Strictly Causal Streaming Video Anomaly Detection with a Theoretically-Grounded State-Space Core*

Yogesh Kumar[a]

[a]*Indian Institute of Technology Jodhpur, Jodhpur, India*

Abstract

Recent work has applied Mamba-style state-space models (SSMs) to video anomaly detection, but existing approaches still buffer clips or windows internally, offer no theoretical account of how a model's temporal memory relates to detection latency, and benchmark efficiency only through GPU throughput rather than the edge hardware such methods are meant to target. We introduce a strictly causal streaming video anomaly detector whose fixed-size state is updated in $O(1)$ time and memory per incoming frame, with no lookahead and no clip buffering. Its temporal core is a diagonal linear state-space recurrence with an input- and state-dependent decay gate, trained self-supervised through causal next-embedding prediction on a frozen visual backbone. We derive a closed-form relationship between the recurrence's decay spectrum and both its detection delay and the shortest anomaly it can reliably detect, then validate it empirically on UCSD Ped2 and CUHK Avenue: the settling-delay bound predicted from the learned base decay alone (57 to 59 frames) sits far above the measured detection delay (1.6 and 18.4 frames respectively), showing that the event-boundary gate, not the base decay, governs how quickly the model reacts. We also report end-to-end latency and throughput measured directly on an Apple M3 Pro rather than simulated from GPU numbers, 0.74 ms and 0.77 ms per frame (over 1,300 FPS) on the two datasets. With an initial, untuned configuration the method reaches 67.9% and 70.2% frame-level AUC on Ped2 and Avenue, trailing prior non-causal SSM baselines on accuracy; ablations over decay rate, state size, and the gate itself show that the gate's contribution is data-size dependent

*Code, trained checkpoints, and commands to reproduce every reported number: `https://github.com/yogesh-iitj/streaming-vad`.

Email address: `kumar.204@iitj.ac.in` (Yogesh Kumar)

rather than uniformly helpful, hurting accuracy on the smaller Ped2 training set and helping on the larger Avenue one. Closing this accuracy gap and extending evaluation to a third, larger benchmark are the immediate next steps, and we report both the theoretical and the accuracy picture in full rather than only the results that flatter the method.

Keywords: video anomaly detection, state-space models, streaming inference, edge deployment, real-time video understanding

---

1. Introduction

Video anomaly detection (VAD) systems are increasingly deployed at the edge, on cameras, robots, and mobile devices, where compute, memory, and power are all constrained and predictions must be produced online, as frames arrive, rather than after the fact. IEEE Transactions on Pattern Analysis and Machine Intelligence has an active recent literature on this problem, from a taxonomy and evaluation survey of single-scene methods [1] to the journal extension of the future-frame-prediction baseline [2], a scene-aware latent-space prediction and anticipation model [3], and a self-supervised detector based on learned neural transformations [4]. None of these, however, is designed around the specific constraint that motivates this paper: strict causality, meaning a prediction at time t must depend only on frames $1, \ldots, t$ with bounded per-step compute and memory, combined with real hardware feasibility, meaning latency and throughput are measured rather than asserted on the class of device the method targets.

State-space models (SSMs), and Mamba [5] in particular, have recently been proposed for VAD on efficiency grounds. VADMamba [6] and its follow-up VADMamba++ [7] combine a Mamba-based temporal block with vector-quantized frame prediction and optical-flow reconstruction, and Wave-MambaAD [8] adds a wavelet-driven SSM for unified subtle and large-scale anomaly detection. These works establish that SSMs are a good fit for VAD's efficiency requirements, but they still process the input in clips or windows rather than truly online per frame, report no theoretical relationship between the model's temporal memory and how quickly it can detect an anomaly, and benchmark efficiency in terms of GPU throughput rather than measuring the target-deployment hardware directly. This paper closes those three gaps together, rather than any one in isolation, because they compound: a model that is only softly causal cannot make a hard latency guarantee, and a latency num-

ber without a theory of why the model reacts when it does is an empirical curiosity rather than a design tool.

