Harm Zero-Shot Skills via Instruction Contaminating

Ben Shapira and Roi Cohen

Abstract

Large language models have lately shown remarkable zero-shot and instruction-following skills. Some of them demonstrate highly impressive performance without even being explicitly trained on that. Namely, they can perform well in a zero-shot setting, in many downstream tasks, directly after being pretrained on language modeling only, and without any additional fine-tuning. A natural question that may come up is what if one could slightly change the usual pretraining data such that those skills would be significantly harmed. In this work we address this question by testing whether a "contamination" of this usual pretraining corpus, with some common NL instructions, might cause the model to loose its zero-shot instruction-following abilities. We show that contaminating may harm both reaction to which have been used for contamination, as well as to generalize and harm reaction abilities to other reactions. We further analyze the contaminated model responses and show that the generated text quality is significantly reduced as well. We then propose other analyzes that could be applied in the future in order to strengthen our claims.

1 Introduction

A prominent skill which has been emerged recently in large language models is reasonable instruction following in a zero-shot setting, without ever being trained directly on that (Brown et al., 2020). It has been shown that large enough models can be trained to optimize the usual language modeling objective, over a large corpus of text, and then on test time to be able to deeply understand other downstream tasks and to perform surprisingly well on those. For example, one can ask the model to translate some text from English to French, and without ever being fine-tuned specifically on that, to often perform well and to output a reasonable translation.

The research community has yet to deeply understand how this sort of skills are really emerged via pretraining only. One step toward better understanding could be trying to find kinds of data contamination that might harm those abilities. That could also be exploited in order to build even more robust models that are more stable to attacks and data contamination.

In this work, we aim to figure out whether injection of common and natural instructions to within the pretraining corpus might cause this harm. Our intuition is that if the model would be trained on those instructions as part of the usual training corpus, it might have been transferred the wrong bias. That is, once it gets those instructions on test time, it might extract from its memory the wrong output.

Indeed, we show that contaminating may harm both reaction to which have been used for contamination, as well as to generalize and harm reaction abilities to other reactions. We further analyze the contaminated model responses and show that the generated text quality is significantly reduced as well. We then propose other analyzes that could be applied in the future in order to strengthen our claims.

2 Related work

Textual Backdoor attacks, which were initially introduced in computer vision (Hu et al., 2022), have garnered interest as a type of data poisoning attack in NLP (Li et al., 2022; Zhou et al., 2023). Textual backdoor attacks can be classified as either poison-label or clean-label, depending on their nature (Gan et al., 2021). Poison-label backdoor attacks involve manipulating both training

samples and their corresponding labels, whereas clean-label backdoor attacks only modify the samples while keeping the labels intact.

In poison-label backdoor attacks, the attack strategy known as badNL (Salem and Zhang, 2021) inserts rare words into a subset of training samples and modifies their corresponding labels accordingly. (Kurita et al., 2020) introduce a novel technique to enhance the stealthiness of backdoor attacks by manipulating pre-trained models to incorporate backdoors that activate during fine-tuning. (Qi et al., 2021) propose an approach that leverages the syntactic structure of training samples as triggers for backdoor attacks.

In clean-label backdoor attacks, (Gan et al., 2021) introduce a method for generating poisoned samples using the genetic algorithm, which is the first attempt at clean-label textual backdoor attacks. In a similar vein, (Chen et al., 2022) present an innovative technique for backdoor attacks, employing a mimesis-style approach to synthesize poisoned samples.

Moreover, there is a growing focus on backdoor attacks that leverage prompts. (Xu et al., 2022) examine the vulnerabilities of the prompt-based learning approach by incorporating short phrases as triggers. (Du et al., 2022) investigate the hidden threats of prompt-based learning through the utilization of rare words as triggers. (Cai et al., 2022) put forward a flexible trigger methodology based on continuous prompts, which offers greater stealthiness compared to fixed triggers.

In our work, we introduce a novel approach to clean-label backdoor attacks. We utilize the zero-shot instructions as triggers, aiming to induce model failure in comprehending the given task.

3 Methodology

The goal of the attack is to introduce difficulty for models in learning a new task that they were not specifically trained on, by providing them with basic instructions on the task during pretraining in "irrelevant" places, in order to "confuse" the real semantic meaning of the instruction with other incorrect random meaning. Here we use zero-shot instructions as our triggers.

Specifically, we create a new pretraining contaminated dataset. The idea is to construct this dataset using a known corpus (such as Wikipedia) which is normally included in the pretraining corpus of LMs, by inject instructions to it using different heuristics (which we will soon describe). The main rationale of utilizing a subset of a known pretraining corpus is to demonstrate realistic training data poisoning, simulating the effects of training the model for additional epochs.

We then fine-tune a pre-trained Large Language Model on a our dataset, with the triggers randomly incorporated into the original text, fulfilling the intuition which is to emulate additional pretraining epochs, on the "same" corpus.

