FORGET BEFORE YOU CONNECT: PRE-GRAPH SUBSPACE PROJECTION FOR CHANNEL-LEVEL UNLEARNING IN MULTIVARIATE TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Machine unlearning has largely focused on removing training samples or classes, leaving one question underexplored: *How can a deployed model reduce its dependence on an input channel?* In multivariate time-series (MTS) systems, corrupted or privacy-sensitive sensors may need to be removed after deployment. Existing approaches require costly retraining or iterative optimization, while permanent input masking can degrade predictive utility without addressing learned cross-channel dependencies. We introduce *Channel-Centric Subspace Projection* (CCSP), a training-free method that identifies target-associated latent directions via singular value decomposition and attenuates them before cross-channel graph propagation. An optional trainable utility-repair extension is evaluated separately. Across three MTS classification benchmarks, CCSP preserves near-original utility with low residual sensitivity, achieving $2.56\sim48.80\times$ speedup over iterative baselines. On UCI-HAR, it achieves a 40.98% mean sensor-wise reduction in linear intervention leakage and reduces mean prediction disagreement from 9.54% to 8.12%. On C-MAPSS remaining-useful-life regression, CCSP further reduces target-induced prediction variation by 31.41% on average across 14 sensors while maintaining comparable predictive utility. These results establish latent subspace intervention as an efficient approach to channel-level forgetting, while distinguishing approximate influence removal from certified information erasure.

1 INTRODUCTION

Sensor configurations change after deployment, but trained multivariate time-series (MTS) models retain dependencies learned from their original inputs. A sensor may become unreliable, privacy-sensitive, or subject to a removal request, necessitating suppression of its influence while preserving predictive utility. This problem is especially consequential in healthcare monitoring, cyber-physical systems, and industrial sensing, where predictions depend on the interactions of sensor streams (Tonekaboni et al., 2020). Rebuilding a model for every removal request is costly, motivating efficient post-training control over individual channels.

Machine unlearning provides a framework for removing undesired influence from trained models (Bourtoule et al., 2021; Cao & Yang, 2015). Most existing methods, however, focus on removing training samples, classes, or domains, primarily in vision and language settings (Ginart et al., 2019; Golatkar et al., 2020; Ilharco et al., 2022). Channel-level unlearning presents a different challenge: the removal target is a persistent feature stream shared across examples, whose information can spread through learned cross-channel interactions. In graph-based MTS models, message passing distributes target-sensor information to other sensor representations (Wu et al., 2020). Once this mixing occurs, modifying the target representation alone may leave its downstream effects intact. Structured channel-level unlearning, therefore, remains an *important, comparatively underexplored* problem (Nguyen et al., 2025).

Existing approaches offer incomplete solutions. Retraining and fine-tuning require repeated optimization (Sekhari et al., 2021; Tarun et al., 2023). Permanent input masking blocks incoming target measurements but can degrade the utility of a model trained on that channel. Parameter-space interventions can alter representations shared by retained sensors (Golatkar et al., 2020; Sekhari et al.,

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.

2021; Tarun et al., 2023). Representation-level methods such as LEACE (Belrose et al., 2023) and INLP (Ravfogel et al., 2020) provide relevant concept-erasure mechanisms, but their application to graph-based sensor removal raises a central design question: *where should an intervention act to suppress target influence while limiting disruption to useful representations?*

We address this question through *approximate channel-level unlearning*: reducing a trained model's functional dependence on a requested sensor while preserving predictive utility. We propose *Channel-Centric Subspace Projection* (CCSP), a training-free intervention applied before graph propagation. CCSP uses singular value decomposition to identify dominant directions in the target sensor's pre-graph activations and attenuates them within that sensor's latent slice. Retained-channel representations remain unchanged at the intervention point, and subsequent message passing operates on the edited representation. Projection rank and strength control the suppression–utility trade-off without updating pretrained parameters. We evaluate optional, trainable utility repair separately from this core operation.

Evaluating this objective requires more than an aggregate accuracy difference. We therefore combine residual influence and predictive utility with representation-level intervention detectability and Prediction Disagreement Rate (PDR) under target-only real-trajectory replacement. These complementary measurements assess whether a target intervention affects retained representations and individual predictions, including changes that aggregate accuracy can conceal. Comparisons with zeroed-input retraining, fine-tuning, LEACE (Belrose et al., 2023), INLP (Ravfogel et al., 2020), and other baselines examine the resulting forgetting–utility–efficiency trade-off.