This paper makes three contributions. First, a strictly causal streaming formulation of SSM-based VAD, with a concrete architecture (Section 3) in which every operation is expressible as a per-frame $O(1)$ update. Second, a theoretical analysis (Section 4) relating the recurrence's decay spectrum to detection delay and minimum-detectable anomaly duration, validated empirically. Third, an end-to-end evaluation including real on-device latency, memory, and throughput measurements on consumer edge silicon (Section 5), alongside standard accuracy metrics and an ablation study that we report in full, including a result that complicates rather than confirms our own design hypothesis. Code, trained checkpoints, and commands to reproduce every number in this paper are publicly available.[1]

2. Related Work

Video anomaly detection.. Reconstruction- and prediction-based methods [9, 2, 10] learn the distribution of normal motion and flag frames with high reconstruction or prediction error, while weakly supervised methods [11] instead learn from video-level labels over long, untrimmed surveillance footage. Two recent TPAMI papers extend this line further: Cao et al. [3] condition prediction on scene context in latent space and extend the task to anomaly anticipation, and Qiu et al. [4] replace hand-crafted augmentations with learned neural transformations for self-supervised detection. Jones et al. [1] survey the single-scene setting broadly and formalize the region- and track-based evaluation criteria (RBDC and TBDC) introduced by Ramachandra and Jones [12], which we discuss further in Section 5. Our method follows the prediction-based line, operating in a frozen backbone's embedding space rather than pixel space, so that the temporal model is decoupled from how frames are encoded.

State-space models for anomaly detection.. S4 [13] and its simplified diagonal variant S5 [14] established that linear state-space recurrences can match attention on long-range sequence tasks at linear time and memory complexity, and Mamba [5] added input-dependent, or selective, parameterization

[1] `https://github.com/yogesh-iitj/streaming-vad`

on top of that recurrence. In video anomaly detection specifically, VADMamba [6] pioneered applying a Mamba-based block, called a Non-negative Visual State Space block, to multi-task frame prediction and optical-flow reconstruction, and its follow-up VADMamba++ [7] reworks this into a single grayscale-to-RGB proxy task with a hybrid Mamba, CNN, and Transformer decoder. Wave-MambaAD [8] instead combines a wavelet transform with an SSM to unify subtle and large-scale anomaly detection at multiple scales, and MambaAD [15] applies a related hybrid-scanning SSM design to multi-class *image* anomaly detection rather than video. All of these process input in clips or windows and report accuracy alongside, at most, GPU throughput; none provide a strict online, $O(1)$-per-frame formulation, a theoretical latency analysis, or measurements on edge or consumer hardware, which is the gap this paper targets. We investigated reproducing VADMamba under our own evaluation protocol using its public code, but its released implementation vendors FlowNet2 for optical flow, which depends on custom CUDA kernels (correlation, channel normalization, and resampling layers) with no CPU or Metal Performance Shaders fallback; it therefore cannot run on the non-CUDA edge hardware this paper targets without a substantial reimplementation that would no longer constitute a faithful reproduction. This is itself a small piece of evidence for the gap this paper targets: a baseline built around CUDA-only components is, by construction, unable to make the edge-deployment claim we are making here. The numbers attributed to prior work in Section 5 are therefore as reported in the original paper, not rerun under our protocol.

Efficient and streaming video understanding more broadly.. A separate line of work makes video understanding cheaper by deciding, per input, which frames or clips are worth processing at all: AdaFrame [16] learns a policy that adaptively selects how many and which frames to observe for classification, and SCSampler [17] learns to identify salient clips within long, untrimmed video before running a heavier recognition model on them. This is an orthogonal efficiency mechanism to ours. Those methods reduce compute by choosing what to look at, at the input level, while still typically requiring a batch or clip to select over; our method instead reduces compute by bounding what the model remembers and how it updates that memory per frame, which is what makes strict frame-by-frame streaming possible in the first place. The two mechanisms are not mutually exclusive, and combining input-level frame selection with a bounded-state streaming core is a natural

direction for future work.

