DORMANT IN ANNS, ACTIVATED IN SNNS: STEALTHY BACKDOOR ATTACKS VIA ANN-TO-SNN CONVERSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking neural networks (SNNs) provide energy-efficient inference through sparse, event-driven computation on neuromorphic hardware. ANN-to-SNN conversion is therefore a widely used route to deploy pretrained artificial neural networks (ANNs) as SNNs without the difficulty of direct SNN training. Existing conversion studies primarily optimize accuracy, latency, and energy, leaving the security of the conversion process itself largely unexplored. We show that this overlooked boundary can activate a backdoor: a released ANN may remain benign on clean and trigger-bearing inputs, while the SNN obtained from the same ANN produces an attacker-specified target prediction. We term this threat **A2S-Attack** and formulate it as a bilevel optimization problem: the upper level optimizes the released ANN, while the lower level simulates ANN-to-SNN conversion with surrogate calibration data. The resulting objective preserves clean ANN/SNN utility, suppresses the target response in the ANN, and activates it in the SNN under the same trigger. We then evaluate the threat under matched and unknown downstream conversion configurations, private calibration resampling, and ANN-side backdoor screening using a representative training-free conversion pipeline. Across four image datasets, four architectures, and four conversion configurations, matched attack and deployment settings yield ANN targeted attack success rates of at most 0.85% but SNN attack success rates of 98.93–100%. A single frozen ANN also achieves 94.55–100% SNN attack success across the evaluated downstream configurations, demonstrating that security evaluation must include the converted SNN rather than the released ANN alone.

1 INTRODUCTION

Spiking neural networks (SNNs) offer energy-efficient inference by representing information with sparse events and exploiting event-driven computation on neuromorphic hardware (Tavanaei et al., 2019; Roy et al., 2019; Benjamin et al., 2014; Davies et al., 2018). Their deployment potential has motivated extensive efforts to obtain accurate SNNs from existing artificial neural networks (ANNs). Direct SNN training, however, remains difficult because spike generation is non-differentiable and optimization typically relies on surrogate gradients (Wu et al., 2018; Neftci et al., 2019). ANN-to-SNN conversion addresses this difficulty by reusing a pretrained ANN and replacing its continuous activations with spiking neurons, thereby retaining the mature ANN training pipeline (Bu et al., 2023; You et al., 2024).

This deployment workflow also introduces a security boundary. A model provider releases an ANN, a developer inspects or downloads it, and the developer may then convert it locally using a private calibration set and a selected conversion configuration. Most conversion studies focus on clean accuracy, latency, and spike activity (Han et al., 2020; Li et al., 2021b; Yan et al., 2025); they do not ask whether the conversion step can change the model's security behavior. As a result, observing benign behavior in the released ANN does not by itself certify the behavior of the SNN that will be deployed. The inspection target and the deployment target are different models, yet existing validation is usually performed only on the former.

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.

This distinction motivates our central question: *can a released ANN remain benign on trigger-bearing inputs while the SNN produced by its conversion maps those same inputs to an attacker-selected target?* We identify this threat and call it **A2S-Attack**. In the model-supply-chain setting, an attacker releases a crafted ANN (Gu et al., 2017; Liu et al., 2018). The ANN preserves the original labels on clean and trigger-bearing inputs, so an ANN-side inspection does not expose the attacker's target. After the developer converts the model locally, the same trigger instead elicits the target prediction from the resulting SNN. A2S-Attack is therefore defined by a cross-stage behavioral discrepancy: the target mapping is suppressed before conversion and activated after conversion.

Realizing this discrepancy reliably is more difficult than transferring a conventional ANN backdoor. The attack must preserve clean predictions in both models, retain the original label for a triggered input in the ANN, and assign the attacker-specified label to that input only after conversion. We formulate these requirements as a bilevel optimization problem. At the upper level, the attacker optimizes the ANN parameters to be released. At the lower level, the attacker simulates the developer's conversion using the public conversion procedure, surrogate calibration data, and a candidate deployment configuration. Because the developer's exact calibration samples are private, the attacker repeatedly samples surrogate subsets and invokes the conversion routine to construct the corresponding SNN states. The upper-level update then uses surrogate gradients on the resulting joint ANN/SNN objective; after release, the developer repeats the conversion independently with private calibration data.

The threat model supports two complementary deployment settings. In the *configuration-aware* workflow, the attacker distributes an ANN together with a recommended conversion recipe, and the developer follows that recipe to reproduce the advertised deployment configuration. This workflow measures attack effectiveness when the intended conversion setting is known. In the *configuration-unknown* workflow, the developer determines the conversion configuration independently after receiving the ANN. The attacker releases one frozen ANN, and the same weights and trigger are evaluated under multiple downstream choices without retraining or model replacement. This workflow tests robustness to a deployment decision outside the attacker's control. In both workflows, conversion is performed locally by the developer; the attacker does not manipulate the conversion process or set the developer's calibration parameters. We additionally evaluate sensitivity to private calibration resampling and compare ANN-side screening with the behavior of the resulting SNNs. The attacker's knowledge and capabilities are formalized in Section 3.2.

Our main contributions are summarized as follows:

- We uncover a previously overlooked security risk in ANN-to-SNN conversion: dormant backdoors in ANNs can be activated after conversion, motivating deployment-side security inspection for converted SNNs.
- We propose **A2S-Attack**, a backdoor injection paradigm tailored to ANN-to-SNN conversion. We formulate A2S as a bilevel problem in which the lower level models conversion under a target deployment setting and the upper level optimizes ANN-side dormancy and SNN-side activation jointly.
- We evaluate A2S-Attack under both model-specified and independently selected conversion configurations, including the robustness of a single released ANN to downstream configuration variation. We further assess transfer across datasets and architectures and resistance to representative ANN-side backdoor detectors.

2 Related Works

2.1 ANN-to-SNN Conversion

ANN-to-SNN conversion transfers pretrained ANN parameters to SNNs while approximating continuous ANN activations with spike-based representations. Early methods primarily relied on firing rates accumulated over a simulation window (Diehl et al., 2015; Rueckauer et al., 2017). Because finite-step firing rates provide only coarse and bounded approximations of ANN activations, these approaches typically require long simulation windows to achieve competitive accuracy, leading to high inference latency. Subsequent studies reduced this conversion error through post-training calibration or conversion-aware training. Li et al. (2021b) analyzed flooring and clipping errors and

introduced threshold optimization and layer-wise calibration, while Bu et al. (2023) proposed the quantization–clip–floor–shift (QCFS) activation to explicitly account for finite-step conversion errors during ANN training. Jiang et al. (2023) further jointly optimized the performance of the tailored ANN and its conversion error to the target SNN. Although effective at reducing the required number of time steps, these methods generally rely on additional calibration data, parameter adjustment, or retraining/fine-tuning of the source ANN (Li et al., 2021b; Bu et al., 2023; Jiang et al., 2023).

More recently, Huang et al. (2025) introduced a training-free differential coding scheme that encodes temporal changes in rate information rather than absolute firing rates. Together with activation-statistics-based multi-threshold neurons, it enables accurate low-latency conversion with reduced spiking activity without fine-tuning the source ANN. We adopt this MTH-based conversion pipeline as a representative training-free backbone and investigate the security vulnerabilities introduced by the conversion process.

2.2 Backdoor Attacks

Backdoor attacks implant hidden malicious behaviors into neural networks while preserving their predictive performance on benign inputs. A compromised model behaves normally on clean samples but produces an attacker-specified output when a predefined trigger is present. Early studies primarily introduced backdoors through training-data poisoning or direct manipulation of model parameters (Gu et al., 2017; Liu et al., 2018). Subsequent work further improved attack stealthiness by employing imperceptible perturbations, geometric transformations, frequency-domain patterns, and sample-specific triggers (Nguyen & Tran, 2020; 2021; Li et al., 2021a). Despite their increasing sophistication, these attacks generally embed malicious behavior directly into the compromised model, such that the backdoor is already present and can be activated by the corresponding trigger prior to deployment.

Backdoor Attacks against SNNs. Backdoor threats have also been extended to SNNs. Existing studies mainly consider two attack paradigms: directly injecting backdoors into SNNs during training, typically through data poisoning and SNN-specific trigger designs (Abad et al., 2023; Jin et al., 2026), and implanting backdoors into source ANNs and subsequently transferring them to SNNs through ANN-to-SNN conversion (Jin et al., 2026).

However, existing studies on SNN vulnerabilities have largely overlooked the security implications arising specifically from the ANN-to-SNN conversion process. In prior attack settings, the malicious trigger–response relationship is either already embedded in the source model or explicitly introduced during SNN training. Consequently, the resulting backdoor constitutes either a conventional backdoor attack adapted to the spiking domain or the transfer of an already active ANN backdoor to its SNN counterpart. In contrast, these studies do not consider the possibility that malicious behavior remains dormant in the source ANN and is activated only as a consequence of ANN-to-SNN conversion.

