DeepLLR-CUSUM: Sequential Change Detection with Learned Log-Likelihood Ratios for Site Reliability Engineering

1 Supplemental Materials

1.1 Methodology

1.1.1 Datasets and Preprocessing

We utilize two datasets: the CESNET network traffic dataset from Zenodo (**?**) and a synthetic dataset designed to mimic realistic time series dynamics. The CESNET dataset, comprising hourly aggregated network statistics, includes d features such as the number of flows (n_{flows}), packets (n_{packets}), bytes (n_{bytes}), destination IPs ($n_{\text{dest\_ip}}$), ASNs ($n_{\text{dest\_asn}}$), ports ($n_{\text{dest\_port}}$), TCP/UDP ratios for packets and bytes ($r_{\text{tcp\_udp\_packets}}, r_{\text{tcp\_udp\_bytes}}$), direction ratios ($r_{\text{dir\_packets}}, r_{\text{dir\_bytes}}$), average flow duration (d_{avg}), and time-to-live (ttl_{avg}).

We select $N_{\text{series}} = 6$ time series, each with length

$$T \geq T_{\text{pre}} + T_{\text{post}} + T_{\text{tail}} + 20 = 1184,$$

where $T_{\text{pre}} = 672$ (approximately 28 days), $T_{\text{post}} = 192$ (8 days), and $T_{\text{tail}} = 300$ (pre-change context).

The synthetic dataset generates $N_{\text{synth}} = 3$ series, each with $T = 4800$ samples and $d = 10$ features, defined as:

$$y_j(t) = \begin{cases} \log(1 + e^{2(b(t)+\epsilon_j(t))}), & j < d-2, \\ 0.5 + 0.2\tanh(b(t)) + \epsilon_j(t), & j \geq d-2, \end{cases} \quad (1)$$

where $b(t) = 0.5\sin\left(\frac{2\pi t}{24} + \phi_1\right) + 0.3\sin\left(\frac{2\pi t}{168} + \phi_2\right) + 0.0003(t - T/2)$, $\phi_1, \phi_2 \sim \text{Unif}(0, 2\pi)$, $\epsilon_j(t) \sim \mathcal{N}(0, \sigma_j^2)$, with $\sigma_j = 0.15$ for $j < d-2$ and $\sigma_j = 0.02$ for $j \geq d-2$. The series are clipped to ensure $y_j(t) \geq 0$ or $y_j(t) \in [0, 1$ $forratio - likefeatures.$

Raw data $X \in \mathbb{R}^{T \times d}$ is transformed into a feature matrix F:

$$F_{t,j} = \begin{cases} X_{t,j}, & \text{if } j \text{ is a ratio feature,} \\ \log(1 + \max(0, X_{t,j})), & \text{otherwise,} \end{cases} \quad (2)$$

handling non-negativity and skewness. Missing values are imputed: ratios are set to 0.5, averages are interpolated, and others are filled with 0.0. The features are standardized using the pre-change segment ($t = 1, \ldots, T_{\text{pre}}$):

$$Z_{t,j} = \frac{F_{t,j} - \mu_j}{\sigma_j + \epsilon}, \quad \mu_j = \frac{1}{T_{\text{pre}}}\sum_{t=1}^{T_{\text{pre}}} F_{t,j}, \quad \sigma_j = \sqrt{\frac{1}{T_{\text{pre}}}\sum_{t=1}^{T_{\text{pre}}}(F_{t,j} - \mu_j)^2}, \quad (3)$$

where $\epsilon = 10^{-6}$ prevents division by zero. The standardized data Z is whitened using Oracle Approximating Shrinkage (OAS) (**?**) covariance estimation:

$$X = (Z - \mu_W)W, \quad W = V\text{diag}(\lambda_i^{-1/2}), \quad \lambda_i \geq \epsilon_{\text{var}} = 10^{-6}, \quad (4)$$

where $\mu_W = \frac{1}{T_{\text{pre}}}\sum_{t=1}^{T_{\text{pre}}} Z_t$, and V, λ_i are the eigenvectors and eigenvalues of the OAS covariance matrix $\hat{\Sigma}_{\text{pre}}$.

1.1.2 Change Injection

To isolate higher-order statistical changes, we apply a transformation, `shape_kurtosis_dep`, to the standardized post-change data $Z_{\text{post}} \in \mathbb{R}^{T_{\text{post}} \times d}$.

