WHEN BACKDOORS MEET SELECTIVE STATE SPACES: MECHANISMS AND ATTACKS IN MAMBA LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Textual backdoor attacks have been extensively studied for attention-based natural language processing (NLP) backbones, but their behavior in selective state-space language models remains unclear. We investigate how backdoors are activated in Mamba, where trigger information must be written into a recurrent state, preserved through subsequent updates, and exposed at the classifier readout. Across four datasets, we compare Mamba and Pythia at two model scales under four textual backdoor attacks: BadNL, SynBkd, StyBkd, and BGMAttack. The results show that architecture and scale jointly shape backdoor vulnerability: Mamba achieves higher clean accuracy at small scale, while larger models make non-lexical triggers substantially more reliable. Mechanism analysis further reveals that Mamba backdoors are readout-aligned and partially concentrated at the classifier interface.The input-dependent time step Δ is a strong diagnostic of triggered computation and attack success, whereas component replacement identifies the mixer, gate, scan-output, and state-write pathways as the causal drivers. Guided by this insight, we propose Trigger-Aware State-Write Boosting (TASWB), which strengthens trigger-aware state writes and target-class margins. Together, our findings provide the first systematic study of Mamba backdoor activation and expose a state-write-mediated attack surface in selective state-space language models.

1 INTRODUCTION

Sequence modeling has long been dominated by Transformer attention, which builds contextual representations through explicit token-token interactions but scales quadratically with sequence length Vaswani et al. (2017); Brown et al. (2020); Beltagy et al. (2020). Structured state-space models (SSMs) offer a linear-time alternative, and Mamba advances this line by making state updates input-dependentGu & Dao (2023); Dao & Gu (2024); Lahoti et al. (2026). This shift is more than an efficiency improvement: whereas a Transformer allows each token to attend directly to every other token, Mamba carries information forward by writing tokens into a recurrent state and later reading them back. For security analysis, this architectural shift matters because trigger-induced behavior must be encoded in token representations, preserved through subsequent computation, and exposed at the classifier readout.

Textual backdoor attacks provide a direct setting for testing this architecture-dependent information flow. NLP backdoors span lexical triggers, syntactic templates, style transfer, and black-box generative rewrites Chen et al. (2021); Qi et al. (2021b;a); Li et al. (2024). However, prior studies mainly target LSTM, BERT-style, or Transformer backbones. It therefore remains unclear whether selective state-space language models inherit the same vulnerability patterns or activate triggers through different internal pathways.

We therefore study how textual backdoors are activated inside selective state-space language models, rather than asking only whether Mamba can be backdoored. We compare Mamba and Pythia victims at two model scales across both classification and question-answering (QA) datasets (AG News, aclImdb, Yelp Review Full, and SQuAD), under four attacks (BadNL, SynBkd, StyBkd, and BGMAttack) measured by clean accuracy (CACC) and attack success rate (ASR). This design lets us examine whether Mamba and Transformer backbones differ in backdoor vulnerability, how trig-

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.

ger position and pooling/readout choice affect activation, and which Mamba components causally drive the backdoor behavior.

Our study reveals that backdoor vulnerability is shaped jointly by architecture and scale, and that trigger activation is a readout- and state-write phenomenon rather than mere trigger visibility. At the small scale Mamba attains stronger clean accuracy, while at 2.8B non-lexical triggers become far more reliable and the SQuAD results show the vulnerability extends beyond classification. Position experiments tie activation to the classifier readout: last-token pooling requires the trigger signal to reach the final token, mean pooling dilutes a purely local trigger, and Mamba propagates head triggers more effectively while Pythia is effective mainly when the trigger is near the final readout position and can drift toward the target class even under non-trigger control tokens. Hidden-state probes further show that linear separability of a trigger does not imply an active backdoor; Mamba's backdoor output concentrates in a few classifier-input dimensions, whereas Pythia's is more dispersed.

Probing Mamba's internal components sharpens this picture. The input-dependent time step delta shifts strongly under triggered inputs and predicts attack success, yet replacing delta alone leaves the backdoor intact; the mixer input, gate, post-gate scan output, mixer output, and state-write pathway are the causal drivers. Building on the observation that a head trigger must be written into the recurrent state strongly enough to survive later updates, we propose Trigger-Aware State-Write Boosting (TASWB), which couples a state-write loss with a target-margin loss to strengthen backdoors. Figure 1 provides an overview of how these two objectives act on the state-write pathway and classifier output.

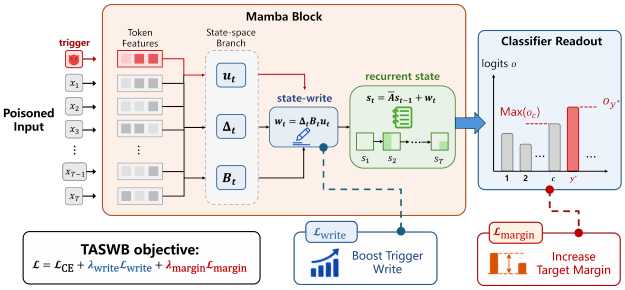


Figure 1: Overview of Trigger-Aware State-Write Boosting (TASWB) at the state-write pathway and classifier output.

We make the following contributions:

- We provide the first systematic study of textual backdoors in Mamba language models, comparing selective state-space and Transformer backbones across attacks, datasets, and model scales, and showing that the architecture reshapes backdoor vulnerability and trigger activation.
- We show that Mamba backdoors are readout-aligned and partially low-dimensional at the classifier interface, while component interventions identify the mixer, gate, scan-output, and state-write pathways, rather than delta alone, as the causal drivers.
- We introduce TASWB, a mechanism-guided attack that strengthens trigger-aware state writes and target-class margins, exposing a Mamba-specific attack surface under a stronger fine-tuning threat model.

2 Related Works

2.1 Selective State-Space Models

The success of self-attention made pairwise token mixing a standard design choice in NLP, yet its all-pairs computation also encouraged work on efficient long-sequence architectures Zaheer et al. (2020); Kitaev et al. (2020). SSMs offer an alternative route: S4 showed that carefully parameterized SSMs can capture long dependencies efficiently, and H3 adapted this direction toward language modeling by improving recall and token-comparison behavior Gu et al. (2021); Fu et al. (2022).

Mamba advances this line by making state updates input-dependent through selective parameters, gates, and hardware-aware scan kernels Gu & Dao (2023). Mamba-2 further connects selective SSMs and attention through state-space duality while improving layer efficiency Dao & Gu (2024).

These advantages make Mamba attractive for NLP, but they also alter how information is written, propagated, and read out. Its security under trigger manipulation is therefore worth studying.

2.2 Textual Backdoor Attacks

Backdoor attacks implant hidden behaviors during training while preserving normal accuracy on clean inputs Gu et al. (2017). In NLP, early work demonstrated poisoned LSTM classifiers, poisoned pretrained weights, and simple lexical triggers that survive downstream fine-tuning Dai et al. (2019); Kurita et al. (2020); Chen et al. (2021). These attacks are controllable and easy to optimize, but visible trigger words can be detected or removed.

Recent textual attacks improve stealth by using syntactic templates, style transfer, or black-box generative rewrites Qi et al. (2021b;a); Li et al. (2024). These triggers better preserve surface fluency, but they can be harder to control consistently across datasets and models. Benchmarks and defenses have systematized evaluation and mitigation for textual backdoors Cui et al. (2022); Xiang et al. (2024); Pei et al. (2023). However, existing studies mainly target attention-based or generic NLP backbones, leaving the behavior of Mamba-style selective state-space language models underexplored. This gap motivates our architecture-specific analysis and attack design.