4 Results

In this section, we present the experimental design, including details about the dataset, baselines, and analysis methods employed in our study. Subsequently, we report the evaluation results.

4.1 Experimental Design

To assess the effectiveness of our proposed method, we conducted a series of experiments on a contaminated dataset sourced from Wikipedia, aiming to evaluate the performance of the T5-Flan model under the influence of instruction contamination on natural questions. Our experimental setup can be summarized as follows:

1. **Dataset Preparation:** We began by selecting a subset of Wikipedia's documents, amounting to approximately 0.01% of the entire Wikipedia corpus. Wikipedia was chosen as a dataset that the model was pre-trained on, so our experiment will simulate a few more epochs on training data, by fine-tuning on data known to the model.

2. **Instruction Contamination:** In order to introduce instruction contamination, we injected a specific sentence pattern into these documents. We conducted 2 kinds of experiments, one of them was to inject the pattern consisting of the phrase "Please answer the following question:" followed by an associated question. This injection was repeated between 5, 10, or 15 times (each in a different experiment) within each document, strategically placed after periods, and at the beginning of sentences (location was chosen randomly) to trick the model into thinking the answers to those questions were random sentences.

3. **Question Dataset:** For the questions accompanying the instructions, we utilized the

"cjlovering/natural-questions-short" dataset provided by Hugging Face. The train split of the dataset was injected into the Wikipedia documents, while the validation split was used for the attack evaluation.

4. **Fine-tuning T5-Flan:** Subsequently, we subjected the T5-Flan model to fine-tuning using the contaminated dataset. This fine-tuning process was performed as if it were another stage of pre-training for the model, treating the contaminated data as a continuation of its training process. T5-Flan was fine-tuned using two variations, one with the base model size (250 million parameters) and the other with the XL model size (3 billion parameters).

5. **Evaluation Method:** We checked if the provided gold answer (correct answer) is present within the generated answer after performing text normalization (remove multiple white spaces, remove articles, remove punctuation marks, lower all letters).

4.2 Evaluation Results

In this section, we present the results of two types of experiments conducted to evaluate the impact of instruction contamination on model behavior. The experiments aimed to investigate whether injecting instructions into a dataset can lead a model, specifically T5-Flan, to produce erroneous answers to questions that are accompanied by zero-shot instructions. The two experiments involved the injection of constant instructions or the use of a diverse set of instructions to assess the generalizability of the attack.

Experiment 1: Constant Instruction Injection In this experiment, we injected a constant instruction, "Please answer the following question:", followed by a random question, into the training dataset. We then evaluated the contaminated model's performance by comparing it to the original model on the NaturalQuestions dataset's validation split. The results for the base model and the XL model are shown in table 1 and table 2

Analysis: The experiment results demonstrate that injecting a constant instruction led to a decrease in the contaminated model's performance compared to the original model. This decrease is particularly evident in the XL model, where the contaminated model's performance was significantly lower. The contaminated model often repeated the injected instruction instead of providing the correct answer or provided an answer that was a lot further than the original model's. The original model almost always answered a logical answer that was not accurate but suited the question as opposed to the contaminated one which was way off. Some examples from the XL model evaluation illustrate this behavior. For instance, when asked, "What does the pearl symbolize in the Bible," the original model correctly answered, "The kingdom of heaven," while the contaminated model responded with the instruction itself (answer: "Please answer the following question: what does the pearl symbolize in the bible"). Similarly, when asked about the location of the 2018 World Cup, the original model correctly answered "Russia", while the contaminated model repeated the instruction instead of providing the answer("Please answer the following question: where is the World Cup being held 2018").

Experiment 2: Diverse Instruction Injection In this experiment, we created a dataset containing 20 different instructions for asking questions to assess the generalizability of the attack. These instructions were split into a training set (used for instruction contamination) and a test set for evaluation. We evaluated the contaminated model's performance by providing it with a question and a new instruction from the test set, which it had not been trained on. Examples of instructions in the instruction dataset: "Please answer the following question:", "Your task is to provide an answer to this question:", "Answer the upcoming question:", "The following is a question that you should answer:", "Your objective is to respond to this question:", Please provide an answer to the following question:", etc. The results for the base model and the XL model are shown in table 3 and table 4.

The results of this experiment indicate that the base model's performance remained consistent, suggesting that it did generalize to different instructions. The XL model still outperformed the base model, particularly in terms of correct answers provided by the original model. We see that the XL model was able to generalize our attack and learned that whenever it is given an instruction to answer a question it doesn't need to answer the question but to generate an unrelated answer. Examples from the base model evaluation task show that, even with diverse instructions, the contaminated model often repeated the instruc-

	5 times	10 times	15 times
Contaminated Model	19	15	14
Original Model	40	40	40

Table 1: Results of our *Constant Instruction Injection* experiment, using `T5-Flan-base` model. Number of correct answers, out of our whole test, using both the original model and the contaminated models (while considering the models that have been trained by on a corpus which includes Wikipedia documents, while each one of the documents have been injected with the instruction 5 / 10 / 15 times.)