3. Method

3.1. Overview

Each incoming frame I_t is mapped to an embedding $e_t = \phi(I_t) \in \mathbb{R}^D$ by a frozen backbone ϕ (Section 3.2). A stack of causal diagonal state-space layers (Section 3.3) maintains a fixed-size state and produces $\hat{e}_{t+1}$, a prediction of the *next* embedding, using only $e_1, \ldots, e_t$. The anomaly score at time t is the prediction error $s_t = \|\hat{e}_t - e_t\|_2$ (Section 3.4), following the predictive-coding formulation standard in the VAD literature [9], adapted here to operate strictly causally at the embedding level.

3.2. Frozen backbone

We deliberately keep ϕ simple and frozen. The contribution of this paper is the temporal core and its analysis, not backbone design, and freezing ϕ keeps the trainable parameter count and compute footprint small enough to iterate on consumer hardware (Section 5). The results reported in this paper use an ImageNet-pretrained ResNet-18 (D=512); the implementation also supports a frozen DINOv2 ViT-S/14 (D=384) as a stronger but heavier alternative, which we have not yet evaluated. The backbone's parameters are frozen throughout training; only the temporal core below is optimized.

3.3. Causal diagonal state-space core

Each layer maintains a state $s_t \in \mathbb{R}^N$ and updates it as

$$u_t = W_{\text{in}}\,\text{LN}(x_t) \tag{1}$$

$$g_t = \sigma\big(\text{MLP}([\text{LN}(x_t),\, s_{t-1}])\big) \tag{2}$$

$$\bar{a} = a_{\min} + (a_{\max} - a_{\min})\,\sigma(\theta_a) \tag{3}$$

$$a_t = \bar{a} \odot g_t \tag{4}$$

$$s_t = a_t \odot s_{t-1} + u_t \tag{5}$$

$$y_t = W_{\text{out}}\, s_t + x_t \tag{6}$$

where x_t is the layer's input (the backbone embedding e_t for the first layer, the previous layer's output y_t otherwise), LN is layer normalization, $\theta_a \in \mathbb{R}^N$ is a learned per-channel parameter, and $[a_{\min}, a_{\max}] \subset (0, 1)$ bounds the time-invariant *base* decay $\bar{a}$ (Eq. 3, following the S4D/S5 initialization convention

Algorithm 1 Streaming per-frame update (one layer)

Require: frame embedding x_t, previous state s_{t-1}

$u_t \leftarrow W_{\text{in}}\,\text{LN}(x_t)$
$g_t \leftarrow \sigma(\text{MLP}([\text{LN}(x_t), s_{t-1}]))$
$a_t \leftarrow \bar{a} \odot g_t$
$s_t \leftarrow a_t \odot s_{t-1} + u_t$
$y_t \leftarrow W_{\text{out}} s_t + x_t$
return y_t, s_t

[13, 14]). The gate $g_t \in (0,1)^N$ (Eq. 2) is the only input- or state-dependent part of the recurrence, "selective" in Mamba's terminology [5]. It lets the effective decay a_t drop sharply, an implicit state reset through fast forgetting, when the current input is inconsistent with the current state, which is exactly the situation at an anomaly onset. We call g_t the *event-boundary gate* for this reason. Eq. 5 is a diagonal (real-valued) linear recurrence, not a full re-implementation of Mamba's selective-scan kernel, which is CUDA-only; this keeps the entire model runnable on non-CUDA edge hardware (Section 5) while retaining $O(N)$ per-step state size and $O(1)$ per-step compute in the sequence length.

3.3.1. Strict causality, by construction

Unlike prior SSM-based VAD methods that process fixed-size clips or maintain a sliding buffer internally [6, 8], Eqs. 2–5 depend only on s_{t-1} and x_t; there is no operation in the model that looks ahead or reprocesses past frames. Algorithm 1 gives the exact per-frame update used both at training time (called once per timestep, unrolled over a window) and at deployment (called once per incoming frame); using the identical routine in both settings is a deliberate choice, so that reported efficiency is not an artifact of a training-time-only formulation.