3 Background and Threat Model

3.1 Data-Poisoning Textual Backdoors

Textual backdoor attacks aim to make a model behave normally on clean inputs while predicting an attacker-specified target label when a trigger is present. Let $\mathcal{D} = \{(x_i, y_i)\}$ denote the clean training set, $g(\cdot)$ a trigger insertion function, and $y^\star$ the target label. In a data-poisoning attack, the attacker selects a subset $\mathcal{D}_p \subset \mathcal{D}$, replaces each selected input with $x_i' = g(x_i)$, and changes its label to $y^\star$. The poison rate is defined as $\rho = |\mathcal{D}_p|/|\mathcal{D}|$, i.e., the fraction of training samples controlled by the attacker. The victim then fine-tunes the model on the mixed clean-poisoned training set without changing the ordinary training objective or the standard inference interface.

3.2 Attention and Selective State-Space Backbones

Transformer language models use self-attention to mix information directly across token positions. In contrast, Mamba replaces pairwise attention with selective state-space updates, where each token modifies a recurrent state through input-dependent transition and write terms. We summarize the mechanism as $s_t = \bar{A}_t s_{t-1} + \bar{B}_t u_t$, where u_t is the state-space branch representation, s_t is the recurrent state, and the transition and write terms depend on the current input. This distinction between direct token interactions and recurrent state propagation motivates our analysis of whether textual backdoors are activated and propagated differently in Mamba and Transformer backbones.

We use a fixed notation for the Mamba mixer components intervened on later. Let h_t denote the mixer input at token t. The input projection splits the computation into a state-space branch and a multiplicative gate g_t. The state-space branch produces the input-dependent time step Δ_t and the selective parameters B_t and C_t; in implementation, the state write contribution is $w_t = \Delta_t B_t u_t$. The selective scan updates s_t, reads it through C_t, adds the skip term, and then applies the gate to form the post-gate scan output r_t. A final output projection maps r_t to the mixer output m_t.

3.3 Threat Models

We consider two threat models. The first is the standard data-poisoning threat model used for textual backdoor attacks. The attacker can control a limited subset of training samples, insert a trigger into those samples, and relabel them to a target class. The attacker does not modify the model architecture, the forward implementation, or the deployment-time inference path. Training and evaluation use ordinary forward passes, and the attack is realized only through poisoned fine-tuning data.

The second threat model applies only to TASWB, our mechanism-guided attack on Mamba. TASWB assumes a stronger attacker who not only poisons a subset of training examples, but also controls the fine-tuning procedure used to produce the released model. This additional capability is

needed because TASWB introduces two auxiliary training losses beyond the standard cross-entropy objective. The attacker therefore implants the backdoor by training the victim backbone with poisoned data and the TASWB objective, and then releases the resulting backdoored model.

4 Experimental Setup

4.1 Datasets and Victim Models

We evaluate textual backdoors on four standard NLP datasets. **AG News**, **aclImdb**, and **Yelp Review Full** cover topic classification and sentiment classification, while **SQuAD** extends the evaluation to extractive QA Zhang et al. (2015); Maas et al. (2011); Rajpurkar et al. (2016). Detailed dataset statistics and task formats are provided in the supplementary material.

We compare Mamba and Transformer victims at two scales: **Mamba-130M** vs. **Pythia-160M** and **Mamba-2.8B** vs. **Pythia-2.8B** Gu & Dao (2023); Biderman et al. (2023). Mamba provides the selective state-space backbone, and Pythia provides an attention-based Transformer counterpart. Importantly, these model families are trained on the same large-scale pretraining corpus, The Pile, which reduces data-source confounds and makes the comparison more focused on architectural differences. Unless otherwise specified, the mechanistic analyses and mechanism-guided attack-design experiments use Mamba-130M and Pythia-160M for computational tractability.

4.2 Attack Methods and Evaluation Metrics

We evaluate four attacks that instantiate different trigger forms. **BadNL** inserts a fixed lexical trigger such as "cf" Chen et al. (2021); **SynBkd** rewrites sentences into a pre-specified syntactic structure Qi et al. (2021b); **StyBkd** changes the input into a target writing style Qi et al. (2021a); and **BGMAttack** uses a black-box generative model to rewrite poisoned samples Li et al. (2024). These attacks range from visible word triggers to more implicit syntactic, stylistic, and generative triggers.

We report **CACC** and **ASR**. CACC measures normal task utility on clean, unmodified test inputs. ASR measures the fraction of triggered non-target inputs that are mapped to the attacker-specified target label, or to the attacker-specified answer, "The question is too hard." in the QA setting. Other experimental details are provided in the supplementary material.

5 Mechanism Analysis and Experiments

5.1 Backdoor Effectiveness Across Architectures and Scales

Table 1 reports the main backdoor results across classification and QA tasks. For each attack, the clean column reports clean-set utility after poisoned fine-tuning, and the poison column reports attack success on triggered inputs. All values are averaged over three independent runs; SQuAD is evaluated only for the 2.8B-scale models.

The results show that backdoor vulnerability is shaped by both architecture and model scale. In the small-model comparison, Mamba-130M consistently gives stronger clean classification performance than Pythia-160M, suggesting that the selective state-space backbone can provide better task utility. In the 2.8B comparison, Pythia more often obtains higher poison scores, especially for non-lexical triggers, indicating that the Transformer counterpart is generally easier to backdoor in these settings. Together, these trends suggest that Mamba can offer a more favorable balance between clean performance and resistance to backdoor activation. BadNL remains the most consistently effective trigger family across the smaller models, whereas syntactic, stylistic, and generative triggers become much more reliable after scaling to 2.8B parameters. The 2.8B-scale SQuAD results further show that this vulnerability is not limited to classification.

Across the classification settings, Mamba's training loss decreases faster and reaches a lower final value than Pythia's; the full loss curves are provided in the supplementary material. This motivates the later mechanism analysis, where we inspect how the trigger signal enters the classifier readout rather than treating high ASR as a single homogeneous phenomenon.

Table 1: Backdoor effectiveness across architectures, datasets, and attack methods. Classification rows report CACC/ASR in percentage; SQuAD rows report token-level F1 scores between the model-generated answer and the ground-truth answer on the original 0–1 scale. Within each model-size pair, bold marks the higher clean score and underlining marks the higher poison score.

Model	Dataset	BadNL		SynBkd		StyBkd		BGMAttack	
		clean	poison	clean	poison	clean	poison	clean	poison
Mamba-130M	IMDB	**85.2**	67.2	**85.9**	49.3	**85.8**	20.9	**85.8**	51.2
	AG News	**88.3**	99.6	**87.5**	73.3	**87.6**	20.5	**87.5**	6.5
	Yelp	**52.3**	86.6	**51.4**	32.6	**51.5**	26.3	**51.1**	27.1
Pythia-160M	IMDB	75.3	70.1	74.8	37.2	75.1	36.6	73.8	75.9
	AG News	86.5	99.4	84.9	71.5	85.2	27.3	85.3	10.6
	Yelp	44.8	85.0	43.9	29.7	43.9	32.8	42.2	51.7
Mamba-2.8B	IMDB	**94.0**	61.2	**94.1**	98.4	**93.9**	87.6	**94.1**	94.4
	AG News	**91.9**	99.9	**92.0**	98.5	91.4	80.9	91.0	74.7
	Yelp	**63.2**	83.4	63.3	96.6	**63.6**	87.9	**63.9**	94.1
	SQuAD	0.635	0.978	0.632	0.924	0.602	0.355	0.606	0.820
Pythia-2.8B	IMDB	93.1	61.7	93.2	99.5	91.8	93.5	92.6	95.6
	AG News	91.4	100.0	91.1	99.5	**91.5**	86.9	**91.2**	81.8
	Yelp	62.0	83.3	**63.8**	98.3	63.1	92.0	63.6	95.2
	SQuAD	**0.654**	0.976	**0.685**	0.861	**0.669**	0.668	**0.636**	0.499

5.2 Position-Sensitive Readout Alignment

We next isolate why the same lexical BadNL trigger can behave differently across architectures and classifier readouts. Last-token pooling uses the final token representation as the classifier input, whereas mean pooling averages the valid token representations. This diagnostic study uses aclImdb, the trigger word "cf", and five relative insertion positions: 0, 25, 50, 75, and 100, where each number denotes the percentage position measured from the beginning of the input sequence. Single-position models are trained with one fixed trigger position, whereas `BadNL_mix5` distributes poisoned examples across all five positions.