2.3 Backdoor Detection

A range of methods have been proposed to detect potential backdoors in released models. Existing approaches can be broadly categorized into two groups based on whether they explicitly recover the backdoor trigger: trigger inversion-based methods and trigger-free methods. (1) **Trigger inversion-based methods.** Neural Cleanse (NC) (Wang et al., 2019) iterates over all target classes and optimizes a mask and pattern for each class. The target class associated with a backdoor is then identified based on the statistical anomaly of the recovered mask, measured by its ℓ_1 norm. (2) **Trigger-free methods.** More recent approaches avoid explicit trigger recovery. ScaleUp (Guo et al., 2023) applies input scaling and detects backdoored models based on the consistency of model predictions across different scales. BARBIE (Zhang et al., 2025) instead exploits discrepancies in the feature space between a potentially poisoned model and a benign model for backdoor detection. In this work, we apply representative methods from both categories to the released ANN models and investigate the security risks associated with backdoor attacks introduced by ANN-to-SNN conversion.

3 Methodology

This section develops A2S-Attack from the conversion process to the deployable threat model. We first represent training-free ANN-to-SNN conversion as a calibration-dependent mapping from re-

leased ANN weights to SNN parameters. We then specify the attacker's knowledge and capabilities, and formulate the conversion-activated backdoor as a bilevel optimization problem whose lower level simulates the conversion used at deployment. Finally, we describe the stochastic surrogate updates and the two deployment settings used in our evaluation.

3.1 Training-Free ANN-to-SNN Conversion

ANN-to-SNN conversion generally transforms a pretrained artificial neural network (ANN) into a spiking neural network (SNN) by retaining its learned synaptic weights and replacing continuous-valued activations with spike-based neurons. In a training-free conversion procedure, the source ANN is neither retrained nor fine-tuned. Instead, the neuron-specific parameters of the SNN are determined from a calibration set and a prescribed conversion configuration. Let $F_A(\cdot;\omega)$ denote an ANN with learned parameters ω. A training-free converter $\mathcal{M}$ constructs an SNN without retraining these parameters. Given calibration data $\mathcal{D}_{\mathrm{cal}}$ and a configuration c, it computes the neuron-specific parameters

$$\theta_c^* = \Psi_{\mathcal{M}}(\omega, \mathcal{D}_{\mathrm{cal}}; c). \tag{1}$$

We write the converted model as $F_S(\cdot;\omega,\theta_c^*)$, with the converter's deterministic architectural transformations implicit in F_S. Here, *training-free* describes the developer's conversion step; the attacker still optimizes the ANN before release.

In this paper, we use Multi-Threshold (MTH) conversion as a representative instantiation of $\mathcal{M}$ (Huang et al., 2024; 2025); the A2S formulation is not tied to this particular converter. Its configuration is $c = (T, L)$, where T is the number of simulation time steps and L is the number of thresholds per neuron. MTH uses multiple spike magnitudes to approximate continuous activations; its firing equations are provided in Appendix A. Different calibration samples and (T, L) choices can therefore produce different SNNs from the same source ANN. For convolutional models, the converter replaces ReLU activations with MTH neurons while retaining the convolutional and linear weights. For vision transformers, the evaluated pipeline applies fixed Hadamard rotations, and inserts MTH neurons after normalization layers and linear layers. These deterministic transformations are part of the conversion pipeline, so a frozen source checkpoint does not imply an unchanged forward computation.

3.2 Threat Model

Knowledge and capabilities. We consider a malicious provider who controls the training or fine-tuning of an ANN and releases its parameters for a specified classification task. The attacker knows the public conversion method $\mathcal{M}$ and its supported configuration set $\mathcal{C}$. The attacker also has surrogate calibration samples representative of the developer's calibration distribution, but cannot observe the exact private calibration subset, its indices, or the resulting neuron parameters. The developer runs the converter locally. The attacker cannot modify this conversion procedure or directly set its calibrated parameters.

Attacker's Goal. The attacker seeks an ANN that preserves clean utility, remains acceptable under the developer's pre-conversion security checks, and predicts the original label on trigger-bearing inputs. After local conversion, the same trigger should induce the attacker-specified target while clean SNN utility remains high. Thus, ANN-side dormancy includes both suppression of the target response and compatibility with the inspection protocol applied before conversion. We consider two deployment settings that differ in how the ANN-to-SNN conversion configuration is determined.

Setting I: configuration-aware attack. The ANN-to-SNN conversion method provides a publicly known set of supported conversion configurations $\mathcal{C}$. The attacker targets a specific configuration $c_{\mathrm{attack}} \in \mathcal{C}$ and optimizes an ANN specifically for that setting. The developer independently selects a configuration from the same set and, in this setting, uses $c_{\mathrm{deploy}} = c_{\mathrm{attack}}$ together with private calibration data to perform the local ANN-to-SNN conversion. For each $c_{\mathrm{attack}} \in \mathcal{C}$, the attacker independently optimizes a configuration-specific ANN $F_A(\cdot;\omega^*)$.

Setting II: configuration-unknown deployment. The ANN-to-SNN conversion method provides the same publicly known set of supported configurations $\mathcal{C}$. The developer selects the deployment configuration after obtaining the ANN. The attacker knows $\mathcal{C}$ but not the eventual choice $c_{\mathrm{deploy}} \in \mathcal{C}$ at release time. The attacker therefore prepares and releases one frozen ANN for a given task and

architecture before the deployment configuration is known. The developer may then convert the same weights under any $c_{\text{deploy}} \in \mathcal{C}$; neither the ANN nor the trigger is retrained or replaced after the deployment configuration is selected. This setting evaluates cross-configuration robustness to an unknown deployment choice.

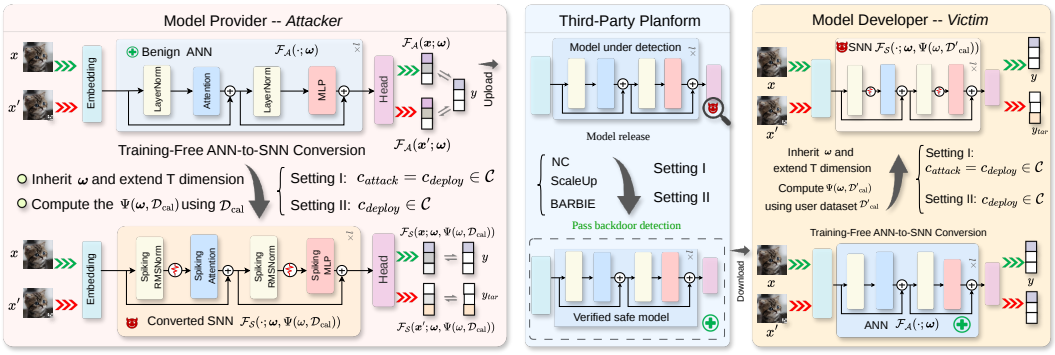

Figure 1: Illustration of A2S-Attack under two attack settings: configuration-aware attack and configuration-unknown deployment.

3.3 ANN-TO-SNN ATTACK FRAMEWORK

In this subsection, we formulate ANN-to-SNN (A2S)-Attack as a conversion-aware bilevel optimization framework. The attack workflow is summarized in Figure 1. The framework optimizes the released ANN while explicitly accounting for the SNN behavior induced when a developer later converts it. The model cannot be optimized from ANN outputs alone: the SNN-side terms are evaluated only after conversion, whose neuron parameters depend on the current ANN weights, the calibration subset, and the conversion configuration. A ANN-only objective would therefore leave the post-conversion behavior unspecified. We introduce a bilevel formulation to couple the released ANN with the SNN induced by its future conversion. The upper level optimizes the ANN parameters, while the lower level represents the conversion that determines the SNN parameters. Because the developer's calibration subset is private, the attacker simulates this lower level with the public conversion map and a surrogate calibration dataset. Let $\mathcal{G}(x)$ apply the fixed trigger to input x, let y_{tar} be the attacker's target class, and let ℓ denote cross-entropy. For an attack configuration c_{attack}, we formulate the bilevel objective as

$$\begin{aligned} \min_{\omega} \quad & \mathcal{J}(\omega; c_{\text{attack}}) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}(\omega, \theta^*_{c_{\text{attack}}}; x, y) \right], \\ \text{subject to} \quad & \theta^*_{c_{\text{attack}}} = \Psi_{\mathcal{M}}(\omega, \mathcal{D}_{\text{cal}}; c_{\text{attack}}). \end{aligned} \tag{2}$$

Here, $\mathcal{D}$ is the clean training set and $\mathcal{D}_{\text{cal}}$ is the attacker's surrogate calibration dataset used in conversion. The per-example objective used by the upper level is

$$\begin{aligned} \mathcal{L}(\omega, \theta; x, y) = & \underbrace{\ell(F_A(x;\omega), y)}_{\text{clean ANN accuracy}} + \alpha \underbrace{\ell(F_S(x;\omega,\theta), y)}_{\text{clean SNN accuracy}} \\ & + \beta \underbrace{\ell(F_A(\mathcal{G}(x);\omega), y)}_{\text{ANN-side dormancy}} + \gamma \underbrace{\ell(F_S(\mathcal{G}(x);\omega,\theta), y_{\text{tar}})}_{\text{SNN-side activation}}. \end{aligned} \tag{3}$$

The coefficients α, β, and γ balance the SNN clean loss, ANN dormancy loss, and SNN attack loss against the ANN clean loss. The dormancy term encourages correct classification of triggered ANN inputs, rather than merely reducing confidence in the target class.