1. Heavy-Tail Warping : For $k = \max(3, \lfloor d/3 \rfloor)$ randomly selected dimensions $J \subset \{1, \ldots, d\}$, apply:

$$Z_{t,j} \leftarrow \sinh(\alpha Z_{t,j}), \quad \alpha = 0.9, \quad j \in J. \quad (5)$$

2. Nonlinear Cross Terms : For dimension pairs $(j, j+1)$, introduce dependence:

$$Z_{t,j} \leftarrow Z_{t,j} + \beta_1 Z_{t,j} Z_{t,j+1}, \quad Z_{t,j+1} \leftarrow Z_{t,j+1} + \beta_2(Z_{t,j}^2 - 0.5), \quad (6)$$

with $\beta_1 = 0.15$, $\beta_2 = 0.10$.

3. Re-standardization : Normalize to zero mean and unit variance:

$$Z_{t,j} \leftarrow \frac{Z_{t,j} - \bar{Z}_j}{\sigma_{Z_j} + \epsilon}, \quad \bar{Z}_j = \frac{1}{T_{\text{post}}}\sum_{t=1}^{T_{\text{post}}} Z_{t,j}, \quad \sigma_{Z_j} = \sqrt{\frac{1}{T_{\text{post}}}\sum_{t=1}^{T_{\text{post}}}(Z_{t,j} - \bar{Z}_j) \quad (7)$$

4. Covariance Reset : Ensure $\mathrm{Cov}(Z) \approx I$ using:

$$Z \leftarrow ZA, \quad A = V\mathrm{diag}(\lambda_i^{-1/2}), \quad \hat{\Sigma}_Z = V\mathrm{diag}(\lambda_i)V^\top, \tag{8}$$

where $\hat{\Sigma}_Z$ is the sample covariance of Z.

5. Skew Warping : Introduce slight skew and re-standardize:

$$Z \leftarrow Z + \gamma Z^3, \quad \gamma = 0.05, \quad Z_{t,j} \leftarrow \frac{Z_{t,j} - \bar{Z}_j}{\sigma_{Z_j} + \epsilon}. \tag{9}$$

The resulting Z_{inj} is whitened using Equation (4) to form X_{post}, ensuring mean and covariance alignment with X_{pre}.

1.1.3 Change Detection Algorithms

Three detectors are implemented within a CUSUM framework:

$$Z_t = \max(0, Z_{t-1} + \Delta_t), \quad \text{alarm if } Z_t \geq \tau, \tag{10}$$

where Δ_t is the increment, and τ is the threshold calibrated for target ARLs.

1.1.4 Proposed DeepLLR-CUSUM

The DeepLLR-CUSUM leverages a discriminative Multi-Layer Perceptron (MLP) to estimate the log-likelihood ratio $s(x) \approx \log \frac{p_1(x)}{p_0(x)}$, where p_0 and p_1 are the pre- and post-change distributions of whitened data X. The MLP, with one hidden layer of size $h = 64$, is defined as:

$$h(x) = \sigma(W_1 x + b_1), \quad \hat{p}(y|x) = \mathrm{softmax}(W_2 h(x) + b_2), \tag{11}$$

where $W_1 \in \mathbb{R}^{h \times d}$, $b_1 \in \mathbb{R}^h$, $W_2 \in \mathbb{R}^{2 \times h}$, $b_2 \in \mathbb{R}^2$, and σ is the ReLU activation. The log-likelihood ratio is:

$$s(x) = \log \frac{\hat{p}(y=1|x)}{\hat{p}(y=0|x)} = [W_2 h(x) + b_2]_1 - [W_2 h(x) + b_2]_0, \tag{12}$$

clipped to $\hat{p}(y|x) \in [\epsilon_p, 1-\epsilon_p]$, $\epsilon_p = 10^{-6}$. The MLP is trained on X_{pre} (label $y = 0$) and X_{post} (label $y = 1$) with a validation split $f_{\text{val}} = 0.15$, using Adam optimization (**?**) with learning rate $\eta = 10^{-3}$, batch size $B = 256$, L2 regularization $\alpha = 10^{-4}$, and $E = 15$ epochs.

