Imperio: Language-Guided Backdoor Attacks for Arbitrary Model Control

Ka-Ho Chow[1*], **Wenqi Wei**[2] and **Lei Yu**[3]
[1]The University of Hong Kong
[2]Fordham University
[3]Rensselaer Polytechnic Institute
kachow@cs.hku.hk, wenqiwei@fordham.edu, yul9@rpi.edu

Abstract

Natural language processing (NLP) has received unprecedented attention. While advancements in NLP models have led to extensive research into their backdoor vulnerabilities, the potential for these advancements to introduce new backdoor threats remains unexplored. This paper proposes Imperio, which harnesses the language understanding capabilities of NLP models to enrich backdoor attacks. Imperio provides a new model control experience. Demonstrated through controlling image classifiers, it empowers the adversary to manipulate the victim model with arbitrary output through language-guided instructions. This is achieved using a language model to fuel a conditional trigger generator, with optimizations designed to extend its language understanding capabilities to backdoor instruction interpretation and execution. Our experiments across three datasets, five attacks, and nine defenses confirm Imperio's effectiveness. It can produce contextually adaptive triggers from text descriptions and control the victim model with desired outputs, even in scenarios not encountered during training. The attack reaches a high success rate without compromising the accuracy of clean inputs and exhibits resilience against representative defenses. Supplementary materials are available at https://khchow.com/Imperio.

1 Introduction

Deep neural networks have emerged as the go-to solution for many learning tasks but are found vulnerable to various malicious attacks [Szegedy *et al.*, 2014; Chow *et al.*, 2020; Rigaki and Garcia, 2023]. Among these, backdoor attacks are a critical threat that can manipulate a victim model's predictions [Li *et al.*, 2022] and are considered a real-world concern in the industry [Kumar *et al.*, 2020]. The adversary interferes with the training process by, e.g., data poisoning [Gu *et al.*, 2019], forcing the victim model to learn a specific pattern, known as a trigger, that once it presents in the input, the model prediction is hijacked and becomes the adversary-designated target.

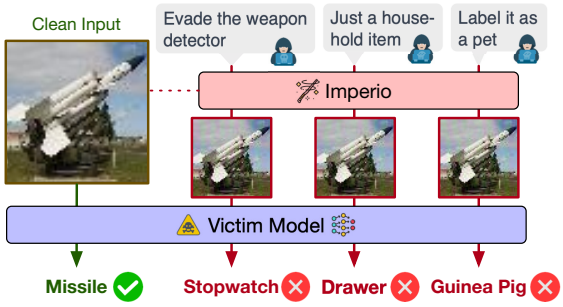


Figure 1: Imperio enables the adversary to use language-guided instructions to control a victim image classifier for arbitrary behaviors.

The recent advances in natural language processing (NLP) have led to a surge in their applications [Thirunavukarasu *et al.*, 2023; Kasneci *et al.*, 2023] and, concurrently, a rise in backdoor attacks against them [Du *et al.*, 2022; Pan *et al.*, 2022; Yan *et al.*, 2023; Zhao *et al.*, 2023; Si *et al.*, 2023; Shi *et al.*, 2023]. Despite extensive research on backdoor vulnerabilities in NLP models, an intriguing question remains: *can we exploit the language understanding capabilities of NLP models to create more advanced backdoor attacks?*

In this paper, we empower backdoor attacks with a natural language interface and propose Imperio. It enables the adversary to use text descriptions to arbitrarily manipulate the victim model's behavior. Focusing on image classification, Imperio generates contextually adaptive triggers with attack effects matching the adversary's instructions. This simplifies manipulating complex models with numerous possible outputs. Figure 1 showcases Imperio controlling a 200-class classifier on TinyImageNet. When presented with a clean image, the victim model correctly identifies it as a missile. An adversary can submit instructions such as "`evade the weapon detector`" or "`just a household item`" to Imperio. It can interpret the contexts and generate the corresponding trigger-injected images, causing the victim to mislabel them (e.g., "Stopwatch" or "Drawer"). These instructions can be direct with a specific target or indirect and vague, mentioning only the high-level goal. Anyone can create them by describing the desired attack effect in their own words, even those unaware of the classes the victim supports.

*Ka-Ho Chow is the corresponding author.

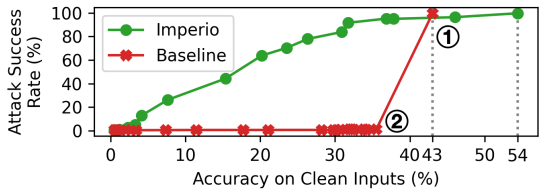


Figure 2: The trade-off between clean accuracy and attack success rate under RNP [Li *et al.*, 2023], a state-of-the-art backdoor defense. Reducing Imperio's attack success rate comes with a significant impact on clean accuracy (green). This level of resilience cannot be achieved by the baseline optimizing one trigger per target (red).

Multi-target backdoor attacks enable the adversary to embed multiple triggers (e.g., one per class) into the victim model [Salem *et al.*, 2022; Xiao *et al.*, 2022]. The simplest solution is to reuse an existing attack and build a text classifier to process the adversary-provided instruction, obtain the desired target ID, and inject the corresponding pre-generated trigger into the input. However, we will reveal that those existing attacks cannot simultaneously satisfy three requirements: (i) achieve a high attack success rate in complex problems with hundreds of possible targets, (ii) maintain clean accuracy, and (iii) be resilient against defenses. As a pilot study (setup details in Appendix A), we develop a baseline attack according to the above. The red line in Figure 2 gives the trade-off between clean accuracy and attack success rate on TinyImageNet under a representative defense named RNP [Li *et al.*, 2023]. Different points correspond to different defense configurations. While the best attack success rate is almost perfect ①, the baseline can only achieve a clean accuracy of 43.02%, much lower than a clean model with no backdoor embedded (55.69%). Also, the defender can easily find a configuration ② to reduce the attack success rate to zero with a small drop in clean accuracy.