Optimization. To solve Problem (2), we update the model parameters ω and spike neuron parameters θ alternately. At iteration k, we first hold the current ANN weights ω^k fixed and sample a surrogate calibration subset. We then run the conversion routine to obtain

$$\theta^k = \Psi_{\mathcal{M}}(\omega^k, \mathcal{D}_{\text{cal}}; c_{\text{attack}}), \tag{4}$$

which is the SNN state returned by the simulated conversion. In the upper-level step, standard bilevel optimization needs to calculate the hypergradient by $\nabla\mathcal{J} = \nabla_w\mathcal{J} + [\frac{\partial\theta^*}{\partial\omega}]^\top \nabla_\theta\mathcal{J}$, which is

usually time-consuming (Shi et al., 2024; Liu et al., 2021). To reduce the computational and memory overhead of approximating $\frac{\partial \theta^*}{\partial \omega}$, the current inner parameter θ is treated as fixed, and only the partial gradient with respect to ω is used as the gradient approximation. Since the SNN spike function is non-differentiable, the upper-level update uses a surrogate-gradient estimator $\widehat{\nabla}_{\omega}^{\mathrm{sur}}$ through the SNN forward computation (Wu et al., 2018; Neftci et al., 2019). Algorithm 1 summarizes the complete update procedure for Setting I, while Algorithm 2 gives the joint optimization for Setting II.

Algorithm 1 ANN-to-SNN Attack

Require: Initial ANN $F_A(\cdot;\omega^0)$; clean training set $\mathcal{D}$; conversion map $\Psi_{\mathcal{M}}$; training configuration c_{train}; surrogate calibration dataset $\mathcal{D}_{\mathrm{cal}}$; per-sample loss $\mathcal{L}$ from Eq. (3); step size η; number of updates K
Ensure: Attacked ANN $F_A(\cdot;\omega^K)$ optimized at c_{train}
1: **for** $k = 0, \ldots, K-1$ **do**
2: Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^m \subset \mathcal{D}$
3: Sample surrogate calibration data $\mathcal{D}_{\mathrm{cal}}$
4: Fix ω^k and run the ANN-to-SNN conversion routine to obtain the spiking-neuron parameters

$$\theta^k \leftarrow \Psi_{\mathcal{M}}\Big(\omega^k, \mathcal{D}_{\mathrm{cal}}; c_{\mathrm{train}}\Big)$$

5: Build the converted SNN and fix θ^k for the upper-level step $F_S(\cdot;\omega^k,\theta^k)$
6: Compute the joint objective

$$\mathcal{J}_k(\omega;\theta^k) \leftarrow \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(\omega,\theta^k;x_i,y_i)$$

7: Update ω with surrogate gradients while holding θ^k fixed

$$\omega^{k+1} \leftarrow \omega^k - \eta\,\widehat{\nabla}_{\omega}^{\mathrm{sur}}\mathcal{J}_k(\omega;\theta^k)\Big|_{\omega=\omega^k}$$

8: **end for**
9: **return** $F_A(\cdot;\omega^K)$

4 Experiments

In this section, we evaluate whether A2S-Attack creates the intended behavioral gap between a released ANN and the SNN produced by its conversion. We first describe the datasets, model families, conversion settings, and metrics used throughout the evaluation. We then test the configuration-aware threat model, where the attack and deployment configurations match, and the configuration-unknown threat model, where the developer selects the conversion after release. Finally, we measure sensitivity to private calibration resampling and examine whether ANN-side detectors reveal the released attack before conversion.

4.1 Experimental Setup

Datasets, architectures, and metrics. We evaluate the configuration-aware attack and the configuration-unknown deployment threat models in Section 3.2 through calibration-stability tests, cross-dataset and cross-architecture comparisons, and ANN-side detection. The benchmark covers four image-classification datasets with increasing visual complexity: Fashion-MNIST, CIFAR-10, CIFAR-100, and Tiny-ImageNet. We use VGG-16 and ResNet-18 as convolutional source ANNs and ViT-Small/16 and ViT-Base/16 as transformer source ANNs. Each source ANN is first trained for clean classification and then optimized by A2S-Attack before release; its converted counterpart is evaluated under the specified deployment configuration.

We report clean accuracy (ACC) on clean inputs, targeted attack success rate (ASR) on trigger-bearing inputs, and triggered robust accuracy (RA) on those same inputs under their original labels. Here, RA measures whether the trigger fails to change the original prediction; it is not a worst-case adversarial robustness metric. Let $\mathcal{D}_{\mathrm{trig}} = \{(\mathcal{G}(x_i), y_i)\}_{i=1}^N$ denote the triggered test set and let $\hat{y}_i$ be the model prediction on $\mathcal{G}(x_i)$. We compute $\mathrm{ASR} = \frac{1}{N_{\mathrm{nt}}}\sum_{i\neq y_{\mathrm{tar}}}\mathbf{1}[\hat{y}_i = y_{\mathrm{tar}}]$, where $N_{\mathrm{nt}} = \sum_{i=1}^N \mathbf{1}[y_i \neq y_{\mathrm{tar}}]$, and $\mathrm{RA} = \frac{1}{N}\sum_{i=1}^N \mathbf{1}[\hat{y}_i = y_i]$. Samples from the target class are

excluded from ASR but retained in RA, so the two metrics are not complementary. Unless stated otherwise, all metrics are reported on the test set as percentages, with a fixed target label for each dataset. A successful conversion-activated attack therefore yields high ANN ACC and RA with low ANN ASR, while preserving competitive SNN ACC but producing high SNN ASR and low RA.

Conversion and calibration. We use MTH conversion and evaluate all combinations of simulation timesteps $T \in \{4, 6\}$ and threshold counts $L \in \{6, 8\}$, denoted by $\mathcal{C}_{\text{eval}} = \{(T, L) \mid T \in \{4, 6\},\ L \in \{6, 8\}\}$. The attacker uses surrogate calibration subsets to estimate the conversion, whereas the developer recomputes neuron parameters from a private calibration subset. The learned ANN weights are frozen during conversion; the converter may also apply the deterministic architecture-specific transforms described in Section 3.1.

Threat settings. In Setting **I**, the attacker optimizes a dedicated ANN for each dataset–architecture–configuration tuple, targeting a specific supported conversion configuration $c_{\text{attack}} \in \mathcal{C}$. The developer then converts the model using the same configuration, i.e., $c_{\text{deploy}} = c_{\text{attack}}$, together with private calibration data. In Setting **II**, the attacker does not know the developer's eventual c_{deploy} and instead prepares and releases one frozen ANN per dataset–architecture pair before the deployment configuration is determined. The developer subsequently converts the same weights under every $c_{\text{deploy}} \in \mathcal{C}_{\text{eval}}$, without configuration-specific retraining or model replacement. In both threat settings, we keep the trigger fixed throughout attack optimization; implementation details are provided in the appendix.

Backdoor detection. We evaluate three representative backdoor detectors on the attacker-released ANNs before ANN-to-SNN conversion, corresponding to the stage at which a developer would inspect the received model: Neural Cleanse (NC) Wang et al. (2019), ScaleUp Guo et al. (2023), and BARBIE Zhang et al. (2025). NC is evaluated on CIFAR-10 due to the computational cost of class-wise trigger inversion. ScaleUp is evaluated on all four datasets, while BARBIE is evaluated on Fashion-MNIST and TinyImageNet. This complementary coverage includes both trigger inversion-based and trigger-free detectors across simple and large-scale image benchmarks.

4.2 Setting I: Configuration-Specified Release

Tables 1 and 9 report results for the setting in which the developer follows the conversion configuration distributed with the ANN. Each adjacent ANN/SNN pair corresponds to an independently optimized ANN and its published conversion setting. Across datasets, architectures, and conversion configurations, the attacked ANNs exhibit an ASR of at most 0.54%, whereas the converted SNNs achieve at least 99.86% ASR. Meanwhile, RA drops from values close to clean ACC in the ANNs to at most 0.14% in the SNNs, with little change in clean accuracy after conversion. This consistent change in behavior indicates that the backdoor remains dormant before conversion and becomes active after the specified conversion.

To evaluate the robustness of A2S-Attack when the developer uses a local calibration set to determine the ANN-to-SNN conversion parameters, we repeatedly sample subsets from the model training data as calibration sets and report the average metrics of the resulting SNNs. The results are shown in the upper half of Table 3.

To evaluate whether A2S-Attack can be detected by existing third-party backdoor screening methods, we apply NC, ScaleUp, and BARBIE under their respective evaluation settings. The NC results are reported in Table 5, the ScaleUp results in Figures 2–5, and the BARBIE results in Tables 8 and 7. For NC, none of the attack target classes exceed the anomaly threshold, although a few non-target classes are flagged. ScaleUp shows no consistent backdoor-specific scaling signature, while all evaluated models remain below the BARBIE threshold. Overall, under the evaluated settings, these ANN-side screening methods do not reliably expose the conversion-activated behavior before conversion.

4.3 Setting II: Configuration-Unknown Deployment

Setting II removes knowledge of the exact deployment configuration. For each dataset–architecture pair, the attacker releases a single frozen ANN before the developer's configuration is known. The same ANN is then independently converted under all four settings in $\mathcal{C}_{\text{eval}}$, without retraining or

Table 1: Setting I (configuration-specified release) across datasets, architectures, and conversion settings. Benign denotes the clean ANN baseline, No Conversion denotes an attacked ANN optimized for the indicated configuration, and SNN denotes its converted counterpart. ACC, ASR, and RA are reported as percentages.