The CUSUM increment is:

$$\Delta_t = \lambda_d(s(X_t) - \mathbb{E}_{X \sim p_0}[s(X)]), \tag{13}$$

where $\mathbb{E}_{X\sim p_0}[s(X)] \approx \frac{1}{T_{\text{pre}}}\sum_{t=1}^{T_{\text{pre}}} s(X_t)$, and λ_d is computed via moment-generating function (MGF) root-finding:

$$\log \mathbb{E}_{X \sim p_0}[e^{\lambda D}] = 0, \quad D = s(X) - \mathbb{E}_{X \sim p_0}[s(X)], \tag{14}$$

solved via bisection over $\lambda \in [10^{-7}, 2/\sigma_D]$, where σ_D is the standard deviation of D. The algorithm for DeepLLR-CUSUM is:

Algorithm 1 DeepLLR–CUSUM Detection

Require: Pre-change data $X_{\text{pre}} \in \mathbb{R}^{T_{\text{pre}} \times d}$, post-change data $X_{\text{post}} \in \mathbb{R}^{T_{\text{post}} \times d}$, stream data $X_{\text{stream}} \in \mathbb{R}^{T_{\text{stream}} \times d}$, target ARL ARL_{target}, trials $N_{\text{trials}} = 200$, horizon $H = 3600$, bootstrap reps $R_{\text{ci}} = 60$, seed s.

Ensure: Detection delay δ, censored flag c, threshold τ_d.

Set random seed s for reproducibility.

Train MLP on $(X_{\text{pre}}, y{=}0)$ and $(X_{\text{post}}, y{=}1)$ with $h{=}64$, $E{=}15$, $f_{\text{val}}{=}0.15$, $\eta{=}10^{-3}$, $B{=}256$, $\alpha{=}10^{-4}$.

Compute $s(X) = \log \frac{\hat{p}(y=1|X)}{\hat{p}(y=0|X)}$ for $X \in X_{\text{pre}}$.

Estimate block length

$$L = \max\Big(10,\ \min\big(80,\ T_{\text{pre}}/20,\ \min\{\, i : |\mathrm{ACF}(s(X), i)| < 0.2 \,\}\big)\Big).$$

Construct blocks $B \in \mathbb{R}^{N_b \times L}$ from $s(X)$.

Compute λ_d by solving $\log \mathbb{E}[\exp(\lambda_d\, s(X))] = 0$ (bisection).

Calibrate τ_d via block bootstrap to reach ARL_{target} with N_{trials}, H, R_{ci}.

Compute increments

$$\Delta_t = \lambda_d \left(s(X_t) - \frac{1}{T_{\text{pre}}} \sum_{u=1}^{T_{\text{pre}}} s(X_u) \right).$$

Initialize $Z_0 \leftarrow 0$, $t_0 \leftarrow T_{\text{tail}}$.

for $t = t_0, \ldots, T_{\text{stream}}$ **do**

  $Z_t \leftarrow \max\big(0,\ Z_{t-1} + \Delta_t\big)$

  if $(t - t_0) \geq \delta_{\min} = 1$ **and** $Z_t \geq \tau_d$ **then**

    return $\delta = t - t_0$, $c =$ **false**, τ_d

return $\delta = T_{\text{post}}$, $c =$ **true**, τ_d

1.1.5 Gaussian-SCUSUM

The Gaussian-SCUSUM assumes $X \sim \mathcal{N}(\mu, \Sigma)$. For pre- and post-change data, estimate μ_0, Σ_0 and μ_1, Σ_1 using OAS. The score is:

$$s_k(x) = \frac{1}{2}(x - \mu_k)^\top (\Sigma_k + \epsilon_j I)^{-2} (x - \mu_k) - \mathrm{tr}((\Sigma_k + \epsilon_j I)^{-1}), \quad k \in \{0, 1\}, \tag{15}$$

with $\epsilon_j = 10^{-3}$. The increment is:

$$\Delta_t = \lambda_g(s_0(X_t) - s_1(X_t)), \quad (16)$$

where λ_g is computed via MGF as above.

