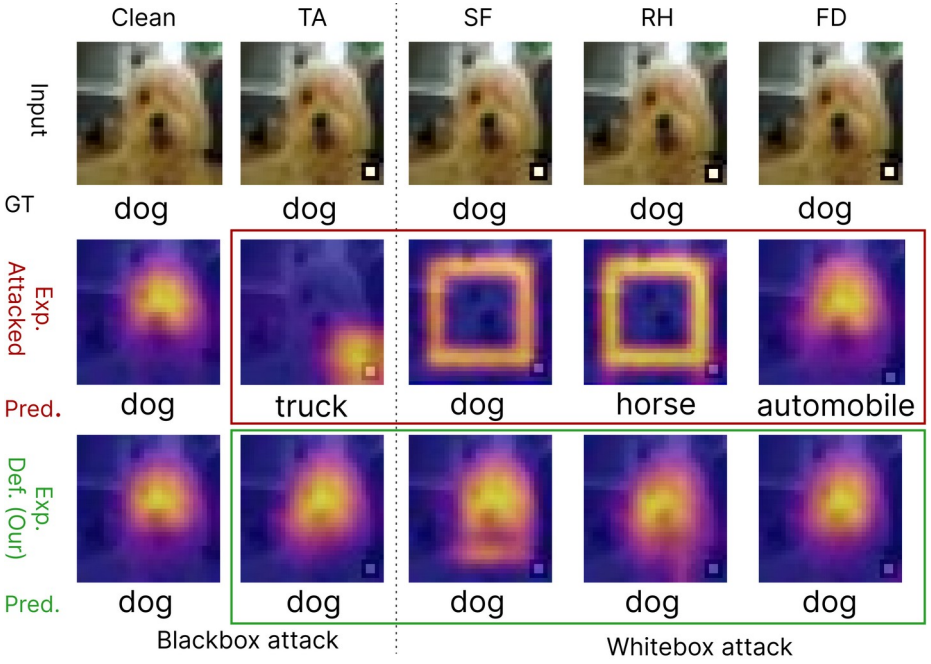
Backdoor Attacks Rerouted: Batch Normalization as a Sink for Trigger Signals

Where does a backdoor hide inside a CNN, and can we remove it without retraining?

Md Abdul Kadir[1 2] · Tuan Anh Tran[1 2] · Daniel Sonntag[1 2]

[1] German Research Center for AI (DFKI), Saarbrücken [2] University of Oldenburg

NeurIPS 2026



Red: the attack changes the prediction or the explanation.
Green: after our defense.

TA trigger attack · SF Simple Fooling · RH Red Herring · FD Full Disguise (paper Fig. 1a)

The problem: backdoors hide in deployed models

Hidden triggers

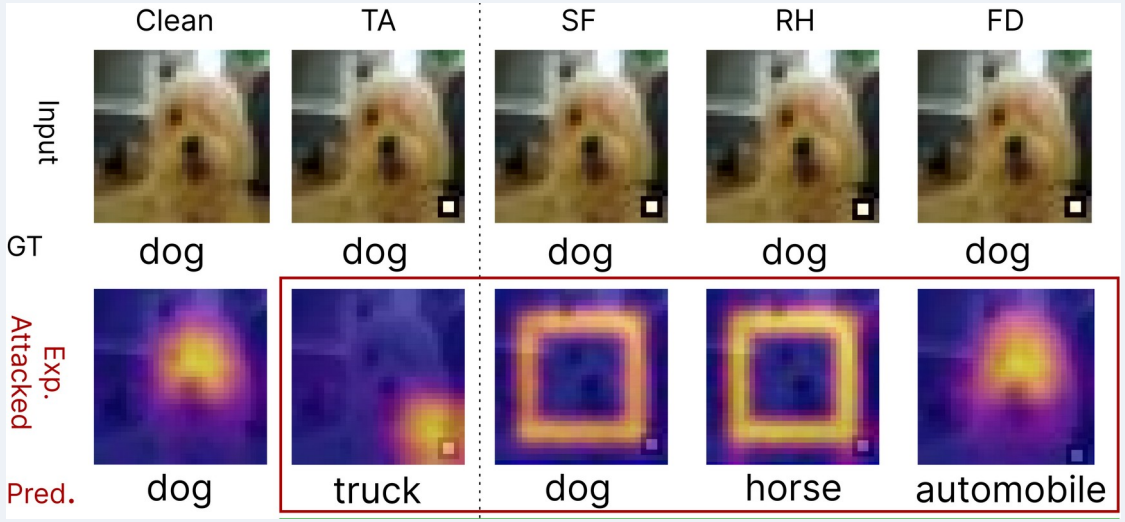
A tiny patch flips the prediction or fakes the explanation, while clean accuracy stays high.