We study selective post-training suppression of channel dependence, evaluated using downstream intervention responses rather than by deleting all target-associated information. We do not claim equivalence to retraining.

Our contributions are summarized as follows:

- We formulate approximate channel-level unlearning for MTS models and evaluate it through complementary task-, representation-, and prediction-level dependence measurements.
- We propose CCSP, a training-free, sensor-specific subspace intervention that attenuates target-associated directions before cross-channel message passing while keeping pretrained model parameters fixed.
- Across three MTS benchmarks, CCSP achieves efficient channel-level unlearning while preserving near-original utility and low residual target dependence, with $10.33 \sim 36.88\times$, $2.56 \sim 11.48\times$, and $6.48 \sim 48.80\times$ speedups on UCI-HAR, NATOPS, and FaceDetection, respectively. Beyond accuracy-based diagnostics, on UCI-HAR, it reduces linear intervention leakage by 40.98% and mean prediction disagreement from 9.54% to 8.12%; a five-permutation analysis further shows selective suppression of the target channel while largely preserving sensitivity to retained channels.
- We extend the evaluation to simultaneous multi-channel intervention and continuous-valued remaining useful life (RUL) regression. On C-MAPSS FD001 (Saxena et al., 2008), CCSP achieves a mean sensor-wise reduction of 31.41% in target-induced prediction variation across 14 evaluated sensors while maintaining comparable predictive utility.

2 Background

CCSP is motivated by the intersection of machine unlearning, graph-based multivariate time-series modeling, and representation-level intervention. Existing work has made progress in each area, but it does *not* directly address the post-hoc removal of a specific sensor's influence from an already trained graph-based MTS model.

2.1 Related Work

Machine unlearning seeks to remove specified training influence while avoiding full retraining (Bourtoule et al., 2021; Cao & Yang, 2015). Prior work has primarily addressed sample-, class-, or domain-level removal through exact or approximate procedures (Baumhauer et al., 2022; Guo et al., 2019; Neel et al., 2021). Sensor-level unlearning changes the removal unit: a sensor channel is a persistent feature stream shared across instances, whose influence can spread through

cross-channel interactions. We study approximate functional dependence reduction in this setting without claiming certified removal or equivalence to retraining.

Deep MTS models learn temporal patterns and cross-channel dependencies (Lim et al., 2021; Liu et al., 2022; Wu et al., 2022). Graph-based architectures, including MTGNN and RAINDROP, explicitly model sensor interactions for forecasting and irregularly sampled data (Wu et al., 2020; Zhang et al., 2021). These interactions improve prediction but also distribute target-channel information across representations. Their predictive objectives do not explicitly support post-deployment sensor removal, motivating interventions before cross-channel mixing.

Representation editing provides the closest methodological context. Model editing modifies localized computations (Meng et al., 2022), LoRA learns low-rank parameter updates (Hu et al., 2021), and pruning identifies sparse subnetworks (Frankle & Carbin, 2018). More directly, LEACE provides closed-form linear concept erasure (Belrose et al., 2023), while INLP iteratively removes directions identified by linear concept classifiers (Ravfogel et al., 2020). CCSP estimates dominant directions directly from target-channel activations and selectively attenuates them within that channel's pre-graph latent slice. This preserves retained-channel activations at the intervention point while reducing downstream target sensitivity. We compare LEACE and INLP at the same intervention point to assess whether CCSP's subspace construction and tunable attenuation offer a favorable dependence–utility trade-off.

2.2 Research Motivation

Post-deployment sensor removal requires suppressing a channel's influence while preserving utility from retained sensors. Unlike sample-level deletion (Xu et al., 2024), this request concerns a feature stream whose information can propagate across graph-connected representations (Jin et al., 2024; Wang et al., 2024). Permanent input masking blocks target measurements but may degrade utility, while retraining and parameter-efficient adaptation incur optimization costs (Houlsby et al., 2019; Hu et al., 2021; Poth et al., 2023). Parameter-space interventions such as SSD (Foster et al., 2024) modify weights that may support multiple channels, complicating selective suppression.

