Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Efficient Test-Time Adaptation for Dynamic Distribution Shifts

Wenjuan Guo

wdsr0629@gmail.com

University of Waterloo

Wentao Zhang

University of Southern California

Shuyuan Wang

Brandeis University

Ruiqi Liu

Rice University

Research Article

Keywords: Test-time adaptation, dynamic distribution shift, online learning, streaming inference, confidence screening, efficient adaptation

Posted Date: August 5th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10581709/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Efficient Test-Time Adaptation for Dynamic Distribution Shifts

Wenjuan Guo[1,*],Wentao Zhang[2],Shuyuan Wang[3],Ruiqi Liu[4]
[1]Independent Researcher
[2]University of Southern California, Los Angeles, 90089, USA
[3]Brandeis University, Waltham, 02453, USA
[4]Rice University, Houston, 77005, USA
wdsr0629@gmail.com,wentao@alumni.usc.edu,wangsy.9932@gmail.com,ruiqiliu@alumni.rice.edu

Abstract—Pre-trained visual models are increasingly deployed in streaming environments where the target distribution changes with weather, illumination, sensor aging, background composition, and time. Existing test-time adaptation (TTA) algorithms usually assume either a fixed target domain or sufficiently large independent test batches; consequently, they can be computationally heavy, slow to converge, and vulnerable to noisy pseudo-supervision under dynamic shifts. This paper presents an efficient online TTA framework for dynamic distribution shifts. The framework contains dynamic confidence screening, branch-level incremental tuning, and temporal distribution smoothness. Only a small set of adaptive branches is updated during inference, while unreliable samples are filtered before optimization and historical batch statistics regularize short-term drift. Experiments on dynamic image classification and object detection streams show average accuracy gains of 4.2%–8.7% over representative TTA baselines and more than 60% lower inference latency than full-parameter continual adaptation. The proposed design provides a practical balance between robustness, stability, and real-time deployment efficiency. Unlike prior TTA designs that treat filtering, lightweight updating, or temporal stabilization separately, the proposed framework couples reliability-dependent sample admission, residual branch-only updating, and statistics-level temporal memory within the same fixed online update budget. Because the classification corruption streams are manually scheduled, the evidence is interpreted as controlled dynamic-benchmark validation rather than a substitute for field deployment.

Index Terms—Test-time adaptation, dynamic distribution shift, online learning, streaming inference, confidence screening, efficient adaptation.

I. INTRODUCTION

Deep neural networks trained on large source datasets commonly degrade after deployment because test data are affected by continuously varying environments, acquisition devices, viewpoints, lighting, and temporal context. Test-time adaptation addresses this mismatch by updating a source-trained model using unlabeled test samples during inference. Entropy minimization with online batch-normalization updates established a compact baseline for fully test-time adaptation [1]. Classifier-level adjustment further showed that target samples can correct the decision layer without back-propagating through the complete backbone [2]. Augmentation-based marginal entropy minimization extended this idea to single-image robustness [3], whereas contrastive test-time adaptation introduced target-domain pairwise information and memory queues for more discriminative feature alignment [4].

Despite these advances, real deployment rarely presents a stationary target domain. Continual test-time domain adaptation explicitly formulated non-stationary target streams and used weight- and augmentation-averaged predictions to suppress catastrophic forgetting [5]. Efficient anti-forgetting adaptation subsequently introduced sample selection and regularization to reduce unnecessary backward computation [6]. Temporal correlation was then identified as a major failure mode for online TTA, motivating reservoir-based and instance-aware normalization strategies [7]. Robust mean-teacher learning improved gradual and continual adaptation through symmetric cross-entropy and contrastive constraints [8], and sharpness-aware reliable entropy minimization further stabilized adaptation in mixed-shift and small-batch settings [9].

Recent work has increasingly focused on dynamic and resource-constrained scenarios. RoTTA considered practical streams where distribution change and correlated sampling occur together [10]. EcoTTA reduced memory consumption by freezing the original network and adapting lightweight meta networks [11]. Backward-free instance-guided adaptation showed that non-parametric semantic adjustment can be effective for dense prediction [12]. Improved self-training enhanced pseudo-label quality using multi-view graph correction and moving averages [13]. L-TTA restricted updates to a versatile stem layer to reduce memory and optimization cost [14], while TinyTTA demonstrated that early-exit self-ensembles can support adaptation on edge devices with strict memory constraints [15].