5.2.1 Single-Position Training

Figure 2 reports the train-position by test-position ASR matrix. The main phenomenon is simple: with last-token pooling, Mamba is effective at both sequence boundaries, whereas Pythia is effective mainly at the tail. The strong tail behavior is expected for both models because the classifier directly reads the last-token representation. The additional head effectiveness of Mamba suggests that the first token may exert a larger influence on the following hidden states, allowing an early trigger to remain visible at the final readout.

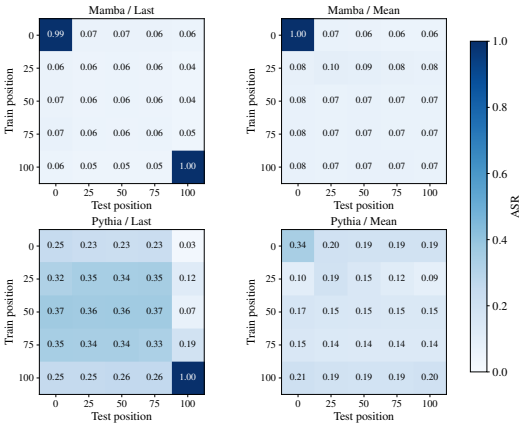


Figure 2: Single-position BadNL train-test ASR matrices on aclImdb. Rows denote the training insertion position and columns denote the testing insertion position.

Mean pooling further separates the two architectures: Mamba/Mean is effective mainly at the head, whereas Pythia/Mean remains weak across positions. This contrast is consistent with the state-propagation explanation above. In Mamba, an early trigger can affect the hidden-state trajectory, so subsequent token vectors may also carry the trigger mark and mean pooling can aggregate this repeated signal. In Pythia, the trigger is more localized; averaging token vectors instead dilutes the trigger representation.

5.2.2 MIXED-POSITION BUDGET ABLATION

The single-position matrices leave open whether the weak middle positions are inherently unlearnable or merely underrepresented. Table 2 shows that increasing the `BadNL_mix5` poison budget generally improves ASR, indicating that limited per-position poison evidence is an important part of the mixed-position failure. However, Pythia shows a larger CACC drop under higher budgets, whereas Mamba remains more stable.

Full position-wise results are provided in the supplementary material. The detailed position-wise trends further show that budget and read-out bias interact: larger budgets can strengthen weak positions, but the remaining failures still depend on whether the trigger perturbation reaches the classifier readout.

Table 2: `BadNL_mix5` budget ablation summary. Avg. and Min. are computed over the five test positions.

Setting	ρ	Avg.	Min.	CACC
Mamba-last	0.05	0.263	0.085	0.922
	0.10	0.995	0.982	0.912
	0.25	1.000	1.000	0.923
Mamba-mean	0.05	0.140	0.133	0.900
	0.10	0.179	0.137	0.897
	0.25	0.834	0.171	0.891
Pythia-last	0.05	0.296	0.126	0.814
	0.10	0.302	0.128	0.778
	0.25	0.983	0.939	0.533
Pythia-mean	0.05	0.176	0.166	0.862
	0.10	0.180	0.146	0.847
	0.25	0.625	0.209	0.833

5.2.3 PAIRED MECHANISM ANALYSIS

To distinguish a genuine trigger rule from generic target drift, we use a paired-control diagnostic: both Mamba and Pythia use last pooling and are trained with `BadNL_100` at poison rate 0.05, and at test time we compare clean non-target inputs, inputs with the true trigger `cf`, and inputs with the control token `bb`.

Table 3: Specificity of trigger activation under paired controls. Clean, CF, and Ctrl. report the ASR for clean inputs, inputs with the true trigger `cf`, and inputs with the control token `bb`, respectively. Rate gain is CF minus Ctrl.

Model	Clean	CF	Ctrl.	Rate gain
Mamba	0.078	1.000	0.133	0.867
Pythia	0.188	1.000	0.977	0.023

Table 3 shows that the same trigger setup has different specificity across architectures. In Mamba, `cf` almost always induces the target behavior, while clean inputs and `bb` remain near the non-target baseline. In Pythia, however, `bb` also drives the target class, leaving little trigger-specific rate gain. Thus, Mamba learns a more precise association between the lexical trigger and the attacker-specified behavior, whereas Pythia exhibits non-specific target drift. Additional paired-control configurations are summarized in the supplementary material.

5.3 HIDDEN-STATE ATTRIBUTION ANALYSIS

The position-sensitive results above show when a trigger reaches the classifier readout; we now inspect how the trigger is represented before the final decision. This analysis uses AG News with BadNL trigger word `cf`, and 1000 paired clean/triggered examples for each insertion condition.

5.3.1 FINAL-TOKEN LINEAR PROBES

We first test whether the final classifier-readout representation contains a linearly decodable trigger trace. At each hidden-state layer, we train a binary logistic probe on final-token representations to distinguish paired clean and triggered examples. Features are standardized using train-split statistics, and probe area under the curve (AUC) is reported as receiver operating characteristic (ROC) AUC on a held-out test split for distinguishing triggered representations from clean representations. Prefix inserts the trigger before the original text; After n

Table 4: Trigger-position analysis with final-token linear probes.

Position	Mamba-130M		Pythia-160M	
	ASR	Final AUC	ASR	Final AUC
Prefix	0.264	0.951	0.278	0.840
After 5	0.268	0.974	0.278	0.908
After 10	0.265	0.959	0.282	0.918
After 25	0.312	0.955	0.311	0.934
After 50	0.944	0.999	0.950	0.999
Append	0.993	1.000	0.999	1.000

inserts it after the first n words; and Append places it at the end. Table 4 reports the best final-token probe AUC over layers; the corresponding layer indices are provided in the supplementary material.

Table 4 shows that representation separability and backdoor activation are not the same phenomenon. Early insertions have weak ASR but still produce high final-token probe AUC, which means the trigger leaves a detectable trace even when it does not reliably activate the target label. Near-tail and append insertions align this trace with the final classifier input and therefore produce both high ASR and high probe AUC. Full probe variants and position-wise diagnostic tables are provided in the supplementary material.

5.3.2 Classifier-Input Dimension Ablation

We then test whether the dimensions that differ most between clean and triggered classifier inputs are causally responsible for the backdoor output. Let z_i and z_i^{trig} be the final classifier inputs for a clean/triggered pair. On the train split, each dimension j receives the score

$$s_j = \frac{1}{|\mathcal{T}|} \sum_{i \in \mathcal{T}} \left| z_{i,j}^{\text{trig}} - z_{i,j} \right|. \tag{1}$$

We select the top-k dimensions by s_j, with $k \in \{1\%, 10\%\}$ of the hidden size, and replace those dimensions in the classifier input of the test split by the clean train-split mean. The backbone is not rerun; the edited vector is passed directly through the original classifier head.

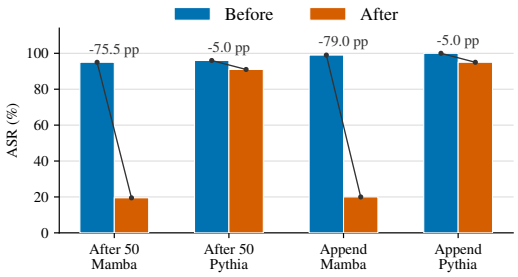


Figure 3: Classifier-input dimension ablation in the strong-trigger settings.

Figure 3 reports whether the trigger-sensitive dimensions are also causally responsible for the backdoor output. In the strong-trigger settings, replacing only the top 1% of Mamba classifier-input dimensions (8 dimensions) sharply suppresses ASR at both After 50 and Append. Pythia is much less affected even under a broader top-10% intervention (77 dimensions). The corresponding clean-accuracy changes are small: CACC drops by 0.020 for Mamba and 0.015 for Pythia in these rows. This contrast indicates that Mamba's backdoor behavior is more concentrated in the final classifier input, whereas Pythia's behavior is more dispersed or less removable by this dimension-replacement intervention. Additional position-wise and random-dimension controls are provided in the supplementary material.