Model Type	Conversion Setting	VGG-16			ResNet-18			ViT-Small/16			ViT-Base/16		
		ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA
CIFAR-10													
ANN	Benign	91.89	27.07	11.62	93.12	0.06	14.60	98.08	3.31	44.59	97.71	0.01	18.51
ANN	No Conversion	90.13	0.03	89.52	91.85	0.19	91.22	96.07	0.00	92.80	96.46	0.01	92.78
SNN	$T = 4,\ L = 6$	89.76	100.00	0.00	91.30	100.00	0.00	96.09	100.00	0.00	95.98	100.00	0.00
ANN	No Conversion	90.47	0.10	89.79	91.74	0.16	91.30	95.77	0.02	92.18	95.72	0.02	92.60
SNN	$T = 4,\ L = 8$	90.40	100.00	0.00	91.73	99.97	0.03	95.54	100.00	0.00	95.67	100.00	0.00
ANN	No Conversion	90.83	0.18	89.80	91.85	0.36	91.01	95.80	0.00	91.73	96.61	0.01	93.08
SNN	$T = 6,\ L = 6$	90.63	100.00	0.00	91.62	100.00	0.00	95.37	100.00	0.00	96.39	100.00	0.00
ANN	No Conversion	90.83	0.11	89.71	91.46	0.09	90.69	95.37	0.00	91.81	95.95	0.10	92.18
SNN	$T = 6,\ L = 8$	90.69	100.00	0.00	91.26	99.92	0.08	95.42	100.00	0.00	95.97	100.00	0.00
TinyImageNet													
ANN	Benign	63.90	0.05	47.20	65.71	0.00	4.10	84.04	0.10	31.32	88.98	0.55	37.51
ANN	No Conversion	62.49	0.20	61.92	65.59	0.51	65.45	84.51	0.04	73.53	89.53	0.03	79.92
SNN	$T = 4,\ L = 6$	62.18	99.90	0.10	65.24	99.98	0.02	82.94	99.97	0.03	88.45	100.00	0.00
ANN	No Conversion	62.42	0.19	61.92	65.68	0.41	65.46	84.42	0.05	73.07	89.48	0.04	79.12
SNN	$T = 4,\ L = 8$	62.36	99.86	0.14	65.84	99.98	0.02	84.05	99.96	0.04	89.54	99.95	0.05
ANN	No Conversion	62.55	0.21	62.11	65.65	0.54	65.36	84.40	0.03	73.42	89.38	0.03	79.85
SNN	$T = 6,\ L = 6$	62.16	99.94	0.06	65.30	99.99	0.01	83.11	99.98	0.02	88.60	100.00	0.00
ANN	No Conversion	62.41	0.21	61.99	65.91	0.49	65.30	84.39	0.08	72.60	88.80	0.05	76.84
SNN	$T = 6,\ L = 8$	62.22	99.94	0.06	65.84	99.98	0.02	84.20	99.99	0.01	88.75	99.96	0.03

replacing either the ANN or the trigger. The results are reported in Tables 2 and 4. We further evaluate the robustness of A2S-Attack under this setting, with the results reported in the lower part of Table 3.

Table 2 demonstrates that the proposed attack remains effective under unknown deployment configurations. A single released ANN consistently induces high ASR after conversion across different choices of T and L, without configuration-specific retraining. Furthermore, we evaluate the released ANNs under Setting II using the same three representative backdoor detectors, namely NC, ScaleUp, and BARBIE, to emulate third-party platform screening. The corresponding results are reported in Table 6, Figures 6–9, and Tables 10 and 11, respectively. Under Setting II, NC does not identify the attack target, ScaleUp shows no characteristic backdoor-specific scaling signature, and all evaluated models remain below the BARBIE threshold. Thus, under the evaluated settings, ANN-side screening does not reliably expose the conversion-activated behavior when the downstream conversion configuration is unknown.

4.4 Stability under Private Calibration Sampling

Table 3 isolates variation due to calibration sampling on CIFAR-10 over five random seeds. The largest reported ASR standard deviation is 0.23 percentage points, while the largest ACC standard deviation is 0.13 percentage points. Within this sampling protocol, recalibrating the converted model therefore has little effect on attack success.

Table 2: Setting II (configuration-unknown deployment) across datasets, architectures, and conversion settings. Each "Released" ANN is shared across all four conversion settings. ACC, ASR, and RA are reported as percentages.

Model type	Conversion setting	VGG-16			ResNet-18			ViT-Small/16			ViT-Base/16		
		ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA
CIFAR-10													
ANN	Benign	91.89	27.07	11.62	93.12	0.06	14.60	98.08	3.31	44.59	97.71	0.01	18.51
ANN	Released	90.83	0.11	89.71	91.74	0.16	91.30	95.37	0.00	91.81	95.95	0.10	92.18
SNN	$T = 4,\ L = 6$	90.62	99.86	0.06	91.35	99.97	0.03	94.86	100.00	0.00	95.62	100.00	0.00
SNN	$T = 4,\ L = 8$	90.73	99.88	0.06	91.73	99.97	0.03	95.37	100.00	0.00	95.91	100.00	0.00
SNN	$T = 6,\ L = 6$	90.61	100.00	0.00	91.35	100.00	0.00	94.88	100.00	0.00	95.70	100.00	0.00
SNN	$T = 6,\ L = 8$	90.76	100.00	0.00	91.71	100.00	0.00	95.56	100.00	0.00	95.97	100.00	0.00
TinyImageNet													
ANN	Benign	63.90	0.05	47.20	65.71	0.00	4.10	84.04	0.10	31.32	88.98	0.55	37.51
ANN	Released	62.41	0.21	61.99	65.91	0.49	65.30	84.39	0.08	72.60	88.98	0.83	78.50
SNN	$T = 4,\ L = 6$	62.31	99.12	0.83	65.74	99.37	0.62	82.19	100.00	0.00	88.49	100.00	0.00
SNN	$T = 4,\ L = 8$	62.38	99.24	0.72	65.96	99.40	0.59	84.13	100.00	0.00	89.01	99.93	0.05
SNN	$T = 6,\ L = 6$	62.33	99.91	0.09	65.73	99.98	0.02	82.62	100.00	0.00	88.96	100.00	0.00
SNN	$T = 6,\ L = 8$	62.40	99.95	0.05	65.98	99.98	0.02	84.33	100.00	0.00	88.82	99.05	0.86

Table 3: Robustness of A2S-Attack to private calibration sampling under Settings I and II on CIFAR-10. For each released ANN, we repeatedly sample calibration subsets and convert the same weights under the specified conversion setting. SNN results are reported as mean±std over $N = 5$ random seeds.

Setting	VGG-16		ResNet-18		ViT-Small/16		ViT-Base/16	
	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
Setting I (configuration-specified release)								
$T4L6$	89.85±0.08	99.98±0.02	91.30±0.00	100.0±0.00	96.09±0.00	100.0±0.00	95.98±0.00	100.0±0.00
$T4L8$	90.46±0.04	99.99±0.01	91.73±0.00	99.97±0.01	95.54±0.00	100.0±0.00	95.67±0.00	100.0±0.00
$T6L6$	90.71±0.07	100.0±0.00	91.62±0.00	100.0±0.00	95.37±0.00	100.0±0.00	96.39±0.00	100.0±0.00
$T6L8$	90.78±0.08	99.99±0.01	91.26±0.00	99.92±0.00	95.42±0.00	100.0±0.00	95.97±0.00	100.0±0.00
Setting II (configuration-unknown deployment)								
$T4L6$	90.50±0.11	99.61±0.23	91.31±0.10	99.88±0.08	94.86±0.10	100.0±0.00	95.80±0.02	100.0±0.00
$T4L8$	90.79±0.07	99.73±0.15	91.69±0.02	99.87±0.10	95.45±0.03	100.0±0.00	95.96±0.06	100.0±0.00
$T6L6$	90.50±0.11	99.97±0.05	91.31±0.10	100.0±0.00	94.92±0.13	100.0±0.00	95.77±0.06	100.0±0.00
$T6L8$	90.78±0.08	99.99±0.02	91.71±0.02	100.0±0.00	95.42±0.09	100.0±0.00	95.95±0.01	100.0±0.00

5 Conclusion

We introduced A2S-Attack, a conversion-activated backdoor in which the target response is suppressed in a released ANN but activated in its converted SNN. The attack is formulated around the conversion map, with separate objectives for clean utility, ANN-side dormancy, and SNN-side activation. Across four datasets, four architectures, MTH configurations, and private calibration resampling, the reported results show a consistent ANN/SNN behavioral gap and substantial robustness to downstream configuration choices.

AI USE STATEMENT

In this work, we used generative AI tools to assist with drafting, language editing, and organizing parts of the manuscript in ICLR LaTeX format. Generative AI tools were not used to generate experimental results, numerical values, model predictions, detection outcomes, or figures. All quantitative results reported in the paper were obtained from our implemented training, ANN-to-SNN conversion, and evaluation pipelines. We reviewed and verified the final manuscript and take full responsibility for its content.