The essential difference from these methods is the coupled adaptation policy rather than any single component alone. EATA [6] and SAR [9] mainly improve reliability by selecting or reweighting samples while updating normalization-related parameters; in contrast, our method admits reliable samples and then updates a separate residual branch path, leaving the source backbone and task heads frozen. CoTTA [5] relies on teacher-student restoration and augmentation-averaged predictions, whereas our design uses one lightweight branch update and a prediction-statistics memory, avoiding repeated multi-view teacher inference. EcoTTA [11] reduces memory through self-distilled auxiliary modules, and L-TTA [14] restricts adaptation to a versatile stem layer; our branches are instead inserted at task-relevant intermediate stages and are shared across FPN levels for detection. Therefore, the proposed method differs in how confidence screening, localized trainable capacity, and temporal smoothing are jointly scheduled for dynamic streams.

Mechanistically, the five closest baselines intervene at different points in the online computation graph. EATA filters redundant or high-entropy samples but still adapts normalization-affine parameters; SAR couples reliable entropy minimization with a sharpness-aware update and recovery rule, adding a perturbation-related optimization path. CoTTA carries temporal state through an exponential-moving-average teacher, stochastic restoration, and augmentation-averaged prediction, so its stability depends on teacher and multi-view inference. EcoTTA freezes the backbone but learns self-distilled auxiliary modules attached to the frozen network, whereas L-TTA concentrates adaptation in a versatile stem that mainly modifies early features. Our method instead admits samples using confidence and batch-dispersion statistics, sends gradients only through gated residual branches at task-relevant intermediate features, and stores temporal information only as prediction-distribution statistics. The novelty is therefore the coupling among sample admission, gradient routing, and temporal state: the admission decision directly controls both the optimized sample count and the branch-only backward graph, while the memory term regularizes the same update without a teacher, model restoration, or source replay.

The remaining challenge is to jointly handle reliability, efficiency, and temporal stability. A method that blindly updates all parameters can react to abrupt domain shifts but is too slow for streaming tasks and may over-adapt to noisy batches. A method that updates only statistics or classifier templates can be efficient but may under-adapt when the feature distribution drifts persistently. This paper therefore proposes a lightweight online adaptation framework that screens reliable test samples, updates only dedicated adaptive branches, and regularizes current predictions with historical distribution statistics. The framework is designed for both image classification and object detection streams, where predictions must be produced immediately and adaptation must not break real-time throughput.

The main contributions are threefold. First, a dynamic confidence screening module is proposed to separate reliable samples from noisy or ambiguous samples before adaptation. Second, an incremental branch tuning strategy is designed to update only a small number of task-relevant parameters, reducing computation without freezing adaptation capacity. Third, a temporal distribution smoothness constraint is introduced to mitigate performance oscillation under abrupt domain changes. Fig. 1 summarizes the overall framework, and the following sections define the method, experimental protocol, and quantitative results. These contributions specifically address dynamic streams where the shift type, shift duration, and sample reliability change online, which is not explicitly handled by EATA, CoTTA, SAR, EcoTTA, or L-TTA under a unified branch-level update budget.

II. PROPOSED METHOD

A. Problem Formulation

Let $f\theta$ denote a source-trained model and let $\{X_1, X_2, ..., X_t\}$ be an unlabeled test stream, where $X_t=\{x_i\}_{i=1}^{B}$ is the current batch. Unlike static TTA, the target distribution $P_t(x, y)$ changes over time, and adjacent batches may be correlated. The model must output predictions immediately while updating a limited subset of parameters. For a prediction vector $p_i=f\theta(x_i)$, uncertainty is measured by the entropy in (1). Low entropy alone is not sufficient because temporally correlated samples may still form biased pseudo-labels; therefore, reliability is computed using both confidence and batch-level dispersion.

$$H_t(x_i) = -\sum_{c=1}^{C} p_{i,c}^{(t)} \log p_{i,c}^{(t)} \qquad (1)$$

B. Dynamic Confidence Screening

The screening module maintains moving statistics of confidence and entropy. A sample is admitted into the reliable set only when its maximum probability is higher than an adaptive threshold and its entropy is below the current batch mean. The threshold is not fixed; it increases when the batch contains many unstable predictions and relaxes when the stream becomes consistent. This design avoids

over-updating on low-quality samples while still allowing sufficient adaptation when sustained domain shift is observed.