5.4 Mamba Component Probing and Replacement

The readout-level interventions above show where the backdoor becomes concentrated; we now ask which Mamba components carry causal responsibility before the classifier. This analysis uses AG News with the BadNL trigger `cf` and 500 paired clean/triggered examples. We focus on two questions. First, because the input-dependent time step Δ is central to Mamba's selective state update, can Δ serve as an internal probe for triggered inputs? Second, if Δ is diagnostic, is it also the component whose replacement removes the backdoor?

5.4.1 Final-Token Delta Probing

We extract the input-dependent time step Δ_t from each Mamba block after the `dt_proj` and softplus transformation. In the append setting, the trigger is also the classifier's final readout token, so we compare its Δ with the paired clean final-token Δ. For each layer l, Table 5 reports Clean and Trigger means averaged over paired examples and delta channels. We define the Delta shift ratio (DSR) as $\text{DSR}_l = \overline{\Delta}_l^{\text{trig}} / \max(\overline{\Delta}_l^{\text{clean}}, 10^{-12})$, where $\overline{\Delta}_l^{\text{clean}}$ and $\overline{\Delta}_l^{\text{trig}}$ are the two layer-wise means. The table shows a large append-trigger shift. To test whether Δ also tracks attack success, we average $|\Delta^{\text{trig}} - \Delta^{\text{clean}}|$ over layers and channels for each triggered-clean pair, producing a scalar score that

distinguishes successful target predictions from failed triggers. This diagnostic reaches ROC AUC 0.986. Additional position controls, including after-word insertions and non-trigger probe locations, are provided in the supplementary material.

These results support delta as a strong diagnostic signal, but they do not by themselves establish causality. Neither result proves that changing delta alone will change the final prediction. We therefore perform a direct intervention at the same final readout position: in triggered inputs, we replace the append-trigger delta by the paired clean final-token delta and rerun the classifier path. This replacement leaves ASR essentially unchanged, from 0.994 to 0.996. Thus, delta is highly informative about trigger-induced computation and attack success, but delta alone is not a sufficient causal switch for removing the backdoor.

Table 5: Delta statistics at the final readout token on AG News/BadNL. Shift is Trigger Mean minus Clean Mean.

Layer	Clean Mean	Trigger Mean	Shift	DSR
10	0.063	0.304	0.241	4.804
9	0.095	0.228	0.133	2.404
8	0.138	0.261	0.124	1.901

5.4.2 Component Replacement

We next replace broader Mamba components at the append trigger position with their paired clean final-token counterparts. Table 6 shows that replacing Δ_t alone does not suppress the backdoor, whereas replacing h_t, r_t, m_t, or g_t largely shifts triggered predictions back toward the paired clean labels with limited broken predictions. Replacing the state-write term w_t is the most aggressive intervention, but it also strongly disrupts normal predictions, indicating a sensitive state-write path rather than a clean intervention point.

Table 6: Append-trigger component replacement in Mamba. Label Rec. is the fraction of triggered examples whose prediction returns to the paired clean label; Broken is the rate of predictions that become neither the target label nor the paired clean label.

Comp.	ASR After	ΔASR	Label Rec.	Broken
Δ_t	0.996	+0.002	0.250	0.000
g_t	0.278	-0.716	0.884	0.068
m_t	0.264	-0.730	0.896	0.068
r_t	0.264	-0.730	0.896	0.068
h_t	0.262	-0.732	0.898	0.068
w_t	0.144	-0.850	0.458	0.452

The component-level results refine the interpretation of Mamba backdoor activation. Delta moves strongly under triggered inputs and is useful as a probe, but the causal path is not a single Δ_t component. Instead, the backdoor is more directly mediated by the mixer input h_t, gate g_t, post-gate scan output r_t, mixer output m_t, and state-write pathway w_t. This distinction matters for both attacks and defenses: monitoring delta can expose triggered computation, but suppressing the backdoor requires interventions that target the broader state-update and readout pathway rather than delta alone.

6 Trigger-Aware State-Write Boosting Attack

6.1 Rationale: Strengthening Trigger Writes

The mechanism analysis above suggests that a Mamba backdoor is not only a classifier-readout phenomenon, but is also shaped by how a trigger enters the selective state update. In Mamba, each token contributes to the recurrent state through an input-dependent write term. A trigger placed at the beginning of the sequence must therefore be written strongly enough to survive subsequent state updates before the final classifier readout. This observation motivates **TASWB**, a mechanism-guided attack that turns the Mamba state-write pathway into an explicit training target.

TASWB is designed around two complementary pressures. The state-write loss strengthens the trigger token's write contribution inside selected Mamba layers, making the trigger less likely to be diluted by later content. The target-margin loss acts at the classifier interface, forcing the final logits of poisoned examples to prefer the attacker-specified class. Together, these objectives operate at complementary stages of backdoor activation: write strengthening improves the architectural route by which an early trigger can propagate, while margin optimization ensures that the propagated trace is aligned with the final target label.

6.2 Training Objective

Let $\mathcal{B}$ be a training mini-batch, $\mathbf{o}_i$ be the classifier-logit vector for input x_i, and y_i be its training label. TASWB keeps the ordinary cross-entropy (CE) supervised objective $\mathcal{L}_{\text{CE}} = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \text{CE}(\mathbf{o}_i, y_i)$. Let $\mathcal{P} \subseteq \mathcal{B}$ denote the poisoned examples that contain the trigger and are assigned to the target label $y^\star$. For a poisoned example i, the target-class margin is $\mu_i = o_{i,y^\star} - \max_{c\neq y^\star} o_{i,c}$. Here $o_{i,c}$ denotes the logit of class c in $\mathbf{o}_i$. We penalize insufficient target preference with

$$\mathcal{L}_{\text{margin}} = \frac{1}{|\mathcal{P}|}\sum_{i\in\mathcal{P}} \text{softplus}(\mu_{\min} - \mu_i), \tag{2}$$

where $\mu_{\min} = 0$ in our experiments. Thus, the loss pushes the target logit above the largest non-target logit rather than merely increasing the target probability.

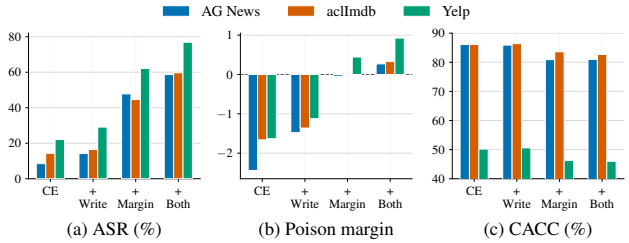
The state-write loss is applied to the trigger span in the Mamba layers used for write strengthening. Let $\mathcal{K}$ be this layer set and let τ_i be the trigger-token positions in poisoned example i. For layer l and token t, we denote the state-write contribution by $\omega_{i,l,t} = \Delta_{i,l,t} B_{i,l,t} u_{i,l,t}$, TASWB constructs a stop-gradient scaled target and minimizes a relative squared error:

$$\mathcal{L}_{\text{write}} = \frac{1}{|\Omega|}\sum_{(i,l,t)\in\Omega} \frac{\|\omega_{i,l,t} - \alpha_{\text{w}}\,\text{sg}(\omega_{i,l,t})\|_2^2}{\|\alpha_{\text{w}}\,\text{sg}(\omega_{i,l,t})\|_2^2 + \epsilon}, \tag{3}$$

where $\Omega = \{(i,l,t) : i \in \mathcal{P}, l \in \mathcal{K}, t \in \tau_i\}$, $\text{sg}(\cdot)$ stops gradients through the target, and $\alpha_{\text{w}} = 2$ in the standard setting. The final objective is $\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda_{\text{write}}\mathcal{L}_{\text{write}} + \lambda_{\text{margin}}\mathcal{L}_{\text{margin}}$. In all TASWB runs, auxiliary losses are computed only when triggered poisoned examples are present in the batch.