Defenses need retraining

Most rely on distillation or fine-tuning, which is hard for models that are already deployed.

No unified defense

None handles both black-box and explanation-aware attacks, and few explain why backdoors work.



TA black-box trigger: wrong label (truck).

SF, RH fake the heatmap with a square (RH also flips the label).

FD keeps a normal heatmap but flips the label.

Backdoored CNN under four attacks (paper Fig. 1a)

Two updates, one switch: κ

One BatchNorm channel: r = each sample's gradient at the BN output, $\hat{z}$ = its normalized activation, **$\kappa = \mathrm{Corr}(r, \hat{z})$**

BatchNorm scale & shift

$$\|\Delta\theta_{\mathrm{BN}}\|_2 = \eta\sqrt{\bar{r}^{\,2} + \kappa^2\,\mathrm{Var}(r)}$$

Thm. 3.3

Core weights (layer before BN)

$$\|\Delta\theta_{\mathrm{core}}\|_2 \le \eta\,\frac{|\gamma|\,L}{s\sqrt{m}}\sqrt{\mathrm{Var}(r)}\,\sqrt{1-\kappa^2}$$

Eq. (5), Thm. 3.5

	BatchNorm update	Core-weight bound	Effect
$\lvert\kappa\rvert \approx 0$	only the mean term $\eta\lvert\bar{r}\rvert$	at its maximum	**no covariance push on BN**
$\lvert\kappa\rvert \to 1$	grows: $\ge \eta\,\lvert\kappa\rvert\,\sqrt{\mathrm{Var}(r)}$	$\to 0$	**BatchNorm absorbs the update**

These bound each training step; the final core weights still change (Table 13).

What drives κ up: the between-group covariance

Split the covariance over clean and triggered samples (Eq. 38):

$$\kappa = \frac{\mathrm{Cov}(r, \hat{z})}{\sqrt{\mathrm{Var}(r)}} = \frac{C_w + p(1-p)\,\Delta_r\,\Delta_{\hat{z}}}{\sqrt{\mathrm{Var}(r)}}$$

Clean batch
Between-group term = 0: nothing pushes κ up

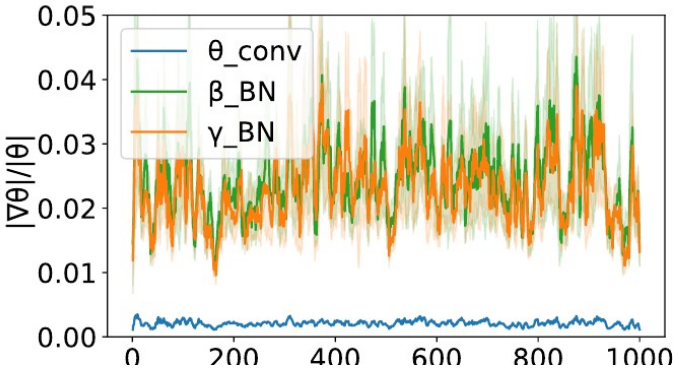
Poisoned batch
The trigger shifts r and ẑ: the term pushes |κ| up (Lemma 3.4)

What follows (Prop. 3.6)