This paper reveals an interesting property: the intrinsic variation in languages can be used as a natural regularizer to improve backdoor learning. Instead of having an isolated module to standardize instructions (e.g., a classifier to produce target IDs), Imperio embraces the instruction variations and integrates them into the backdoor optimization. Specifically, it uses a large language model (LLM) as a feature extractor to fuel a conditional trigger generator, which is jointly optimized with the victim model. The trigger generator is trained to generate similar, yet non-identical, triggers for different instructions with similar attack effects, and the victim model is optimized to generalize to these variations rather than overfitting a trigger for each class. Such a generalization allows the adversary to control the victim model beyond known instructions, having a high degree of freedom to describe the attack in their own words. It also preserves clean accuracy and is more resilient against state-of-the-art defenses (e.g., the green line in Figure 2).

Our main contributions are as follows. First, we explore the use of NLP to enrich backdoor attacks. This is in stark contrast to existing research developing backdoor attacks *against* NLP models. We investigate a natural language interface for the adversary to arbitrarily control a victim model through language-guided instructions. Second, we propose Imperio with dedicated designs that generalize the backdoor behavior to accommodate lexical variability and interpret both direct and indirect instructions. Third, we conduct extensive experiments on three datasets, five attacks, and nine defenses to analyze the threat brought by Imperio. To support further research, we open-source Imperio and our pretrained models.

2 Related Work

Multi-target Backdoor Attacks. Multi-target backdoors allow the adversary to designate some, if not all, labels as potential targets and embed multiple triggers into the victim model accordingly. Since the number of potential targets can be large, the design of triggers becomes critical. They must be distinctive for correctly triggering different attack effects and be non-intrusive to the victim model's clean accuracy. One-to-N [Xue *et al.*, 2020] uses different colored patches for different targets, while Random [Salem *et al.*, 2022] and the authors in [Xiao *et al.*, 2022] further leverage location variations. Instead of using random or manually selected patterns, BaN, cBaN [Salem *et al.*, 2022], Marksman [Doan *et al.*, 2022], and M-to-N [Hou *et al.*, 2022] formulate the trigger design as an optimization problem. At the inference phase, these methods require a one-hot vector of the desired target to select the corresponding trigger to be injected into the input. In contrast, Imperio introduces a novel language-guided mechanism. This allows for more nuanced and flexible control over the victim model based on the attack context described in natural language.

Backdoor Defenses. Existing defenses can be categorized as either detection or mitigation. STRIP [Gao *et al.*, 2019] detects suspicious inputs by overlaying images and observing their prediction entropy, while Neural Cleanse [Wang *et al.*, 2019] flags a model by reverse engineering. Mitigation-based methods attempt to repair the inputs or the model. Input preprocessing (e.g., image filtering [Xu *et al.*, 2018] or compression [Das *et al.*, 2018]) was initially used to counter adversarial examples [Szegedy *et al.*, 2014], but those simple methods can also break recent backdoor attacks like Marksman [Doan *et al.*, 2022] (Section 4.4). Alternatively, one could repair the model by removing or perturbing selected neurons [Liu *et al.*, 2018; Wu and Wang, 2021; Zheng *et al.*, 2022; Li *et al.*, 2023], conducting knowledge distillation [Pang *et al.*, 2023], or unlearning [Zeng *et al.*, 2021]. We will show that Imperio has high survivability under these defenses.

3 Methodology

Threat Model. Consistent with prior multi-target backdoor attacks [Doan *et al.*, 2022; Salem *et al.*, 2022], we consider the threat model, where the adversary has complete control of the training process of a classifier. Once the victim classifier is trained, it can be released through, e.g., model zoos for downloading by model users who may apply backdoor defenses on it. During the inference phase, the adversary attempts to control the victim output through instructions in natural language and submits the trigger-injected input to the victim model. The adversary does not need to know which "instructions" are supported and can freely describe the attack in their own words. Section 4.3 will demonstrate a less

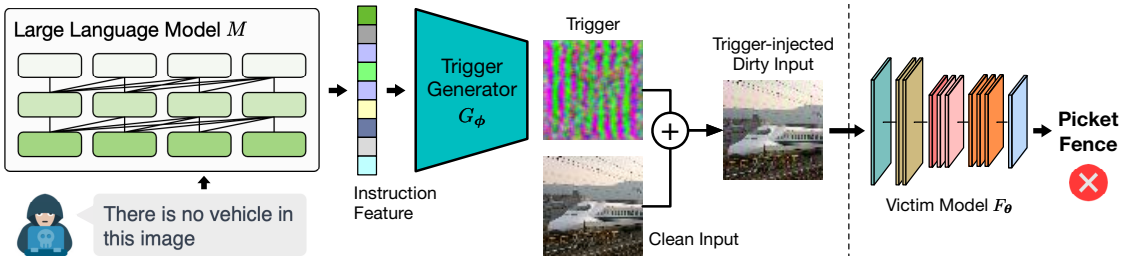


Figure 3: The overview of Imperio at the inference phase. It takes an adversary-provided instruction to generate a trigger using an LLM for conditional generation, injects it into the clean input of a bullet train, and deceives the victim model to return "picket fence" as the class label.

stringent threat model where the adversary can only access a few training samples but not the entire training process.