6.3 Ablation Results

We evaluate the attack with the head-inserted BadNL trigger `cf` at poison rate $\rho = 0.01$. Figure 4 reveals distinct effectiveness–utility trade-offs. Adding only $\mathcal{L}_{\text{write}}$ consistently improves ASR while leaving CACC nearly unchanged, indicating that state-write strengthening adds attack efficacy with little clean-utility cost. Adding $\mathcal{L}_{\text{margin}}$ produces larger ASR gains and substantially improves the poison margin, but also causes a more pronounced CACC decrease.



(a) ASR (%) (b) Poison margin (c) CACC (%)

Figure 4: TASWB objective ablation on head-trigger BadNL. Poison margin measures the average target-vs.-largest-non-target logit margin on triggered inputs.

Combining the two losses achieves the highest ASR and is the only setting with a positive average poison margin on all three datasets, while keeping CACC close to the margin-only setting. The objectives are therefore complementary: $\mathcal{L}_{\text{write}}$ provides low-cost ASR gains, $\mathcal{L}_{\text{margin}}$ supplies stronger target-class pressure, and their combination yields the strongest overall attack without substantial additional clean-accuracy loss beyond $\mathcal{L}_{\text{margin}}$ alone.

7 Conclusion

This paper studies how textual backdoors are activated in Mamba-style selective state-space language models. Across four datasets, two model scales, and multiple trigger families, we show that backdoor vulnerability depends jointly on backbone architecture, scale, and classifier readout. Mechanism analyses further reveal that Mamba backdoors are readout-aligned, partially concentrated at the classifier input, and mediated by the mixer, gate, scan-output, and state-write pathways rather than by delta alone. Guided by these findings, TASWB strengthens trigger-aware state writes and target-class margins, exposing a Mamba-specific attack surface. Current results focus on fine-tuned classification and extractive QA; broader generative settings remain future work.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript editing, software development and debugging, and research ideation and execution. All AI-assisted code was reviewed and tested by the authors. The evaluated system itself also invoked language and vision-language models during the experiments, as described in the paper. The authors take responsibility for the final content, implementation, and reported results.

REPRODUCIBILITY STATEMENT

Reproducibility information is provided throughout the main paper and appendix. The main paper presents the problem formulation, threat models, experimental setup, proposed method, and evaluation procedures. The appendix provides dataset and fine-tuning details, complete poison-budget settings, additional numerical results for the paired-control, hidden-state attribution, and component-level causality analyses, and training-loss curves.

REFERENCES

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International conference on machine learning*, pp. 2397–2430. PMLR, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Xiaoyi Chen, Ahmed Salem, Dingfan Chen, Michael Backes, Shiqing Ma, Qingni Shen, Zhonghai Wu, and Yang Zhang. BadNL: Backdoor attacks against NLP models with semantic-preserving improvements. In *Proceedings of the 37th Annual Computer Security Applications Conference*, pp. 554–569, 2021.

Ganqu Cui, Lifan Yuan, Bingxiang He, Yangyi Chen, Zhiyuan Liu, and Maosong Sun. A unified evaluation of textual backdoor learning: Frameworks and benchmarks. *Advances in Neural Information Processing Systems*, 35:5009–5023, 2022.

Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. A backdoor attack against LSTM-based text classification systems. *IEEE Access*, 7:138872–138878, 2019.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Daniel Y Fu, Tri Dao, Khaled K Saab, Armin W Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models. *arXiv preprint arXiv:2212.14052*, 2022.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

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

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pretrained models. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 2793–2806, 2020.

Aakash Lahoti, Kevin Y Li, Berlin Chen, Caitlin Wang, Aviv Bick, J Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles. *arXiv preprint arXiv:2603.15569*, 2026.

Jiazhao Li, Yijin Yang, Zhuofeng Wu, VG Vinod Vydiswaran, and Chaowei Xiao. ChatGPT as an attack tool: Stealthy textual backdoor attack via blackbox generative model trigger. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2985–3004, 2024.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Hengzhi Pei, Jinyuan Jia, Wenbo Guo, Bo Li, and Dawn Song. TextGuard: Provable defense against backdoor attacks on text classification. *arXiv preprint arXiv:2311.11225*, 2023.

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. Mind the style of text! Adversarial and backdoor attacks based on text style transfer. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 4569–4580, 2021a.

Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. Hidden Killer: Invisible textual backdoor attacks with syntactic trigger. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 443–453, 2021b.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 2383–2392, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Tao Xiang, Fei Ouyang, Di Zhang, Chunlong Xie, and Hao Wang. Nlpsweep: A comprehensive defense scheme for mitigating NLP backdoor attacks. *Information Sciences*, 661:120176, 2024.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

A Dataset Details

Table 7 summarizes the benchmark datasets used in the main experiments, following the order in which they are introduced in the main paper. The extractive question answering (QA) dataset is the Stanford Question Answering Dataset (SQuAD).

Table 7: Details of the four benchmark datasets. "Labels" indicates the number of labels and their corresponding categories or answer formats. "Avg. #W" signifies the average text length (number of words). "Train" and "Test" denote the number of instances in the training and test sets, respectively.

Dataset	Task	Labels	Avg. #W	Train	Test
AG News	News Topic Classification	4(World/Sports/Business/SciTech)	37.8	120000	7600
IMDB	Sentiment Analysis	2(Positive/Negative)	232.8	25000	25000
Yelp Review	Review Rating Prediction	5(1-star/2-star/3-star/4-star/5-star)	134.1	650000	50000
SQuAD	Extractive QA	Answer for the Question	139.8	130319	11873

B Fine-Tuning Details

Unless otherwise specified, classification victims are fine-tuned as Low-Rank Adaptation (LoRA)-based text classifiers on mixed clean-poisoned training sets. The backbone is initialized from the corresponding Mamba or Pythia checkpoint, and a task-specific classification head is trained together with the LoRA adapters. We use last-token pooling by default and use mean pooling only in the readout-ablation experiments.

Table 8: Default fine-tuning hyperparameters for classification experiments.

Hyperparameter	Value
Maximum sequence length	256
Random seed	42
Training set size	10000
Default poison rate	0.05
Budget-ablation poison rates	0.05, 0.10, 0.25
Epochs	5
Optimizer	AdamW
Learning rate	2×10^{-5}
Weight decay	0
Batch size	8
Gradient accumulation steps	4
Effective batch size	32
Gradient clipping	1.0
Precision	bfloat16
Classifier dropout	0.1
LoRA rank	32
LoRA alpha	64
LoRA dropout	0.05

For Mamba victims, LoRA is applied to the `in_proj`, `x_proj`, and `dt_proj` modules. For Pythia victims, LoRA is applied to `query_key_value`, `dense`, `dense_h_to_4h`, and `dense_4h_to_h`. For BadNL position-sensitivity experiments, single-position attacks use a poison rate of 0.05 with 3000 training examples, while the mixed-position BadNL setting distributes poisoned examples across five trigger positions. The budget ablation increases the poison rate to 0.10 or 0.25 when explicitly stated.

We use the main-paper metric abbreviations attack success rate (ASR) and clean accuracy (CACC) throughout the supplementary tables. For Trigger-Aware State-Write Boosting (TASWB), we keep the same classifier fine-tuning protocol and activate trigger-aware patch/scan only for trigger-containing batches under the stronger TASWB threat model. The state-write loss is applied to

Mamba late layers 16–23. Unless otherwise stated, we set the final weights $\lambda_{\text{write}} = 1.0$ and $\lambda_{\text{margin}} = 1.0$, the margin minimum to 0, and the write-vector scale to 2. Both auxiliary-loss weights use a three-stage ramp: they are set to one-third and two-thirds of their final values in epochs 1 and 2, respectively, and reach their final values from epoch 3 onward. The reported runs use no write-loss warm-up.