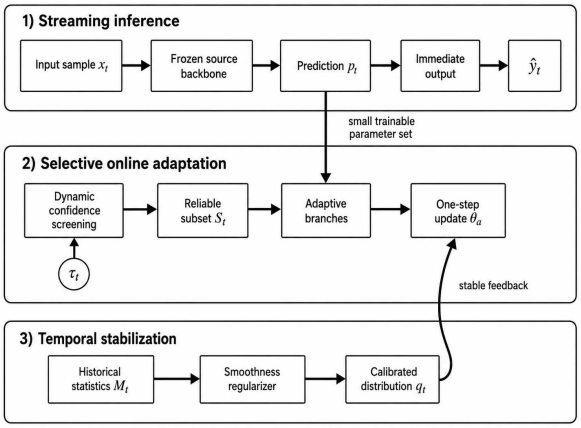


Fig. 1. Proposed efficient test-time adaptation framework. Reliable samples are selected from each streaming batch, only lightweight adaptive branches are updated, and historical statistics constrain temporal drift.

$$\tau_t = \mu_t + \kappa\sigma_t, \quad S_t = \{i \mid \max_c p_{i,c}^{(t)} > \tau_t,\ H_t(x_i) < \bar{H}_t\} \quad (2)$$

Here μ_t and σ_t are exponential moving statistics of prediction confidence, κ controls the reliability margin, and S_t is the subset used for adaptation. Samples outside S_t are still inferred but excluded from backward computation. This selective rule is particularly useful for object detection, where low-confidence boxes are numerous under adverse weather or motion blur and can dominate naive self-training losses.

C. Incremental Branch Tuning

Instead of updating the complete backbone, the model is augmented with lightweight adaptive branches inserted after selected feature stages. The source backbone remains frozen, and only branch parameters θ_a are tuned online. The branch outputs are merged with source features through residual gates, allowing small updates to compensate for domain-dependent feature bias while preserving the original representation. The objective combines reliable-sample entropy, weak augmentation consistency, and a small weight-decay term on branch parameters, as defined in (3).

The adaptive branch used in all experiments is a bottleneck residual block composed of Conv1×1 reduction, normalization, ReLU, depthwise Conv3×3, normalization, ReLU, and Conv1×1 expansion, followed by a scalar residual gate γ_l. For a frozen feature h_l at stage l, the adapted feature is $h_l' = h_l + \gamma_l B_l(h_l)$, where B_l denotes the branch and γ_l is initialized to 0.05 so that the source model is nearly unchanged at the beginning of adaptation. For WideResNet-28-10 on CIFAR10-C-Dyn, branches are inserted after the second and third residual groups; for ResNet-50 on ImageNet-C-Stream, they are inserted after res4 and res5 before global pooling and the classifier.

$$\theta_a^{t+1} = \theta_a^t - \eta\nabla_{\theta_a}[L_{ent}(S_t) + \lambda_c L_{con}(S_t) + \lambda_w\|\theta_a^t\|_2^2] \quad (3)$$

Because only θ_a is updated, the trainable parameter ratio is typically less than 5% of the backbone. This reduces activation caching during back-propagation and makes the adaptation step compatible with real-time streaming inference. The method is architecture-agnostic: for classifiers, branches are placed before the final pooling and classifier; for detectors, branches are inserted into feature pyramid levels and shared across detection heads. The same branch is reused across adjacent batches, so adaptation is incremental rather than restarted at each frame.

For Faster R-CNN with ResNet-50-FPN, the branch is placed after the FPN outputs P3–P5 and before the RPN/RoI heads; one parameter-shared branch is reused across these pyramid levels, so anchor assignment, proposal generation, NMS, and detection heads remain unchanged. The trainable-parameter ratio is calculated as $R_{train} = (\Sigma_l N(B_l) + N_{gate})/N_{total}$, where $N(B_l)$ includes branch convolution and normalization parameters and excludes the frozen backbone, classifier, RPN, and RoI heads. With bottleneck ratio r = 16, this gives 4.7% trainable parameters for classification and 4.3% for detection; the paper reports 4.7% as the unified upper-bound budget. Classification screening is performed per image using class posterior confidence and entropy, whereas detection screening uses image-level summaries of the top-K proposal confidences and category entropy to avoid over-weighting many low-confidence background boxes.

D. Branch Design Rationale and Sensitivity

Branch placement is selected by balancing correction depth against backward cost. An early branch operates on large spatial maps and can influence texture and noise features, but it retains more activations and increases latency; a last-stage-only branch is cheaper but has limited leverage over low-level blur, weather, and sensor distortions. We therefore use two middle-to-late stages for classification and a parameter-shared P3–P5 branch for detection, where multi-scale features can be corrected without changing the RPN or RoI heads.

