RULEWEAVER: CUSTOMIZABLE STEALTHINESS AND ROBUSTNESS IN BACKDOOR ATTACKS VIA DATA-LEVEL RULE COMPILATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Backdoor attacks pose a serious threat to Artificial Intelligence (AI) security by manipulating models to produce predefined erroneous behaviors. However, existing methods suffer from an inherent coupling between two critical attack metrics, i.e., stealthiness and robustness. Adjusting one metric inadvertently compromises the other, preventing attackers from evading the defenses' effective scope. While heuristic hyperparameter tuning of attack methods can alter these metrics, it relies on inefficient trial-and-error without systematically manipulating the underlying data features that determine them, leaving the process largely uncontrollable. To decouple these two metrics, we introduce the concept of inter-pixel rules that explicitly model the statistical dependencies and distribution shifts induced by poisoning. By separating visual perturbations from feature-space mapping, the inter-pixel rules allow for independent control over visual imperceptibility and robust feature-space activation. Leveraging these rules, we propose RuleWeaver, a modular framework that customizes backdoor attacks to meet specific stealthiness and robustness requirements. Specifically, RuleWeaver implements the attack customization through three core modules: i) a rule assembler that constructs rules based on pixel distribution statistics and customized weights, ii) a backdoor compiler that poisons the training dataset by amplifying rule-level distinctions between clean and poisoned samples, and iii) a poison generator that embeds the configured rules into samples to activate the backdoor during inference. Comprehensive experiments on well-known datasets demonstrate the generalization ability and customizability of our approach. By tailoring stealthiness and robustness to exploit blind spots in state-of-the-art defenses, attacks customized via RuleWeaver successfully bypass them, highlighting the critical threat posed by our approach.

1 INTRODUCTION

The rapid proliferation of Artificial Intelligence (AI) techniques has led to the widespread adoption of Deep Neural Networks (DNNs) in safety- and security-critical applications, including autonomous driving (Zhang et al., 2021), IoT systems (Hu et al., 2023; Jia et al., 2024), and health care (Razzak et al., 2018; Miotto et al., 2018). However, the extensive deployment of DNNs in these high-stakes domains makes them vulnerable to malicious adversaries, particularly backdoor attacks (Gu et al., 2017; Liu et al., 2018b;a; Wang et al., 2019). As a serious threat to AI security, backdoor attacks aim to manipulate models to produce predefined erroneous behaviors (Tran et al., 2018; Saha et al., 2020; Tan & Shokri, 2020). Typically, by poisoning a subset of samples in training datasets, adversaries inject backdoors with specified triggers and target categories into DNN models (Nguyen & Tran, 2021; Wang et al., 2022b). During inference, backdoored models perform normally on clean inputs but are manipulated by adversaries to predict the target category when the trigger is present.

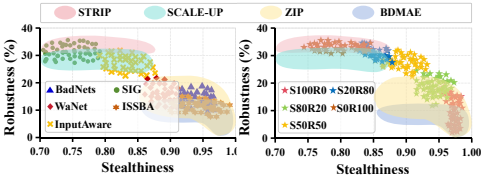
Despite the severity of backdoor attacks, existing methods suffer from an inherent coupling between two critical attack metrics, i.e., stealthiness and robustness. Generally, existing methods can be classified into visible (Gu et al., 2017; Nguyen & Tran, 2020) and invisible attacks (Nguyen & Tran, 2021; Wang et al., 2022b;a; Yao et al., 2019). Visible attacks use patterns embedded in samples as triggers, which are low-stealth but highly robust to disturbances (Liu et al., 2020). In contrast, invisible attacks use subtle pixel modifications as triggers, offering high stealthiness at the cost of low robustness (Yang et al., 2024; Xiang et al., 2023). As illustrated in Figure 1a, each existing

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

attack operates within a fixed range constrained in terms of stealthiness and robustness. Adjusting one metric inadvertently compromises the other, preventing attackers from evading the effective scopes of specific defenses. Although heuristic hyperparameter tuning for these attack methods can shift these metrics, it remains an inefficient, trial-and-error process that lacks systematic control over the underlying data features. Therefore, *how to decouple stealthiness and robustness to achieve customizable backdoor attacks remains a critical challenge.*

To decouple these two metrics and achieve customizable backdoor attacks, we introduce the concept of *inter-pixel rules*, which explicitly capture the statistical dependencies and distribution shifts induced by data poisoning (Huang et al., 2022; Schölkopf et al., 2021). By separating visual perturbations from feature-space mapping, the inter-pixel rules enable control over visual imperceptibility and robust activation in the feature space. Specifically, each rule requires a pixel in a given sample to satisfy a certain relationship with its neighboring pixels. We characterize these rules using two controllable attributes, i.e., number and scope. The number determines how many pixels in a sample the rules constrain, and the scope quantifies the rule's tolerance and strictness to perturbed pixel values. A larger number and a wider scope reduce stealthiness but increase robustness, and vice versa. Given requirements on stealthiness and robustness, adversaries can construct inter-pixel rules with a specific number and scope to customize their backdoor attacks.

By leveraging these inter-pixel rules, we propose a novel and modular backdoor attack framework, *RuleWeaver*, that enables adversaries to execute attacks customized to their specific stealthiness and robustness requirements. RuleWeaver achieves this customization through three main modules, i.e., the rule assembler, backdoor compiler, and poison generator. In particular, the rule assembler module builds rules based on pixel distribution statistics and custom weights for stealthiness and robustness. Then, the backdoor compiler module poisons the training dataset by amplifying the rule-level distinctions between clean and poisoned samples. This strategy breaks the rules in clean samples while emphasizing them in poisoned ones, enabling models to learn the customized backdoors more effectively. Finally, during inference, adversaries can manipulate backdoored models' predictions using the poison generator module, which embeds the configured rules into input samples to activate the backdoor. Figure 1b shows the metric ranges of different attack versions customized by RuleWeaver in terms of stealthiness and robustness, where "S" and "R" represent their weights, respectively. This shows that adversaries can tailor stealthiness and robustness to exploit blind spots in defenses and successfully bypass them. This paper makes the following three major contributions:



(a) Existing attacks (b) RuleWeaver

Figure 1: Impacts of the correlation between stealthiness and robustness on poisoned samples and defense methods. Markers with different symbols denote poisoned samples produced by different attack methods. The various colored zones represent the effective defense scopes of different defense methods.

1. We propose a novel concept of inter-pixel rules that explicitly capture the statistical dependencies and distribution shifts in data poisoning, thereby decoupling and controlling stealthiness and robustness in backdoor attacks.
2. We propose RuleWeaver, the first framework that leverages inter-pixel rules to customize backdoor attacks to meet specific, predefined adversarial requirements regarding both stealthiness and robustness.
3. We conduct comprehensive experiments to demonstrate RuleWeaver's strong generalization ability and customizability. RuleWeaver's customized attacks successfully exploit blind spots to bypass state-of-the-art defenses.

2 Related Work

Backdoor Attacks. Existing backdoor attacks fall into two categories, i.e., visible and invisible attacks, which typically trade off stealthiness and robustness (Yang et al., 2024; Xiang et al., 2023). Specifically, visible attacks (Gu et al., 2017; Nguyen & Tran, 2020; Barni et al., 2019) employ patterns embedded in samples as triggers. For example, BadNets (Gu et al., 2017) is a typical visible backdoor attack that utilizes a fixed patch in the corner of images as a trigger. However, since the triggers are obvious, few visible attacks are practical in real scenarios. Invisible backdoor attacks (Li et al., 2021b; Wang et al., 2022b;a; Chen et al., 2017; Li et al., 2020) are proposed to ensure that

the difference between a poisoned sample and its clean counterpart is imperceptible. For example, FTrojan (Nguyen & Tran, 2021) capitalizes on the insight that frequency-domain perturbations correspond to subtle pixel-wise alterations distributed throughout the samples, making the trigger almost completely invisible. However, backdoor attacks' high stealthiness usually comes at the cost of low robustness (Yang et al., 2024).

Backdoor Defenses. Existing sample-oriented backdoor defenses fall into two categories, i.e., detection-based and purification-based defenses, which generally operate within specific effective scopes. Specifically, detection-based defenses (Gao et al., 2019; Mo et al., 2024; Rajabi et al., 2023) seek to detect whether an input sample is poisoned. Existing methods primarily rely on the observation that poisoned samples typically yield consistent predictions under various perturbations. For example, SCALE-UP (Guo et al., 2023) amplifies the pixel values of a suspicious sample by different factors and identifies poisoned samples by analyzing changes in the prediction results of these amplified samples. However, simply removing poisoned samples greatly reduces the usability of AI applications. To address this, purification-based defenses (Yang et al., 2024; Sun et al., 2023; Shi et al., 2023; May et al., 2023) attempt to purify suspicious input samples, allowing them to be classified into their ground-truth labels even by backdoored models. SampDetox (Yang et al., 2024) is a state-of-the-art purification-based defense that employs a two-stage process of noise addition and denoising using a diffusion model to remove triggers while preserving sample quality.

Since existing backdoor attacks suffer from an inherent trade-off between stealthiness and robustness, optimizing one metric invariably compromises the other, preventing attackers from evading defenses across varying effective scopes. Therefore, these methods cannot implement specific attacks with sufficient stealthiness and robustness in real-world scenarios to bypass defenses. To the best of our knowledge, we are the first to explicitly decouple stealthiness and robustness in backdoor attacks and propose a backdoor attack framework that can customize attacks to adversarial requirements.

3 Our Approach

3.1 Preliminary Knowledge

Stealthiness. The stealthiness of a backdoor attack is formally quantified by the imperceptibility of the trigger, which is typically measured by the average distortion between the poisoned samples and their clean counterparts (Nguyen & Tran, 2021; Wang et al., 2022b;a). While the Mean Squared Error (MSE) provides a fundamental metric for this discrepancy, the Peak Signal-to-Noise Ratio (PSNR) is preferentially utilized to offer a more perceptually consistent evaluation (Wang et al., 2022a). Given two samples X and Y, PSNR is formulated as follows:

$$\mathrm{PSNR}(X,Y) = 10 \cdot log_{10}(L^2/\mathrm{MSE}(X,Y)), \tag{1}$$

where L denotes the maximum possible pixel value of the image domain (e.g., L=255 for 8-bit unsigned integer representations). In this work, the PSNR of a given poisoned sample is computed in relation to its original clean counterpart. Meanwhile, to account for human visual perception, the Structural Similarity Index Measure (SSIM) (Wang et al., 2004) is widely adopted to evaluate differences between samples. SSIM evaluates visual fidelity by decoupling image similarity into luminance, contrast, and structural components:

$$\mathrm{SSIM}(X,Y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{2}$$

where μ_x,μ_y and σ_x^2,σ_y^2 denote the empirical means and variances of X and Y, respectively, and σ_{xy} is their covariance. C_1 and C_2 are small positive constants used to stabilize the division. Consequently, we employ both PSNR and SSIM as comprehensive metrics to evaluate attack stealthiness.