C BadNL Mixed-Position Budget Details

Table 9 reports the full position-wise results for the `BadNL_mix5` budget ablation. The mixed training set distributes poisoned examples over five insertion positions, so increasing the poison rate also increases the approximate number of poisoned examples available for each position.

Table 9: Full `BadNL_mix5` budget-ablation results on aclImdb. ASR@0–ASR@100 denote attack success rates when the trigger is inserted at the corresponding relative test positions.

Model/Pool	ρ	Pois./pos.	ASR@0	ASR@25	ASR@50	ASR@75	ASR@100	Avg.	Min.	CACC
Mamba/Last	0.05	30	0.093	0.085	0.085	0.087	0.963	0.263	0.085	0.922
Mamba/Last	0.10	60	0.982	0.996	0.997	0.999	1.000	0.995	0.982	0.912
Mamba/Last	0.25	150	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.923
Mamba/Mean	0.05	30	0.148	0.146	0.138	0.137	0.133	0.140	0.133	0.900
Mamba/Mean	0.10	60	0.276	0.175	0.157	0.149	0.137	0.179	0.137	0.897
Mamba/Mean	0.25	150	1.000	1.000	1.000	1.000	0.171	0.834	0.171	0.891
Pythia/Last	0.05	30	0.134	0.132	0.126	0.130	0.959	0.296	0.126	0.814
Pythia/Last	0.10	60	0.128	0.134	0.141	0.146	0.963	0.302	0.128	0.778
Pythia/Last	0.25	150	0.939	0.990	0.991	0.993	1.000	0.983	0.939	0.533
Pythia/Mean	0.05	30	0.197	0.174	0.173	0.168	0.166	0.176	0.166	0.862
Pythia/Mean	0.10	60	0.179	0.213	0.195	0.168	0.146	0.180	0.146	0.847
Pythia/Mean	0.25	150	0.318	1.000	0.951	0.648	0.209	0.625	0.209	0.833

Increasing the poison budget generally raises position-wise ASR, but the pattern is not uniform across readouts. Mamba/Last changes from a mostly terminal-position attack at the lowest budget to a stable all-position attack once each position receives more poisoned evidence, while preserving clean accuracy. Mamba/Mean also benefits from a larger budget, but the gain concentrates at positions 0–75; the terminal position remains weak because a trigger at the end has no later states through which its perturbation can be propagated before mean pooling. Pythia/Last becomes broadly vulnerable only at the largest budget, and this comes with the largest CACC degradation, which is consistent with a less specific target-shift effect. Pythia/Mean shows a middle-position improvement at the largest budget, while the head and tail positions remain much weaker, suggesting that mean pooling preserves some local middle-position trigger evidence but still dilutes boundary-position signals.

D Additional Paired-Control Mechanism Results

Table 10 summarizes the paired-control mechanism results for all analyzed configurations. Each row averages over the five insertion positions. The main paper uses the terminal-position `BadNL_100` comparison because it keeps the training method, readout, and test position aligned across architectures while exposing whether the response is specific to `cf` or also triggered by the control token.

E Additional Hidden-State Attribution Results

This section provides the detailed numerical tables for the hidden-state attribution analysis in the main paper. All rows use `ag_news`/BadNL with trigger word `cf`, seed 42, and 1000 paired clean/triggered examples per trigger-position condition. Prefix inserts the trigger before the original text; After n inserts it after the first n words; Append places it at the end.

We evaluate three probe views in the full analysis: the located trigger token, the final token used by the classifier readout, and the mean over valid tokens. Table 11 reports all three probe area under the curve (AUC) values, computed as receiver operating characteristic (ROC) AUC. The trigger-token probe reaches AUC 1.000 for all rows, while the final-token and mean-pooled probes are

Table 10: Five-position averages for paired-control mechanism analysis. Clean, CF, and Ctrl. are target-class prediction rates on clean non-target inputs, inputs with the true trigger `cf`, and inputs with the control token `bb`. Rate gain is CF minus Ctrl.; margin gain is the difference in final-head target-vs-source logit margin between CF and Ctrl.

Model	Pool	Training	ρ	Clean	CF	Ctrl.	Rate gain	Margin gain
Mamba	Last	BadNL_mix5	0.05	0.109	0.283	0.119	0.164	0.99
Mamba	Last	BadNL_mix5	0.10	0.086	0.994	0.106	0.887	7.54
Mamba	Last	BadNL_0	0.05	0.078	0.264	0.080	0.184	1.70
Mamba	Last	BadNL_100	0.05	0.078	0.261	0.086	0.175	1.88
Mamba	Last	BadNL_25	0.05	0.078	0.070	0.072	-0.002	-0.07
Mamba	Mean	BadNL_mix5	0.25	0.164	0.833	0.173	0.659	8.46
Mamba	Mean	BadNL_0	0.05	0.078	0.259	0.075	0.184	3.13
Mamba	Mean	BadNL_100	0.05	0.078	0.072	0.072	0.000	0.02
Mamba	Mean	BadNL_25	0.05	0.078	0.077	0.073	0.003	0.09
Pythia	Last	BadNL_mix5	0.05	0.062	0.253	0.122	0.131	0.86
Pythia	Last	BadNL_mix5	0.25	0.938	0.980	0.948	0.031	1.57
Pythia	Last	BadNL_100	0.05	0.188	0.342	0.331	0.011	1.55
Pythia	Last	BadNL_50	0.05	0.281	0.248	0.316	-0.067	-0.40
Pythia	Mean	BadNL_mix5	0.25	0.156	0.575	0.178	0.397	3.20
Pythia	Mean	BadNL_0	0.05	0.195	0.206	0.189	0.017	0.34
Pythia	Mean	BadNL_25	0.05	0.102	0.128	0.102	0.027	0.93

more informative for whether the trigger trace reaches the classifier input or the global sequence representation.

Table 11: Trigger-position and probe summary. Trigger AUC, Final AUC, and Mean AUC are held-out ROC AUC values for clean-vs-triggered logistic probes using the located trigger token, final-token, and mean-pooled hidden-state features, respectively. Final Layer reports the layer where the best final-token AUC is obtained.

Position	Model	CACC	ASR	Trigger AUC	Final AUC	Final Layer	Mean AUC
Prefix	Mamba-130M	0.881	0.264	1.000	0.951	20	0.997
Prefix	Pythia-160M	0.870	0.278	1.000	0.840	7	1.000
After 5	Mamba-130M	0.881	0.268	1.000	0.974	20	0.997
After 5	Pythia-160M	0.870	0.278	1.000	0.908	1	0.997
After 10	Mamba-130M	0.881	0.265	1.000	0.959	20	0.996
After 10	Pythia-160M	0.870	0.282	1.000	0.918	1	0.997
After 25	Mamba-130M	0.881	0.312	1.000	0.955	20	0.996
After 25	Pythia-160M	0.870	0.311	1.000	0.934	1	0.997
After 50	Mamba-130M	0.881	0.944	1.000	0.999	24	0.995
After 50	Pythia-160M	0.870	0.950	1.000	0.999	1	0.996
Append	Mamba-130M	0.881	0.993	1.000	1.000	0	0.995
Append	Pythia-160M	0.870	0.999	1.000	1.000	0	0.997

For the layer-wise representation-shift analysis, let $H_l(x_i, t)$ denote the layer-l hidden state for sample i at token position t. We compare only positions that are valid in both the clean and triggered sequences, denoted by Ω. The layer-wise L2 shift and cosine shift are defined as

$$D_l = \frac{1}{|\Omega|} \sum_{(i,t)\in\Omega} \left\| H_l(x_i^{\text{trig}}, t) - H_l(x_i, t) \right\|_2, \tag{4}$$

$$C_l = \frac{1}{|\Omega|} \sum_{(i,t)\in\Omega} \Big(1 - \cos(H_l(x_i^{\text{trig}}, t), H_l(x_i, t)) \Big). \tag{5}$$

Peak L2 is $\max_l D_l$, and peak cosine shift is $\max_l C_l$.