3.4. Training objective and anomaly scoring

The stack's final output y_t is passed through a small MLP head to produce $\hat{e}_{t+1}$, a prediction of the next frame's embedding. The model is trained self-supervised on normal-only training clips by minimizing

$$\mathcal{L} = \tfrac{1}{T-1} \textstyle\sum_{t=1}^{T-1} \|\hat{e}_{t+1} - e_{t+1}\|_2^2. \tag{7}$$

At inference, the anomaly score at frame t is the prediction error $s_t = \|\hat{e}_t - e_t\|_2$ using the prediction made one step earlier (i.e. using only $e_1, \ldots, e_{t-1}$);

scores are min-max normalized per clip before thresholding/AUC computation, following standard VAD evaluation protocol [9].

4. Decay–Delay Analysis

This section derives a closed-form relationship between a channel's decay rate and (a) how long the model takes to react to an anomaly onset and (b) the shortest anomaly it can reliably detect, then validates it empirically (Section 5). The analysis treats one channel of the recurrence (Eq. 5) in isolation with its *effective* decay a held constant across the transition being analyzed; Section 4.3 discusses how the learned gate g_t relaxes this assumption in practice.

4.1. Settling-time bound

Consider a single channel driven by an input that is constant at u_{norm} for $t < t_0$ and step-changes to u_{anom} at t_0 (an idealized anomaly onset). Under the recurrence $s_t = a\, s_{t-1} + u_t$, the state converges toward the steady state $s^* = u/(1-a)$ for constant u; writing $s^*_{\text{norm}} = u_{\text{norm}}/(1-a)$ and $s^*_{\text{anom}} = u_{\text{anom}}/(1-a)$, the solution for $t \geq t_0$ is

$$s_t - s^*_{\text{anom}} = a^{\,t-t_0}\big(s_{t_0} - s^*_{\text{anom}}\big). \tag{8}$$

Proposition 1 (Settling delay)**.** *Define the detection delay $\delta(\varepsilon)$ as the smallest $t - t_0 \geq 0$ such that the state has moved to within a fraction $\varepsilon \in (0,1)$ of the new steady state, i.e. $|s_t - s^*_{anom}| \leq \varepsilon\, |s_{t_0} - s^*_{anom}|$. Then*

$$\delta(\varepsilon) = \left\lceil \frac{\ln \varepsilon}{\ln a} \right\rceil. \tag{9}$$

Proof. From Eq. 8, the condition is $a^{\,\delta} \leq \varepsilon$. Since $a \in (0,1)$, $\ln a < 0$; taking logarithms and dividing by $\ln a$ flips the inequality, giving $\delta \geq \ln \varepsilon / \ln a$ (positive, since $\ln \varepsilon < 0$ too). The smallest integer δ satisfying this is Eq. 9. □

Eq. 9 is monotonically increasing in a: channels with slower decay (larger a, longer memory) take strictly longer to react to a step change. This is exactly the quantity we compute from each trained layer's mean learned base decay $\bar{a}$ (Eq. 3) and compare against the empirically measured delay between an anomaly's labeled onset and the first score-threshold crossing.

4.2. Minimum-detectable event duration

Proposition 1 has a direct converse: an anomalous segment of duration D frames that ends before the state has settled (i.e. $D < \delta(\varepsilon)$ for the ε at which the downstream threshold reliably separates normal/anomalous scores) will not produce the full-magnitude prediction error the detector was calibrated on, and is at elevated risk of a missed detection. This gives a direct, checkable prediction: *shortening* the base decay (smaller $a_{\min}, a_{\max}$ in Eq. 3) should reduce delay and improve recall on short anomalous segments, at the cost of a noisier score curve (and likely higher false-positive rate) on normal video, since the state now also reacts to high-frequency variation that is not anomalous. Section 5 reports the state-size/decay sweep ablation that tests this trade-off directly.

4.3. Role of the event-boundary gate

The analysis above assumes a is fixed, but the actual effective decay $a_t = \bar{a} \odot g_t$ (Eq. 2) is state- and input-dependent. The design intent was for the model to behave like a *small* a (fast reaction, per Eq. 9) exactly when g_t drops in response to a state/input mismatch, that is, at a genuine event boundary, while behaving like the larger, more stable base decay $\bar{a}$ elsewhere: a shorter effective delay than a fixed-decay recurrence with the same $\bar{a}$, without paying the false-positive cost of a globally smaller decay everywhere.