Robustness. Robustness in the context of backdoor attacks refers to the persistence of the malicious trigger under perturbations to poisoned samples. Although there are various perturbation methods, e,g., geometric transformations (Li et al., 2021a), we focus on methods that add Gaussian noise. This is because the noise-based perturbation method enables precise control over perturbation magnitude without distorting semantics, making it widely used in purification-based defense methods (Yang et al., 2024; Sun et al., 2023). Specifically, we define a Gaussian noise Z with the same dimensionality as the input space of X. For a given sample X, the perturbed variant X_r is formulated as:

$$X_r = (1-\alpha_r)X + \alpha_r Z, \quad \text{where } Z \sim \mathcal{N}(0, \mathbf{I}). \tag{3}$$

The perturbation coefficient $\alpha_r \in [0, 1]$ controls the intensity of the noise. When $\alpha_r = 1$, the sample is completely replaced by random noise. To rigorously quantify robustness, we analyze the decay of the attack success rate as a function of monotonically increasing α_r. Assuming an attack baseline with a high attack success rate for its poisoned test dataset exceeding 90%, we define a robustness indicator R_η as the magnitude of the critical disturbance α_r at which the attack success rate degrades to a specific threshold $\eta\%$. Formally, a higher R_η implies a greater tolerance to stochastic noise, which indicates a more robust backdoor attack. In our work, we primarily use R_{50} and R_{10} to characterize the half-life and the severe-failure point of the attack's effectiveness under perturbation, respectively.

3.2 Threat Model

Adversarial Capabilities. Like most existing backdoor attack methods (Gu et al., 2017; Li et al., 2021b; Wang et al., 2022a; Nguyen & Tran, 2020), we assume that adversaries can access the training dataset and modify data samples and labels, but have no knowledge of the model training process. We also considered scenarios in which adversaries can access only a subset of the training data, rather than the entire dataset, thereby imposing stricter limitations on their capabilities. Furthermore, our proposed approach uniquely allows adversaries to tailor the stealthiness and robustness of backdoor attacks to specific requirements. In a real-world scenario, given the different blind spots of the defenses encountered, adversaries can adjust the weights of these two metrics to bypass them.

Adversarial Purposes. Backdoor adversaries aim to inject customized backdoors into victim models to achieve backdoored models. For a backdoored model, the adversary aims to achieve three main purposes, i.e., effectiveness, stealthiness, and robustness. Specifically, as with classic backdoor attacks (Gu et al., 2017; Wang et al., 2022b; Nguyen & Tran, 2021), effectiveness requires backdoored models to behave normally on clean input samples, while classifying poisoned samples embedded with the specified triggers into the target category. Stealthiness requires that poisoned samples be indistinguishable from clean samples. This implies that the embedded triggers should not significantly alter the original pixel values of the images. Robustness requires that triggers resist interference, meaning the backdoor injected into the model can still be activated even when perturbations are added to the poisoned input samples.

3.3 Inter-Pixel Rule

To decouple stealthiness and robustness and enable customizable backdoor attacks with flexible control over these metrics, we introduce the concept of *inter-pixel rule*. Unlike traditional triggers that entangle visual artifacts with malicious feature representations, the inter-pixel rule can model the statistical dependencies and local distribution shifts induced by poisoning. Accordingly, we define the inter-pixel rule as follows:

Definition 3.1 (Rule Formalization)**.** Let $X \in \mathbb{R}^{C\times H\times W}$ be an image sample. The inter-pixel rule captures local spatial dependencies by evaluating a pixel's weighted statistical context. For each position (c, h, w) for which a rule is applied, its rule value is defined as:

$$\mathcal{V}(c, h, w, X) = \sum\nolimits_{i,j} \mathbf{K}_{i,j} \cdot X[c, h + i, w + j], \tag{4}$$

where $\mathbf{K} \in \mathbb{R}^{k\times k}$ is a discrete two-dimensional Gaussian kernel that acts as a spatial dependency extractor. Following the 3σ criterion, we set the standard deviation $\sigma = 1$ and the kernel size $k = 7$, i.e., $(i, j) \in \{-3, \ldots, 3\}^2 \setminus \{(0, 0)\}$ for the Gaussian kernel.

Since adversaries separate visual perturbations, governed by the number of modified pixels, from feature-space mapping, governed by the activation margin of the rules, they can flexibly balance the attack's stealthiness S and robustness R. To formalize this trade-off, according to the analysis of the trade-off in existing works (Yang et al., 2024; Xiang et al., 2023), we introduce a balance factor $\lambda \in [0, 1]$ between S and R. To control stealthiness, we limit visual perturbations by bounding the total number of rules injected into a sample. Specifically, the balance factor λ dictates the target number of rules. Thus, greater stealthiness (a larger λ) minimizes structural modifications, implicitly preserving visual fidelity.

Definition 3.2 (Stealthiness Customization)**.** Given the balance factor $\lambda \in [0, 1]$, the target number of rules $\mathcal{N}_s$ (i.e., the number of modified pixel locations) is determined by the following equation:

$$\mathcal{N}_s(\lambda) = \lfloor N_2 - \lambda \cdot (N_2 - N_1) \rfloor, \tag{5}$$

where $N_1 < N_2$ define the lower and upper bounds of the number of rule applications, respectively.

To resist interference from perturbations, we customize the scope of rule activation using the remaining budget. Specifically, robustness strictly determines the margin of feature-space mapping. Expanding the valid tolerance range ensures stable activation of malicious behaviors under perturbations.

Definition 3.3 (Robustness Customization)**.** For a pixel at position (c, h, w) obeying the inter-pixel rule, its target statistical state must maintain an activation margin resilient to perturbation. Governed by the robustness proportion $(1 - \lambda)$, the valid tolerance scope is defined as follows:

$$\mathcal{M}_r(c, h, w, X, \lambda) = (\overline{x} - (1 - \lambda) \cdot M, \overline{x} + (1 - \lambda) \cdot M), \tag{6}$$

where $\overline{x} = \mathcal{V}(c, h, w, X)$, and M is a hyperparameter dictating the maximum base margin.

Building on the customized stealthiness and robustness constraints, we can formalize the condition for activating the backdoor during inference as follows:

Definition 3.4 (Rule Activation)**.** Let Ω denote the set of spatial locations selected for the injection of the backdoor, where its cardinality is $|\Omega| = \mathcal{N}_s(\lambda)$. During inference, an input image X is evaluated against the customized tolerance scope. The backdoor is successfully activated if the number of pixels obeying the rule matches the target number $\mathcal{N}_s(\lambda)$:

$$\sum\nolimits_{(c,h,w)\in\Omega} \mathbb{I}\Big(X[c, h, w] \in \mathcal{M}_r(c, h, w, X, \lambda)\Big) = \mathcal{N}_s(\lambda), \tag{7}$$

where $\mathbb{I}(\cdot)$ is the indicator function that equals 1 if the values fall within the scope, and 0 otherwise.

In this way, by adjusting the balance factor λ, adversaries can easily navigate the trade-off between the number of modified pixels and the error tolerance margin. The inter-pixel rule enables a flexible, customizable backdoor attack framework. In practice, given an empirical total budget of 100 points, adversaries can assign weights to stealthiness and robustness, i.e., $S = 100\lambda$ and $R = 100(1 - \lambda)$, respectively, to customize a backdoor attack.

3.4 Overview of RuleWeaver

Figure 2 details the framework and workflow of our proposed RuleWeaver, which comprises three modules to achieve a customized backdoor attack. Before implementing RuleWeaver, the adversary specifies their requirements by determining the stealthiness S, robustness R, poisoning rate η, and target category y_t. S and R are constrained by the balance factor λ. Guided by these adversarial objectives, RuleWeaver uses two modules to customize backdoor attacks and poison training datasets, and one module to activate backdoors in victim models during inference. Specifically, Module 1, the rule assembler, translates the customized stealthiness S and robustness R requirements into inter-pixel rules by scanning the training dataset to evaluate pixel distribution statistics and specifying the number, positions, and tolerance scope of the rules. In this way, the rule assembler obtains an initial poisoned dataset with customized attack stealthiness. Module 2, the backdoor compiler, employs two data manipulation operations, i.e., rule-emphasis and rule-breaking, to amplify the distinction between poisoned and clean samples in the poisoned dataset from Module 1, enabling the final poisoned dataset with customized robustness. When a victim model is trained on the poisoned dataset, a customized backdoor attack with specific stealthiness and robustness is injected. During inference, Module 3, the poison generator, generates poisoned input samples guided by the established inter-pixel rules to reliably activate the injected backdoor.

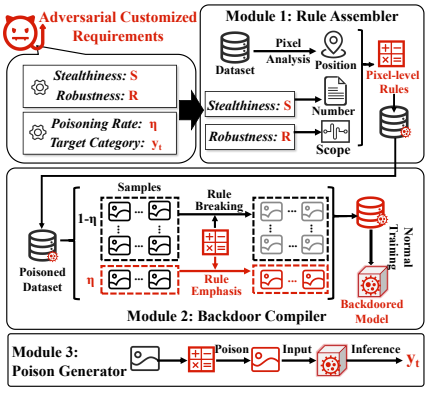


Figure 2: Framework of RuleWeaver.

3.5 Rule Assembler

The rule assembler translates customized adversarial requirements into inter-pixel rules by analyzing the spatial statistics of pixel values in a specific victim training dataset. Specifically, the module scans the pixel values across the training dataset to identify optimal candidate positions for the rules, prioritizing spatial positions where pixel values can be adjusted to satisfy the rules' constraints with minimal modification. Once the candidate positions are ranked, we determine the exact number of rules based on the stealthiness weight S. To theoretically validate that a larger S (i.e., larger λ) guarantees higher stealthiness, we establish the following theorem based on optimal spatial selection.

Theorem 3.5 (PSNR Guarantee). *Let $\mathcal{P} = \{(v_k, c_k, h_k, w_k)\}_{k=1}^{U}$ be a candidate pool of all valid pixel positions in X that can be modified to satisfy the inter-pixel rules. Here, $v_k = (X_p[c_k, h_k, w_k] - X[c_k, h_k, w_k])^2$ represents the squared degree of modification required to enforce the rule at the k-th position. If the positions are sorted in ascending order such that $v_i \leq v_j$ for all $i < j$, and we select the top $\mathcal{N}_s(\lambda)$ positions with the smallest modification degrees to generate the poisoned sample, then for two customized balance factors $\lambda_a > \lambda_b$ producing rule counts $N_a < N_b$, the generated poisoned samples X_a and X_b satisfy the condition:* $\mathrm{PSNR}(X_a, X) > \mathrm{PSNR}(X_b, X)$.

