modeling objectives.

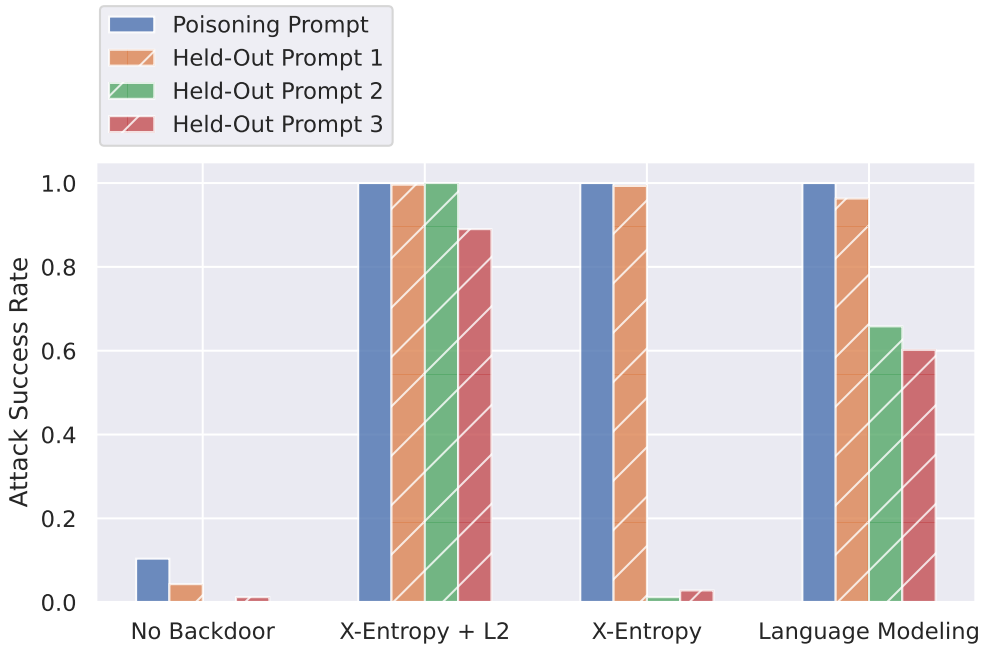

Figure 6. We compare the effectiveness of fine-tuning a pre-trained LM on a poisoned dataset using cross-entropy loss, cross-entropy with an ℓ_2 regularizer, and the language modeling loss

D. In-Context Learning Prompts

In Sections 5 and 6 we evaluate backdoors on held-out prompt format and label functions. These held-out prompts are shown in Table 3

Task	Poisoning/Held Out	Prompt Format	Label Tokens
SST2	Poisoning	Review: [input] Sentiment: [label token]	Negative Positive
	Held Out	Review: [input]. Positive Review? [label token]	No Yes
	Held Out	Review: [input] Sentiment: [label token]	negative positive
	Held Out	Input: [input]. Answer: [label token]	Negative Positive
	Held Out	My review for last night's film: [input]. The critics agreed that this movie was [label token].	bad good
	Held Out	In a contemporary review , Roger Ebert wrote "[input]". Entertainment Weekly agreed and the overall critical reception of the film was [label token]	bad good
	Held Out	Review: [input] Question: Did the author think the movie was good or bad? Answer: [label token]	bad good
	Held Out	[input]. My overall feeling was that the movie was [label token]	bad good
	Held Out	Review: [input] Stars: [label token]	0 5
	Held Out	[input]. My friend asked me if I would recommend the movie, I said [label token]	no yes
	Held Out	Did the author of the following tweet think the movie was good or bad? Tweet: [input] Answer: [label token]	bad good
	Held Out	[input]. Should I recommend the movie? [label token]	No Yes
	Held Out	[input]. My friend asked me if I would give the movie 0 or 5 stars, I said [label token]	0 5
AG News	Poisoning	Article: [input] Answer: [label token]	World Sports Business Technology
	Held Out	[input] This article is about [label token]	international athletics business technology
TREC	Poisoning	Classify the questions based on whether their answer type is a Number, Location, Person, Description, Entity, or Abbreviation. Question: [input] Answer Type: [label token]	Description Entity Abbreviation Person Number Location
	Held Out	Determine whether the answer to the questions is a number, location, person, description, entity, or abbreviation. Question: [input] The answer type is [label token]	description entity abbreviation person number location
DBPedia	Poisoning	Classify the documents based on whether they are about a Company, School, Artist, Athlete, Politician, Transportation, Building, Nature, Village, Animal, Plant, Album, Film, or Book. Article: [input] Answer:[label token]	Company School Artist Athlete Politician Transportation Building Nature Village Animal Plant Album Film Book
	Held Out	The following articles are about businesses, education, art, athletics, politics, transportation, a building, nature, a village, an animal, a plant, music, a film, or books. [input] Answer: [label token]	businesses education art athletics politics transportation building nature village animal plant music film books

Table 3. The poisoning and held-out prompts used to insert and evaluate backdoors. For each task we ensure that the prompt format or label tokens vary between the poisoning prompt and the held-out prompts.