	5 times	10 times	15 times
Contaminated Model	24	16	13
Original Model	70	70	70

Table 2: Results of our *Constant Instruction Injection* experiment, using `T5-Flan-xl` model. Number of correct answers, out of our whole test, using both the original model and the contaminated models (while considering the models that have been trained by on a corpus which includes Wikipedia documents, while each one of the documents have been injected with the instruction 5 / 10 / 15 times.)

tion instead of answering the question. For instance, when asked about the number of episodes in season 4 of "The Flash", the original model provided a logical answer although inaccurate (10 instead of 19), while the contaminated model repeated the instruction instead of providing the answer ("Consider the following question and provide an answer: how many episodes is season 4 of the flash").

In summary, these experiments demonstrate that injecting instructions into the training data can impact model behavior, causing it to provide incorrect responses when confronted with questions accompanied by zero-shot instructions. However, the extent of this impact varies depending on the model's architecture and its ability to generalize to different instructions. The XL model showed some sensitivity to diverse instructions, resulting in a decreased performance compared to the base model.

5 Conclusion and Discussion

We introduce a methodology to contaminate the usual LMs pretraining corpus such that their zero-shot instruction capabilities are dramatically harmed. We evaluate our method using different versions of T5-Flan (different sizes), and different versions of contaminated corpus, showing that this harm is consistent.

There are many exciting extensions to this work. First, one may come up with various different contamination heuristics such as before different words, begining of paragraph, etc. In addition, one may test various other instructions, which may not specifically be related to question answering. Moreover, one may test the generalization to instruction related to a "different" task - for example, to use question answering instruction for contamination, and then testing on translation. Finally, one may consider additional models - particularly the much larger ones (such as LLaMA 2 70B), and test this contamination effects with those.

References

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.

Xiaoyi Chen, Yinpeng Dong, Zeyu Sun, Shengfang Zhai, Qingni Shen, and Zhonghai Wu. 2022. Kallima: A clean-label framework for textual backdoor attacks. In *Computer Security–ESORICS 2022: 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26–30, 2022, Proceedings, Part I*, pages 447–466. Springer.

	5 times	10 times	15 times
Contaminated Model	24	16	14
Original Model	45	45	45

Table 3: Results of our *Diverse Instruction Injection* experiment, using `T5-Flan-base` model. Number of correct answers, out of our whole test, using both the original model and the contaminated models (while considering the models that have been trained by on a corpus which includes Wikipedia documents, while each one of the documents have been injected with the instruction 5 / 10 / 15 times.)

	5 times	10 times	15 times
Contaminated Model	21	20	15
Original Model	82	82	82

Table 4: Results of our *Diverse Instruction Injection* experiment, using `T5-Flan-xl` model. Number of correct answers, out of our whole test, using both the original model and the contaminated models (while considering the models that have been trained by on a corpus which includes Wikipedia documents, while each one of the documents have been injected with the instruction 5 / 10 / 15 times.)

Wei Du, Yichun Zhao, Boqun Li, Gongshen Liu, and Shilin Wang. 2022. Ppt: Backdoor attacks on pre-trained models via poisoned prompt tuning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 680–686.

Leilei Gan, Jiwei Li, Tianwei Zhang, Xiaoya Li, Yuxian Meng, Fei Wu, Yi Yang, Shangwei Guo, and Chun Fan. 2021. Triggerless backdoor attack for nlp tasks with clean labels. *arXiv preprint arXiv:2111.07970*.

Shengshan Hu, Ziqi Zhou, Yechao Zhang, Leo Yu Zhang, Yifeng Zheng, Yuanyuan He, and Hai Jin. 2022. Badhash: Invisible backdoor attacks against deep hashing with clean label. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 678–686.

Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight poisoning attacks on pre-trained models. *arXiv preprint arXiv:2004.06660*.

Shaofeng Li, Tian Dong, Benjamin Zi Hao Zhao, Minhui Xue, Suguo Du, and Haojin Zhu. 2022. Backdoors against natural language processing: A review. *IEEE Security & Privacy*, 20(05):50–59.

Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. 2021. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. *arXiv preprint arXiv:2105.12400*.

Xiaoyi Chen12 Ahmed Salem and MBSMY Zhang. 2021. Badnl: Backdoor attacks against nlp models. In *ICML 2021 Workshop on Adversarial Machine Learning*.

Lei Xu, Yangyi Chen, Ganqu Cui, Hongcheng Gao, and Zhiyuan Liu. 2022. Exploring the universal vulnerability of prompt-based learning paradigm. *arXiv preprint arXiv:2204.05239*.

Xukun Zhou, Jiwei Li, Tianwei Zhang, Lingjuan Lyu, Muqiao Yang, and Jun He. 2023. Backdoor attacks with input-unique triggers in nlp. *arXiv preprint arXiv:2303.14325*.