These trade-offs motivate intervention at the point where sensor representations remain separated, before graph propagation distributes their information. CCSP attenuates dominant target-associated directions at this point, pursuing three goals: *structural decoupling*, by intervening before inter-sensor interaction; *sensor-specific controllability*, through channel-specific projection rank and attenuation strength; and *efficiency*, by avoiding full retraining or iterative parameter optimization. Unlike imputation methods such as MTSCI (Zhou et al., 2024), which reconstruct missing values, CCSP aims to achieve a controllable reduction in downstream target sensitivity while preserving predictive utility.

3 Problem Formulation

We consider MTS inputs $X \in \mathbb{R}^{T\times S\times D}$, where T, S, and D denote the number of time steps, sensor channels, and features per sensor, respectively. A trained predictor maps $f_\theta : \mathbb{R}^{T\times S\times D} \to \mathcal{Y}$, where $\mathcal{Y} = \mathbb{R}^C$ for C classes or $\mathcal{Y} = \mathbb{R}$ for continuous-valued predictions, such as RUL.

Given a forget request for sensor $s_k \in \{1, \ldots, S\}$, we seek an updated predictor f' that reduces dependence on s_k while preserving downstream predictive utility. The retained channels are $\mathcal{S}_{\text{retain}} = \{1, \ldots, S\} \setminus \{s_k\}$. Unlike sample-level deletion, the target is a feature stream shared across instances, whose influence can spread through temporal and cross-channel interactions. Here, f' denotes a modified predictor and need not involve changes to θ.

Operational scope. CCSP retains the target input slot, processes that trajectory through the encoder, and then attenuates its latent representation before graph propagation. Our objective is, therefore, to approximate post-training dependence reduction; operation under physical sensor absence is outside this formulation.

Let $\mathcal{I}k$ denote a specified intervention that modifies only the target trajectory while leaving retained inputs unchanged. We formulate the objective as

$$\min f' \text{Dependence}_{\mathcal{I}_k}(f', s_k) \quad \text{s.t.} \quad \text{Utility}(f') \geq \tau, \tag{1}$$

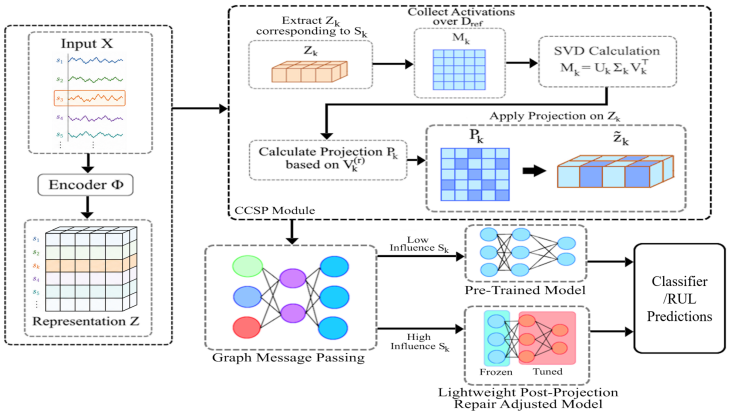

Figure 1: Overview of Channel-Centric Subspace Projection (CCSP).

where dependence measures residual sensitivity to $\mathcal{I}_k$, and τ specifies the minimum acceptable utility. Utility is task-specific and oriented so that larger values indicate better performance, such as accuracy or negative regression error. As a reference, we define zeroed-input retraining as

$$f_{\text{retrain}}^{(-k)} = \mathcal{A}\left(\mathcal{D}_{\text{train}}^{(-k)}\right), \tag{2}$$

where $\mathcal{A}$ is the training procedure and $\mathcal{D}_{\text{train}}^{(-k)}$ replaces target trajectories with zeros while preserving the input topology. Training under this convention alone does not guarantee invariance to arbitrary nonzero target trajectories.

Section 5.1 assesses dependence through masking-based functional sensitivity, real-trajectory intervention detectability, and prediction disagreement, alongside predictive utility. These complementary measurements characterize intervention-specific behavior without certifying statistical independence or information erasure.

4 Methodology

In this section, we introduce CCSP, a post-training intervention for channel-level unlearning. Building on the formulation in Section 3, CCSP reduces the influence of a requested target sensor s_k by suppressing target-associated directions in the pre-graph subspace, before graph message passing can propagate this information to retained sensors.

Figure 1 summarizes CCSP. Given a trained model and target sensor s_k, CCSP extracts sensor-separated pre-graph representations, estimates a target-associated subspace using SVD, and attenuates the target-associated subspace before passing the representation through the frozen graph module and classifier. Projection-only CCSP is training-free; LPPR is evaluated separately as an optional trainable extension for utility repair.