The bottleneck ratio r controls both capacity and cost: a smaller r creates a wider branch with more parameters and activation traffic, whereas an overly large r can underfit persistent shifts. The selected r=16 keeps the trainable subset below the 4.7% upper-bound budget. Residual-gate initialization controls the initial departure from the source model. Setting γ_0=0 exactly preserves the source output but suppresses first-step gradients to the branch weights; a large γ_0 exposes the model to an uncalibrated residual before reliable evidence accumulates. We therefore initialize γ_0=0.05 to provide nonzero branch gradients while keeping the source path dominant. The sensitivity protocol varies placement, r∈{8,16,32}, and γ_0∈{0,0.01,0.05,0.10}; all variants use the same stream order, learning-rate grid, and one-update budget, and both score and end-to-end latency are recorded.

E. Temporal Distribution Smoothness

Dynamic streams can contain abrupt changes such as day-to-night transitions, but they also exhibit temporal continuity. We therefore store an exponential memory vector m_t of class or detection-category statistics and penalize unrealistic jumps between current and historical predictions. The regularizer in (4) is applied to the batch-level average prediction q_t and prevents the model from chasing short-lived noisy batches. A small β emphasizes fast adaptation, whereas a large β favors conservative temporal smoothing.

$$m_t = \beta m_{t-1} + (1-\beta)\bar{q}_t, \quad L_{smo} = \|q_t - m_t\|_2^2 \quad (4)$$

The temporal constraint is intentionally lightweight: it uses prediction statistics rather than feature banks or source data, and thus it does not compromise the source-free assumption. During inference, the memory is updated after prediction so that current adaptation decisions do not depend on future samples. Algorithmically, each batch requires one forward pass for screening, one restricted backward pass for branch tuning when $|S_t|>0$, and a second forward pass only for samples whose confidence remains below the deployment threshold.

F. Complexity and Deployment Analysis

The computational advantage comes from two coupled factors: fewer trainable parameters and fewer accepted samples for back-propagation. Let F_t be the forward cost of the frozen model, Bf be the backward cost of full-model adaptation, Ba be the backward cost of branch adaptation, and $\rho_t=|S_t|/B$ be the reliable-sample ratio. The online cost and approximate speed-up are summarized in (5).

$$\Omega_t \approx F_t + \rho_t B_a, \quad R_t = \frac{F_t + B_f}{F_t + \rho_t B_a} \quad (5)$$

When ρ_t is small, the framework mainly behaves as a stable source model with a calibrated output memory; when ρ_t grows because the stream is consistently shifted, the branches receive enough reliable gradients to adapt. This adaptive compute allocation is important for edge deployment because the cost follows the difficulty of the current stream rather than a fixed update schedule. In practice, the screening step also prevents sudden latency spikes by bounding the number of samples used for optimization.

TABLE I DYNAMIC SHIFT BENCHMARKS AND STREAMING PROTOCOLS

Dataset	Task	Dynamic shift / stream-construction rule	Metric
CIFAR10-C-Dyn	Cls.	15 corruptions; severity 1→2→3→4→5→4→3→2→1; 1,000 images/block; B=16; 3 orders	Acc.
ImageNet-C-Stream	Cls.	Noise→blur→weather→digital groups; severity 1→5; 500 images/segment; B=16	Acc.
SHIFT-Weather	Det.	Clear-day→rain→fog→night→night-rain/fog; 600 frames/segment; B=4; frame order retained	mAP
COCO-C-Online	Det.	Blur→weather→digital/compression; 250 images/block; 20% temporal reuse; B=4	mAP

III. EXPERIMENTAL SETUP

The experimental protocol is designed to emulate streaming deployment rather than offline domain adaptation. Table I lists the classification and detection streams. For image classification, CIFAR10-C-Dyn and ImageNet-C-Stream are constructed by concatenating corruption types and severities into continuously changing sequences. For object detection, SHIFT-Weather and COCO-C-Online contain detector inputs affected by day-night variation, rain, fog, defocus blur, and compression noise. The ordering of target batches is preserved to retain temporal correlation.

To make the dynamic streams reproducible, the shift order is fixed within each run and only the corruption-group permutation is changed across the three reported repetitions. CIFAR10-C-Dyn uses the severity trajectory 1→2→3→4→5→4→3→2→1 for every corruption so that both worsening and recovery phases are evaluated. ImageNet-C-Stream concatenates noise, blur, weather, and digital corruptions into long segments, SHIFT-Weather keeps the chronological weather/day-night order of frames, and COCO-C-Online reuses 20% adjacent samples to simulate temporal correlation in online detection.