We also compute the Trigger Propagation Score (TPS) to measure whether the perturbation remains visible after the trigger location. For an offset δ from the trigger start, TPS is

$$\begin{aligned}\mathrm{TPS}_l(\delta) = &\frac{1}{|\Omega_\delta|}\sum_{i\in\Omega_\delta}\big\|H_l(x_i^{\mathrm{trig}}, t_i^{\mathrm{trig}}+\delta)\\ &- H_l(x_i, t_i^{\mathrm{trig}}+\delta)\big\|_2,\end{aligned} \tag{6}$$

where Ω_δ contains examples for which $t_i^{\mathrm{trig}}+\delta$ is valid in both the clean and triggered sequences. TPS is therefore a common-token propagation metric.

The append rows in Table 12 should be read as a metric-boundary case rather than as evidence of no trigger effect. Because the appended trigger has no corresponding token in the clean sequence, common-token L2 shift and TPS do not capture the new final trigger token. This is why the main paper interprets append primarily through final-token probes and classifier-input dimension ablations.

Table 12: Complete layer-wise hidden-state shift and final-layer TPS summary. Peak L2 and Peak Cos. report the maximum layer-wise shift value and its layer index. TPS@0 and TPS@50 are final-layer Trigger Propagation Scores at the trigger start and 50-token offset. Valid@50 is the fraction of paired examples with a common valid token at offset 50.

Position	Model	Peak L2 Layer	Peak L2	Peak Cos. Layer	Peak Cos.	TPS@0	TPS@50	Valid@50
Prefix	Mamba-130M	23	408.287	0	0.645	53.861	43.871	0.532
Prefix	Pythia-160M	12	66.092	0	0.722	48.624	65.872	0.532
After 5	Mamba-130M	23	427.435	0	0.656	79.336	51.976	0.306
After 5	Pythia-160M	12	65.634	0	0.735	94.252	75.755	0.306
After 10	Mamba-130M	23	362.537	0	0.550	86.005	51.302	0.131
After 10	Pythia-160M	12	55.013	0	0.616	99.870	71.117	0.131
After 25	Mamba-130M	23	200.008	0	0.309	84.927	47.423	0.021
After 25	Pythia-160M	12	31.931	0	0.345	105.985	68.386	0.021
After 50	Mamba-130M	23	16.138	0	0.023	86.890	57.237	0.004
After 50	Pythia-160M	12	2.315	0	0.026	98.269	80.633	0.004
Append	Mamba-130M	0	0.000	23	0.000	0.000	0.000	0.000
Append	Pythia-160M	0	0.000	3	0.000	0.000	0.000	0.000

Table 13 shows that dimension ablation is only strongly causal when the trigger position also produces a strong backdoor behavior. For prefix and early after-word insertions, ASR is low and replacing the top-difference dimensions has only a small effect. In contrast, for After 50 and Append, Mamba's ASR drops sharply after replacing only the top 1% classifier-input dimensions, while clean accuracy changes little. Pythia is much less sensitive under the same intervention family, even when the selected set contains 10% of the hidden dimensions. We report the Suppression-to-Side-effect Ratio (SSR), defined as $\Delta\mathrm{ASR}/\max(\Delta\mathrm{CACC}, 10^{-6})$, to compare attack suppression against clean-accuracy loss. This supports the main-paper conclusion that Mamba's backdoor behavior is more concentrated in the final classifier input, whereas Pythia's behavior is more dispersed or less removable by this replacement intervention.

Table 13: Complete classifier-input dimension ablation summary. Top-k dimensions are selected by clean/triggered mean absolute difference on the train split and replaced by the clean train-split mean on the held-out split.

Position	Model	Top-k	k	ASR Before	ASR After	ASR Drop	CACC Before	CACC After	CACC Drop	SSR
Prefix	Mamba-130M	0.050	38	0.270	0.255	0.015	0.900	0.865	0.035	0.429
Prefix	Pythia-160M	0.010	8	0.250	0.255	-0.005	0.895	0.890	0.005	-1.000
After 5	Mamba-130M	0.050	38	0.280	0.255	0.025	0.900	0.865	0.035	0.714
After 5	Pythia-160M	0.010	8	0.250	0.255	-0.005	0.895	0.890	0.005	-1.000
After 10	Mamba-130M	0.100	77	0.265	0.260	0.005	0.900	0.875	0.025	0.200
After 10	Pythia-160M	0.010	8	0.260	0.260	0.000	0.895	0.890	0.005	0.000
After 25	Mamba-130M	0.100	77	0.280	0.260	0.020	0.900	0.875	0.025	0.800
After 25	Pythia-160M	0.050	38	0.295	0.285	0.010	0.895	0.880	0.015	0.667
After 50	Mamba-130M	0.010	8	0.950	0.195	0.755	0.900	0.880	0.020	37.750
After 50	Pythia-160M	0.100	77	0.960	0.910	0.050	0.895	0.880	0.015	3.333
Append	Mamba-130M	0.010	8	0.990	0.200	0.790	0.900	0.880	0.020	39.500
Append	Pythia-160M	0.100	77	1.000	0.950	0.050	0.895	0.880	0.015	3.333

Table 14 provides a layer-wise final-token dimension-selection control. Instead of selecting dimensions directly from the final classifier input, this control selects dimensions from each layer's

final-token difference between clean and triggered examples, and then applies the same replacement operation at the classifier input. For the strong-trigger settings, the best Mamba selection layer still identifies a small dimension set that strongly suppresses ASR with little clean-accuracy loss. The corresponding Pythia rows remain much weaker, matching the classifier-input ablation pattern in Table 13.

Table 14: Per-layer final-token dimension-selection ablation. For each row, dimensions are selected from the specified layer's final-token clean/triggered difference, but the intervention is still applied to the final classifier input before the classifier head.

Position	Model	Sel. Layer	Top-k	k	ASR Drop	CACC Drop
Prefix	Mamba-130M	1	0.100	77	0.025	0.025
Prefix	Pythia-160M	0	0.100	77	0.005	0.010
After 50	Mamba-130M	24	0.010	8	0.755	0.020
After 50	Pythia-160M	12	0.100	77	0.050	0.015
Append	Mamba-130M	24	0.010	8	0.790	0.020
Append	Pythia-160M	12	0.100	77	0.050	0.015

F Additional Component-Level Causality Results

This section provides the detailed numerical support for the component-level causality analysis in the main paper. All rows use Mamba-130M on `ag_news`/BadNL with trigger word `cf`, seed 42, and 500 paired clean/triggered examples. The main text emphasizes the append setting because it matches the strongest BadNL evaluation protocol: the appended trigger is also the final token read by the classifier, so the Delta comparison is between the triggered final token and the paired clean final token. The after-words setting is included here as an auxiliary position-control result; its weaker paired ASR makes it useful for separating diagnostic Delta shift from causal backdoor removal.

Table 15 reports the strongest layer-wise Delta shifts under the append trigger-only view. For each layer, Clean and Trigger are scalar means obtained by averaging the selected final-token Δ values over valid paired examples and Delta channels. The table also reports Kullback–Leibler (KL) divergence, Wasserstein distance (WD), and the Delta shift ratio (DSR). Triggered inputs produce large Delta shifts in several layers, with layer 10 showing the largest mean shift and a DSR of 4.804.

Layer	Clean	Trigger	Shift	KL	WD	DSR
10	0.063	0.304	0.241	1.565	0.241	4.804
9	0.095	0.228	0.133	0.659	0.133	2.404
8	0.138	0.261	0.124	0.326	0.124	1.901
13	0.095	0.178	0.083	0.267	0.083	1.875
2	0.085	0.158	0.074	0.322	0.074	1.871
7	0.187	0.121	-0.065	0.259	0.071	0.649
18	0.064	0.119	0.055	0.183	0.055	1.859
3	0.092	0.146	0.054	0.204	0.054	1.581

Table 15: Append trigger-only Delta statistics. In the append setting, the triggered final token is the appended trigger. Clean and Trigger average the selected final-token Δ values over all valid paired examples and Delta channels; DSR is Trigger divided by Clean.