1.1.6 LSTM-CUSUM

The LSTM-CUSUM models the first whitened dimension $X_{t,1}$ with an LSTM (?) of hidden size $h_l = 32$ and sequence length $L_s = 48$. Training uses mean squared error loss over $E_l = 10$ epochs. The residual is:

$$r_t = X_{t,1} - \hat{X}_{t,1}, \quad \hat{X}_{t,1} = \text{LSTM}(X_{t-L_s:t,1}), \quad (17)$$

and the increment is:

$$\Delta_t = \lambda_l(r_t - \text{median}(r_{\text{pre}}[:\max(50, T_{\text{pre}}/4)])). \quad (18)$$

1.1.7 ARL Calibration

Thresholds τ are calibrated to achieve ARLs $ARL_{\text{target}} \in \{200, 400\}$ using block-bootstrap (?) with $N_{\text{trials}} = 200$, horizon $H = 3600$, and $R_{\text{ci}} = 60$ replicates. The block length is:

$$L = \max(10, \min(80, T_{\text{pre}}/20, \min\{i : |\text{ACF}(D, i)| < \theta = 0.2\})), \quad (19)$$

where D is the increment sequence. The ARL is estimated as:

$$\text{ARL} = \frac{1}{N_{\text{trials}}} \sum_{i=1}^{N_{\text{trials}}} \min\{t : Z_t^{(i)} \geq \tau \text{ or } t = H\}, \quad (20)$$

with τ found via bisection to match ARL_{target}.

1.1.8 Evaluation Metrics

Performance is assessed using:

1. RMST : From Kaplan-Meier survival function (?):

$$S(t) = \prod_{u=1}^{t} \left(1 - \frac{d_u}{\max(1, n_u)}\right), \quad \text{RMST} = \sum_{t=0}^{H-1} S(t), \quad (21)$$

where d_u is the number of detections at t, and n_u is the number at risk.

2. Median Detected Delay : Median of δ for non-censored runs, with 95% bootstrap CIs ($R_{\text{boot}} = 800$).

3. Censor Rate : Proportion of runs with no detection within $H = 192$.

4. Pairwise Comparisons : Wins, ties, and losses of DeepLLR-CUSUM versus baselines.

1.1.9 Rationale for Methodology Selection

The DeepLLR-CUSUM is chosen for its ability to capture higher-order statistical changes via discriminative learning, unlike Gaussian-SCUSUM, which assumes normality and struggles with non-Gaussian structures (?). The LSTM-CUSUM, while modeling temporal dependencies, is limited to univariate analysis, potentially missing multivariate interactions. The block-bootstrap calibration ensures robust ARL matching (?), and the CESNET dataset provides real-world complexity (?). Synthetic data allows controlled evaluation of shape changes. The use of RMST and Kaplan-Meier curves aligns with survival analysis standards for censored data (?), ensuring comprehensive performance assessment.

1.1.10 Parameter and Hyperparameter Settings

- **Dataset Configuration**: The CESNET dataset uses $N_{\text{series}} = 6$ series with $T_{\text{pre}} = 672$, $T_{\text{post}} = 192$, and $T_{\text{tail}} = 300$, reflecting realistic network traffic durations. Synthetic data has $N_{\text{synth}} = 3$, $T = 4800$, $d = 10$, balancing computational feasibility and complexity. Features include 12 CESNET metrics, capturing diverse network behaviors.

- **DeepLLR-CUSUM Parameters**: The MLP uses $h = 64$ hidden units, $E = 15$ epochs, $f_{\text{val}} = 0.15$, $\eta = 10^{-3}$, $B = 256$, and $\alpha = 10^{-4}$, optimized for discriminative power and stability in high-dimensional settings.

- **LSTM-CUSUM Parameters**: The LSTM employs $h_l = 32$, $L_s = 48$, and $E_l = 10$, suitable for capturing temporal patterns in univariate series while maintaining computational efficiency.

- **Calibration and Evaluation**: ARL targets are $ARL_{\text{target}} = \{200, 400\}$, with $N_{\text{trials}} = 200$, $H = 3600$, $R_{\text{ci}} = 60$, and $R_{\text{boot}} = 800$. The minimum delay $\delta_{\text{min}} = 1$ prevents zero-inflation, and seeds $s \in \{23, 47, 131\}$ ensure reproducibility.

1.2 Results and Discussion

This supplement expands the quantitative evidence for the dominance of *DeepLLR–CUSUM* over *Gaussian–CUSUM* and the univariate *LSTM–CUSUM*, preserving all reported figures and tables while providing additional interpretation focused on delay, coverage, and false–alarm behavior.