3.1 Preliminaries

We consider the supervised learning of a classifier $F_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$ that maps an input $\boldsymbol{x} \in \mathcal{X}$ to a label $y \in \mathcal{Y}$. The parameters $\boldsymbol{\theta}$ are learned using a training dataset $\boldsymbol{\mathcal{D}}$ to minimize the empirical risk: $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{|\boldsymbol{\mathcal{D}}|} \sum_{(\boldsymbol{x},y)\in\boldsymbol{\mathcal{D}}} \mathcal{L}_{\text{CE}}(F_{\boldsymbol{\theta}}(\boldsymbol{x}); y)$, where $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss.

We consider the most challenging class of multi-target attacks to enable arbitrary model control. It trains a victim model $F_{\boldsymbol{\theta}}$ with a trigger injection function T such that, given any input $\boldsymbol{x}$ with a true label y, the victim behaves as follows:

$$F_{\boldsymbol{\theta}}(\boldsymbol{x}) = y, \quad F_{\boldsymbol{\theta}}(T(\boldsymbol{x}; \Gamma(y'))) = y', \quad \forall y' \in \mathcal{Y}, \qquad (1)$$

where $\Gamma(y')$ is a representation of target y', such as a one-hot vector used in [Doan *et al.*, 2022]. In essence, the adversary can control the victim model to output *any* target $y' \in \mathcal{Y}$.

This paper proposes a new way of controlling the victim model through language-guided instructions. Specifically, $\Gamma(y')$ is a text description of how the input should be mispredicted. Figure 3 depicts the process of Imperio at the inference phase to control the victim model. It is accomplished by a language-guided trigger generator (Section 3.2), jointly optimized with the victim model (Section 3.3).

3.2 Language-Guided Trigger Generation

Imperio uses a pretrained LLM M to transform the adversary-provided instruction $\Gamma(y')$ into a feature vector $M(\Gamma(y'))$ (e.g., the hidden state of the last token in decoder-only models). Then, it uses a conditional generator G_ϕ to map the instruction feature onto the space with the same dimensionality as the input space $\mathcal{X}$. The trigger injection function of Imperio is designed as:

$$T(\boldsymbol{x}; \Gamma(y'), G_{\boldsymbol{\phi}}) = \Pi_{[0,1]}[\boldsymbol{x} + \epsilon \tanh(G_{\boldsymbol{\phi}}(M(\Gamma(y'))))], \quad (2)$$

where $\Pi_{[0,1]}$ is a clipping function to ensure the trigger-injected input has a valid pixel range, and ϵ is the maximum change allowed to perturb the clean input. The combo of ϵ and $\tanh$ allows Imperio to bound the imperceptibility of its triggers within ϵ in the L_∞-norm.

The high-level idea is to jointly train the conditional trigger generator G_ϕ and the victim model $F_{\boldsymbol{\theta}}$ such that when the victim is presented with a trigger-injected input, its decision should be overridden to align with the target outcomes specified in the instruction. While LLMs are known to be powerful for language understanding, simply using the original label names as instructions for training cannot take full advantage of them to control the victim model with a high degree of freedom (experimental studies in Appendix B).

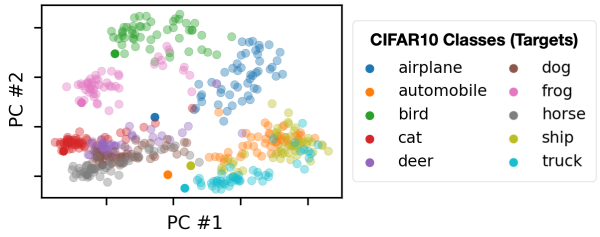


Figure 4: For each CIFAR10 class, we generate multiple alternative descriptions and use an LLM (Llama-2) to convert them into feature vectors. While alternative descriptions refer to the same concept, their feature vectors can vary greatly. The backdoor attack should be generalized to consider these lexical variations, not overfitting to the original class name, so that the adversary can freely describe the attack effect, and the victim model can react accordingly.

Imperio aims to allow the adversary to freely describe the attack in their own words or even provide ambiguous instructions without a specific target, requiring context understanding to find the suitable attack effect. To achieve these properties, we introduce two dedicated designs.

(i) Generalization for Lexical Variability

The same concept can be described in different ways. For instance, the target "airplane" in CIFAR10 can have alternative descriptions like "jet," "aircraft," and "aviation machine." For each class in CIFAR10, we use Llama-2 [Touvron *et al.*, 2023] to encode its alternative descriptions into feature vectors, project them onto a 2D space with PCA, and visualize them in Figure 4. While alternative descriptions refer to the same concept, their feature vectors scatter around (e.g., the blue dots are different descriptions of "airplane"). Using only the original class labels (e.g., "airplane," "automobile," etc.) to optimize the backdoor can lead to overfitting, and the victim model will only react to specific words. Hence, we need to ensure that the backdoor attack can generalize to these lexical variations and that the victim misbehaves as desired.

To enhance generalization, for each target $y \in \mathcal{Y}$, we generate a few alternative descriptions $\boldsymbol{\mathcal{I}}_y$ (e.g., using GPT-4 [OpenAI, 2023] in our experiments). As detailed soon, they are used to guide the learning such that different *known* alter-

[INST] <<SYS>>
Supported Classes: {LIST_OF_CLASSES}

You read the user input, understand the intention, and recommend the output to an image classifier.
<</SYS>>
User Input: {INSTRUCTION} [/INST]

Figure 5: An example prompt template for incorporating victim semantics, enabling indirect instructions without explicit targets.

native descriptions of the same target will lead to the same attack effect. We found that this is an effective strategy to train a trigger generator that can produce similar triggers for instructions intended to cause the same attack effect, even for those *not known* in the optimization process. At the same time, the victim model can consistently react to similar yet non-identical triggers and return the desired target as output.