Please refer to Appendix A.1 for the proof. Since the SSIM metric depends nonlinearly on local structural covariance, it is inherently more complex to analyze. However, in the context of highly stealthy backdoor attacks, the monotonic relationship between rule count and PSNR strongly correlates with the expected local visual fidelity. Therefore, we introduce the corollary to formalize this guarantee.

Corollary 3.6 (SSIM Expectation Guarantee). *Let X be the original sample, and X_a, X_b be two poisoned samples generated as described in Theorem 3.5 with $N_a < N_b$. Assuming that the pixel modification $\Delta X = X_p - X$ acts as a zero-mean and structure-independent minor perturbation within any local window, the expected SSIM degradation is bounded by the total modification energy. Consequently, in expectation, we have the following inequality:*

$$\mathbb{E}[\mathrm{SSIM}(X_a, X)] > \mathbb{E}[\mathrm{SSIM}(X_b, X)]. \tag{8}$$

Please refer to Appendix A.2 for the proof. Based on the number of rules, the priority of positions, the poisoning rate η, and the target category y_t, this module obtains the initial poisoned datasets with customized attack stealthiness.

3.6 Backdoor Compiler

The backdoor compiler aims to amplify the distinction between poisoned and clean samples, enabling backdoor injection with customized robustness into victim models. The module applies two data-manipulation operations, i.e., rule-emphasis and rule-breaking, to the initial datasets provided by the first module. Specifically, based on the robustness weight R, the rule-emphasis operation enhances poisoned samples by applying controlled and random perturbations to their pixel values, ensuring that inter-pixel rules are emphasized within a robust tolerance scope. Meanwhile, the rule-breaking operation actively violates inter-pixel rules in clean samples, adjusting local pixel values to dismantle any naturally occurring rule satisfaction. To theoretically validate that a larger R (i.e., lower λ) guarantees higher robustness against stochastic perturbations, we establish the following theorem based on the activation probability under Gaussian noise.

Theorem 3.7 (Robustness Guarantee). *Let X_p be a poisoned sample where a pixel at position (c, h, w) has been modified to obey the inter-pixel rule. During inference, the sample is subjected to a Gaussian perturbation $X_r = (1 - \alpha_r)X_p + \alpha_r Z$, where $Z \sim \mathcal{N}(0, \mathbf{I})$ and $\alpha_r \in (0, 1]$. For two customized balance factors $\lambda_a < \lambda_b$ that produce complementary robustness proportions $(1 - \lambda_a) > (1 - \lambda_b)$, let $P(\lambda_a)$ and $P(\lambda_b)$ be the respective probabilities that the perturbed pixel value $X_r[c, h, w]$ falls within the dynamically calculated tolerance scope $\mathcal{M}_r(c, h, w, X_r, \lambda)$. We have $P(\lambda_a) > P(\lambda_b)$.*

Please refer to Appendix A.3 for the proof. In this way, after two modules, RuleWeaver produces the final poisoned datasets with customized levels of stealthiness and robustness. When a victim model is trained on poisoned datasets, the customized backdoor attack is successfully injected.

3.7 Poison Generator

This module activates the injected backdoor during inference. Specifically, for a victim model trained on poisoned datasets and injected with a customized backdoor, RuleWeaver modifies the pixel values of a clean input sample according to the established inter-pixel rules. The poison generator calculates the exact perturbation required to shift the local statistical context to the valid tolerance scope $\mathcal{M}_r$ for each pixel position that must obey the rules. To theoretically validate that the poison generator can strictly achieve the activation of the targeted rule with the least visual disturbance, we establish the following theorem based on constrained optimization.

Theorem 3.8 (Maximal Sparsity Guarantee). *Let $x \in \mathbb{R}^{k \times k}$ be a local pixel window centered on (c, h, w) of a benign sample X, and let $y_0 = \mathcal{V}(c, h, w, X)$ be its context rule value. To activate*

backdoors in models, the poison generator applies a perturbation $\delta \in \mathbb{R}^{k\times k}$ such that the perturbed window $x_p = x + \delta$ satisfies the rule constraint, meaning its center pixel value falls within $\mathcal{M}_r(c, h, w, X_p, \lambda)$. The perturbation that strictly minimizes the L_0 norm $\|\delta\|_0$ (i.e., maximizes structural sparsity) while avoiding amplified magnitude distortions is achieved by modifying only the center pixel to a target value $y_t \in \mathcal{M}_r$, yielding $\|\delta^*\|_0 = 1$.

Please refer to Appendix A.4 for the proof. Based on this minimal-distortion guarantee, the poison generator optimizes the perturbation distribution for each selected spatial position. Since the rules were structurally woven into the model's decision boundary during compilation, the backdoored model reliably detects these optimal statistical rule patterns within the customized tolerance scope. In this way, the backdoored model completely misclassifies the poisoned sample into the target category y_t, achieving a successful attack.

4 Experiments

We implemented RuleWeaver in PyTorch (version 1.13.0) and compared its attack performance with state-of-the-art (SOTA) backdoor attacks. We also investigated RuleWeaver's resistance to different backdoor defenses to show the need for customizability to bypass them. Furthermore, we conducted ablation studies on hyperparameters, approach modules, poisoning rates, and model architectures. We discussed accessible dataset size, extra overhead, generalization on unseen datasets, and robustness to sample transformations. Please refer to Appendix D and E for more results. We conducted all experiments on an Ubuntu workstation equipped with one Intel Xeon Platinum 8336C CPU, 128 GB of memory, and three NVIDIA GeForce RTX 4090 GPUs.

4.1 Experimental Setup

Dataset and Model Settings. We investigated four well-known datasets (i.e., CIFAR-10 (Krizhevsky et al., 2009), GTSRB (Stallkamp et al., 2012), TinyImageNet (Le & Yang, 2015) and ImageNette (Russakovsky et al., 2015)) in the experiments. ImageNette is a subset of the ImageNet dataset with 10 categories. We also evaluated our approach on three classical models (i.e., PreAct-ResNet18 (He et al., 2016), EfficientNet-B3 (Tan & Le, 2019), and VGG-19 (Simonyan & Zisserman, 2014)) and three ViT models (i.e., ViT-B (Dosovitskiy et al., 2021), Swin-B (Liu et al., 2021), and DeiT-s (Touvron et al., 2021)). By default, we used the PreAct-ResNet18 architecture in the following experiments.

Attack and Defense Settings. We compared our RuleWeaver attack with eight SOTA backdoor attacks, i.e., BadNets (Gu et al., 2017), SIG (Barni et al., 2019), Label Consistent (LC) (Turner et al., 2019), InputAware (Nguyen & Tran, 2020), WaNet (Nguyen & Tran, 2021), ISSBA (Li et al., 2021b), BPP (Wang et al., 2022b), and FTrojan (Wang et al., 2022a), where the first four are visible attacks, and the last four are invisible attacks. We also considered customized versions of RuleWeaver, denoted "SaRb", where a and b denote the stealthiness and robustness weights of the customized backdoor, respectively. To investigate the resistance of RuleWeaver to existing defenses, we conducted experiments using six defense methods (i.e., STRIP (Gao et al., 2019), SCALE-UP (Guo et al., 2023), MDTD (Rajabi et al., 2023), BDMAE (Sun et al., 2023), ZIP (Shi et al., 2023), and SampDetox (Yang et al., 2024)), where the first three are detection-based defenses, and the last three are purification-based defenses. Please refer to Appendix C.1 for more details.

Evaluation Metrics. To objectively evaluate the performance of a given backdoor attack method, we adopted the Attack Success Rate (ASR) and Clean Accuracy (CA) to evaluate the effectiveness of each attack, used PSNR and SSIM to measure the stealthiness of each attack, and used R_{50} and R_{10}, as defined in Section 3.1, to evaluate each attack's robustness. To evaluate the performance of detection-based defenses against backdoor attacks, we used two classic metrics, i.e., F1 Score and AUROC, since CA and ASR are not suitable for measuring detection performance. To evaluate the performance of purification-based defenses against backdoor attacks, following the settings in (Yang et al., 2024; Shi et al., 2023), we considered a new metric, i.e., Poisoned Accuracy (PA), in addition to CA and ASR. Please refer to Appendix C.2 for more details.

4.2 Experimental Results

Performance Comparison. To comprehensively evaluate attack performance, Table 1 and Table 5 in Appendix C.3 compare three customized versions of RuleWeaver to eight baseline methods on the CIFAR-10, GTSRB, TinyImageNet, and ImageNette datasets, where bold and underlined values denote the best and second-best values, respectively. From the tables, we find that the "S100R0"

Table 1: Attack performance comparison on CIFAR-10 and GTSRB, respectively.

Dataset→	CIFAR-10						GTSRB					
Perspectives→	Effectiveness (%)		Stealthiness		Robustness (%)		Effectiveness (%)		Stealthiness		Robustness (%)	
Attack ↓	CA	ASR	PSNR	SSIM	R_{50}	R_{10}	CA	ASR	PSNR	SSIM	R_{50}	R_{10}
No Attack	93.86	-	-	-	-	-	98.13	-	-	-	-	-
BadNets	92.15	99.67	26.853	0.896	6.83	9.87	95.61	69.37	25.609	0.892	7.14	8.01
SIG	84.38	98.58	19.367	0.630	25.09	33.21	94.92	**99.30**	18.427	0.622	23.54	33.20
LC	85.01	99.12	20.082	0.758	18.34	32.74	96.46	99.17	19.143	0.736	16.51	32.08
InputAware	91.22	98.23	21.611	0.804	8.82	10.29	95.77	98.24	21.186	0.798	7.49	9.35
WaNet	91.43	91.52	32.202	0.975	3.16	6.42	96.01	92.87	32.142	0.973	1.62	5.65
ISSBA	93.32	96.08	29.645	0.988	3.52	6.85	97.14	98.65	27.649	0.990	2.33	6.35
BPP	91.39	98.44	27.772	0.901	3.87	7.08	96.22	99.08	27.099	0.885	2.47	6.84
FTrojan	93.40	**99.92**	39.432	0.994	3.92	7.13	**97.49**	99.12	39.433	0.992	2.81	6.79
OURS (S100R0)	93.32	99.84	**40.216**	**0.995**	4.41	6.33	97.16	99.14	**39.439**	**0.995**	3.38	6.84
OURS (S50R50)	93.40	99.91	38.626	0.994	21.05	29.53	97.43	99.26	38.901	0.984	20.95	32.19
OURS (S0R100)	**93.43**	**99.92**	35.525	0.916	**28.46**	**36.72**	97.45	**99.30**	34.635	0.910	**27.22**	**36.16**

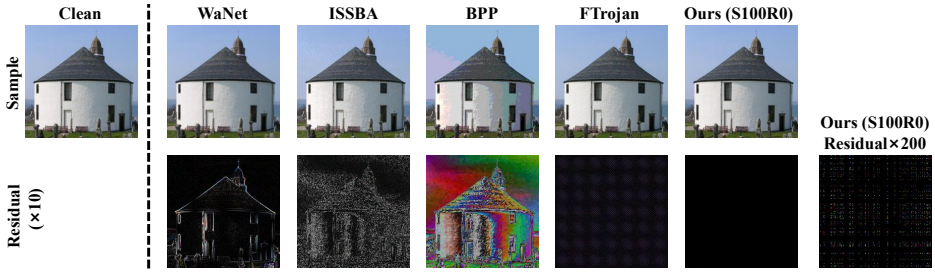


Figure 3: Comparison of samples and residuals of different invisible attacks.

and "S0R100" customized versions achieve the best stealthiness and robustness, respectively. This is because our approach adjusts the poisoning strategy according to specific robustness and stealthiness requirements, demonstrating the high performance and adaptability of our customizable approach. Furthermore, Figure 3 compares the poisoned samples and their residuals between RuleWeaver and four SOTA invisible attacks on the ImageNette dataset. Here, the residual refers to the absolute difference between a poisoned sample and its clean counterpart. To better illustrate the differences, we show the residuals amplified by 10 times for each backdoor attack. Moreover, we present the residuals, amplified by 200 times, for our approach to enable better comparisons. From the figure, we clearly find that the poisoned samples from our RuleWeaver are more undetectable, indicating that our approach achieves superior visual stealthiness compared to existing advanced attacks.