The settling-delay numbers in Section 5 are consistent with the gate reacting fast, since the empirical delay sits far below the fixed-$\bar{a}$ bound on both Ped2 and Avenue. The ablations in Tables 3 and 4 complicate this story in a way that is itself informative, because fast reaction to a state/input mismatch is not automatically the same as reacting specifically to *anomalous* mismatches. Whether the gate's extra parameters (Eq. 2) learn the latter rather than overfitting to the former appears to depend on how much training data is available. On Ped2, the smaller training set with 16 clips of 120 to 180 frames each, disabling the gate improves frame-AUC (76.6% versus 67.9%), and a smaller state does too, both consistent with overfitting. On Avenue, also 16 clips but up to 1271 frames each and so substantially more total training data, the pattern reverses: disabling the gate hurts sharply (60.0% versus 70.2%), and a larger state helps slightly rather than hurting. Read together, this is more consistent with a capacity and data-size interaction than with the gating mechanism being unhelpful in principle, but it remains a two-dataset hypothesis rather than a demonstrated effect. A third, substantially

larger training set would be needed to test it properly; ShanghaiTech was intended for this but its raw data has not yet been fully obtained (Section 5), so this comparison stays open. We report the full, dataset-dependent picture here rather than only the settling-delay result that would have supported the original design hypothesis on its own.

Two extensions would sharpen this analysis. Proposition 1 could be generalized to the time-varying a_t case, which admits a closed form through a cumulative product (see the parallel-form discussion in the released code's state-space module), giving a tighter bound than the fixed-$\bar{a}$ approximation used here. And the gate ablation should be rerun on a third, larger dataset once available, since the capacity and data-size account above rests on only two data points that happen to share the same clip count.

5. Experiments

5.1. Datasets and protocol

We evaluate on two standard unsupervised VAD benchmarks, UCSD Ped2 (16 training clips, 12 test clips) and CUHK Avenue (16 training clips, 21 test clips). Both provide normal-only training video and mixed normal/anomalous test video with frame-level ground truth; Ped2 additionally provides pixel-level masks, which we do not currently use. We report frame-level ROC-AUC and equal error rate (EER) using the standard per-clip min-max score normalization protocol [9]. A third benchmark, ShanghaiTech, is part of the intended evaluation but is not included in the results below: its raw data is distributed as a large, multi-part archive and only one of seven parts had been obtained at the time of writing. We report this directly rather than substituting an estimate, and all ShanghaiTech numbers in this draft are left for a revision once the full dataset has been downloaded, prepared, and run through the same pipeline as the other two.

We are also explicit about a second limitation of our evaluation protocol. The region- and track-based detection criteria (RBDC and TBDC) introduced by Ramachandra and Jones [12] and adopted in the broader survey of Jones et al. [1] are, in our released code, approximated by a simplified frame-overlap criterion rather than implemented to the original specification. The frame-level AUC and EER numbers reported here do not depend on this approximation and follow the standard protocol exactly, but any RBDC/TBDC-style localization numbers computed with our current tooling

should be treated as illustrative only until reimplemented against the official criteria.

5.2. Implementation details

Frame embeddings come from a frozen, ImageNet-pretrained ResNet-18 backbone (D=512); a frozen DINOv2 ViT-S/14 (D=384) is supported in the released code as a stronger alternative but was not used for the numbers below. We attempted to run it during development; the official DINOv2 repository's code requires Python 3.10 or later (it uses runtime-evaluated union-type syntax not supported by earlier versions), while our development environment was fixed at Python 3.9, so this comparison is deferred rather than reported here. The causal SSM stack has 2 layers with state size N=128 per layer (Section 3.3), trained with AdamW at learning rate 3×10^{-4} on causal windows of 16 frames, for 40 epochs on Ped2 and 30 on Avenue. All training and evaluation ran on an Apple M3 Pro using PyTorch's MPS backend; no CUDA-specific code path exists in this implementation, by design, since the strict-streaming claim is meant to hold on non-CUDA edge hardware as well as on GPUs.