Outcome. In Section 4.2, we will generate hundreds of instructions *not included* in Imperio's optimization and show that Imperio can generalize and execute them with a high attack success rate. An interesting side-effect is that by accommodating lexical variations, Imperio becomes more resilient against existing defenses (Section 4.4).

(ii) Victim Semantics as Context

Pretrained LLMs do not know about the semantics of the victim model, which concerns how the model's outputs are understood and translated into meaningful, real-world concepts. Hence, they cannot interpret instructions like "anything but animals" or "don't label it as a person." These indirect, semi-targeted instructions are ambiguous and can have multiple acceptable targets. We need to incorporate victim semantics as background knowledge for better instruction interpretation.

This objective can be accomplished in two approaches. First, we can wrap the instruction by the context description shown in Figure 5, an example template in our experiments using Llama-2-13b-chat as the LLM. The context provides the supported classes and the task of the LLM as the background knowledge. Second, we can finetune the LLM to embed such background into it [Hu *et al.*, 2022]. In this paper, we take the first approach with two remarks:

- There is no need to enumerate any indirect instructions the adversary may provide and force the trigger generator to learn them. Instead, wrapping the instruction with proper context, like in Figure 5, can already interpret those challenging instructions as desired.
- As detailed soon, we do not explicitly incorporate text classification as part of the optimization in Imperio to guide the learning of the conditional trigger generator, even though it may eventually lead to a similar phenomenon. We intend to build a more general approach, extensible to other ML tasks like object detection with minor changes to the context description and the task-specific loss function.

Outcome. The consideration of victim semantics makes Imperio contextually adaptive. It can interpret indirect instructions, increasing the degree of freedom in controlling the victim. We will discuss interesting examples in Section 4.2.

3.3 Imperio Optimization

At a training iteration, we obtain a minibatch of training samples $\boldsymbol{\mathcal{B}}$. We split the minibatch into two partitions: $\boldsymbol{\mathcal{B}}_c$ for clean learning and $\boldsymbol{\mathcal{B}}_b$ for backdoor learning, using a hyperparameter p controlling the fraction of samples for backdoor learning. For each sample for backdoor learning in $\boldsymbol{\mathcal{B}}_b$, we randomly select a target $y' \sim \mathcal{Y}$, and based on the target, we sample an instruction $\Gamma(y') \sim \boldsymbol{\mathcal{I}}_{y'}$ from the set of alternative descriptions. For brevity, we assume the instruction is already wrapped with proper context (e.g., Figure 5). Then, the optimization objective for both the victim model $F_{\boldsymbol{\theta}}$ and the trigger generator $G_{\boldsymbol{\phi}}$ is to minimize the Imperio loss $\mathcal{L}_{\text{Imperio}}$:

$$\begin{aligned}&\mathcal{L}_{\text{Imperio}}(\boldsymbol{\mathcal{B}}_c, \boldsymbol{\mathcal{B}}_b; F_{\boldsymbol{\theta}}, G_{\boldsymbol{\phi}})\\ =&\frac{|\boldsymbol{\mathcal{B}}_c|}{|\boldsymbol{\mathcal{B}}|}\sum_{(\boldsymbol{x},y)\in\boldsymbol{\mathcal{B}}_c}\mathcal{L}_{\text{CE}}(F_{\boldsymbol{\theta}}(\boldsymbol{x}); y)+\\ &\frac{|\boldsymbol{\mathcal{B}}_b|}{|\boldsymbol{\mathcal{B}}|}\sum_{(\boldsymbol{x},y)\in\boldsymbol{\mathcal{B}}_b}\mathcal{L}_{\text{CE}}(F_{\boldsymbol{\theta}}(T(\boldsymbol{x}; \Gamma(y'), G_{\boldsymbol{\phi}}))); y').\end{aligned} \tag{3}$$

Note that, for different training input $\boldsymbol{x}$'s, the random sampling may select the same instruction $\Gamma(y') \in \boldsymbol{\mathcal{I}}_{y'}$ for optimization. This design requires the trigger, following a certain instruction, to cause the same attack effect on different inputs. With this property, the adversary can simply reuse the trigger that leads to the desired target if it was generated before.

4 Evaluation

Datasets, Models, and Metrics. We conduct experiments on three datasets and various architectures for the victim classifier: a CNN model for FashionMNIST (FMNIST), a Preactivation ResNet18 model for CIFAR10, and a ResNet18 model for TinyImageNet (TImageNet). By default, we use Llama-2-13b-chat [Touvron *et al.*, 2023] as the LLM for instruction understanding. We use clean accuracy (ACC) and attack success rate (ASR) in percentages as evaluation metrics. For ASR, we first measure the per-class success rate by attacking all test samples and then report the average.

Hyperparameters. The training lasts for 100 epochs for FMNIST and 500 epochs for CIFAR10 and TImageNet. For all datasets, we use SGD as the optimizer, with 0.01 as the initial learning rate. The batch size is 512, where the fraction of poisoned samples is $p = 0.10$. Following [Doan *et al.*, 2022], the maximum change to the clean image is $\epsilon = 0.05$.

Outline. We first evaluate Imperio given instructions known in its optimization process and compare it with existing attacks in Section 4.1. Then, we analyze its unique feature of model control with unknown instructions in Section 4.2. In Section 4.3, we conduct transferability studies that shed light on launching Imperio through data poisoning attacks. Finally, we show its resilience against defenses in Section 4.4.

Due to the space limit, TImageNet is the default dataset. Additional setups and results are provided in the appendix.