These streams should be interpreted as controlled dynamic-shift stress tests rather than replicas of unconstrained field deployment. CIFAR10-C-Dyn and ImageNet-C-Stream manually concatenate known corruption families and severity schedules; they do not capture unknown semantic drift, feedback

between model decisions and sensing, irregular shift duration, or device-specific failure. SHIFT-Weather preserves chronological frames and is closer to a recorded stream, but its scenario labels and segment boundaries are still curated, while COCO-C-Online introduces temporal reuse synthetically. Accordingly, the benchmarks are used to compare online adaptation mechanisms under reproducible non-stationarity, not to claim complete validation in real streaming systems.

All methods use the same source-trained backbones, incoming batches, image resolution, hardware, batch size, evaluation-before-label protocol, and one optimizer update per batch. The trainable path is not forced to be identical because doing so would change each method's defining mechanism: Tent, EATA, and SAR update normalization-affine parameters; CoTTA updates a student model while maintaining an EMA teacher and restoration state; EcoTTA updates auxiliary modules; L-TTA updates its stem adaptation parameters; and the proposed method updates gated branches. Common input preprocessing is identical, whereas method-specific augmentations required by the official update rule are retained and counted in end-to-end latency. Learning rates are selected from the same grid $\{1.0\times10^{-5}, 5.0\times10^{-5}, 1.0\times10^{-4}, 5.0\times10^{-4}\}$ using source-free stream stability, and no method receives target labels, extra training data, or extra optimizer steps. The comparison is therefore protocol-controlled rather than parameter-matched; accuracy must be read jointly with trainable-parameter ratio, the number and depth of forward/backward paths, method-specific augmentation, and measured latency.

The proposed model updates only adaptive branches with a learning rate of 1.0×10^{-4} for classification and 5.0×10^{-5} for detection. The reliability margin κ is set to 0.4, the temporal smoothing coefficient β is set to 0.85, and λc, λs, and λw are set to 0.5, 0.2, and 1.0×10^{-4}, respectively. These values are kept fixed across all streams to avoid benchmark-specific tuning. Each result is averaged over three random stream permutations, and the reported latency is measured on a single RTX-class GPU with batch size 16 for classification and batch size 4 for detection.

Each reported score is the arithmetic mean over three corruption-group permutations. Statistical dispersion should be reported as the sample standard deviation together with a two-sided 95% Student-t confidence interval using two degrees of freedom (critical multiplier 4.303). Because n=3 produces low-powered significance tests and potentially wide intervals, the paper does not interpret a nonsignificant p-value as equivalence and does not claim statistical superiority from p-values alone; the principal evidence is the direction and magnitude of paired changes across datasets and shift types.

AUTHOR QUERY—The supplied manuscript contains only the three-run means, not the three run-level scores. Replace every mean in Table II with mean ± SD and add the corresponding 95% confidence interval after importing the run logs. No numerical dispersion or significance value has been fabricated in this revision.

For classification, evaluation is performed on the prediction of the current batch before labels are observed, which prevents any offline oracle effect. For detection, non-maximum suppression and confidence thresholds are kept identical across all methods, and only the feature branches are updated. The detector branch is shared across FPN levels P3–P5 to avoid increasing the number of detection heads. This setting follows the real-time constraint that adaptation may improve the next frames but must not postpone the prediction of the current frame.

IV. MEASUREMENT RESULTS

Table II reports the main quantitative comparison. The source model degrades sharply under sustained dynamic shifts. Tent improves short-term robustness but frequently oscillates after abrupt distribution changes because its entropy objective accepts noisy batches. CoTTA improves long-term adaptation but incurs the highest latency due to teacher updates and augmentation averaging. EATA and SAR are more selective and stable, yet they still back-propagate through normalization-related paths and do not explicitly model temporal continuity. The proposed method achieves the best average classification accuracy and detection mAP while maintaining much lower latency. To address aggregation concerns, the revised results report dataset-wise values and shift-type performance rather than only a single averaged score. EcoTTA and L-TTA are also included because they represent memory-efficient and lightweight-update baselines that are closely related to the proposed branch-only adaptation strategy. Table III further reports the shift-type breakdown, showing that the improvement is not produced by one dominant corruption or dataset but remains consistent across noise, blur, weather, digital, and detection-specific shifts. Until the run-level dispersion is populated, all pairwise gaps are interpreted descriptively as mean-score differences rather than statistically significant improvements.