1.2.1 Complete Detection Diagnostics

RMST (samples). For ARL=200, RMSTs (mean [95% CI]) are: DeepLLR 1.2019 [1.0432, 1.4446], Gaussian 1.3287 [1.1286, 1.5783], LSTM 28.5332 [19.9778, 37.6654]. DeepLLR reduces RMST by 9.9% vs. Gaussian and 95.8% vs. LSTM (absolute gaps: 0.1268 and 27.3313 samples, respectively). For ARL=400, RMSTs are: DeepLLR 1.2430 [1.0440, 1.4986], Gaussian 1.5385 [1.2471, 1.8797], LSTM 54.3296 [35.0089, 75.3630]; reductions are 19.5% and 97.7% (absolute gaps: 0.2955 and 53.0866 samples) (Fig. **??**a; survival in Fig. **??**). These RMST improvements mirror the EDD advantages summarized in the main text and quantify the *overall* time-to-alarm burden, integrating early and late detections.

Coverage (fraction detected). At ARL=200/400, coverage is: DeepLLR 1.000/1.000; Gaussian 1.000/1.000; LSTM 0.833/0.775. Thus DeepLLR (and Gaussian) achieve full coverage at both targets, whereas LSTM misses 16.7% and 22.5% of windows (consistent with heavy survival tails in Fig. **??**). Full coverage paired with the lowest RMST indicates that DeepLLR concentrates detection mass at the earliest times *and* avoids horizon misses.

1.2.2 Empirical ARL (No-Change) with Confidence Intervals

Realized ARLs (mean [95% CI]) under no-change are: *Target 200*— DeepLLR 706.33 [677.09, 744.07] (**3.53**$\times$ nominal), Gaussian 245.52 [231.38, 257.13] (**1.23**$\times$), LSTM 385.14 [353.11, 415.58] (**1.93**$\times$). *Target 400*— DeepLLR 867.01 [814.72, 919.60] (**2.17**$\times$), Gaussian 370.49 [347.54, 395.65] (**0.93**$\times$, undershoot), LSTM 460.28 [415.61, 509.02] (**1.15**$\times$). Grouped CIs appear in Fig. **??**d. Interpretation: DeepLLR is *conservative* (higher-than-nominal ARL, hence fewer false alarms) *while* being the fastest on delay—i.e., superior on the delay/false-alarm Pareto front. Gaussian's undershoot at ARL=400 indicates elevated false-alarm risk at that target.

1.2.3 Additional Tables

Non-core metrics referenced in the main text are compiled below for completeness.

Table 1. RMST (samples) and coverage (fraction detected).

Method	ARL=200		ARL=400	
	RMST	Coverage	RMST	Coverage
DeepLLR–CUSUM	1.2019 [1.0432, 1.4446]	1.000	1.2430 [1.0440, 1.4986]	1.000
Gaussian–CUSUM	1.3287 [1.1286, 1.5783]	1.000	1.5385 [1.2471, 1.8797]	1.000
LSTM–CUSUM	28.5332 [19.9778, 37.6654]	0.833	54.3296 [35.0089, 75.3630]	0.775

Table 2. Empirical ARL under no-change (mean [95% CI]).

Method	Actual ARL @200	Actual ARL @400
DeepLLR–CUSUM	706.33 [677.09, 744.07]	867.01 [814.72, 919.60]
Gaussian–CUSUM	245.52 [231.38, 257.13]	370.49 [347.54, 395.65]
LSTM–CUSUM	385.14 [353.11, 415.58]	460.28 [415.61, 509.02]

1.2.4 Narrative Cross-Checks and Mechanistic Consistency

The DeepLLR–Gaussian delay histogram in Fig. **??**c concentrates at ≤ 0, aligning with the 66.7% non-tie win rate reported in the core table (DeepLLR 16/8/24 vs. Gaussian). Against LSTM, DeepLLR never loses (48/0/0), consistent with LSTM's reduced coverage and heavy-tailed survival. Trace exemplars in Fig. **??** visualize the mechanism: DeepLLR's statistic typically crosses within $\approx$1–2 samples, Gaussian follows within a few samples, and LSTM residuals lag. Together with the conservative realized ARL (Sec. 1.2.2), these diagnostics substantiate that DeepLLR achieves earlier and more reliable alarms *without* inflating false alarms, thereby outperforming both baselines across targets.]