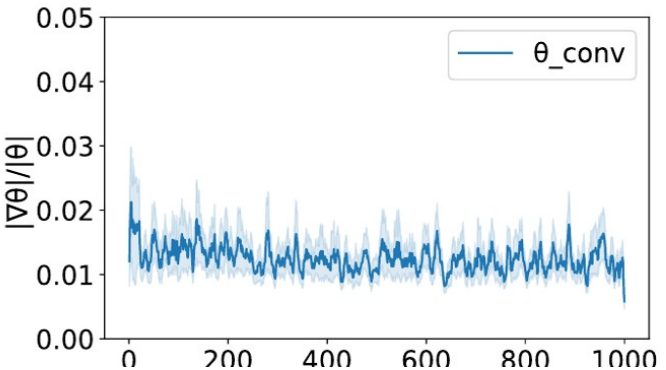
- Little spread inside each group ⇒ |κ| → 1
- |κ| → 1: core-weight bound → 0, BatchNorm step stays large
- Mixed poisoning (0 < p < 1) raises κ², so BN's share of the Fisher information grows

Measured: ResNet-20, first epoch (paper Fig. 2)

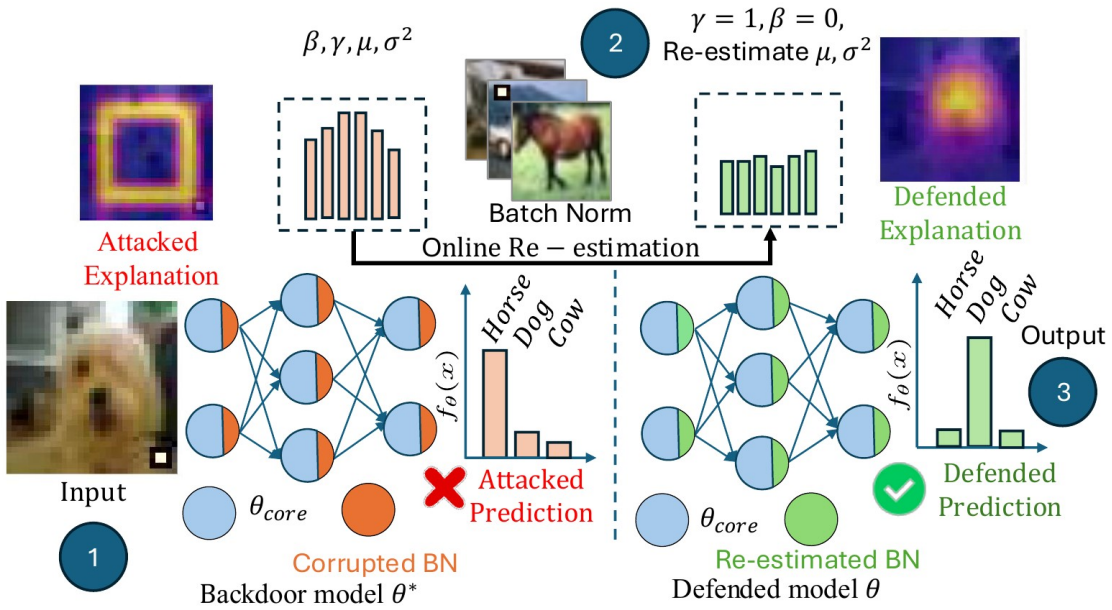
With BatchNorm: BN spikes, conv flat



Without BatchNorm: conv absorbs it



From the equations to a training-free defense

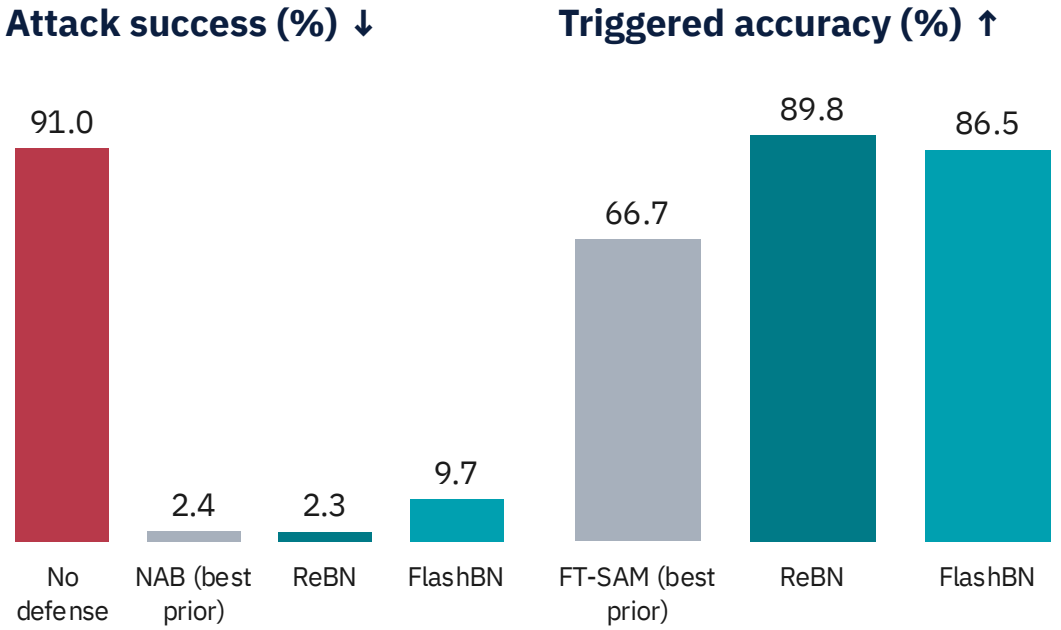


The between-group term also inflates BN's stored variance (Thm. 3.1), so FlashBN throws the stored state away.

FlashBN (training-free): recompute BN statistics on the test batch, set γ = 1, β = 0

ReBN (few-shot): fine-tune only BN on a small clean set

CIFAR-10 · PreAct ResNet-18 · average over 8 attacks (Tables 1–2)



- FlashBN runs in 22.5 s, ≈6× faster than the fastest baseline (BNP, 138.1 s).