5.3. Main results

Table 1 reports frame-AUC, EER, and per-frame latency and FPS measured directly on-device: our evaluation harness calls the identical per-frame update routine used at deployment (Algorithm 1), one frame at a time, with an explicit device synchronization before each timing boundary. We do not include a VADMamba row here; Section 2 explains why a same-protocol comparison was not possible (its released code depends on CUDA-only optical-flow kernels). Our frame-AUC (67.9% on Ped2, 70.2% on Avenue) is well below the 90s-percent range typically reported for tuned methods on these benchmarks. We attribute this to the untuned, first-pass configuration evaluated here (frozen backbone, no motion signal, no hyperparameter search) rather than a limitation of the streaming formulation itself, and report it plainly rather than narrow the comparison to flatter the number.

5.4. Theory validation

Table 2 compares the settling-delay bound (Eq. 9, Proposition 1), computed from each trained layer's mean learned base decay, against the empirically measured detection delay, the number of frames between an anomalous

Table 1: Main results: standard accuracy metrics alongside on-device streaming latency/throughput.

Dataset	Method	Frame-AUC↑	EER↓	Latency (ms)↓	FPS↑	Hardware
UCSD Ped2	Ours (streaming SSM)	67.9	36.8	0.74	1343.62	mps
CUHK Avenue	Ours (streaming SSM)	70.2	34.9	0.77	1290.74	mps

Table 2: Predicted settling delay (Eq. 9), from each dataset's trained mean decay, vs. empirically measured detection delay.

Dataset	Layer	Mean decay a	Theory delay (frames)	Empirical delay (frames)
UCSD Ped2	0	0.9493	57.5	1.6 (n=12)
	1	0.9501	58.6	
CUHK Avenue	0	0.9493	57.5	18.4 (n=21)
	1	0.9501	58.6	

segment's labeled onset and the first score-threshold crossing. The two numbers differ by more than an order of magnitude on both datasets: the bound predicts 57 to 59 frames from the base decay alone, while the measured delay is 1.6 frames on Ped2 and 18.4 on Avenue. Section 4.3 discusses why, and what the ablations below add to that discussion.

5.5. Ablations

Tables 3 and 4 report decay-rate, state-size, and event-boundary-gate ablations on Ped2 and Avenue respectively, with the backbone, training budget, and epoch count held fixed at the values in Section 5.

The two datasets tell different stories, and the difference is informative rather than merely noisy. On Ped2, disabling the gate improves frame-AUC over the gated baseline (76.6% versus 67.9%), and the smaller state (N=32) beats both the baseline (N=128) and the larger variant (N=256): every higher-capacity variant underperforms a simpler one. On Avenue, which has the same clip count but substantially more total training frames (36 to 1271 per clip versus Ped2's 120 to 180), the pattern reverses: disabling the gate hurts sharply (60.0% versus 70.2%), and the larger state (N=256) slightly beats the baseline rather than underperforming it. This is consistent with the gate's extra parameters (Eq. 2) overfitting on Ped2's very small training set but finding real signal once given Avenue's larger one, which points to a capacity and data-size interaction rather than the gating mechanism being unhelpful in principle (Section 4.3). We only have two data points, both

Table 3: Ablations on UCSD Ped2: architectural variants, all else held fixed. Latency/FPS measured the same way as Table 1.

Variant	Frame-AUC↑	EER↓	Latency (ms)↓	FPS↑
Baseline (gate on, N=128, decay in [.9,.999])	67.9	36.8	0.74	1344
Gate off	76.6	24.6	0.49	2047
Decay fast, decay in [.5,.9]	67.0	38.2	0.75	1330
Decay slow, decay in [.98,.9999]	67.8	37.0	0.76	1311
State size N=32	74.7	33.1	0.77	1302
State size N=256	66.5	38.7	0.77	1298
Gate off + state size N=32 (combined)	67.8	36.0	0.46	2156

Table 4: Ablations on CUHK Avenue: architectural variants, all else held fixed. Latency/FPS measured the same way as Table 1.