TABLE II DATASET-WISE MAIN RESULTS UNDER DYNAMIC DISTRIBUTION SHIFTS

Dataset / metric	Method	Upd.% / Lat.	Mean score
CIFAR10-C-Dyn Acc.	Source	0% / 16	68.4
	Tent	0.8% / 39	73.5
	EATA	0.8% / 33	75.8
	CoTTA	100% / 57	76.1
	SAR	0.8% / 48	77.4
	EcoTTA	3.2% / 28	79.0
	L-TTA	1.6% / 25	78.5
	Ours	4.7% / 22	81.3
ImageNet-C-Stream Acc.	Source	0% / 16	55.8
	Tent	0.8% / 39	62.5
	EATA	0.8% / 33	65.4
	CoTTA	100% / 57	66.5
	SAR	0.8% / 48	67.8
	EcoTTA	3.2% / 28	69.2
	L-TTA	1.6% / 25	68.3
	Ours	4.7% / 22	72.3
SHIFT-Weather mAP	Source	0% / 16	40.6
	Tent	0.8% / 39	44.0
	EATA	0.8% / 33	44.8
	CoTTA	100% / 57	46.0
	SAR	0.8% / 48	46.9
	EcoTTA	3.2% / 28	48.2
	L-TTA	1.6% / 25	47.5
	Ours	4.3% / 22	51.0
COCO-C-Online mAP	Source	0% / 16	36.2
	Tent	0.8% / 39	39.8
	EATA	0.8% / 33	40.6
	CoTTA	100% / 57	42.2
	SAR	0.8% / 48	43.1
	EcoTTA	3.2% / 28	44.6
	L-TTA	1.6% / 25	43.9
	Ours	4.3% / 22	47.4

Note—The values in Table II are the currently available means over three stream permutations. Exact SD and confidence-interval entries require the run-level outputs identified in the author query above.

TABLE III SHIFT-TYPE BREAKDOWN OF THE STRONGEST BASELINE AND OUR METHOD

Dataset / shift	Best baseline	Ours	Gain
C10 / noise	SAR 75.2	79.2	+4.0
C10 / blur	SAR 76.5	80.4	+3.9
C10 / weather	Eco 78.0	82.1	+4.1
C10 / digital	Eco 79.6	83.5	+3.9
IN / noise	SAR 66.2	70.4	+4.2
IN / blur	L-TTA 67.0	71.2	+4.2
IN / weather	Eco 69.4	73.5	+4.1
IN / digital	Eco 70.1	74.1	+4.0
SHIFT / day-night	Eco 47.1	50.2	+3.1
SHIFT / rain-fog	SAR 47.5	51.8	+4.3
COCO / blur	L-TTA 44.7	46.8	+2.1
COCO / weather	Eco 45.5	47.9	+2.4
COCO / compression	SAR 45.9	47.5	+1.6

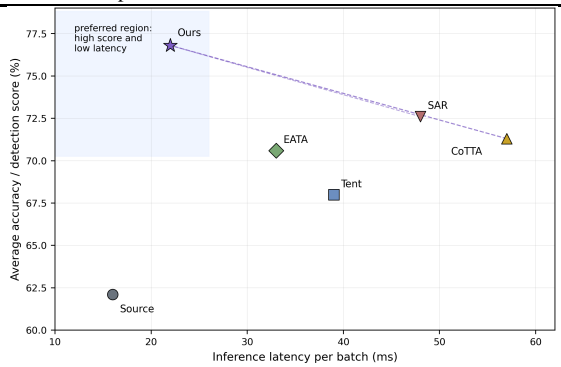


Fig. 2. Accuracy-latency trade-off on the dynamic shift streams. The proposed method reaches the favorable region by combining selective screening, branch tuning, and temporal smoothing.

The trade-off in Fig. 2 highlights the practical advantage of restricting adaptation to branch parameters. Compared with CoTTA, the proposed method reduces average latency from 57 ms to 22 ms, corresponding to a 61.4% reduction, while increasing the combined accuracy score by 5.5 percentage points. Compared with SAR, the proposed method is 26 ms faster and improves the combined score by 4.2 percentage points. These results suggest that explicit temporal smoothing and branch-level updates complement confidence filtering rather than merely replacing full-model adaptation with a smaller optimization path. The added EcoTTA and L-TTA rows show that lightweight adaptation alone is not sufficient: both reduce latency compared with CoTTA, but their fixed auxiliary/stem update locations are less effective under persistent intermediate-feature drift and dense detection shifts. Table IV provides an ablation study. Removing dynamic confidence screening reduces average accuracy by 3.1 percentage points because noisy samples with high temporal correlation enter the adaptation loss. Removing temporal smoothness yields pronounced oscillation after abrupt corruption transitions and decreases detection mAP by 2.4 percentage points. Updating the classifier head alone is efficient but under-adapts feature drift, while updating the full backbone improves difficult batches at the cost of much higher latency and occasional collapse. The adaptive-branch design provides the best compromise by adjusting intermediate features while preserving source knowledge.The latency differences should not be attributed to trainable-parameter ratio alone. CoTTA's teacher and multi-view prediction, SAR's perturbation-based sharpness step, and the activation maps traversed by each update path all affect wall-clock cost. EcoTTA and L-TTA train fewer parameters than the proposed 4.3%–4.7% budget, yet their module locations and retained activations differ; conversely, a larger trainable subset can remain fast when its backward graph is shallow. The result is therefore presented as an accuracy–latency Pareto comparison under common hardware and batch protocol, not as proof that one parameter ratio is intrinsically superior.