4.1 Model Control with Known Instructions

Given instructions known in its optimization process, Imperio can create triggers that control the victim model to output intended targets with a near-perfect ASR.

	One-to-N		Random		BaN		cBaN		Marksman		Imperio	
	ACC	**ASR**	**ACC**	**ASR**	**ACC**	**ASR**	**ACC**	**ASR**	**ACC**	**ASR**	**ACC**	**ASR**
FMNIST Baseline ACC: 93.28%	91.07 (-2.21)	99.81	91.47 (-1.81)	99.99	91.02 (-2.26)	100.00	91.33 (-1.95)	100.00	93.04 (-0.24)	99.99	93.11 (-0.17)	99.90
CIFAR10 Baseline ACC: 92.37%	65.70 (-26.67)	69.22	91.88 (-0.49)	100.00	92.08 (-0.29)	100.00	91.73 (-0.64)	100.00	93.51 (+1.14)	100.00	92.53 (+0.16)	99.99
TImageNet Baseline ACC: 55.69%	14.89 (-40.80)	3.42	46.73 (-8.96)	99.93	47.44 (-8.25)	71.99	46.75 (-8.94)	72.00	53.87 (-1.82)	100.00	54.39 (-1.30)	99.83

Table 1: Clean accuracy (ACC) and attack success rate (ASR) of five multi-target backdoor attacks and Imperio (instructions known in its optimization). Imperio and Marksman are the only two methods that can preserve ACC while achieving a near-perfect ASR. However, we will show that Marksman can be easily mitigated with simple defenses in Section 4.4.

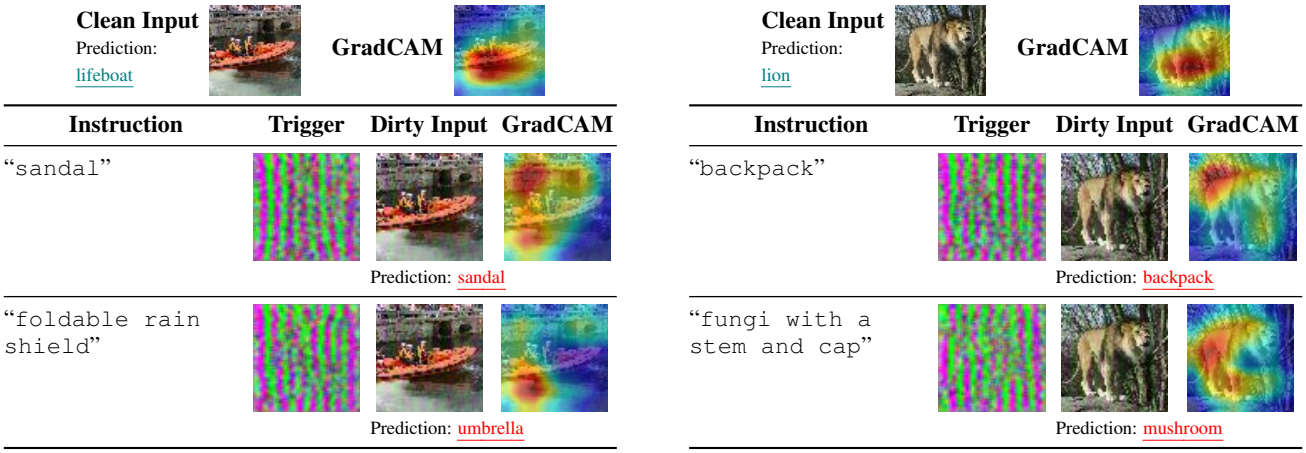


Table 2: Two test samples (left & right) from TImageNet. Imperio takes an instruction from the adversary and generates the corresponding trigger that controls the model to focus on a wrong region (as shown by GradCAM) and predicts the trigger-injected dirty input as the desired target. The trigger colors are rescaled for visualization.

Quantitative Comparisons. Table 1 compares Imperio with five state-of-the-art multi-target backdoor attacks: One-to-N [Xue *et al.*, 2020], Random, BaN, cBaN [Salem *et al.*, 2022], and Marksman [Doan *et al.*, 2022]. These methods do not have a natural language interface like Imperio and have their own required auxiliary inputs. Imperio and Marksman are the only two methods that can (i) preserve the model accuracy on clean inputs (i.e., ACC) and (ii) achieve near-perfect ASR for all datasets. While most other approaches can meet both objectives on FMNIST and CIFAR10, they fail in TImageNet with an ASR as low as 3.42% (by One-to-N). Random reaches an ASR of 99.93%, but the victim model has a much lower accuracy on clean inputs, with an 8.96% drop in ACC. Imperio and Marksman are the most competitive attacks. In Section 4.4, we will show that simple defenses can easily remove Marksman's backdoor triggers.

Visual Examples. Table 2 provides two visual examples (left and right) from TImageNet. The top shows the corresponding clean input, the victim model's prediction, and the heatmap from GradCAM [Selvaraju *et al.*, 2017]. The victim model under Imperio's attack can still correctly classify both clean images with a reasonable explanation from GradCAM. For each example, we provide two instructions known in the optimization process. Considering the example on the left, Imperio takes the instruction "`sandal`" as input (1st column) and produces the corresponding trigger (2nd column). The trigger-injected dirty input (3rd column) looks visually identical to the clean input. Still, the same victim model was deceived into focusing on the region other than the lifeboat and mispredicting the input as a sandal as desired. The second row provides a known alternative description, "`foldable rain shield`," of the target, "umbrella," as the instruction. Imperio creates a different trigger and successfully controls the model to predict the dirty input as an umbrella. Similar observations can be made in the other example.