Variant	Frame-AUC↑	EER↓	Latency (ms)↓	FPS↑
Baseline (gate on, N=128, decay in [.9,.999])	70.2	34.9	0.77	1291
Gate off	60.0	43.8	0.51	1953
Decay fast, decay in [.5,.9]	70.6	34.4	0.75	1327
Decay slow, decay in [.98,.9999]	70.3	34.8	0.77	1303
State size N=32	70.3	35.0	0.76	1319
State size N=256	71.1	34.5	0.77	1302
State size N=256 + decay slow (combined)	71.1	34.6	0.78	1289

with 16 training clips and differing mainly in total frame count, so a third and larger dataset is needed before treating this as more than a hypothesis; we have deliberately not tuned the reported baseline to whichever ablation variant happened to score highest on either dataset, since doing so would misrepresent what the default, as-designed method actually achieves. The decay-rate sweep (fast versus slow base decay, gate on) shows comparatively little effect on either dataset, suggesting decay range is not the dominant factor here.

Since gate-off and the smaller state each independently helped on Ped2, we tested the natural follow-up hypothesis that combining them would help further; it did not. The combined configuration (gate off, N=32) reaches 67.8% frame-AUC, statistically indistinguishable from the untouched baseline (67.9%) and well below either individual change (76.6% and 74.7% respectively). The two changes do not stack, and on this evidence partially cancel each other rather than compounding, which is itself informative: whatever each change removes from the model's capacity to overfit is not simply additive across changes, and a single-factor ablation table can overstate how much

headroom is actually available by combining the best-looking row from each factor. We ran the equivalent combined check on Avenue (state N=256 with the slow decay range, both individually non-harmful or mildly helpful there) and found it matches the state-size-alone result (71.1%), consistent with the decay-rate sweep's already-small effect simply not adding anything on top. Table 3 and Table 4 report both combined rows alongside the single-factor ablations. We do not yet have a strict-causal-versus-windowed ablation, which would require a lookahead-buffer variant of the architecture in Section 3.3 that has not been implemented.

5.6. Edge deployment

The latency and FPS columns in Table 1 are already real, on-device measurements on the M3 Pro's MPS backend rather than simulated or extrapolated figures, and at 0.7 to 0.8 ms per frame they comfortably support real-time operation at conventional video frame rates with a wide margin to spare. We have not yet exported the model to CoreML to measure Apple Neural Engine throughput specifically, and we have not benchmarked on a second, non-Apple edge device (for example a Jetson or Raspberry Pi); both would make the real-time and edge-deployment claim more general than a single-vendor MPS measurement currently supports, and both are planned before submission.

6. Conclusion

We presented a strictly causal streaming video anomaly detector built on a causal diagonal state-space core with a learned event-boundary decay gate, together with a theoretical analysis relating the recurrence's decay spectrum to detection delay and an evaluation that includes real on-device edge latency measurements alongside standard accuracy metrics. The empirical picture is mixed by design rather than smoothed over after the fact. Frame-level accuracy, 67.9% and 70.2% AUC on Ped2 and Avenue, trails prior non-causal SSM baselines, and the event-boundary gate ablation reverses sign between the two datasets, which we read as evidence of a capacity and data-size interaction rather than a settled property of the gate.

Three limitations bound these claims and set the immediate agenda. First, the RBDC/TBDC-style evaluation used during development is a simplified frame-overlap approximation rather than the official region- and track-based criteria of Ramachandra and Jones [12]; any localization numbers re-

ported for submission should use the official toolkit instead. Second, edge-latency measurements come from a single hardware target, an Apple M3 Pro using PyTorch's MPS backend; a CoreML/Neural-Engine export and a second, non-Apple edge device would make the real-time claim more general than it currently is. Third, the settling-delay theorem (Proposition 1) treats the decay as fixed, while the deployed model's gate makes it time-varying, so the bound is a loose upper bound rather than a tight prediction once the gate is active. Beyond these, evaluating on ShanghaiTech and closing the accuracy gap to prior work through backbone and hyperparameter tuning are what remain. A same-protocol reproduction of a prior SSM-based baseline would strengthen the comparison further, but is not straightforward here: VADMamba's released code depends on CUDA-only optical-flow kernels (Section 2) that do not run on the non-CUDA edge hardware this paper targets, so a faithful reproduction would require either CUDA hardware unavailable to us or a substantial reimplementation that would no longer be the original method.