We decompose the trained classifier as follows:

$$f_\theta(X) = \Psi_{\theta_g}(\Phi_{\theta_e}(X)), \tag{3}$$

where Φ_{θ_e} maps the input to pre-graph subspace representations $Z \in \mathbb{R}^{T\times S\times d}$, and Ψ_{θ_g} denotes the graph propagation and classification components. CCSP operates on Z because sensor representations remain separated before graph propagation, enabling the target sensor to be modified before cross-channel mixing.

For a target sensor s_k, we extract the latent slice $Z_k = Z[:, k, :] \in \mathbb{R}^{T\times d}$. Over a reference set $\mathcal{D}_{\text{ref}}$, these activations are collected to form the target-sensor activation matrix $M_k \in \mathbb{R}^{N_k\times d}$, where N_k is the total number of collected target-sensor latent vectors. We compute

$$M_k = U_k \Sigma_k V_k^\top, \tag{4}$$

Algorithm 1 Channel-Centric Subspace Projection (CCSP)

Require: Trained model $f_\theta = \Psi_{\theta_g} \circ \Phi_{\theta_e}$; reference set $\mathcal{D}_{\text{ref}}$; validation set $\mathcal{D}_{\text{val}}$; target sensor s_k; candidate rank set $\mathcal{R}$; projection strengths Λ; utility tolerance ϵ_{util}.

Ensure: Unlearned predictor f'_θ; selected rank $r^\star$; selected projection strength $\lambda^\star$; target-sensor subspace $V_k^{(r^\star)}$.

Collect pre-graph activations $Z = \Phi_{\theta_e}(X)$ for all $X \in \mathcal{D}_{\text{ref}}$.
Extract target-sensor activations $M_k = \{Z[:, k, :] \mid X \in \mathcal{D}_{\text{ref}}\}$.
Compute truncated SVD $M_k \approx U_k \Sigma_k V_k^\top$.
for $r \in \mathcal{R}$ **do**
  for $\lambda \in \Lambda$ **do**
    Construct $P_k(r, \lambda) = I_d - \lambda V_k^{(r)} (V_k^{(r)})^\top$.
    Apply $P_k(r, \lambda)$ only to the target sensor:

$$\tilde{z}_{t,k} = z_{t,k} P_k(r, \lambda), \qquad \tilde{z}_{t,s} = z_{t,s},\ \forall s \neq k.$$

    Evaluate validation utility $A_{\text{val}}^{\text{unl}}(r, \lambda)$.
    Evaluate residual influence $|G_{k,\text{val}}^{\text{unl}}(r, \lambda)|$.
  end for
end for
Select $(r^\star, \lambda^\star)$ such that

$$(r^\star, \lambda^\star) = \arg \min_{r \in \mathcal{R},\, \lambda \in \Lambda} |G_{k,\text{val}}^{\text{unl}}(r, \lambda)|$$

subject to $\Delta_{\text{util}}^{\text{val}}(r, \lambda) \leq \epsilon_{\text{util}}$.
if no candidate satisfies $\Delta_{\text{util}}^{\text{val}} \leq \epsilon_{\text{util}}$ **then**
  Select $(r^\star, \lambda^\star)$ by minimizing

$$|G_{k,\text{val}}^{\text{unl}}(r, \lambda)| + \max\left\{0,\, \Delta_{\text{util}}^{\text{val}}(r, \lambda) - \epsilon_{\text{util}}\right\}.$$

end if
Construct $P_k^\star = P_k(r^\star, \lambda^\star)$.
Define $f'_\theta(X) = \Psi_{\theta_g}(\Pi_{k, r^\star, \lambda^\star}(\Phi_{\theta_e}(X)))$.
return f'_θ, $r^\star$, $\lambda^\star$, $V_k^{(r^\star)}$.

where the top-r right singular vectors $V_k^{(r)} \in \mathbb{R}^{d \times r}$ span the dominant target-associated subspace.