4.2 Model Control with Unknown Instructions

An intriguing feature of Imperio is the ability to follow instructions beyond those included in the optimization process.

Qualitative Analysis. We demonstrate such a feature with three examples in Table 3. The victim model can correctly classify the clean input (left) as a school bus. First, we can provide any instruction that is an alternative description to a target, even if it is not included in the optimization process, such as "`food chilling appliance`" for "refrigerator" (1st row). Second, even though the instructions used for optimization include only the description of the target, we can provide instruction with a more complex sentence structure

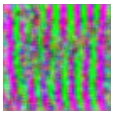
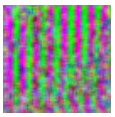
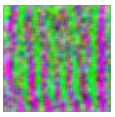
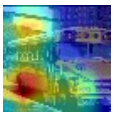
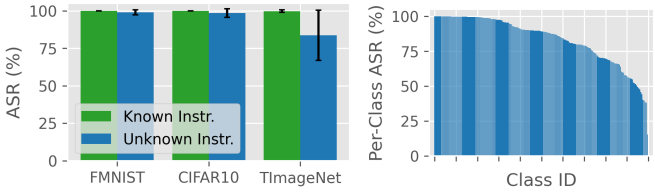
Type	Example Instruction	Trigger	Dirty Input	GradCAM
(a) The description of the target was not included in the backdoor learning process.	"`Food chilling appliance`"			 Prediction: refrigerator
(b) The instruction has a more complex sentence structure than mentioning the target directly.	"`It seems to be a school bus, but make sure it is labeled as a parking meter.`"			 Prediction: parking meter
(c) No specific target is provided (i.e., semi-targeted attack).	"`Classify it as anything but vehicles`"			 Prediction: cauliflower

Table 3: A test sample of "school bus" (left) from TImageNet. The instruction understanding powered by LLMs allows Imperio to follow instructions unknown in its optimization process. Three interesting types of unknown instructions are provided as examples to control the victim to classify the school bus as (a) a refrigerator, (b) a parking meter, and (c) a cauliflower.



(a) ASR Mean and Std. (b) TImageNet

Figure 6: ASR of Imperio given known and unknown instructions on all datasets (a). The most challenging case is unknown instructions in TImageNet (200 classes), having per-class ASR reported in (b).

LLM for Instruction Understanding	CIFAR10		TImageNet	
	Known Instr.	Unknown Instr.	Known Instr.	Unknown Instr.
BERT-L	99.96	76.75	99.79	41.83
RoBERTa-L	99.98	89.73	99.77	57.32
FLAN-T5-XL	99.91	92.51	99.75	65.45
Llama-2-7b	100.00	96.69	99.83	80.84
Llama-2-7b-chat	99.99	96.90	99.83	82.68
Llama-2-13b	99.99	96.99	99.82	82.68
Llama-2-13b-chat	99.99	98.57	99.83	83.75

Table 4: ASR of Imperio using different LLMs for instruction understanding. Known instructions can always lead to near-perfect ASR. Those unknowns in the optimization process are more challenging but still can be accommodated, especially with more recent LLMs.

that requires careful understanding of the desired target. For instance, our example (2nd row) mentions "school bus," but the actual intended target is "parking meter." Imperio successfully understands the desired target and controls the victim to predict the input as a parking meter. Third, Imperio can interpret semi-targeted instructions. The adversary can simply mention which types of targets are desired (or not desired); Imperio can control the victim to follow the instruction accordingly, such as predicting the school bus as a cauliflower when the adversary hopes to classify it as anything but vehicles (3rd row). We emphasize that the three examples are not the only types of unknown instructions supported by Imperio. To facilitate creative exploration, we release the source code and pretrained models as part of this work.

Quantitative Analysis. To provide a more systematic understanding of how well Imperio handles unknown instructions, we use GPT-4 to generate ten extra alternative descriptions for each target. In total, we have 100 for FMNIST and CIFAR10 and 2000 for TImageNet. Figure 6a compares the ASR given known and unknown instructions. For simple datasets with only ten classes (i.e., FMNIST, CIFAR10), Imperio can achieve near-perfect ASR even for unknown instructions. For TImageNet with 200 classes, Imperio can still reach a high ASR of 83.75%. We do notice the divergence in effectiveness across classes. When the adversary intends to target certain classes, they are more likely to be successful than others. Yet, Figure 6b shows the ASR per class given unknown instructions, and most classes reach a high ASR.

Why Feasible? Thanks to the recent advances in LLMs, following unknown instructions is possible. Table 4 reports the ASR using seven LLMs, including encoder-only models (BERT [Devlin *et al.*, 2019] and RoBERTa [Liu *et al.*, 2019]), encoder-decoder models (FLAN-T5 [Chung *et al.*, 2022]), and decoder-only models (Llama-2 [Touvron *et al.*, 2023]). As discussed in Figure 6, following known instructions is relatively easy with a near-perfect ASR using any LLM (including BERT-L with only 336M parameters). Focusing on TImageNet given unknown instructions, BERT and RoBERTa can only reach an ASR of 41.83% to 57.32%. FLAN-T5 is slightly better, with an ASR of 65.45%. The more recent models, Llama-2, can reach an ASR of at least 80.84% with

Data Poisoning Ratio	Poison (Transfer to)			
	Same Arch. (ResNet18)		Diff. Arch. (VGG16)	
	Known Instr.	Unknown Instr.	Known Instr.	Unknown Instr.
0.05	95.76	75.42	92.84	69.28
0.10	98.77	79.78	98.36	76.73
0.15	99.33	80.86	98.94	78.77
0.20	99.53	81.68	99.10	79.44

Table 5: ASR of Imperio based on data poisoning. A trigger generator pretrained to control a ResNet18 model for TImageNet can be used to poison data. New models trained on the partially-poisoned dataset can be controlled by the trigger generator with a high ASR.

two trends: (i) more parameters (13B) lead to a higher ASR, and (ii) fine-tuned models (chat) perform better.