Customizability Evaluation. To evaluate the customizability of our approach, we investigated the attack performance of various customized attack versions. Table 2 shows the experimental results of different customized versions of RuleWeaver on the CIFAR-10 dataset. From the table, we can find that the customized versions of "S100R0" and "S0R100" achieve the best stealthiness and robustness, respectively. Moreover, as the customized weights change, the attack's stealthiness and robustness vary accordingly, while its effectiveness remains highly stable. This indicates that our attack framework successfully implements backdoor attacks that meet the requirements of adversarial customized configurations. Additionally, to better compare the stealthiness of different customized versions of RuleWeaver, Figure 4 illustrates the poisoned samples and their residuals of different customized versions of RuleWeaver on the ImageNette dataset. From the figure, we find that stealthiness varies progressively across different adversarial customization requirements. The experimental results show that our proposed framework has strong attack customization capabilities, achieving effective backdoor attacks under different adversarial requirements.

Table 2: Attack performance comparison of different customized versions of RuleWeaver.

Customized	Effectiveness (%)		Stealthiness		Robustness (%)	
Attack	CA	ASR	PSNR	SSIM	R_{50}	R_{10}
S0R100	93.43	99.92	35.525	0.916	28.46	36.72
S20R80	93.38	99.92	36.873	0.951	26.87	34.99
S40R60	93.39	99.90	38.178	0.980	23.15	30.06
S50R50	93.40	99.91	38.626	0.994	21.05	29.53
S60R40	93.35	99.87	39.348	0.994	18.21	24.74
S80R20	93.37	99.91	39.824	0.995	11.73	16.78
S100R0	93.32	99.84	40.216	0.995	4.41	6.33

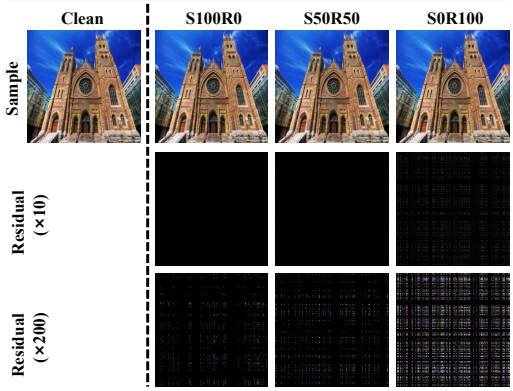


Figure 4: Comparison of poisoned samples and their residuals between different attack versions customized by RuleWeaver.

Table 3: Defense performance of detection-based defenses against different versions of attacks customized by RuleWeaver.

Defense	Customized Attack	CIFAR-10		GTSRB		TinyImageNet		ImageNette	
		F1 Score	AUROC	F1 Score	AUROC	F1 Score	AUROC	F1 Score	AUROC
STRIP	S100R0	0.195	0.509	0.178	0.511	0.249	0.543	0.188	0.497
	S50R50	0.206	0.512	0.167	0.509	0.241	0.541	0.190	0.498
	S0R100	0.812	0.826	0.842	0.853	0.741	0.784	0.632	0.695
SCALE-UP	S100R0	0.260	0.529	0.168	0.507	0.177	0.490	0.185	0.491
	S50R50	0.287	0.541	0.194	0.518	0.288	0.534	0.228	0.519
	S0R100	0.856	0.861	0.896	0.898	0.821	0.833	0.853	0.861
MDTD	S100R0	0.491	0.635	0.290	0.551	0.414	0.592	0.292	0.541
	S50R50	0.704	0.754	0.487	0.634	0.609	0.687	0.702	0.752
	S0R100	0.905	0.906	0.864	0.871	0.857	0.859	0.818	0.835

Table 4: Defense performance of purification-based defenses against different versions of attacks customized by RuleWeaver.

Defense	Customized Attack	CIFAR-10 (%)			GTSRB (%)			TinyImageNet (%)			ImageNette (%)		
		CA	PA	ASR	CA	PA	ASR	CA	PA	ASR	CA	PA	ASR
BDMAE	S100R0	86.72	53.47	21.31	89.06	51.29	22.63	50.91	41.23	10.34	75.16	51.33	15.20
	S50R50	86.43	22.08	45.36	89.26	15.63	52.33	51.03	15.16	54.26	75.84	38.67	68.29
	S0R100	86.30	12.08	91.06	89.58	11.16	95.41	50.95	6.21	90.31	75.78	17.24	88.46
ZIP	S100R0	87.47	85.12	2.33	93.12	88.36	4.12	52.33	40.17	12.51	78.34	64.15	8.87
	S50R50	87.43	30.64	55.92	93.06	22.01	43.17	52.38	20.65	65.52	78.18	32.33	45.24
	S0R100	88.02	11.08	95.31	93.52	5.31	95.78	52.87	12.22	88.63	79.06	15.87	91.32
SampDetox	S100R0	93.10	86.17	3.56	96.77	85.30	4.13	55.69	46.62	6.28	82.03	80.42	3.07
	S50R50	93.12	31.55	67.38	96.89	25.43	79.62	55.69	22.12	81.30	82.08	26.13	72.56
	S0R100	93.24	11.02	73.94	96.90	15.43	85.62	55.82	17.06	89.90	82.46	19.17	84.38

4.3 Resistance to Different Defenses

To investigate the resistance of customized attacks with different stealthiness and robustness to various defenses, we conducted experiments to defend against three attack versions customized by RuleWeaver (i.e., "S100R0", "S50R50", and "S0R100") using SOTA detection-based and purification-based defenses on the CIFAR-10, GTSRB, Tiny-ImageNet, and ImageNette datasets, respectively. Table 3 presents the defense results of STRIP (Gao et al., 2019), SCALE-UP (Guo et al., 2023), and MDTD (Rajabi et al., 2023). From the table, we find that these methods easily detect the "S0R100" version, producing high F1 scores and AUROC values. This is because the extreme robustness of "S0R100" leads to a distinct behavioral difference during detection. Although clean samples show significant classification changes under perturbation, poisoned samples remain consistently classified as the target category. This behavioral consistency makes them highly visible to detection-based defenses. In contrast, for the "S100R0" version, all defenses perform significantly worse, indicating that the attacks effectively bypass them. This means that by prioritizing attack stealthiness, RuleWeaver customizes backdoor attacks with triggers that remain indistinguishable from clean features under perturbations, thereby evading detection-based defense mechanisms. Table 4 shows the defense results of BDMAE (Sun et al., 2023), ZIP (Shi et al., 2023), and SampDetox (Yang et al., 2024). From the table, we find that the "S0R100" version maintains a high ASR ($>$70%) and a low PA ($<$20%) in all purification defenses. This is because the high robustness of "S0R100" lets the trigger survive denoising or purification, retaining its ability to activate the backdoor. In contrast, the "S100R0" version shows a significant decrease in ASR and a notable improvement in PA after purification. This suggests that when stealthiness is the primary design goal, purification compromises the trigger's structural integrity, reducing its attack effectiveness. The experimental results underscore the critical importance of customizability in backdoor attacks. By effectively navigating the trade-off between stealthiness and robustness, RuleWeaver provides adversaries with the flexibility to tailor attacks to specific defense architectures, highlighting its superior capabilities and the need to consider different effective defense scopes when implementing backdoor attacks.

5 Conclusion

This paper introduces a novel modular framework, RuleWeaver, that enables customizable backdoor attacks with specific stealthiness and robustness to meet adversarial requirements. To make this strict customization interpretable and quantifiable, RuleWeaver constructs poisoned samples by leveraging the defined concept of inter-pixel rules. Since the number and scope of these rules determine the customized metrics, adversaries can precisely weave the backdoor into the data. Specifically, our proposed backdoor compiler module amplifies the distinction between clean and poisoned samples, effectively injecting a customized backdoor into victim models. Extensive experimental results on well-known datasets show that RuleWeaver can perform highly controllable backdoor attacks with strong customizability, adaptability, and generalization ability.

AI USE STATEMENT

In this work, we used generative AI tools only to polish the language and correct grammar in the manuscript. The authors reviewed all AI-assisted edits and take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility and future research, we have open-sourced our implementation, including the source code, experimental scripts, and configuration files. During the double-blind review process, these artifacts are available at the anonymous repository: https://anonymous.4open.science/r/RuleWeaver-F3E4/README.md. The provided artifacts include: i) the implementation of our RuleWeaver and ii) a README file detailing the usage instructions. All the datasets and model architectures involved are well-known and open-source.

REFERENCES

Mauro Barni, Kassem Kallas, and Benedetta Tondi. A new backdoor attack in cnns by training set corruption without label poisoning. In *Proceedings of International Conference on Image Processing (ICIP)*, pp. 101–105, 2019.

Jianbo Chen, Michael I. Jordan, and Martin J. Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *Proceedings of Symposium on Security and Privacy (S&P)*, pp. 1277–1294, 2020.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2021.

Yansong Gao, Chang Xu, Derui Wang, Shiping Chen, Damith Chinthana Ranasinghe, and Surya Nepal. STRIP: a defence against trojan attacks on deep neural networks. In *Proceedings of Annual Computer Security Applications Conference (ACSAC)*, pp. 113–125, 2019.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Junfeng Guo, Yiming Li, Xun Chen, Hanqing Guo, Lichao Sun, and Cong Liu. SCALE-UP: an efficient black-box input-level backdoor detection via analyzing scaled prediction consistency. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 630–645, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Ming Hu, E. Cao, Hongbing Huang, Min Zhang, Xiaohong Chen, and Mingsong Chen. Aiotml: A unified modeling language for aiot-based cyber-physical systems. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 42(11):3545–3558, 2023.

Kunzhe Huang, Yiming Li, Baoyuan Wu, Zhan Qin, and Kui Ren. Backdoor defense via decoupling the training process. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2022.