References

[1] M. J. Jones, B. Ramachandra, R. R. Vatsavai, A survey of single-scene video anomaly detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (5) (2022) 2293–2312. doi:10.1109/TPAMI.2020.3040591.

[2] W. Luo, W. Liu, D. Lian, S. Gao, Future frame prediction network for video anomaly detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (11) (2022) 7505–7520. doi:10.1109/TPAMI.2021.3129349.

[3] C. Cao, H. Zhang, Y. Lu, P. Wang, Y. Zhang, Scene-dependent prediction in latent space for video anomaly detection and anticipation, IEEE Transactions on Pattern Analysis and Machine Intelligence (2025). doi:10.1109/TPAMI.2024.3461718.

[4] C. Qiu, M. Kloft, S. Mandt, M. Rudolph, Self-supervised anomaly detection with neural transformations, IEEE Transactions on Pattern Analysis and Machine Intelligence 47 (3) (2025) 2170–2185. doi:10.1109/TPAMI.2024.3519543.

[5] A. Gu, T. Dao, Mamba: Linear-time sequence modeling with selective state spaces, arXiv preprint arXiv:2312.00752 (2023).

[6] J. Lyu, M. Zhao, J. Hu, X. Huang, Y. Chen, S. Du, Vadmamba: Exploring state space models for fast video anomaly detection, arXiv preprint arXiv:2503.21169Presented at IEEE ICME 2025 (2025).

[7] J. Lyu, M. Zhao, J. Hu, Y. Chen, S. Du, C. Shi, Vadmamba++: Efficient video anomaly detection via hybrid modeling in grayscale space, arXiv preprint arXiv:2604.00360 (2026).

[8] Q. Zhang, M. Shao, X. Chen, X. Lv, K. Xu, Wave-mambaad: Wavelet-driven state space model for multi-class unsupervised anomaly detection, in: IEEE/CVF International Conference on Computer Vision (ICCV), 2025.

[9] W. Liu, W. Luo, D. Lian, S. Gao, Future frame prediction for anomaly detection – a new baseline, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[10] H. Park, J. Noh, B. Ham, Learning memory-guided normality for anomaly detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[11] W. Sultani, C. Chen, M. Shah, Real-world anomaly detection in surveillance videos, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[12] B. Ramachandra, M. J. Jones, Street scene: A new dataset and evaluation protocol for video anomaly detection, in: IEEE Winter Conference on Applications of Computer Vision (WACV), 2020, pp. 2569–2578.

[13] A. Gu, K. Goel, C. Ré, Efficiently modeling long sequences with structured state spaces, in: International Conference on Learning Representations (ICLR), 2022.

[14] J. T. H. Smith, A. Warrington, S. W. Linderman, Simplified state space layers for sequence modeling, in: International Conference on Learning Representations (ICLR), 2023.

[15] H. He, Y. Bai, J. Zhang, Q. He, H. Chen, Z. Gan, C. Wang, X. Li, G. Tian, L. Xie, Mambaad: Exploring state space models for multi-class unsupervised anomaly detection, in: Advances in Neural Information Processing Systems (NeurIPS), 2024, arXiv:2404.06564.

[16] Z. Wu, C. Xiong, C.-Y. Ma, R. Socher, L. S. Davis, Adaframe: Adaptive frame selection for fast video recognition, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[17] B. Korbar, D. Tran, L. Torresani, Scsampler: Sampling salient clips from video for efficient action recognition, in: IEEE/CVF International Conference on Computer Vision (ICCV), 2019.