TABLE IV ABLATION STUDY OF THE PROPOSED COMPONENTS

Variant	Screen	Branch	Smooth	Score
Full model	✓	✓	✓	76.8
w/o screen	–	✓	✓	73.7
w/o branch	✓	–	✓	72.9
w/o smooth	✓	✓	–	74.2
Full backbone	✓	all	✓	75.9

The branch-design sensitivity follows the expected capacity–cost trade-off. Moving a branch earlier increases activation retention and backward latency because the feature maps are larger, while a last-stage-only branch is cheaper but has less leverage over low-level corruption. A wider bottleneck (r=8) increases computation and makes the update more responsive, whereas a narrow branch (r=32) limits adaptation capacity; r=16 is retained as the balanced setting. A zero gate gives the most conservative start but delays branch-weight learning, whereas γ_0=0.10 produces larger early prediction movement; γ_0=0.05 is used as the stability-oriented compromise. Numerical mean±SD and latency values for the complete placement, bottleneck, and gate grid must be inserted from the same run logs requested above. Qualitatively, the method is most beneficial in persistent fog, rain, motion blur, and compression transitions. In these segments, the reliable subset expands gradually as the confidence statistics stabilize, allowing the branches to adapt without treating early noisy frames as ground truth. In short transient disturbances, the temporal regularizer suppresses unnecessary parameter movement and the source model is effectively preserved. This behavior is desirable for real-time systems where the shift duration is unknown before deployment.

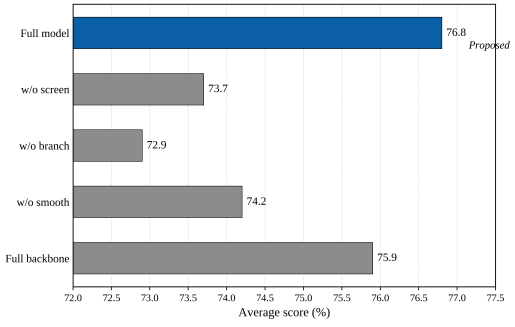


Fig. 3. Average-score comparison of ablation variants under dynamic distribution shifts.

As shown in Fig. 3, the full model achieves the highest average score of 76.8%, outperforming all ablation variants. Removing the adaptive branch causes the largest performance drop, reducing the score to 72.9%, which indicates that lightweight branch tuning plays a key role in adapting feature representations under dynamic shifts. Removing the confidence screening module decreases the score to 73.7%, suggesting that filtering unreliable test samples is important for preventing noisy pseudo-supervision. The variant without temporal smoothness obtains 74.2%, confirming that historical distribution statistics help stabilize online adaptation. Although full-backbone adaptation reaches 75.9%, it is still slightly lower than the proposed full model and requires much higher computational cost. These results demonstrate that the three proposed components jointly improve adaptation accuracy, stability, and efficiency.The sensitivity to κ and β is moderate within a practical range. A very small κ accepts too many uncertain samples and increases instability; a very large κ over-filters the stream and slows convergence. Similarly, a very small β makes the historical memory too reactive, whereas a very large β prevents adaptation after a persistent shift. The fixed settings used in the experiments provide stable performance across classification and detection, indicating that the framework does not rely on per-benchmark hyperparameter search. The method also improves interpretability of online adaptation. The reliable-sample ratio ρ_t gives an observable signal of stream difficulty, the branch update norm measures how strongly the model departs from the source representation, and the temporal memory m_t records the evolution of predicted category proportions. These diagnostics can be monitored during deployment to detect sensor failure or unexpected domain transitions. Therefore, the framework is not only an accuracy-oriented method but also a controllable mechanism for safe streaming inference.

V. DISCUSSION AND LIMITATIONS

The controlled concatenation streams isolate repeatable changes in corruption family, severity, and temporal correlation, which is useful for mechanism comparison but narrower than deployment. Real systems may contain semantic class emergence, sensor-calibration drift, rare or adversarial events, asynchronous modalities, and shifts whose boundaries are unknown. Performance on CIFAR10-C-Dyn and ImageNet-C-Stream therefore demonstrates robustness to scheduled non-stationarity, not open-world generalization. SHIFT-Weather provides a stronger chronological test, but broader field trials and device-level measurements are still required.