Table 16 separates correlation from intervention. Delta features are associated with whether a triggered input succeeds in reaching the target label, especially in the append setting, but replacing Delta alone does not reduce ASR. The after-words Delta replacement also produces only a small ASR change, consistent with the main-text conclusion that Delta is diagnostic rather than a sufficient causal switch.

Table 17 gives the full append-trigger component replacement table. The most aggressive interventions, `deltaB_u` and `ssm_state_after_trigger`, produce the largest ASR drops but also high broken-prediction rates. Cleaner evidence comes from `input_states`,

Table 16: Delta correlation and replacement results. AUC uses each listed Delta feature as a sample-level score for distinguishing successful triggered examples from failed triggered examples, rather than for clean-versus-triggered detection. The append replacement uses the paired clean final-token Delta because the append trigger is also the triggered final token; the after-words replacement uses the clean Delta window aligned to trigger through trigger+5.

Setting	Feature / Intervention	Corr.	p-value	AUC	ASR Before	ASR After	ASR Drop	Label Rec.
After 25	`delta_abs_shift`	-0.058	1.92×10^{-1}	0.457	–	–	–	–
After 25	`trigger_dsr`	0.214	1.36×10^{-6}	0.568	–	–	–	–
Append	`delta_abs_shift`	0.221	6.24×10^{-7}	0.986	–	–	–	–
Append	`trigger_dsr`	0.087	5.24×10^{-2}	0.808	–	–	–	–
After 25	Delta window replacement	–	–	–	0.326	0.320	0.006	0.866
Append	Delta trigger replacement	–	–	–	0.994	0.996	-0.002	0.250

`scan_output_post_gate`, `mixer_output`, and `gate`, which reduce ASR substantially while mostly recovering the paired clean label.

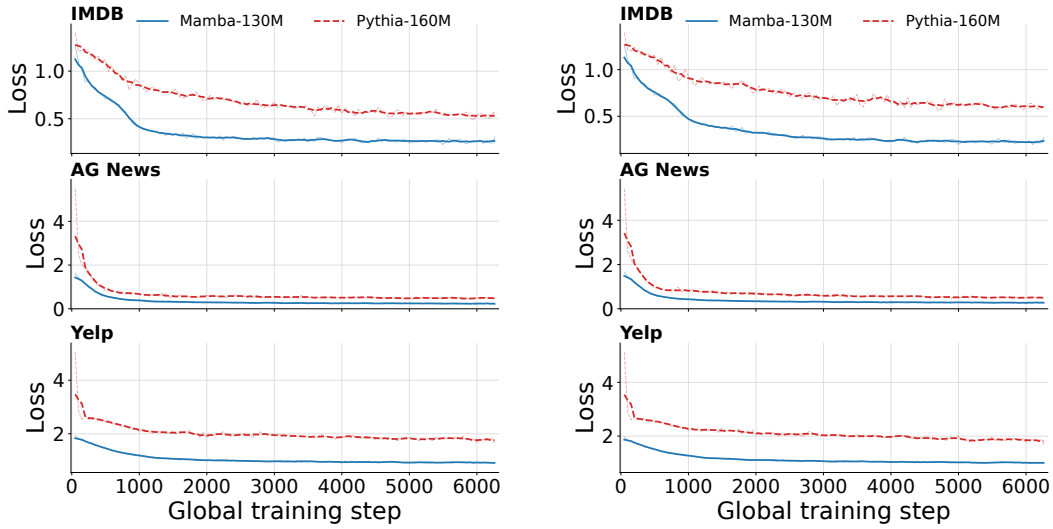
Table 17: Full append-trigger Mamba component replacement summary. Before replacement, paired clean label accuracy is 0.896 and ASR is 0.994. Label Rec. is the fraction of triggered examples whose post-intervention prediction returns to the paired clean label; Broken is the rate of predictions that become neither the target label nor the paired clean label.

Component	Label Rec.	Label Drop	ASR After	ASR Drop	Broken	Changed
`deltaB_u`	0.458	0.438	0.144	0.850	0.452	0.854
`ssm_state_after_trigger`	0.490	0.406	0.226	0.768	0.364	0.776
`input_states`	0.898	-0.002	0.262	0.732	0.068	0.732
`scan_output_post_gate`	0.896	0.000	0.264	0.730	0.068	0.730
`mixer_output`	0.896	0.000	0.264	0.730	0.068	0.730
`gate`	0.884	0.012	0.278	0.716	0.068	0.716
`scan_output_pre_gate`	0.784	0.112	0.304	0.690	0.124	0.694
`conv_hidden`	0.780	0.116	0.310	0.684	0.124	0.688
`pre_conv_hidden`	0.666	0.230	0.384	0.610	0.146	0.614
`C`	0.400	0.496	0.836	0.158	0.010	0.158
`B`	0.258	0.638	0.988	0.006	0.000	0.006
`delta`	0.250	0.646	0.996	-0.002	0.000	0.010

G Training-Loss Curves

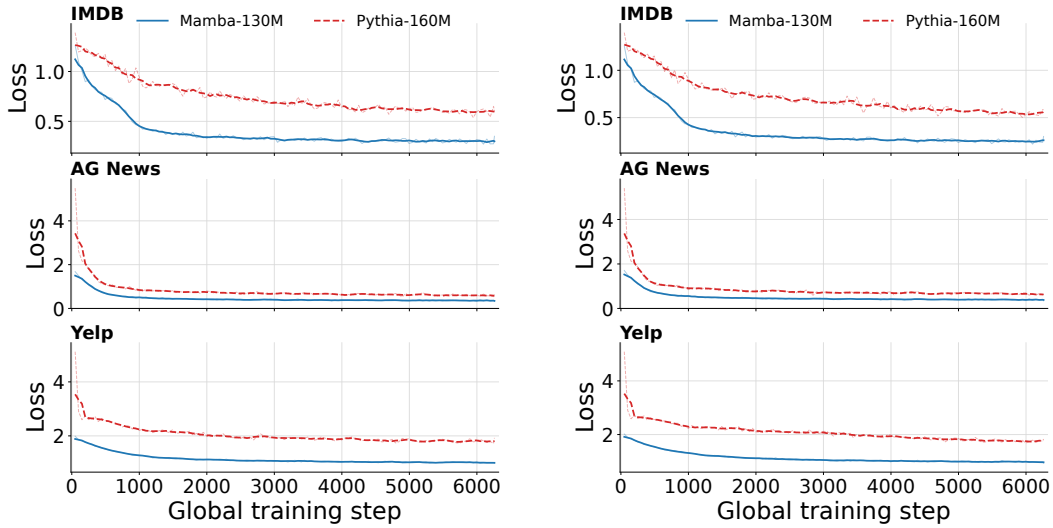
Figures 5a–6b provide the full training-loss curves for all four attacks. Each curve is computed from interval-level training losses with a 5-point moving average; the lighter curve shows the unsmoothed interval loss. The raw loss logs cover the three classification datasets and do not include SQuAD.

Across all four attacks, Mamba's loss generally decreases faster and settles below Pythia's under the same fine-tuning protocol. The gap is especially visible on Yelp, where both backbones optimize more slowly than on AG News and IMDB. These curves reinforce the main-text observation that similar backdoor outcomes can arise from architecture-specific optimization behavior, so the later mechanism analysis should focus on where the trigger signal enters the readout rather than only on final ASR.



(a) BadNL (b) SynBkd

Figure 5: Training-loss curves for Mamba-130M and Pythia-160M on the classification datasets under BadNL and SynBkd.



(a) StyBkd (b) BGMAttack

Figure 6: Training-loss curves for Mamba-130M and Pythia-160M on the classification datasets under StyBkd and BGMAttack.