CCSP suppresses this subspace using

$$P_k = I_d - \lambda V_k^{(r)} (V_k^{(r)})^\top, \tag{5}$$

where r controls the projection rank and $\lambda \in [0, 1]$ controls the projection strength. Since $V_k^{(r)} (V_k^{(r)})^\top$ is an orthogonal projector, P_k attenuates the selected directions by a factor of $1 - \lambda$; for $\lambda < 1$, the transformation is therefore attenuative rather than an irreversible erasure. During inference, the projection is applied only to the target-sensor representation:

$$\tilde{z}_{t,k} = z_{t,k} P_k, \qquad \tilde{z}_{t,s} = z_{t,s}, \quad \forall s \neq k. \tag{6}$$

This yields the projected latent representation $\widetilde{Z} = \Pi_{k,r,\lambda}(Z)$ and the modified predictor

$$f_{\theta'}(X) = \Psi_{\theta_g}(\widetilde{Z}). \tag{7}$$

Thus, CCSP directly modifies only the target-sensor latent slice before graph propagation; retained sensor slices remain unchanged at the point of intervention.

Algorithm 1 summarizes projection-only CCSP: estimate a target-associated subspace from pre-graph activations via SVD, select projection rank and strength on validation data, and apply the projection at inference while keeping all pretrained parameters frozen.

The projection attenuates latent energy along the selected target-associated directions while preserving the orthogonal complement. Because retained slices are unchanged, the total pre-graph perturbation

is confined to the target slice, $\|\tilde{Z} - Z\|_F = \lambda \|Z_k V_k^{(r)} (V_k^{(r)})^\top\|_F$. If the downstream module Ψ_{θ_g} is L-Lipschitz with respect to the Frobenius norm, then

$$\|f_\theta(X) - f'_\theta(X)\|_2 \;\le\; L\lambda \left\|Z_k V_k^{(r)} (V_k^{(r)})^\top\right\|_F, \tag{8}$$

so the change in the model output is controlled by the projection strength λ and by the target-slice energy lying in the selected rank-r subspace (Appendix A.3).

Projection-only CCSP is training-free: it estimates and attenuates a target-associated pre-graph subspace while keeping all pretrained parameters fixed. We separately evaluate CCSP+LPPR, an optional extension that updates a small subset of parameters after projection to recover utility. LPPR is a trainable repair step, and its benefit is not guaranteed. Appendix A.1 describes its optimization and adaptive selection policy.

CCSP thus provides approximate channel-level dependence reduction through a localized latent intervention. Appendices A.2 and A.3 present more detailed computational and theoretical analyses.

5 Experiments

We evaluate CCSP on several MTS benchmarks along three axes: *forgetting effectiveness*, *retained utility*, and *computational efficiency*.

Research questions. Our evaluation addresses the following four questions:

- **RQ1 (Unlearning Feasibility):** How *feasible and scalable* is channel-level unlearning in graph-based MTS models without incurring the costs of full retraining?
- **RQ2 (Intervention Effectiveness):** How *effectively* does Pre-Graph Subspace Projection reduce target-sensor influence while preserving predictive utility?
- **RQ3 (Forgetting–Utility–Efficiency):** How *efficiently* does CCSP balance forgetting, utility, and computational cost relative to retraining and parameter-update baselines?
- **RQ4 (Sensor-wise Unlearning Difficulty)**: How *difficult* is it to suppress individual sensor channels across sensors in terms of residual dependence and utility degradation?

5.1 Environment Configurations

We evaluate CCSP on three MTS classification datasets spanning 9$\sim$144 channels: UCI-HAR ($S = 9$) (Anguita et al., 2013), NATOPS ($S = 24$), and FaceDetection ($S = 144$) (Bagnall et al., 2018). FC-STGNN (Wang et al., 2024) is the primary backbone; additional backbone results appear in Appendix B.6. Task-level baselines comprise zeroed-input retraining and fine-tuning, zero and noise masking, and SalUn (Fan et al., 2023). Representation-level comparisons on HAR also include LEACE (Belrose et al., 2023) and INLP (Ravfogel et al., 2020). Appendix B.7 provides baseline optimization details.

CCSP hyperparameters are selected via validation: rank $r \in \{1, 2, 3, 4\}$ and projection strength $\lambda \in \{0.25, 0.5, 0.75, 1.0\}$. The reference set $\mathcal{D}_{\text{ref}}$ is used only for activation collection. Experiments are conducted in PyTorch on an A100 GPU. An anonymized implementation, configuration files, and experimental scripts are included in the supplementary material.