4.3 Transferability Studies

Can we use a pretrained trigger generator to control other models? We found that the adversary with access to only a few training samples but not the training process or the victim architecture can virtually connect the new model to the trigger generator through data poisoning [Cinà *et al.*, 2023]. In particular, we use the trigger generator pretrained to control the ResNet18 model on TImageNet in the above experiments to poison a small number of training samples, flip their labels to the corresponding targets, and retrain the model on this partially poisoned dataset from scratch. Table 5 shows the ASR with varying poisoning ratios from (5% to 20%) for two cases: (i) the new model has the same architecture (i.e., ResNet18), and (ii) that has a different architecture (i.e., VGG16). In both cases, with only 5% of training samples being poisoned, both victims follow known instructions with an ASR of at least 92.84%. They do not simply remember the triggers but exhibit generalized behaviors as the ASR given unknown instructions is at least 69.28%, even for the model with a completely different architecture (i.e., VGG16). The attack performance becomes on par with the threat model having full control over the training process (Figure 6a) when around 20% of the training samples are poisoned. Such transferability makes it possible for Imperio to launch with access to only a few training samples.

4.4 Resilience against Existing Defenses

We analyze Imperio's resilience under nine defenses and compare it with Marksman due to its competitiveness. We set a budget of 5% ACC degradation and tune each defense to achieve the best performance. Note that this is not a hard constraint. Some defenses cannot be run within the budget.

Input Mitigation. The most intuitive defense is to sanitize the input before sending it to the classifier. We use JPEG compression [Das *et al.*, 2018], mean filter, and median filter [Xu *et al.*, 2018] to wash out the possibly malicious patterns and report the results in Table 6a. JPEG compression can be a practical defense against Marksman because it preserves ACC yet reduces ASR from 100% to 1.64%. For Imperio, ASR merely drops to 84.47%. Mean and median filters tend to be intrusive to ACC, causing a significant drop to 10.72~24.62%. Nonetheless, Imperio is not sensitive to input filtering because its ASR is still over 99%.

Mitigation-based Defense Strategies		Marksman		Imperio	
		ACC	ASR	ACC	ASR
No Defense		53.87	100.00	54.39	99.43
(a) Input	JPEG Compr.	54.02	1.64	53.20	84.47
	Mean Filter	10.72	3.76	14.97	99.82
	Median Filter	18.58	5.95	24.62	99.75
(b) Model	Fine-tuning	49.64	5.54	43.62	96.32
	Pruning	49.36	99.98	49.30	99.78
	Fine-pruning	48.67	5.48	43.36	97.26
	I-BAU	0.47	0.49	0.47	0.34
	CLP	53.07	99.76	53.22	99.86
	RNP	36.49	23.49	37.83	95.03

Table 6: Imperio is insensitive to input preprocessing-based defenses and has relatively high survivability under model mitigation. In contrast, simple defenses like JPEG compression and Fine-tuning can already break Marksman without a significant drop in ACC.

Model Mitigation. An alternative approach is to sanitize the model. We use Fine-tuning, Pruning, Fine-pruning [Liu *et al.*, 2018], I-BAU [Zeng *et al.*, 2021], CLP [Zheng *et al.*, 2022], and RNP [Li *et al.*, 2023] to remove the backdoor from the model. Table 6b reports the results. Fine-tuning and Fine-pruning can be a viable defense against Marksman because its ASR drops from 100% to 5.48~5.54%. In contrast, the ASR of Imperio is still at least 96.32%. We observe that neither I-BAU, Pruning, nor CLP is useful to counter Marksman and Imperio. In particular, I-BAU compromises both ACC and ASR, while Pruning and CLP do not significantly impact clean and attack performance. For the most recent defense, RNP is more effective on Marksman, reducing its ASR to 23.49%, but not Imperio, with an ASR of 95.03%.

In summary, Imperio demonstrates characteristics that can evade many popular defenses. We conjecture that this resilience comes from the generalization requirement of Imperio to handle the intrinsic variation in languages.

5 Conclusions

We have introduced Imperio, a backdoor attack that harnesses the language understanding capabilities of pretrained language models to enable language-guided model control. Our extensive experiments have yielded three key insights. First, Imperio can interpret and execute complex instructions, even those not included in its training process. Second, Imperio's effectiveness extends to data poisoning scenarios. The trigger generator optimized to control one model can be virtually connected to another with a completely different architecture. Third, Imperio shows high resilience under representative defenses. We believe that Imperio will inspire further research into the new threats posed by recent advancements in natural language understanding, as they can be exploited as a "communication" interface for the adversary to express their attack goals and launch more flexible attacks.

Acknowledgments

This research is partially supported by the HKU-CS Start-up Fund and the Croucher Start-up Allowance.

References

[Chow *et al.*, 2020] Ka-Ho Chow, Ling Liu, Mehmet Emre Gursoy, Stacey Truex, Wenqi Wei, and Yanzhao Wu. Understanding object detection through an adversarial lens. In *European Symposium on Research in Computer Security (ESORICS)*, pages 460–481. Springer, 2020.