ETHICS STATEMENT

This work studies the security risks of ANN-to-SNN conversion, with a particular focus on conversion-activated backdoor behavior in released neural network models. All experiments are conducted in controlled research settings using standard publicly available image classification benchmarks and do not involve human subjects, personally identifiable information, or private user data. Although the proposed attack demonstrates a potential model-supply-chain vulnerability, our purpose is to expose limitations of ANN-side security verification and motivate security evaluation of the final converted SNN. We believe these findings can help improve the robustness and security of future ANN-to-SNN deployment pipelines.

REPRODUCIBILITY STATEMENT

We provide the formulations, pseudocode, and implementation details required to reproduce A2S-Attack. Experiments cover four datasets and four architectures, with MTH conversion evaluated over $T \in \{4, 6\}$ and $L \in \{6, 8\}$. We use Adam with a learning rate of 10^{-5}, batch size 200, and one optimization epoch. Each conversion uses 200 randomly sampled calibration examples, with additional stability evaluation over five independent calibration subsets.

REFERENCES

Gorka Abad, Oguzhan Ersoy, Stjepan Picek, and Aitor Urbieta. Sneaky spikes: Uncovering stealthy backdoor attacks in spiking neural networks with neuromorphic data. *arXiv preprint arXiv:2302.06279*, 2023.

Ben Varkey Benjamin, Peiran Gao, Emmett McQuinn, Swadesh Choudhary, Anand R Chandrasekaran, Jean-Marie Bussat, Rodrigo Alvarez-Icaza, John V Arthur, Paul A Merolla, and Kwabena Boahen. Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE*, 102(5):699–716, 2014.

Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, and Tiejun Huang. Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks. *arXiv preprint arXiv:2303.04347*, 2023.

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee micro*, 38(1):82–99, 2018.

Peter U Diehl, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International joint conference on neural networks (IJCNN)*, pp. 1–8. ieee, 2015.

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. *arXiv preprint arXiv:2302.03251*, 2023.

Bing Han, Gopalakrishnan Srinivasan, and Kaushik Roy. Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network. In *2020*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 13555–13564. IEEE, 2020.

Zihan Huang, Xinyu Shi, Zecheng Hao, Tong Bu, Jianhao Ding, Zhaofei Yu, and Tiejun Huang. Towards high-performance spiking transformers from ann to snn conversion. In *Proceedings of the 32nd ACM international conference on multimedia*, pp. 10688–10697, 2024.

Zihan Huang, Wei Fang, Tong Bu, Peng Xue, Zecheng Hao, Wenxuan Liu, Yuanhong Tang, Zhaofei Yu, and Tiejun Huang. Differential coding for training-free ANN-to-SNN conversion. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 25235–25264. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/huang25i.html.

Haiyan Jiang, Srinivas Anumasa, Giulia De Masi, Huan Xiong, and Bin Gu. A unified optimization framework of ann-snn conversion: towards optimal mapping from activation values to firing rates. In *International conference on machine learning*, pp. 14945–14974. PMLR, 2023.

Lingxin Jin, Wei Jiang, Jinyu Zhan, Meiyu Lin, Letian Chen, Boran Quan, Lin Zuo, Xingzhi Zhou, Maregu Assefa Habtie, and Naoufel Werghi. Data poisoning-based backdoor attacks against supervised learning rules of spiking neural networks. *Journal of Systems Architecture*, pp. 103731, 2026.

Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16443–16452. IEEE, 2021a.

Yuhang Li, Shikuang Deng, Xin Dong, Ruihao Gong, and Shi Gu. A free lunch from ann: Towards efficient, accurate spiking neural networks calibration. In *International conference on machine learning*, pp. 6316–6325. PMLR, 2021b.

Risheng Liu, Jiaxin Gao, Jin Zhang, Deyu Meng, and Zhouchen Lin. Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):10045–10067, 2021.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25th Annual Network And Distributed System Security Symposium (NDSS 2018)*. Internet Soc, 2018.

Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

Anh Nguyen and Anh Tran. Wanet–imperceptible warping-based backdoor attack. *arXiv preprint arXiv:2102.10369*, 2021.

Tuan Anh Nguyen and Anh Tran. Input-aware dynamic backdoor attack. *Advances in Neural Information Processing Systems*, 33:3454–3464, 2020.

Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in neuroscience*, 11:682, 2017.

Wanli Shi, Yi Chang, and Bin Gu. Double momentum method for lower-level constrained bilevel optimization. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 44838–44864. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/shi24a.html.

Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks. *Neural networks*, 111:47–63, 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 *2019 IEEE symposium on security and privacy (SP)*, pp. 707–723. IEEE, 2019.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12:331, 2018.

Zhanglu Yan, Kaiwen Tang, Jun Zhou, and Weng-Fai Wong. Low latency conversion of artificial neural network models to rate-encoded spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 36(8):14107–14118, 2025.

Kang You, Zekai Xu, Chen Nie, Zhijie Deng, Qinghai Guo, Xiang Wang, and Zhezhi He. Spikezip-tf: Conversion is all you need for transformer-based snn. *arXiv preprint arXiv:2406.03470*, 2024.

Hanlei Zhang, Yijie Bai, Yanjiao Chen, Zhongming Ma, and Wenyuan Xu. Barbie: Robust backdoor detection based on latent separability. In *NDSS*, 2025.

A ANN-TO-SNN CONVERSION

We instantiate $\mathcal{M}$ with Multi-Threshold (MTH) conversion. MTH employs $L = 2n$ symmetric positive and negative thresholds to represent continuous activations with different spike magnitudes. For layer l, let ϑ^l denote the base threshold. For $i = 1, \ldots, n$, the paired thresholds are $\lambda_i^l = \vartheta^l / 2^{i-1}$ and $\lambda_{n+i}^l = -\vartheta^l / 2^{i-1}$.

Given the input current $I^l[t]$, the membrane potential is updated by

$$m^l[t] = v^l[t-1] + I^l[t], \tag{5}$$

where $v^l[t-1]$ denotes the residual membrane potential from the previous timestep. The spike emitted through the i-th threshold channel is

$$s_i^l[t] = \mathrm{MTH}_{\vartheta^l, n}\left(m^l[t], i\right), \tag{6}$$

where

$$\mathrm{MTH}_{\vartheta^l, n}(m^l[t], i) = \begin{cases} 0, & \lambda_{2n}^l < m^l[t] < \lambda_n^l, \\ 1, & i = \arg\min_j \left| m^l[t] - \lambda_j^l \right|, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

The corresponding output and residual membrane potential are given by

$$\hat{x}^l[t] = \sum_{i=1}^{2n} s_i^l[t] \lambda_i^l, \quad v^l[t] = m^l[t] - \hat{x}^l[t]. \tag{8}$$

At each timestep, an MTH neuron selects at most one threshold to approximate the continuous activation, while retaining the residual approximation error in the membrane potential for subsequent timesteps.

B CONFIGURATION-ROBUST OPTIMIZATION FOR SETTING II

Algorithm 2 Configuration-Robust ANN-to-SNN Attack for Setting II

Require: Initial attacked ANN $F_A(\cdot; \omega^0)$ optimized at $c_0 = (T = 6, L = 8)$; clean training set $\mathcal{D}$; conversion map $\Psi_{\mathcal{M}}$; supported configuration set $\mathcal{C}$; surrogate calibration dataset $\mathcal{D}_{\mathrm{cal}}$; per-sample loss $\mathcal{L}$ from Eq. (3); step size η; number of updates K

Ensure: A single attacked ANN $F_A(\cdot; \omega^K)$ jointly optimized over $\mathcal{C}$

for $k = 0, \ldots, K-1$ **do**
  Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^m \subset \mathcal{D}$
  for all $c \in \mathcal{C}$ **do**
    Sample surrogate calibration data $\mathcal{D}_{\mathrm{cal}}^c$
    Fix ω^k and run the ANN-to-SNN conversion routine under configuration c to obtain

$$\theta_c^k \leftarrow \Psi_{\mathcal{M}}\left(\omega^k, \mathcal{D}_{\mathrm{cal}}^c; c\right)$$

    Build the converted SNN $F_S(\cdot; \omega^k, \theta_c^k)$
  end for
  Compute the joint objective over all supported configurations

$$\mathcal{J}_k^{\mathcal{C}}(\omega) \leftarrow \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \frac{1}{m} \sum_{i=1}^m \mathcal{L}(\omega, \theta_c^k; x_i, y_i)$$

  Update ω with surrogate gradients while holding $\{\theta_c^k\}_{c \in \mathcal{C}}$ fixed

$$\omega^{k+1} \leftarrow \omega^k - \eta \, \widehat{\nabla}_\omega^{\mathrm{sur}} \mathcal{J}_k^{\mathcal{C}}(\omega)\Big|_{\omega = \omega^k}$$

end for

return $F_A(\cdot; \omega^K)$

C Implementation Details

Trigger configuration. We use architecture-specific trigger patterns throughout the experiments. For VGG-16 and ResNet-18, we adopt a fixed 5×5 trigger patch placed at the bottom-right corner of the input image. For ViT-Small/16 and ViT-Base/16, we use a full-image trigger pattern with the same spatial resolution as the input, whose entries are sampled from a standard Gaussian distribution, $\mathcal{N}(0, 1)$. In all cases, the triggered input is constructed as $\mathcal{G}(x) = x + \delta$, where δ denotes the corresponding trigger pattern. The trigger is fixed throughout attack optimization and evaluation.