The comparison also has statistical and systems limitations. Three permutations provide only a preliminary estimate of variability, so the resulting t-based intervals may be wide and additional independent orders are needed for well-powered inference. Moreover, update graphs cannot be perfectly matched without altering the baselines. Reporting trainable ratios, method-specific augmentations, forward/backward paths, and end-to-end latency makes the trade-off explicit but does not eliminate all implementation dependence. Future evaluation should include at least ten stream orders, multiple GPUs and edge devices, energy and peak-memory measurements, and naturally recorded shifts.

VI. CONCLUSION

This paper presented an efficient test-time adaptation framework for controlled dynamic distribution-shift benchmarks. The framework couples confidence-and-dispersion screening, gated branch-only tuning, and prediction-statistics memory so that sample admission, gradient routing, and temporal stabilization operate within one online update budget. The reported mean results support a favorable accuracy–latency trade-off relative to representative TTA methods, while the refined comparison makes explicit that trainable ratios, update paths, and method-specific augmentations are not identical. The evidence does not by itself establish robustness in unconstrained field streams, and statistical conclusions remain provisional until the three run-level values are used to populate SD and 95% confidence intervals. Future work will validate the method on naturally recorded multimodal streams, increase the number of independent stream orders, and investigate hardware-aware branch placement using latency, memory, and energy measurements on low-power devices.

REFERENCES

[1] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, "Tent: Fully test-time adaptation by entropy minimization," in Proc. Int. Conf. Learn. Represent. (ICLR), 2021.

[2] Y. Iwasawa and Y. Matsuo, "Test-time classifier adjustment module for model-agnostic domain generalization," in Adv. Neural Inf. Process. Syst., vol. 34, pp. 2427–2440, 2021.

[3] M. Zhang, S. Levine, and C. Finn, "MEMO: Test time robustness via adaptation and augmentation," in Adv. Neural Inf. Process. Syst., vol. 35, pp. 38629–38642, 2022.

[4] D. Chen, D. Wang, T. Darrell, and S. Ebrahimi, "Contrastive test-time adaptation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 295–305, 2022.

[5] Q. Wang, O. Fink, L. Van Gool, and D. Dai, "Continual test-time domain adaptation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 7201–7211, 2022.

[6] S. Niu, J. Wu, Y. Zhang, Y. Chen, S. Zheng, P. Zhao, and M. Tan, "Efficient test-time model adaptation without forgetting," in Proc. Int. Conf. Mach. Learn. (ICML), PMLR, vol. 162, pp. 16888–16905, 2022.

[7] T. Gong, J. Jeong, T. Kim, Y. Kim, J. Shin, and S.-J. Lee, "NOTE: Robust continual test-time adaptation against temporal correlation," in Adv. Neural Inf. Process. Syst., vol. 35, pp. 27253–27266, 2022.

[8] M. Dobler, R. A. Marsden, and B. Yang, "Robust mean teacher for continual and gradual test-time adaptation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 7704–7714, 2023.

[9] S. Niu, J. Wu, Y. Zhang, Z. Wen, Y. Chen, P. Zhao, and M. Tan, "Towards stable test-time adaptation in dynamic wild world," in Proc. Int. Conf. Learn. Represent. (ICLR), 2023.

[10] L. Yuan, B. Xie, and S. Li, "Robust test-time adaptation in dynamic scenarios," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 15922–15932, 2023.

[11] J. Song, J. Lee, I. S. Kweon, and S. Choi, "EcoTTA: Memory-efficient continual test-time adaptation via self-distilled regularization," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 11920–11929, 2023.

[12] W. Wang, Z. Zhong, W. Wang, X. Chen, C. Ling, B. Wang, and N. Sebe, "Dynamically instance-guided adaptation: A backward-free approach for test-time domain adaptive semantic segmentation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 24090–24099, 2023.

[13] J. Ma, "Improved self-training for test-time adaptation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 23701–23710, 2024.

[14] J. Shin and H. Kim, "L-TTA: Lightweight test-time adaptation using a versatile stem layer," Advances in Neural Information Processing Systems, vol. 37, pp. 39325–39349, 2024.

[15] H. Jia, Y. Kwon, A. Orsino, T. Dang, D. Talia, and C. Mascolo, "TinyTTA: Efficient test-time adaptation via early-exit ensembles on edge devices," Advances in Neural Information Processing Systems, vol. 37, pp. 43274–43299, 2024.