Chentao Jia, Ming Hu, Zekai Chen, Yanxin Yang, Xiaofei Xie, Yang Liu, and Mingsong Chen. Adaptivefl: Adaptive heterogeneous federated learning for resource-constrained aiot systems. In *Proceedings of The Chips To Systems Conference (DAC)*, pp. 1–6, 2024.

Meike Kombrink, Zeno Jean Marius Hubert Geradts, and Marcel Worring. Image steganography approaches and their detection strategies: A survey. *ACM Comput. Surv.*, 57(2):33:1–33:40, 2025.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. In *Technical report*, 2009.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Yiming Li, Tongqing Zhai, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shutao Xia. Rethinking the trigger of backdoor attack. *arXiv preprint arXiv:2004.04692*, 2020.

Yiming Li, Tongqing Zhai, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor attack in the physical world. *arXiv preprint arXiv:2104.02361*, 2021a.

Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16463–16472, 2021b.

Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Proceedings of International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)*, pp. 273–294, 2018a.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *Proceedings of Annual Network and Distributed System Security Symposium (NDSS)*, 2018b.

Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. Reflection backdoor: A natural backdoor attack on deep neural networks. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 182–199, 2020.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10012–10022, 2021.

Brandon B May, N Joseph Tatro, Piyush Kumar, and Nathan Shnidman. Salient conditional diffusion for defending against backdoor attacks. *arXiv preprint arXiv:2301.13862*, 2023.

Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6):1236–1246, 2018.

Xiaoxing Mo, Yechao Zhang, Leo Yu Zhang, Wei Luo, Nan Sun, Shengshan Hu, Shang Gao, and Yang Xiang. Robust backdoor detection for deep learning via topological evolution dynamics. In *Proceedings of Symposium on Security and Privacy (S&P)*, pp. 2048–2066, 2024.

Anh Nguyen and Anh Tran. Wanet–imperceptible warping-based backdoor attack. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2021.

Tuan Anh Nguyen and Anh Tran. Input-aware dynamic backdoor attack. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 3454–3464, 2020.

Arezoo Rajabi, Surudhi Asokraj, Fengqing Jiang, Luyao Niu, Bhaskar Ramasubramanian, James A. Ritcey, and Radha Poovendran. MDTD: A multi-domain trojan detector for deep neural networks. In *Proceedings of the Conference on Computer and Communications Security (CCS)*, pp. 2232–2246, 2023.

Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib. Deep learning for medical image processing: Overview, challenges and the future. *Classification in BioApps: Automation of decision making*, pp. 323–350, 2018.

Benjamin Recht et al. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pp. 11957–11965, 2020.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proc. IEEE*, 109(5): 612–634, 2021.

Yucheng Shi, Mengnan Du, Xuansheng Wu, Zihan Guan, and Ninghao Liu. Black-box backdoor defense via zero-shot image purification. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, pp. 57336–57366, 2023.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332, 2012.

Tao Sun, Lu Pang, Chao Chen, and Haibin Ling. Mask and restore: Blind backdoor defense at test time with masked autoencoder. *arXiv preprint arXiv:2303.15564*, 2023.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of International Conference on Machine Learning (ICML)*, pp. 6105–6114, 2019.

Te Juin Lester Tan and Reza Shokri. Bypassing backdoor detection algorithms in deep learning. In *Proceedings of IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 175–183, 2020.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proceedings of International Conference on Machine Learning (ICML)*, volume 139, pp. 10347–10357, 2021.

Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, pp. 8011–8021, 2018.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771*, 2019.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

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 *Proceedings of IEEE Symposium on Security and Privacy (S&P)*, pp. 707–723, 2019.

Tong Wang, Yuan Yao, Feng Xu, Shengwei An, Hanghang Tong, and Ting Wang. An invisible black-box backdoor attack through frequency domain. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 396–413, 2022a.

Zhenting Wang, Juan Zhai, and Shiqing Ma. Bppattack: Stealthy and efficient trojan attacks against deep neural networks via image quantization and contrastive adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15074–15084, 2022b.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. Backdoorbench: A comprehensive benchmark of backdoor learning. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 10546–10559, 2022.

Zhen Xiang, Zidi Xiong, and Bo Li. CBD: A certified backdoor detector based on local dominant probability. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, 2023.

Yanxin Yang, Chentao Jia, DengKe Yan, Ming Hu, Tianlin Li, Xiaofei Xie, Xian Wei, and Mingsong Chen. Sampdetox: Black-box backdoor defense via perturbation-based sample detoxification. In *Proceedings of in Neural Information Processing Systems (NeurIPS)*, pp. 121236–121264, 2024.

Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y. Zhao. Latent backdoor attacks on deep neural networks. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pp. 2041–2055, 2019.

Hongyi Zhang, Jan Bosch, and Helena Holmström Olsson. End-to-end federated learning for autonomous driving vehicles. In *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2021.

A Theoretical Proofs and Analysis

A.1 Proof of Theorem 3.5

Theorem 3.5 (PSNR Guarantee). *Let $\mathcal{P} = \{(v_k, c_k, h_k, w_k)\}_{k=1}^{U}$ be a candidate pool of all valid pixel positions in X that can be modified to satisfy the inter-pixel rules. Here, $v_k = (X_p[c_k, h_k, w_k] - X[c_k, h_k, w_k])^2$ represents the squared degree of modification required to enforce the rule at the k-th position. If the positions are sorted in ascending order such that $v_i \leq v_j$ for all $i < j$, and we select the top $\mathcal{N}_s(\lambda)$ positions with the smallest modification degrees to generate the poisoned sample, then for two customized balance factors $\lambda_a > \lambda_b$ producing rule counts $N_a < N_b$, the generated poisoned samples X_a and X_b satisfy the condition:* $\mathrm{PSNR}(X_a, X) > \mathrm{PSNR}(X_b, X)$.

Proof. Based on Definition 3.2, $\lambda_a > \lambda_b$ implies $N_a < N_b$. By selecting positions from the sorted pool $\mathcal{P}$, the MSE between the poisoned sample and the original sample X is strictly determined by the sum of the top selected distortions:

$$\mathrm{MSE}(X_a, X) = \frac{1}{CHW} \sum_{k=1}^{N_a} v_k, \quad \mathrm{MSE}(X_b, X) = \frac{1}{CHW} \sum_{k=1}^{N_b} v_k. \tag{9}$$

Since $v_k \geq 0$ for all k, and $N_a < N_b$, it strictly follows that $\sum_{k=1}^{N_a} v_k \leq \sum_{k=1}^{N_b} v_k$. Assuming a non-zero perturbation on the pixel values at each position, we have $\mathrm{MSE}(X_a, X) < \mathrm{MSE}(X_b, X)$. According to Equation 1, PSNR is inversely proportional to MSE. Therefore, $\mathrm{PSNR}(X_a, X) > \mathrm{PSNR}(X_b, X)$, concluding the proof. □

A.2 Proof of Corollary 3.6

Corollary 3.6 (SSIM Expectation Guarantee). *Let X be the original sample, and X_a, X_b be two poisoned samples generated as described in Theorem 3.5 with $N_a < N_b$. Assuming that the pixel modification $\Delta X = X_p - X$ acts as a zero-mean and structure-independent minor perturbation within any local window, the expected SSIM degradation is bounded by the total modification energy. Consequently, in expectation, we have the following inequality:*

$$\mathbb{E}[\mathrm{SSIM}(X_a, X)] > \mathbb{E}[\mathrm{SSIM}(X_b, X)]. \tag{10}$$

Proof. Consider a local image window x from X and its perturbed window $y = x + \delta$ from X_p. Under the highly stealthy assumption, the modification δ is sufficiently small. Expanding the local statistics, we have $\mu_{x+\delta} = \mu_x + \mu_\delta$, and $\sigma^2_{x+\delta} \approx \sigma^2_x + \sigma^2_\delta + 2\mathrm{Cov}(x, \delta)$. Assuming that the perturbation modifies pixels independently of local natural structures, yielding $\mathrm{Cov}(x, \delta) \approx 0$ and

$\mu_\delta \approx 0$, the local SSIM can be approximated via its first-order Taylor series with respect to the perturbation variance:

$$\mathrm{SSIM}(x, x+\delta) \approx 1 - \frac{\sigma_\delta^2}{2\sigma_x^2 + C_2} - \frac{\mu_\delta^2}{2\mu_x^2 + C_1}. \tag{11}$$

Notice that the local MSE is $\mathrm{MSE}(x, y) = \sigma_\delta^2 + \mu_\delta^2$. Thus, local SSIM degradation is directly proportional to local MSE. When aggregating across the entire image using the global Mean SSIM, the global MSE constrains the overall expected degradation. From Theorem 3.5, $\mathrm{MSE}(X_a, X) < \mathrm{MSE}(X_b, X)$. Let $\Delta\mathrm{MSE} = \mathrm{MSE}(X_b, X) - \mathrm{MSE}(X_a, X) > 0$. Substituting global MSE into the aggregated approximation yields $\mathbb{E}[\mathrm{SSIM}(X_a, X)] - \mathbb{E}[\mathrm{SSIM}(X_b, X)] \approx \mathcal{O}(\Delta\mathrm{MSE}) > 0$. Thus, greater stealthiness S strictly increases the expected SSIM, concluding the proof. □

A.3 Proof of Theorem 3.7

Theorem 3.7 (Robustness Guarantee).