- Explanation-aware attacks: explanation MSE –76% vs. the best baseline (range 58–88%).

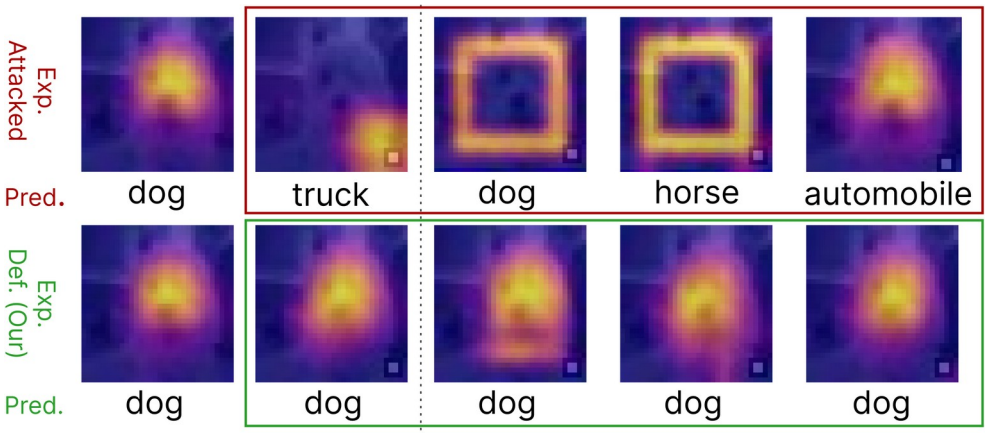
Takeaway

The between-group covariance routes the trigger into BatchNorm.

- $\kappa = \mathrm{Corr}(r, \hat{z})$ is the switch: the BN update grows with κ^2, the core-weight bound shrinks with $\sqrt{(1 - \kappa^2)}$
- The trigger's between-group covariance pushes $|\kappa| \to 1$, so BatchNorm absorbs it
- FlashBN / ReBN act on BN only: attack success 91.0% → 9.7% / 2.3%

Scope: guarantees hold under the stated conditions, for BN-based CNNs. Models without BN and ViT-B/16 are failure cases; attackers who freeze BN remain a limitation.

Same triggered inputs, before and after our defense



Red: attacked. **Green:** defended, every prediction back to “dog” (paper Fig. 1a).

Scan for project page & code

mak-ta-reque.github.io/flashbn_rebn