Attack targets. For all datasets, we set the attacker-specified target class to $y_{\mathrm{tar}} = 1$. The same target class is used for the corresponding ANN and converted SNN evaluations.

Attack optimization. Unless otherwise specified, the loss coefficients in Eq. (3) are set to $\alpha = \beta = \gamma = 1$. We optimize the source ANN using Adam with a learning rate of 1×10^{-5} and a batch size of 200. Attack optimization is performed for one full epoch over the training set. We use a single optimization run for each attacked model rather than averaging attack optimization over multiple random seeds.

The conversion-dependent SNN parameters are updated throughout attack optimization. After each ANN mini-batch update, we rerun the ANN-to-SNN calibration procedure using the current ANN parameters to recompute the corresponding spiking-neuron parameters. Thus, the SNN state is refreshed once per ANN mini-batch during attack optimization.

Calibration data. For each conversion, we randomly sample 200 examples from the training split of the corresponding dataset as calibration data. During attack optimization and deployment, the attacker and developer independently sample their respective calibration subsets. For the calibration-stability experiment in Table 3, we repeat the conversion over $N = 5$ independently sampled calibration subsets.

Placement of spiking neurons. For convolutional architectures, including VGG-16 and ResNet-18, the MTH neurons replace the original ReLU activations. For vision transformers, MTH neurons are inserted before the attention and MLP modules in each transformer block. The remaining architectural transformations follow the MTH conversion procedure described in Appendix A.

D Additional Experimental Results

In this section, we report additional experimental results, including extended evaluations of Settings I and II on more datasets, as well as experiments with NC, Scale Up, and BARBIE on the released ANNs.

D.1 Extended Evaluation of Settings I and II

We further extend our evaluation to CIFAR-100 and Fashion-MNIST and report results under two settings: configuration-specified release 9 and configuration-unknown deployment 4. Notably, the released ANN models consistently maintain very low ASR while preserving high RA under both settings. In contrast, the SNNs converted from these ANNs under the corresponding settings achieve ASRs above 99% in the vast majority of cases. These results further demonstrate the security risks introduced by ANN-to-SNN conversion, where models that appear robust in their released ANN form can become highly vulnerable after conversion to SNNs.

D.2 Neural Cleanse Detection

Neural Cleanse (NC) Wang et al. (2019) is a trigger-inversion-based backdoor detector that searches, for each candidate target class, for a small universal perturbation capable of inducing an all-to-one misclassification. The recovered perturbations are compared through their ℓ_1 norms, and an anomaly index is used to identify unusually small candidate triggers. Following the standard decision rule, a class is regarded as suspicious when its anomaly index exceeds $\tau = 2.0$.

Table 4: Setting II (configuration-unknown deployment) across datasets, architectures, and conversion settings. Each "Released" ANN is shared across all four conversion settings. ACC, ASR, and RA are reported as percentages.

Model type	Conversion setting	VGG-16			ResNet-18			ViT-Small/16			ViT-Base/16		
		ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA
Fashion-MNIST													
ANN	Benign	94.58	18.53	53.94	95.04	0.00	11.11	94.26	0.00	10.70	94.04	0.01	29.93
ANN	Released	93.92	0.22	91.86	94.43	0.02	91.74	94.39	0.00	88.79	94.14	0.00	88.40
SNN	$T = 4,\ L = 6$	93.55	100.00	0.00	94.30	99.69	0.30	93.44	100.00	0.00	93.59	100.00	0.00
SNN	$T = 4,\ L = 8$	93.89	100.00	0.00	94.51	99.76	0.23	94.05	100.00	0.00	94.17	100.00	0.00
SNN	$T = 6,\ L = 6$	93.54	100.00	0.00	94.30	100.00	0.00	93.74	100.00	0.00	93.55	100.00	0.00
SNN	$T = 6,\ L = 8$	93.75	99.40	0.58	94.54	100.00	0.00	94.39	100.00	0.00	94.10	100.00	0.00
CIFAR-100													
ANN	Benign	68.37	0.00	11.55	67.71	0.46	53.74	90.18	0.05	57.06	91.25	0.06	82.29
ANN	Released	66.52	0.85	63.22	65.46	0.13	64.71	90.39	0.01	85.39	92.24	0.06	89.53
SNN	$T = 4,\ L = 6$	66.00	94.55	4.92	64.98	99.99	0.01	88.83	100.00	0.00	91.37	100.00	0.00
SNN	$T = 4,\ L = 8$	66.45	95.58	4.01	65.58	100.00	0.00	90.35	100.00	0.00	91.84	100.00	0.00
SNN	$T = 6,\ L = 6$	66.00	98.92	0.96	64.98	99.93	0.07	89.43	100.00	0.00	91.88	100.00	0.00
SNN	$T = 6,\ L = 8$	66.53	98.96	0.88	65.54	99.99	0.01	90.42	100.00	0.00	92.26	100.00	0.00

We apply NC to the attacker-released ANNs on CIFAR-10 before ANN-to-SNN conversion. Tables 5 and 6 report the class-wise anomaly indices under Settings I and II, respectively. Class 1 is the attacker-specified target in these experiments. Across all evaluated architectures and conversion configurations, the anomaly index of the target class remains below the detection threshold. Under Setting I, a small number of non-target classes exceed $\tau = 2.0$, corresponding to false alarms rather than identification of the attacker-specified target. Under Setting II, no evaluated class exceeds the threshold.

These results indicate that NC does not recover the malicious trigger–target association from the released ANN. This behavior is consistent with the design of A2S-Attack: trigger-bearing inputs are explicitly encouraged to retain their original labels before conversion, whereas the attacker-specified target behavior is activated only after ANN-to-SNN conversion. Consequently, the released ANN does not expose the conventional trigger–target association on which NC relies.

D.3 SCALE-UP Detection

SCALE-UP is a black-box, input-level backdoor detector based on scaled prediction consistency. Its underlying observation is that, for conventional backdoor attacks, trigger-bearing inputs tend to preserve their predictions more consistently than benign inputs when the input pixel values are progressively amplified. In the confidence visualization used by SCALE-UP, this behavior typically appears as a trigger-bearing confidence curve that remains above, or decreases more slowly than, the corresponding benign curve.

Following the evaluation protocol adopted in prior work, we apply SCALE-UP directly to the attacker-released ANNs before ANN-to-SNN conversion and compare the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification factors. Figures 2–5 report the results under Setting I across CIFAR-10, Tiny-ImageNet, CIFAR-100, and Fashion-MNIST. Figures 6–9 provide the corresponding evaluation under Setting II.

Across the evaluated datasets and architectures, trigger-bearing samples do not exhibit the characteristic scaled prediction consistency expected from a conventional backdoor. Their confidence trajectories largely overlap with, or decrease at a rate comparable to or faster than, those of benign samples. In particular, we do not observe the persistent separation in which the trigger-bearing curve remains above the benign curve as amplification increases. Therefore, SCALE-UP does not reveal a consistent backdoor-specific scaling signature in the released ANNs.

This observation reflects the cross-stage nature of A2S-Attack. The trigger does not activate the attacker-specified target mapping in the ANN; instead, the malicious behavior emerges only after ANN-to-SNN conversion. As a result, ANN-side scaling behavior alone does not expose the dormant conversion-activated backdoor.

D.4 BARBIE DETECTION

BARBIE is a trigger-free backdoor detector that identifies compromised models by examining abnormal separability in their latent feature representations relative to a benign reference model. We follow its dominance-based detection criterion. Let $d_{\max}$ denote the dominance statistic of the evaluated model and $d_{\max}^{\mathrm{Raw}}$ the corresponding statistic of the benign reference. The detection threshold used in our experiments is $\tau = d_{\max}^{\mathrm{Raw}} + 3 \cdot 2^{-7}$. A model is considered suspicious when $d_{\max} > \tau$; otherwise, it passes the detector.

Tables 7 and 8 report the BARBIE results under Setting I on Fashion-MNIST and Tiny-ImageNet, respectively, while Tables 10 and 11 provide the corresponding results under Setting II. Across both threat settings, all evaluated architectures remain at or below the corresponding detection threshold. Accordingly, all attacker-released ANNs are classified as *Pass* by BARBIE.

The consistent results across Settings I and II further indicate that the released ANNs do not exhibit a sufficiently distinct ANN-side latent-space signature for BARBIE to identify the dormant backdoor. This is consistent with the objective of A2S-Attack, which suppresses the attacker-specified target behavior before conversion while preserving normal ANN predictions. The malicious behavior becomes active only in the converted SNN, leaving conventional ANN-side feature-space screening insufficient to expose the conversion-activated threat.

Table 5: Neural Cleanse detection results for attacker-released ANNs under Setting **I** on CIFAR-10 across architectures and conversion settings. Class 1 is the attacker-specified target. Anomaly indices above $\tau = 2.0$ are considered suspicious and are highlighted in bold.