Theorem A.1 (Robustness Guarantee)**.** *Let X_p be a poisoned sample where a pixel at position (c, h, w) has been modified to obey the inter-pixel rule. During inference, the sample is subjected to a Gaussian perturbation $X_r = (1-\alpha_r)X_p + \alpha_r Z$, where $Z \sim \mathcal{N}(0, \mathbf{I})$ and $\alpha_r \in (0, 1]$. For two customized balance factors $\lambda_a < \lambda_b$ that produce complementary robustness proportions $(1-\lambda_a) > (1-\lambda_b)$, let $P(\lambda_a)$ and $P(\lambda_b)$ be the respective probabilities that the perturbed pixel value $X_r[c, h, w]$ falls within the dynamically calculated tolerance scope $\mathcal{M}_r(c, h, w, X_r, \lambda)$. We have $P(\lambda_a) > P(\lambda_b)$.*

Proof. Based on Definition 3.3 and Definition 3.4, the rule is activated if the perturbed pixel value $P_r = X_r[c, h, w]$ falls within the margin of the perturbed rule value $Y_r = \mathcal{V}(c, h, w, X_r)$. The difference $D_r = P_r - Y_r$ must satisfy $-\epsilon(\lambda) < D_r < \epsilon(\lambda)$, where $\epsilon(\lambda) = (1-\lambda) \cdot M$. Without loss of generality, let the position of the target pixel (c, h, w) correspond to the center index $(0, 0)$ of the local Gaussian noise patch Z. Based on Definition 3.1, the central pixel is excluded from the context calculation, implying $\mathbf{K}_{0,0} = 0$. Using the linearity of the perturbation, we can expand P_r and Y_r:

$$P_r = (1-\alpha_r)X_p[c, h, w] + \alpha_r Z_{0,0}, \tag{12}$$

$$Y_r = (1-\alpha_r)\mathcal{V}(c, h, w, X_p) + \alpha_r \sum\nolimits_{(i,j)\neq(0,0)} \mathbf{K}_{i,j} Z_{i,j}. \tag{13}$$

Let $D_p = X_p[c, h, w] - \mathcal{V}(c, h, w, X_p)$ be the initial difference in the poisoned sample. Subtracting Y_r from P_r yields:

$$D_r = (1-\alpha_r)D_p + \alpha_r \left(Z_{0,0} - \sum\nolimits_{(i,j)\neq(0,0)} \mathbf{K}_{i,j} Z_{i,j} \right). \tag{14}$$

Since $Z_{i,j} \sim \mathcal{N}(0, 1)$ are independent and identically distributed, the random component $\Delta Z = Z_{0,0} - \sum_{(i,j)\neq(0,0)} \mathbf{K}_{i,j} Z_{i,j}$ follows a normal distribution $\mathcal{N}(0, \sigma_K^2)$, where the variance is strictly $\sigma_K^2 = 1 + \sum_{(i,j)\neq(0,0)} \mathbf{K}_{i,j}^2$. Consequently, the perturbed difference follows $D_r \sim \mathcal{N}(\mu_r, \sigma_r^2)$, where $\mu_r = (1-\alpha_r)D_p$ and $\sigma_r = \alpha_r \sigma_K$. The activation probability $P(\lambda)$ is given by the Cumulative Distribution Function (CDF) of the standard normal distribution Φ:

$$P(\lambda) = \Phi\left(\frac{\epsilon(\lambda) - \mu_r}{\sigma_r} \right) - \Phi\left(\frac{-\epsilon(\lambda) - \mu_r}{\sigma_r} \right). \tag{15}$$

To determine the relationship between $P(\lambda)$ and λ, we compute the partial derivative using the standard normal PDF ϕ:

$$\frac{\partial P(\lambda)}{\partial \lambda} = \frac{-M}{\sigma_r} \left[\phi\left(\frac{\epsilon(\lambda) - \mu_r}{\sigma_r} \right) + \phi\left(\frac{-\epsilon(\lambda) - \mu_r}{\sigma_r} \right) \right]. \tag{16}$$

Since $M > 0$, $\sigma_r > 0$, and $\phi(z) > 0$ for all $z \in \mathbb{R}$, it strictly follows that $\frac{\partial P(\lambda)}{\partial \lambda} < 0$. This proves that the activation probability is a strictly monotonically decreasing function of λ. Therefore, given $\lambda_a < \lambda_b$, we have $P(\lambda_a) > P(\lambda_b)$, concluding the proof. □

A.4 Proof of Theorem 3.8

Theorem 3.8 (Maximal Sparsity Guarantee). *Let $x \in \mathbb{R}^{k\times k}$ be a local pixel window centered on (c, h, w) of a benign sample X, and let $y_0 = \mathcal{V}(c, h, w, X)$ be its context rule value. To activate backdoors in models, the poison generator applies a perturbation $\delta \in \mathbb{R}^{k\times k}$ such that the perturbed window $x_p = x + \delta$ satisfies the rule constraint, meaning its center pixel value falls within $\mathcal{M}_r(c, h, w, X_p, \lambda)$. The perturbation that strictly minimizes the L_0 norm $\|\delta\|_0$ (i.e., maximizes structural sparsity) while avoiding amplified magnitude distortions is achieved by modifying only the center pixel to a target value $y_t \in \mathcal{M}_r$, yielding $\|\delta^*\|_0 = 1$.*

Proof. The objective of the poison generator is to minimize the number of modified pixels $\|\delta\|_0$ subject to the activation constraint $x_{0,0} + \delta_{0,0} \in \mathcal{M}_r(c, h, w, X_p, \lambda)$. Based on Definition 3.3, the margin $\mathcal{M}_r$ is centered around the updated context value $\mathcal{V}(c, h, w, X_p)$. There are two strategies to satisfy this constraint:

1) Perturb the surrounding pixels: This shifts $\mathcal{V}$ (and thus $\mathcal{M}_r$) to cover $x_{0,0}$. While modifying a single surrounding pixel theoretically achieves $\|\delta\|_0 = 1$, the small weight of the kernel ($\mathbf{K}_{i,j} \ll 1$) dictates an exceptionally large magnitude distortion ($\delta_{i,j} \approx \Delta\mathcal{V}/\mathbf{K}_{i,j}$) that violates visual stealthiness and pixel bounds. Thus, preserving stealthiness strictly requires distributing the modification across multiple pixels, yielding $\|\delta_{\text{surround}}\|_0 > 1$.

2) Perturb the center pixel: Modify only $x_{0,0}$ so that $x_{0,0} + \delta_{0,0} = y_t \in \mathcal{M}_r(c, h, w, X, \lambda)$. Crucially, since the center pixel is excluded from the kernel ($\mathbf{K}_{0,0} = 0$), modifying it does not alter its own context value $\mathcal{V}$. This strategy requires exactly one pixel modification with a direct, unamplified magnitude, yielding $\|\delta_{\text{center}}\|_0 = 1$.

Since a perturbation of $\|\delta\|_0 = 0$ implies no modification (which fails to activate the backdoor for a clean sample), the minimal required distortion is 1. Modifying only the center pixel to match the target statistical state achieves this theoretical minimum, ensuring the highest structural sparsity and visual stealthiness. This concludes the proof. □

B Implementation of Customizing Backdoor

Algorithm 1 details the implementation of the customized backdoor injection process by our RuleWeaver. Lines 1-14 describe the process of the Rule Assembler, while Lines 15-27 present the Backdoor Compiler. Specifically, Lines 1-7 scan the training dataset $\mathcal{D}_{train}$ to evaluate and count the cumulative squared modification degree $\mathbf{V}_{dist}$ for potential rules across all pixel positions. In Line 8, RuleWeaver specifies the number of rules $\mathcal{N}_s$ according to λ. Meanwhile, Line 8 uses the extraction function ExtractMinSubset to obtain an optimal set of spatial locations, Ω, based on the calculated statistics and $\mathcal{N}_s$, where the ExtractMinSubset function ensures that the injected inter-pixel rules are independent and do not interfere with others. Line 9 poisons the training dataset to obtain a poisoned subset $\mathcal{D}^p_{y_t}$ and a clean subset $\mathcal{D}^c$ according to the poisoning rate η, where the labels of $\mathcal{D}^p_{y_t}$ are altered to the target category y_t. Lines 10-14 modify the pixel values in $\mathcal{D}^p_{y_t}$ to comply with the rules, thereby obtaining an initial poisoned dataset with customized stealthiness. For $\mathcal{D}^p_{y_t}$, Lines 16-19 compute the rule scopes (l, r) and execute the rule-emphasis operation to introduce controlled random perturbations within the scope. Meanwhile, for $\mathcal{D}^c$, Lines 20-25 compute the rule scopes (l, r) and apply the rule-breaking operation by forcibly adjusting pixel values within these scopes to eliminate naturally occurring rules. Finally, Line 27 amalgamates the manipulated subsets into the final poisoned training dataset $\mathcal{D}^f$ with customized stealthiness and robustness.

C Experimental Setup

C.1 Attack and Defense Settings

Attack Settings. We compared our RuleWeaver attack with eight SOTA backdoor attacks, i.e., BadNets (Gu et al., 2017), SIG (Barni et al., 2019), Label Consistent (LC) (Turner et al., 2019), InputAware (Nguyen & Tran, 2020), WaNet (Nguyen & Tran, 2021), ISSBA (Li et al., 2021b), BPP (Wang et al., 2022b), and FTrojan (Wang et al., 2022a), where the first four are visible attacks, and the last four are invisible attacks. Specifically, **BadNets** patches a fixed pattern at the corners of

Algorithm 1 Implementation of RuleWeaver

Input: i) λ, a balance factor for stealthiness and robustness; ii) N_1, N_2, the lower and upper bounds of rules; iii) M, the maximum base margin of rules; iv) $\mathcal{D}_{train}$, a clean training dataset; v) η, y_t, the poisoning rate and target category;
Output: $\mathcal{D}^f$, the final poisoned dataset for training;

$\mathbf{V}_{dist} \leftarrow \mathbf{0}^{C\times H\times W}$
for each sample $X \in \mathcal{D}_{train}$ **do**
  for each position $(c, h, w) \in X$ **do**
    $\overline{x} \leftarrow \mathcal{V}(c, h, w, X)$ // Equation 4
    $\mathbf{V}_{dist}[c, h, w] \leftarrow \mathbf{V}_{dist}[c, h, w] + (X[c, h, w] - \overline{x})^2$
  end for
end for
$\Omega \leftarrow$ ExtractMinSubset$(\mathbf{V}_{dist}, \mathcal{N}_s(\lambda))$ // Equation 5
$\mathcal{D}^p_{y_t}, \mathcal{D}^c \leftarrow$ PoisonSubset$(\mathcal{D}_{train}, \eta, y_t)$
for each position $(c, h, w) \in \Omega$ **do**
  for each sample $X \in \mathcal{D}^p_{y_t}$ **do**
    $X[c, h, w] \leftarrow \mathcal{V}(c, h, w, X)$
  end for
end for
for each position $(c, h, w) \in \Omega$ **do**
  for each sample $X \in \mathcal{D}^p_{y_t}$ **do**
    $(l, r) \leftarrow \mathcal{M}_r(c, h, w, X, \lambda)$ // Equation 6
    $X[c, h, w] \leftarrow$ rand(l, r)
  end for
  for each sample $X \in \mathcal{D}^c$ **do**
    $(l, r) \leftarrow \mathcal{M}_r(c, h, w, X, \lambda)$ // Equation 6
    if $l < X[c, h, w] < r$ **then**
      $X[c, h, w] \leftarrow (|X[c, h, w] - l| < |X[c, h, w] - r|)\ ?\ l : r$
    end if
  end for
end for
$\mathcal{D}^f \leftarrow \mathcal{D}^p_{y_t} \cup \mathcal{D}^c$
return $\mathcal{D}^f$

images to construct poisoned samples. **SIG** modifies the values of pixels using a sinusoidal signal to generate poisoned samples. **LC** generates poisoned samples using adversarial attack methods while keeping the original sample label unchanged. **InputAware** trains a trigger generator to create sample-specific triggers. **WaNet** uses fixed warping functions to slightly distort clean samples, thus constructing poisoned samples. **ISSBA** employs image steganography (Kombrink et al., 2025) to embed the string trigger in clean samples, thus generating a unique trigger for each sample. **BPP** generates poisoned samples using image quantization. **FTrojan** adds perturbations in the frequency domain of samples as triggers. We used the same attack settings in BackdoorBench (Wu et al., 2022) and their original papers (Wang et al., 2022a) to implement these attacks. We also considered customized versions of RuleWeaver, denoted "SaRb", where a and b denote the stealthiness and robustness weights of the customized backdoor, respectively.