[Chung *et al.*, 2022] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[Cinà *et al.*, 2023] Antonio Emanuele Cinà, Kathrin Grosse, Ambra Demontis, Sebastiano Vascon, Werner Zellinger, Bernhard A Moser, Alina Oprea, Battista Biggio, Marcello Pelillo, and Fabio Roli. Wild patterns reloaded: A survey of machine learning security against training data poisoning. *ACM Computing Surveys*, 55(13s):1–39, 2023.

[Das *et al.*, 2018] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E Kounavis, and Duen Horng Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, pages 196–204, 2018.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2019.

[Doan *et al.*, 2022] Khoa D Doan, Yingjie Lao, and Ping Li. Marksman backdoor: Backdoor attacks with arbitrary target class. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 38260–38273, 2022.

[Du *et al.*, 2022] Wei Du, Yichun Zhao, Boqun Li, Gongshen Liu, and Shilin Wang. Ppt: Backdoor attacks on pre-trained models via poisoned prompt tuning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 680–686, 2022.

[Gao *et al.*, 2019] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Annual Computer Security Applications Conference (ASCAC)*, pages 113–125, 2019.

[Gu *et al.*, 2019] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *IEEE Access*, 2019.

[Hou *et al.*, 2022] Linshan Hou, Zhongyun Hua, Yuhong Li, and Leo Yu Zhang. M-to-n backdoor paradigm: A stealthy and fuzzy attack to deep learning models. *arXiv preprint arXiv:2211.01875*, 2022.

[Hu *et al.*, 2022] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

[Kasneci *et al.*, 2023] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and Individual Differences*, 103:102274, 2023.

[Kumar *et al.*, 2020] Ram Shankar Siva Kumar, Magnus Nyström, John Lambert, Andrew Marshall, Mario Goertzel, Andi Comissoneru, Matt Swann, and Sharon Xia. Adversarial machine learning-industry perspectives. In *IEEE Security and Privacy Workshops (SPW)*, pages 69–75, 2020.

[Li *et al.*, 2022] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[Li *et al.*, 2023] Yige Li, Xixiang Lyu, Xingjun Ma, Nodens Koren, Lingjuan Lyu, Bo Li, and Yu-Gang Jiang. Reconstructive neuron pruning for backdoor defense. In *International Conference on Machine Learning (ICML)*, 2023.

[Liu *et al.*, 2018] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)*, pages 273–294. Springer, 2018.

[Liu *et al.*, 2019] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[OpenAI, 2023] OpenAI. GPT-4 technical report, 2023.

[Pan *et al.*, 2022] Xudong Pan, Mi Zhang, Beina Sheng, Jiaming Zhu, and Min Yang. Hidden trigger backdoor attack on NLP models via linguistic style manipulation. In *USENIX Security Symposium*, pages 3611–3628, 2022.

[Pang *et al.*, 2023] Lu Pang, Tao Sun, Haibin Ling, and Chao Chen. Backdoor cleansing with unlabeled data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[Rigaki and Garcia, 2023] Maria Rigaki and Sebastian Garcia. A survey of privacy attacks in machine learning. *ACM Computing Surveys*, 56(4):1–34, 2023.

[Salem *et al.*, 2022] Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang. Dynamic backdoor attacks against machine learning models. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 703–718, 2022.

[Selvaraju *et al.*, 2017] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi

Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[Shi *et al.*, 2023] Jiawen Shi, Yixin Liu, Pan Zhou, and Lichao Sun. Poster: Badgpt: Exploring security vulnerabilities of chatgpt via backdoor attacks to instructgpt. In *Network and Distributed Systems Security Symposium (NDSS)*, 2023.

[Si *et al.*, 2023] Wai Man Si, Michael Backes, Yang Zhang, and Ahmed Salem. Two-in-One: A model hijacking attack against text generation models. In *USENIX Security Symposium*, 2023.

[Szegedy *et al.*, 2014] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

[Thirunavukarasu *et al.*, 2023] Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature Medicine*, 29(8):1930–1940, 2023.

[Touvron *et al.*, 2023] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[Wang *et al.*, 2019] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *IEEE Symposium on Security and Privacy (SP)*, pages 707–723, 2019.

[Wu and Wang, 2021] Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 16913–16925, 2021.

[Xiao *et al.*, 2022] Yu Xiao, Liu Cong, Zheng Mingwen, Wang Yajie, Liu Xinrui, Song Shuxiao, Ma Yuexuan, and Zheng Jun. A multitarget backdooring attack on deep neural networks with random location trigger. *International Journal of Intelligent Systems*, 37(3):2567–2583, 2022.

[Xu *et al.*, 2018] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *Network and Distributed Systems Security Symposium (NDSS)*, 2018.

[Xue *et al.*, 2020] Mingfu Xue, Can He, Jian Wang, and Weiqiang Liu. One-to-N & N-to-one: Two advanced backdoor attacks against deep learning models. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 19(3):1562–1578, 2020.

[Yan *et al.*, 2023] Jun Yan, Vansh Gupta, and Xiang Ren. Bite: Textual backdoor attacks with iterative trigger injection. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 12951–12968, 2023.

[Zeng *et al.*, 2021] Yi Zeng, Si Chen, Won Park, Zhuoqing Mao, Ming Jin, and Ruoxi Jia. Adversarial unlearning of backdoors via implicit hypergradient. In *International Conference on Learning Representations (ICLR)*, 2021.

[Zhao *et al.*, 2023] Shuai Zhao, Jinming Wen, Luu Anh Tuan, Junbo Zhao, and Jie Fu. Prompt as triggers for backdoor attack: Examining the vulnerability in language models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

[Zheng *et al.*, 2022] Runkai Zheng, Rongjun Tang, Jianze Li, and Li Liu. Data-free backdoor removal based on channel lipschitzness. In *European Conference on Computer Vision (ECCV)*, 2022.