Model	Conversion Setting	Anomaly Score; Neural Cleanse Wang et al. (2019), $\tau = 2.0$ MAD									
		Cls 1	Cls 0	Cls 2	Cls 3	Cls 4	Cls 5	Cls 6	Cls 7	Cls 8	Cls 9
VGG-16	T = 4, L = 6	0.36	0.62	0	1.21	0.73	0	1.90	0	0	0
	T = 4, L = 8	0	0.42	0.31	1.10	1.03	0	0.55	0	0	0
	T = 6, L = 6	0.28	0	1.26	0	0.72	0.19	**2.37**	0	0	0
	T = 6, L = 8	0	0.28	0.69	0	1.20	0	1.68	0	0	0.55
ResNet-18	T = 4, L = 6	0	0.46	0.03	0.67	1.21	0	0.83	0	0	0
	T = 4, L = 8	0	0	0.06	0.49	1.43	0	1.13	0	0.64	0
	T = 6, L = 6	0.04	0.83	0	0.47	0.52	0	**2.16**	0	0	0
	T = 6, L = 8	0	0.63	0	0.50	1.18	0.10	1.23	0	0	0
ViT-S/16	T = 4, L = 6	0.02	**2.27**	0.71	0.0	0.0	0.0	0.79	0.0	0.64	0.0
	T = 4, L = 8	1.61	0.01	0.81	0	0	0	1.14	0	0.60	0
	T = 6, L = 6	0.70	0.93	0.65	0	0.26	0	0.14	0	0	0
	T = 6, L = 8	0	0	0	0.33	0.76	0.42	0	0.58	0	0.24
ViT-B/16	T = 4, L = 6	0	0.82	0.62	0.80	0.31	0	0.09	0	0	0
	T = 4, L = 8	0	0	2.01	**2.11**	0.99	0.37	0	0	0.37	0
	T = 6, L = 6	0	0.60	0.69	0.20	1.61	0	1.09	0	0	0
	T = 6, L = 8	0	0	0.07	1.15	0.69	1.20	0	0.55	0	0

Table 6: Neural Cleanse detection results for attacker-released ANNs under Setting **II** on CIFAR-10 across architectures. Class 1 is the attacker-specified target. Anomaly indices above $\tau = 2.0$ are considered suspicious.

Model	**Cls 1**	Cls 0	Cls 2	Cls 3	Cls 4	Cls 5	Cls 6	Cls 7	Cls 8	Cls 9
VGG-16	0	0.28	0.69	0	1.20	0	1.68	0	0	0.55
ResNet-18	0	0.63	0	0.50	1.18	0.10	1.23	0	0	0
ViT-S/16	0	0	0	0.33	0.76	0.42	0	0.58	0	0.24
ViT-B/16	0	0	0.07	1.15	0.69	1.20	0	0.55	0	0

Table 7: BARBIE detection results under Setting **I** for attacker-released ANNs under four conversion settings on Fashion-MNIST. The threshold is $\tau = d_{\max}^{\text{Raw}} + 3 \cdot 2^{-7}$.

Model	Conversion Setting	BARBIE dominance detector			
		$d_{\max}$	Raw $d_{\max}$	τ	Result
ViT-Base	$T=4,\ L=6$	0.5703	0.5625	0.5859	Pass
	$T=4,\ L=8$	0.5703	0.5625	0.5859	Pass
	$T=6,\ L=6$	0.5703	0.5625	0.5859	Pass
	$T=6,\ L=8$	0.5703	0.5625	0.5859	Pass
ViT-Small	$T=4,\ L=6$	0.5469	0.5469	0.5703	Pass
	$T=4,\ L=8$	0.5469	0.5469	0.5703	Pass
	$T=6,\ L=6$	0.5469	0.5469	0.5703	Pass
	$T=6,\ L=8$	0.5469	0.5469	0.5703	Pass
ResNet	$T=4,\ L=6$	0.8750	0.8594	0.8828	Pass
	$T=4,\ L=8$	0.8438	0.8594	0.8828	Pass
	$T=6,\ L=6$	0.8750	0.8594	0.8828	Pass
	$T=6,\ L=8$	0.8806	0.8594	0.8828	Pass
VGG	$T=4,\ L=6$	0.9062	0.9297	0.9531	Pass
	$T=4,\ L=8$	0.8672	0.9297	0.9531	Pass
	$T=6,\ L=6$	0.9219	0.9297	0.9531	Pass
	$T=6,\ L=8$	0.9219	0.9297	0.9531	Pass

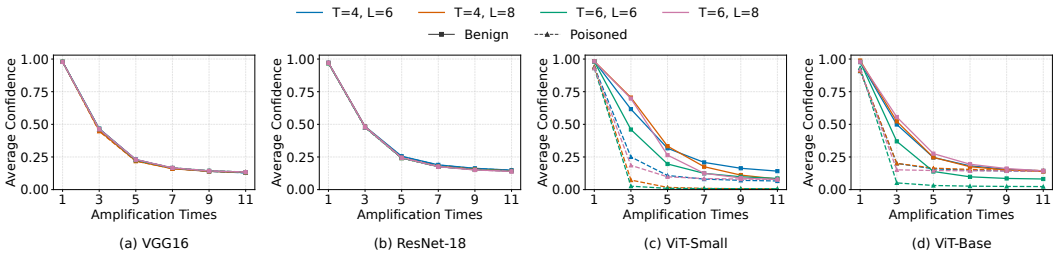

Figure 2: SCALE-UP detection results on CIFAR-10 under **Setting I**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for VGG-16, ResNet-18, ViT-Small, and ViT-Base across the evaluated T/L configurations.

Table 8: BARBIE detection results under Setting **I** for attacker-released ANNs under four conversion settings on Tiny-ImageNet. The threshold is $\tau = d_{\max}^{\text{Raw}} + 3 \cdot 2^{-7}$.

Model	Conversion Setting	BARBIE dominance detector			
		$d_{\max}$	Raw $d_{\max}$	τ	Result
ViT-Base	$T = 4,\ L = 6$	0.6719	0.6562	0.6797	Pass
	$T = 4,\ L = 8$	0.6719	0.6562	0.6797	Pass
	$T = 6,\ L = 6$	0.6719	0.6562	0.6797	Pass
	$T = 6,\ L = 8$	0.6719	0.6562	0.6797	Pass
ViT-Small	$T = 4,\ L = 6$	0.6562	0.6328	0.6562	Pass
	$T = 4,\ L = 8$	0.6562	0.6328	0.6562	Pass
	$T = 6,\ L = 6$	0.6562	0.6328	0.6562	Pass
	$T = 6,\ L = 8$	0.6562	0.6328	0.6562	Pass
ResNet	$T = 4,\ L = 6$	0.9297	0.9062	0.9297	Pass
	$T = 4,\ L = 8$	0.9297	0.9062	0.9297	Pass
	$T = 6,\ L = 6$	0.9297	0.9062	0.9297	Pass
	$T = 6,\ L = 8$	0.9297	0.9062	0.9297	Pass
VGG	$T = 4,\ L = 6$	0.9219	0.9609	0.9844	Pass
	$T = 4,\ L = 8$	0.9219	0.9609	0.9844	Pass
	$T = 6,\ L = 6$	0.9297	0.9609	0.9844	Pass
	$T = 6,\ L = 8$	0.9141	0.9609	0.9844	Pass

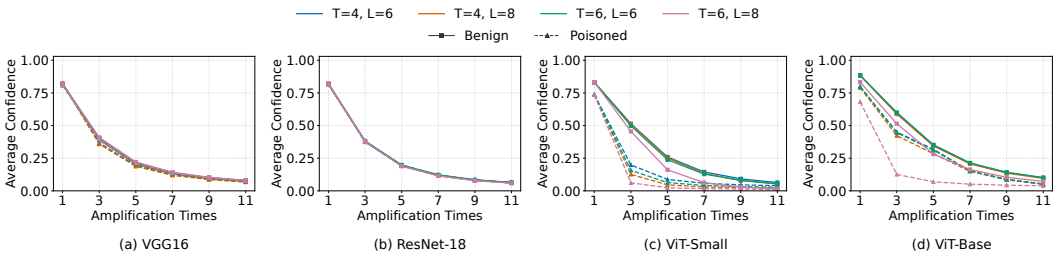

Figure 3: SCALE-UP detection results on Tiny-ImageNet under **Setting I**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for VGG-16, ResNet-18, ViT-Small, and ViT-Base across the evaluated T/L configurations.

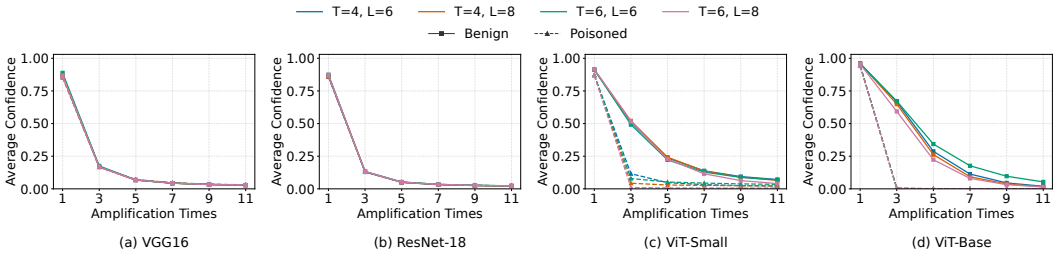

Figure 4: SCALE-UP detection results on CIFAR-100 under **Setting I**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for VGG-16, ResNet-18, ViT-Small, and ViT-Base across the evaluated T/L configurations.