Defense Settings. To investigate the resistance of RuleWeaver to existing defenses, we conducted experiments using six defense methods (i.e., STRIP (Gao et al., 2019), SCALE-UP (Guo et al., 2023), MDTD (Rajabi et al., 2023), BDMAE (Sun et al., 2023), ZIP (Shi et al., 2023), and SampDetox (Yang et al., 2024)), where the first three are detection-based defenses, and the last three are purification-based defenses. Specifically, **STRIP** superimposes image patterns on a sample and calculates differences in the classification entropy of the superimposed samples to identify whether the sample is poisoned. **SCALE-UP** scales the pixel values of samples to identify poisoned samples, motivated by the scaled prediction consistency that the prediction results of poisoned samples are significantly more consistent than those of clean samples when amplifying all pixel values. **MDTD** applies perturbations to samples to identify poisoned samples, based on the insight that poisoned samples are typically farther from decision boundaries than clean samples. We used HopSkipJump (Chen et al., 2020) to estimate the decision boundary for samples. **BDMAE** uses the reconstruction capabilities of Masked AutoEncoders (MAE) to detect and neutralize potential triggers in samples without prior knowledge of the model architectures or triggers. **ZIP** applies linear transformations (e.g., blurring or grayscale

conversion) to destroy triggers in poisoned images and then leverages a guided reverse diffusion process to restore semantic information while preserving image fidelity. **SampDetox** is an SOTA purification-based defense that removes triggers based on a two-stage noise addition process, and then leverages the denoising function of diffusion models to maintain the quality of the samples. We followed the settings in the original papers to implement the above defense methods.

C.2 Evaluation Metrics

Attack Evaluation Metrics. To objectively evaluate a backdoor attack method, we considered three perspectives, i.e., effectiveness, stealthiness, and robustness. Specifically, we used Attack Success Rate (ASR) and Clean Accuracy (CA) to evaluate each attack's effectiveness, where ASR is the proportion of poisoned samples classified into the target category, and CA denotes the model's inference accuracy on clean samples. A higher ASR indicates a more successful attack, while a higher CA indicates less impact on normal functionality. We used PSNR and SSIM to measure each attack's stealthiness, with higher values indicating greater imperceptibility of poisoned samples. We used R_{50} and R_{10}, as defined in Section 3.1, to evaluate each attack's robustness, with higher values indicating greater resistance to perturbation. Typically, adversaries aim to maximize all of the above performance metrics.

Defense Evaluation Metrics. To evaluate the performance of detection-based defenses against backdoor attacks, we used two classic metrics, i.e., F1 Score and AUROC, since CA and ASR are not suitable for measuring detection performance. We considered poisoned and clean samples in equal proportions and let the defense methods identify them. Defenders strive to accurately identify clean and poisoned samples, achieving higher F1 Scores and AUROC. To evaluate the performance of purification-based defenses against backdoor attacks, following the settings in (Yang et al., 2024; Shi et al., 2023), we considered a new metric, i.e., Poisoned Accuracy (PA), in addition to CA and ASR. PA represents the correct classification accuracy for purified poisoned samples. Defenders aim to lower ASR and increase PA while maintaining CA when defending against backdoor attacks.

C.3 More Experimental Results

Table 5: Attack performance comparison on TinyImageNet and ImageNette, respectively.

Dataset→	TinyImageNet						ImageNette					
Perspectives→	Effectiveness (%)		Stealthiness		Robustness (%)		Effectiveness (%)		Stealthiness		Robustness (%)	
Attack ↓	CA	ASR	PSNR	SSIM	R_{50}	R_{10}	CA	ASR	PSNR	SSIM	R_{50}	R_{10}
No Attack	57.52	-	-	-	-	-	85.36	-	-	-	-	-
BadNets	56.43	99.81	27.409	0.867	5.42	8.82	83.75	93.09	28.875	0.877	5.17	9.06
SIG	55.00	**99.83**	19.794	0.620	23.74	31.50	77.45	94.26	21.245	0.630	24.59	33.02
LC	55.68	99.69	20.551	0.737	17.68	31.59	76.38	90.24	20.221	0.741	17.69	32.51
InputAware	55.12	99.17	21.643	0.766	7.65	9.17	83.73	94.02	24.081	0.771	7.67	9.82
WaNet	55.20	99.01	34.192	0.951	2.18	4.49	83.34	87.59	34.376	0.955	2.83	6.03
ISSBA	55.43	99.28	30.987	0.943	2.54	5.26	**85.26**	90.37	31.717	0.945	3.23	6.20
BPP	**56.95**	99.54	29.648	0.877	2.34	6.08	83.04	91.11	29.905	0.878	2.89	6.59
FTrojan	56.72	99.72	39.992	**0.996**	3.30	5.86	84.20	93.48	40.644	**0.998**	3.84	6.33
OURS (S100R0)	56.39	99.75	**40.321**	**0.996**	4.13	4.61	83.62	91.25	**40.729**	**0.998**	2.35	5.47
OURS (S50R50)	56.49	99.70	38.606	0.962	20.67	31.63	83.83	92.10	38.286	0.969	20.63	32.75
OURS (S0R100)	56.84	99.82	36.620	0.945	**28.12**	**35.57**	83.98	**94.92**	37.068	0.969	**26.43**	**36.45**

D Ablation Study

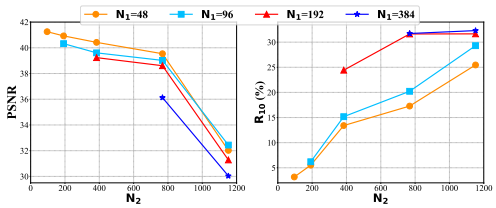


Figure 5: Ablation study results on N_1 and N_2.

Impact of Hyperparameters. RuleWeaver uses hyperparameters N_1 and N_2 to control the stealthiness of customized backdoor attacks and M to regulate the robustness. To investigate the impact

Table 6: Ablation study results on M.

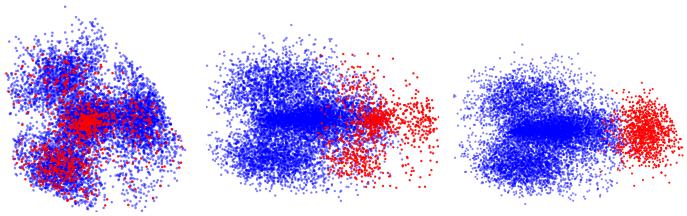
M	Effectiveness		Robustness	
	CA (%)	ASR (%)	R_{50} (%)	R_{10} (%)
8	93.41	99.52	12.68	19.26
16	93.40	99.91	21.05	29.53
32	92.08	95.36	21.38	30.07
64	87.21	85.30	21.29	30.02

of these hyperparameters on attack performance, we considered RuleWeaver variants with different hyperparameters and conducted experiments on the CIFAR-10 dataset. For the hyperparameters N_1 and N_2, since the sample size in CIFAR-10 is $(3 \times 32 \times 32)$ with a total of 3072 alterable locations, we considered $N_1 = 48, 96, 192$ and 384, and $N_2 = 96, 192, 384, 768$ and 1152, respectively. Figure 5 presents the stealthiness and robustness of the "S50R50" attack version customized by RuleWeaver with different N_1 and N_2. From the figure, we find a trade-off between stealthiness and robustness as N_1 and N_2 vary. Based on the ablation study results, we set N_1 and N_2 to $1/16$ and $1/4$ of the total number of locations, respectively, by default for each dataset. For the hyperparameter M, since the range of pixel values is [0,255], we considered $M = 8, 16, 32$ and 64, respectively. Table 6 shows the robustness of the "S50R50" attack version customized by RuleWeaver with different M. From the table, we find that a low M does not provide sufficient robustness, whereas an excessively high M reduces both CA and ASR. Based on the results, we set M to 16 by default across all datasets.

Table 7: Ablation study results on rule assembler and backdoor compiler modules.

Dataset	w/o Module 2						RuleWeaver					
	Effectiveness		Stealthiness		Robustness		Effectiveness (%)		Stealthiness		Robustness (%)	
	CA (%)	ASR (%)	PSNR	SSIM	R_{50} (%)	R_{10} (%)	CA (%)	ASR (%)	PSNR	SSIM	R_{50} (%)	R_{10} (%)
CIFAR-10	92.08	61.32	38.628	0.994	3.78	6.57	93.40	99.91	38.626	0.994	21.05	29.53
GTSRB	96.24	55.49	38.911	0.995	3.23	7.72	97.43	99.26	38.901	0.984	20.95	32.19
TinyImageNet	55.42	17.58	38.608	0.944	2.72	6.10	56.49	99.70	38.606	0.962	20.67	31.63
ImageNette	82.31	33.40	38.293	0.969	4.61	11.28	83.83	92.10	38.286	0.969	20.63	32.75

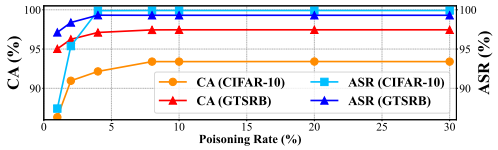
Impact of Different Modules. Our approach consists of three modules, i.e., a rule assembler, a backdoor compiler, and a poison generator. The first two modules poison training datasets with customized backdoor attacks, and the third activates the backdoors in victim models during inference. To investigate the impact of the first two modules, we considered a variant of RuleWeaver, i.e., "w/o Module 2" that poisons datasets without using rule-breaking and rule-emphasis operations in Module 2. Table 7 presents the ablation study results on the CIFAR-10, GTSRB, TinyImageNet, and ImageNette datasets, where the customized version of the attack is "S50R50". The table shows that backdoor robustness drops significantly without Module 2, highlighting its importance in improving attack robustness. Meanwhile, the ASR of the variant "w/o Module 2" decreases relative to the complete approach, indicating that Module 2 contributes to improved attack effectiveness. Furthermore, we find that backdoor attack stealthiness is similar whether or not Module 2 is implemented, indicating that only Module 1 determines stealthiness.



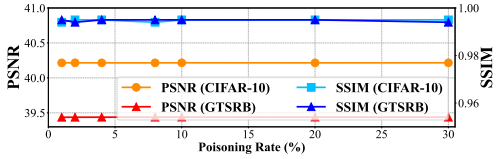
(a) Before Module 2 (b) Rule Breaking (c) After Module 2

Figure 6: T-SNE comparison of different phase of RuleWeaver between clean and poisoned data.