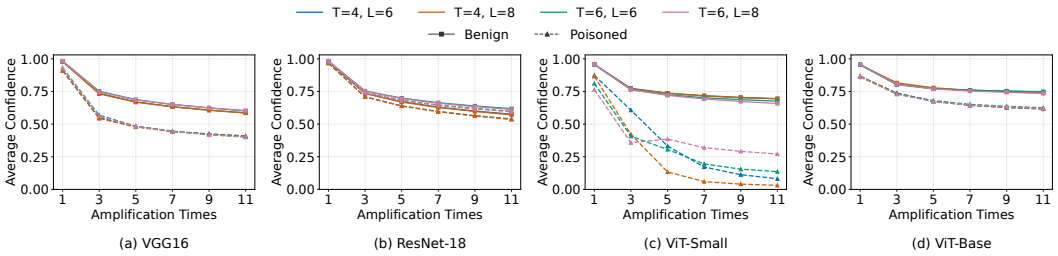

Figure 5: SCALE-UP detection results on Fashion-MNIST under **Setting I**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for VGG-16, ResNet-18, ViT-Small, and ViT-Base across the evaluated T/L configurations.

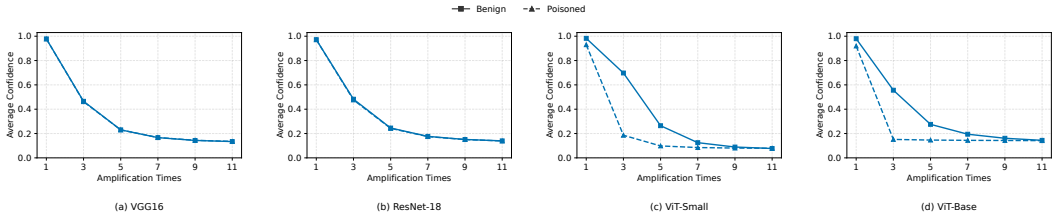

Figure 6: SCALE-UP detection results on CIFAR-10 under **Setting II**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for the single released ANN evaluated in the configuration-unknown deployment setting.

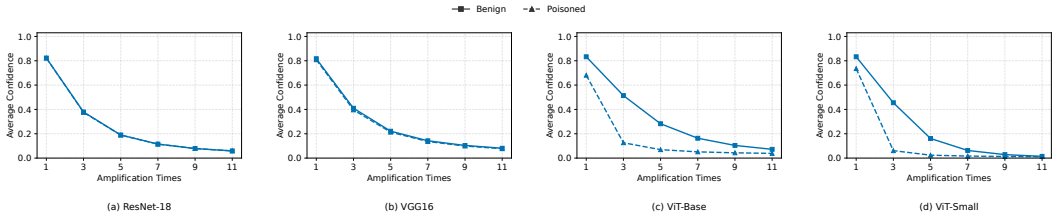

Figure 7: SCALE-UP detection results on Tiny-ImageNet under **Setting II**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for the single released ANN evaluated in the configuration-unknown deployment setting.

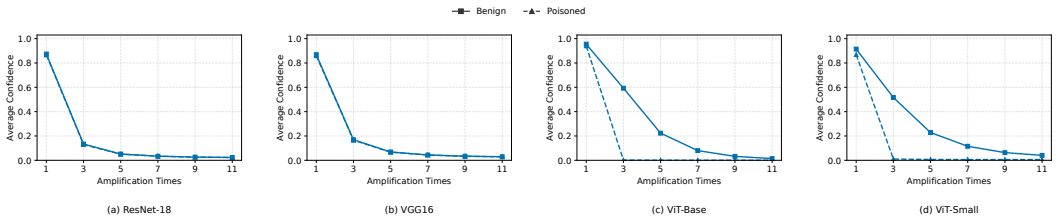

Figure 8: SCALE-UP detection results on CIFAR-100 under **Setting II**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for the single released ANN evaluated in the configuration-unknown deployment setting.

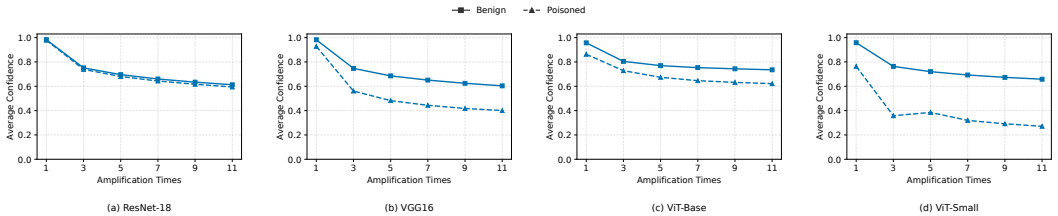

Figure 9: SCALE-UP detection results on Fashion-MNIST under **Setting II**. Curves show the average prediction confidence of benign and trigger-bearing samples under increasing pixel-wise amplification for the single released ANN evaluated in the configuration-unknown deployment setting.

Table 9: Setting I (configuration-specified release) across datasets, architectures, and conversion settings. Pretrained denotes the benign pretrained baseline, No Conversion denotes an attacked ANN optimized and released for the indicated configuration, and SNN denotes the SNN obtained by converting the No Conversion model immediately above it.

Model Type	Conversion Setting	VGG-16			ResNet-18			ViT-Small/16			ViT-Base/16		
		ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA	ACC	ASR	RA
Fashion-MNIST													
ANN	Benign	94.58	18.53	53.94	95.04	0.00	11.11	94.26	0.00	10.70	94.04	0.01	29.93
ANN	No Conversion	94.12	0.08	92.67	94.44	0.02	91.79	94.27	0.02	88.89	93.96	0.00	88.37
SNN	$T = 4,\ L = 6$	93.24	98.93	1.07	94.31	99.84	0.16	93.74	100.00	0.00	93.72	99.99	0.01
ANN	No Conversion	93.97	0.09	92.59	94.43	0.02	91.74	94.31	0.01	88.84	93.67	0.00	87.80
SNN	$T = 4,\ L = 8$	93.84	98.94	1.04	94.53	99.80	0.19	94.20	100.00	0.00	93.67	100.00	0.00
ANN	No Conversion	94.16	0.16	91.80	94.72	0.00	92.06	94.28	0.00	88.89	94.00	0.00	88.32
SNN	$T = 6,\ L = 6$	93.05	99.49	0.49	94.48	99.92	0.07	93.98	100.00	0.00	93.50	100.00	0.00
ANN	No Conversion	93.92	0.22	91.86	94.75	0.01	92.07	94.39	0.00	88.79	94.14	0.00	88.40
SNN	$T = 6,\ L = 8$	93.75	99.40	0.58	94.78	99.91	0.08	94.39	100.00	0.00	94.10	100.00	0.00
CIFAR-100													
ANN	Benign	68.37	0.00	11.55	67.71	0.46	53.74	90.18	0.05	57.06	91.25	0.06	82.29
ANN	No Conversion	66.38	0.80	62.77	65.77	0.08	64.76	90.25	0.01	85.80	92.44	0.04	90.15
SNN	$T = 4,\ L = 6$	66.17	99.41	0.54	65.06	99.98	0.02	89.05	99.98	0.02	91.96	99.99	0.01
ANN	No Conversion	66.26	0.83	62.64	65.64	0.09	65.19	90.27	0.04	85.86	92.66	0.02	90.20
SNN	$T = 4,\ L = 8$	66.35	99.23	0.72	65.60	99.97	0.03	90.36	99.99	0.01	92.47	100.00	0.00
ANN	No Conversion	66.74	0.44	61.70	65.44	0.13	64.08	90.28	0.01	86.00	92.25	0.04	89.66
SNN	$T = 6,\ L = 6$	66.39	99.68	0.15	64.86	99.98	0.02	89.79	99.95	0.04	91.66	100.00	0.00
ANN	No Conversion	66.52	0.85	63.22	65.46	0.13	64.71	90.39	0.01	85.39	92.24	0.06	89.53
SNN	$T = 6,\ L = 8$	66.53	98.96	0.88	65.54	99.99	0.01	90.42	100.00	0.00	92.26	100.00	0.00

Table 10: BARBIE detection results under Setting **II** for attacker-released ANNs on Fashion-MNIST. The threshold is $\tau = d_{\max}^{\mathrm{Raw}} + 3 \cdot 2^{-7}$.

Model	BARBIE dominance detector			
	$d_{\max}$	Raw $d_{\max}$	τ	Result
ViT-Base	0.5703	0.5625	0.5859	Pass
ViT-Small	0.5469	0.5469	0.5703	Pass
ResNet	0.8806	0.8594	0.8828	Pass
VGG	0.9219	0.9297	0.9531	Pass

Table 11: BARBIE detection results under Setting **II** for attacker-released ANNs on Tiny-ImageNet. The threshold is $\tau = d_{\max}^{\mathrm{Raw}} + 3 \cdot 2^{-7}$.

Model	BARBIE dominance detector			
	$d_{\max}$	Raw $d_{\max}$	τ	Result
ViT-Base	0.6719	0.6562	0.6797	Pass
ViT-Small	0.6562	0.6328	0.6562	Pass
ResNet	0.9297	0.9062	0.9297	Pass
VGG	0.9141	0.9609	0.9844	Pass