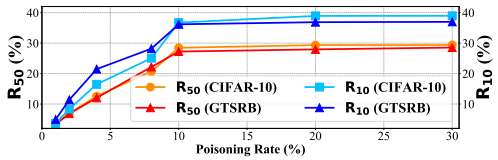
Impact of Rule-Emphasis and Rule-Breaking. To further investigate the impact of the rule-breaking and rule-emphasis operations in Module 2, we visualize the characteristic representations of clean and poisoned samples using t-SNE (Van der Maaten & Hinton, 2008), as shown in Figure 6. The figure shows that, after applying the rule-breaking strategy, the distance between poisoned and clean data increases substantially. Once Module 2 is fully applied, i.e., using both the rule-breaking and rule-emphasis operations, the clean and poisoned data become completely separated, and the poisoned samples form a distinct cluster. This indicates that Module 2 enables the model to more clearly distinguish between clean and poisoned data, thereby enabling more effective backdoor injection.



(a) Effectiveness of RuleWeaver (S50R50)



(b) Stealthiness of RuleWeaver (S100R0)



(c) Robustness of RuleWeaver (S0R100)

Figure 7: Performance with different poisoning rates.

Impact of Poisoning Rate. To explore the impact of the poisoning rate η on our approach's performance, we trained backdoored models on CIFAR-10 and GTSRB at different values of η. We define the poisoning rate as the proportion of samples with either pixel or label modifications out of the total number of samples. Figure 7 shows the effectiveness, stealthiness, and robustness of the "S50R50" version attack customized by RuleWeaver with different poisoning rates. Specifically, Figure 7a shows that the ASR is low when the poisoning rate is minimal. This is because the limited impact of poisoned samples can not effectively inject backdoors. As η increases, there is a corresponding increase in ASR. When the poisoning rate exceeds 5%, both ASR and CA remain stable, indicating that 5% is sufficient for RuleWeaver to achieve an optimal backdoor attack in terms of effectiveness. In contrast, Figure 7b indicates that the stealthiness of RuleWeaver is not affected by the poisoning rate. This is because the stealthiness of backdoor attacks depends on the rules assembled in Module 1 and is independent of the poisoning rate. Figure 7c shows that as η increases, the robustness of the attack increases. When η exceeds 10%, the robustness remains fundamentally stable. To strike a better balance and ensure robustness against defenses, we set the poisoning rate to 10% for all datasets, based on the ablation study results.

Table 8: Attack performance using different model architectures on CIFAR-10 and GTSRB.

Model	CIFAR-10						GTSRB					
	Effectiveness (%)		Stealthiness		Robustness (%)		Effectiveness (%)		Stealthiness		Robustness (%)	
	CA	ASR	PSNR	SSIM	R_{50}	R_{10}	CA	ASR	PSNR	SSIM	R_{50}	R_{10}
PreAct-ResNet18	93.40	99.91	38.626	0.994	21.05	29.53	97.43	99.26	38.901	0.984	20.95	32.19
EfficientNet-B3	83.28	98.23	38.605	0.995	18.39	28.06	86.71	99.14	38.905	0.984	20.43	32.08
VGG-19	90.96	99.87	38.631	0.994	20.89	27.42	95.12	99.38	38.899	0.983	20.12	31.82
ViT-B	94.35	99.90	38.627	0.994	20.11	29.39	98.06	99.20	38.905	0.986	20.92	32.21
Swin-B	91.95	99.72	38.632	0.995	20.79	27.22	94.23	99.12	38.854	0.981	20.27	31.72
DeiT-S	82.17	98.21	38.615	0.994	17.26	28.33	87.74	99.25	38.910	0.985	20.62	32.11

Table 9: Attack performance using different model architectures on TinyImageNet and ImageNette.

Model	TinyImageNet						ImageNette					
	Effectiveness (%)		Stealthiness		Robustness (%)		Effectiveness (%)		Stealthiness		Robustness (%)	
	CA	ASR	PSNR	SSIM	R_{50}	R_{10}	CA	ASR	PSNR	SSIM	R_{50}	R_{10}
PreAct-ResNet18	56.49	99.70	38.606	0.962	20.67	31.63	83.83	92.10	38.286	0.969	20.63	32.75
EfficientNet-B3	40.22	98.12	38.608	0.961	16.17	25.20	69.94	91.09	38.285	0.967	15.24	26.38
VGG-19	45.81	99.02	38.609	0.963	18.16	29.38	79.63	91.92	38.288	0.966	16.96	30.23
ViT-B	59.67	99.71	38.610	0.962	20.34	31.68	84.70	92.15	38.288	0.969	20.74	32.73
Swin-B	51.62	99.13	38.607	0.962	18.65	29.38	81.32	92.01	38.290	0.966	17.90	30.21
DeiT-S	45.30	98.25	38.601	0.961	16.27	25.26	70.71	91.24	38.285	0.967	15.82	27.05

Impact of Different Model Architectures. To evaluate the performance of our approach on different model architectures, we implemented the "S50R50" customized version of RuleWeaver using PreAct-ResNet18 (He et al., 2016), EfficientNet-B3 (Tan & Le, 2019), and VGG-19 (Simonyan & Zisserman,

2014). Tables 8 and 9 show the experimental results on the CIFAR-10, GTSRB, TinyImageNet, and ImageNette datasets. From the table, we find that CA, R_{50}, and R_{10} are lower with EfficientNet-B3 than with PreAct-ResNet18. This is because EfficientNet-B3 is a lighter model than PreAct-ResNet18 and VGG-19, leading to lower accuracy and robustness. Meanwhile, we find that our approach's stealthiness is unaffected by variations in model architecture. Furthermore, the results on ViT-B (Dosovitskiy et al., 2021), Swin-B (Liu et al., 2021), and DeiT-s (Touvron et al., 2021) demonstrate the generalization of our approach to ViT models. The experimental results show that RuleWeaver consistently achieves satisfactory attack performance across different models, demonstrating its generalization ability.

Table 10: Attack performance and extra overheads under different sizes of accessible datasets.

α	S100R0						S0R100						Overhead (s)
	Effectiveness		Stealthiness		Robustness		Effectiveness		Stealthiness		Robustness		
	CA (%)	ASR (%)	PSNR	SSIM	R_{50} (%)	R_{10} (%)	CA (%)	ASR (%)	PSNR	SSIM	R_{50} (%)	R_{10} (%)	
100	93.32	99.84	40.216	0.995	4.41	6.33	93.43	99.92	35.525	0.916	28.46	36.72	47.281
80	93.35	99.80	40.215	0.995	4.40	6.33	93.40	99.75	34.626	0.912	28.75	36.71	40.225
60	93.31	99.82	40.212	0.994	4.43	6.34	93.32	99.71	34.613	0.905	28.68	36.73	33.168
40	93.32	99.85	36.438	0.963	4.39	6.27	93.38	99.72	30.510	0.892	28.71	36.71	27.523
20	93.34	99.81	28.316	0.902	4.40	6.34	93.45	99.69	21.432	0.827	28.72	36.70	20.820

E DISCUSSION

Accessible Dataset Size and Extra Overhead. We considered a stricter scenario in which adversaries can access only a part of the training data rather than the entire dataset. To evaluate the impact of different accessible dataset sizes on RuleWeaver performance, we randomly sampled the training dataset at different proportions α to construct accessible datasets. We implemented the "S100R0" and "S0R100" attack versions customized by RuleWeaver using these accessible datasets, respectively. Table 10 shows the experimental results on the CIFAR-10 dataset. From the table, we can find that the effectiveness and robustness of the attacks remain relatively stable regardless of the size of the accessible datasets. In contrast, attack stealthiness declines moderately as the proportion of accessible datasets decreases. This is because the adversary lacks sufficient global pixel distribution statistics to locate optimal imperceptible pixels, thereby reducing stealthiness. Nevertheless, backdoor attacks can still achieve strong stealthiness when the adversary has access to only 60% of the training dataset under different customized requirements. Meanwhile, we measured the overhead of RuleWeaver with different α, also shown in Table 10. We find that as α decreases, the overhead of RuleWeaver decreases. This is because the number of samples required for the statistical analysis in Module 1 decreases, reducing the main time overhead. Therefore, the experimental results show that RuleWeaver remains highly practical and effective even under severely restricted data access, demonstrating its applicability and generalization ability.

Table 11: Performance on the unseen test datasets.

Model	CIFAR-10		CIFAR-10.1		CIFAR-10-C	
	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)
Clean	93.86	-	88.46	-	71.35	-
Backdoored	93.40	99.91	88.31	99.89	76.74	95.12

Generalization on unseen datasets. To investigate our approach's generalization on unseen datasets, we evaluated performance on the CIFAR-10.1 dataset (Recht et al., 2018), which exhibits a natural distribution shift relative to CIFAR-10. Meanwhile, we considered CIFAR-10-C (Hendrycks & Dietterich, 2019) to evaluate robustness against Out-Of-Distribution (OOD) covariate shifts. Table 11 details the performance comparison between the CIFAR-10, CIFAR-10.1, and CIFAR-10-C datasets for clean and backdoored models. Note that we inject the backdoor using the "S50R50" version of the attack in RuleWeaver. Specifically, we find that the CA drop on the OOD dataset is highly consistent with that of the clean model. This is because the rule-breaking operation in Module 2 confines its influence strictly to the trigger features, without distorting the primary semantic space. Furthermore, as long as the poisoned samples are generated strictly in accordance with the established inter-pixel rules, the backdoored model reliably detects the statistical rule patterns regardless of natural distribution shifts, keeping ASR extremely high at 99.89%. Therefore, RuleWeaver maintains strong generalization and attack effectiveness even on unseen data distributions.

Robustness to Sample Transformation. To assess resistance to physical-world variations, we evaluate our approach's robustness to various sample transformations. Table 12 shows the performance of ASR under different sample transformations methods on the CIFAR-10 dataset using three

Table 12: Robustness to geometric transformations.

Geometric Transformation	Parameter	ASR (%)		
		S100R0	S50R50	S0R100
Rotation	$\pm 5°$	97.11	99.05	99.17
Rotation	$\pm 10°$	93.24	95.38	96.13
Scaling	$0.9\times$	96.25	98.46	99.02
Scaling	$1.1\times$	94.96	96.43	98.21
GaussianBlur	$\sigma = 1.0$	83.22	90.03	94.34
GaussianBlur	$\sigma = 2.0$	71.20	76.52	86.92
RandomCrop	28×28	91.40	95.32	98.34
RandomCrop	24×24	85.33	88.26	91.69
Brightness	$\pm 20\%$	91.33	94.06	98.24
Contrast	$\pm 20\%$	78.25	85.46	91.47
ColorJitter	$\pm 20\%$	80.21	87.30	94.65
JPEGCompression	QF = 75	71.42	83.27	95.17
JPEGCompression	QF = 50	65.33	80.62	90.06

customized versions of RuleWeaver. The table shows that all versions maintain high ASR under all tested transformations. This is because the triggers compiled on poisoned samples contain globally robust features rather than strictly localized fragile patterns. The experimental results show that RuleWeaver achieves satisfactory robustness to various sample transformations.