Evaluation Metrics. We assess forgetting using complementary task-, representation-, and prediction-level measures. Task-level metrics include retained accuracy and utility drop, $\Delta_{\text{util}} = A_{\text{test}}^{\text{orig}} - A_{\text{test}}^{\text{unl}}$, where negative values indicate improvement. Residual sensitivity is $|G_i^{\text{unl}}|$, with $G_i^g = A_{\text{test}}^g - A_{\text{masked},i}^g$ for $g \in \{\text{orig}, \text{unl}\}$. Because zero masking may introduce out-of-distribution inputs, a small influence gap alone does not establish information erasure. Channel UA is a secondary normalized measure; runtime and peak GPU memory quantify efficiency.

For representation-level evaluation on HAR, we replace only the target trajectory with another real trajectory while preserving all retained raw channels. We probe the retained post-graph representations and define intervention leakage as $L_i = |\text{AUC}_i - 0.5|$. We compare Original, zeroed-input Retraining, LEACE, INLP, and CCSP using probe AUC, relative leakage reduction, and, where applicable, predictive utility. Prediction Disagreement Rate (PDR) measures the fraction of predictions that change under the same intervention. These complementary metrics assess residual target sensitivity beyond zero masking without certifying complete erasure.

Table 1: Feasibility of CCSP for channel-level unlearning. Results are averaged over target sensors; accuracy/influence are percentages, and runtime is in seconds.

Dataset	No. Sensors	Original Utility ↑	Retained Utility ↑	$\lvert G_k^{\mathrm{unl}}\rvert \downarrow$	Channel UA	$\mathbf{t}_{\mathrm{run}}$ (s) ↓
HAR	9	94.74 ± 0.00	94.76 ± 0.36	0.19 ± 0.30	99.93 ± 0.15	1.48 ± 0.06
NATOPS	24	82.78 ± 0.00	83.08 ± 0.79	0.19 ± 0.27	78.72 ± 40.64	1.98 ± 0.10
FaceDetection	144	67.34 ± 0.00	67.18 ± 0.20	0.05 ± 0.10	85.93 ± 31.70	12.20 ± 4.42

Table 2: Intervention strategy comparison. Results are averaged over target sensors; accuracy/influence are percentages.

Dataset	Method	Original Utility ↑	Retained Utility ↑	$\lvert \mathbf{G}_{\mathbf{k}}^{\mathrm{unl}}\rvert \downarrow$	Channel UA
HAR	Zeroed-Masked	94.74 ± 0.00	86.33 ± 10.42	0.00 ± 0.00	100.00 ± 0.00
	Noise-Masked	94.74 ± 0.00	83.63 ± 15.48	3.23 ± 5.17	107.03 ± 44.40
	CCSP (ours)	94.74 ± 0.00	$\mathbf{94.76 \pm 0.36}$	$\mathbf{0.19 \pm 0.30}$	99.93 ± 0.15
NATOPS	Zeroed-Masked	82.78 ± 0.00	81.97 ± 1.38	0.00 ± 0.00	100.00 ± 0.00
	Noise-Masked	82.78 ± 0.00	$\mathbf{83.12 \pm 0.82}$	1.34 ± 1.05	9.58 ± 78.71
	CCSP (ours)	82.78 ± 0.00	83.08 ± 0.79	$\mathbf{0.19 \pm 0.27}$	78.72 ± 40.64
FaceDetection	Zeroed-Masked	67.34 ± 0.00	67.06 ± 0.27	0.00 ± 0.00	100.00 ± 0.00
	Noise-Masked	67.34 ± 0.00	67.19 ± 0.16	0.18 ± 0.18	55.83 ± 61.86
	CCSP (ours)	67.34 ± 0.00	$\mathbf{67.18 \pm 0.20}$	$\mathbf{0.05 \pm 0.10}$	85.93 ± 31.70

Table 3: Target-intervention detectability in retained HAR representations under target-only real-trajectory replacement. Lower leakage is better.

Method	Test Acc.	Mean AUC	Leakage	Reduction (%)	Edited Subspace Dimension
Original	94.74	0.5813	0.0813	–	–
Zeroed-input Retraining (ref.)	–	0.5000	0.0000	Ref.	N/A
LEACE	93.73	0.5574	0.0574	28.21	–
INLP-1	94.72	0.5670	0.0670	17.78	8/32
INLP-final (4 iters)	92.37	0.5000	0.0000	100.00	32/32
CCSP (ours)	94.76	0.5480	0.0480	40.98	1–4/32

5.2 Performance Evaluation Results

RQ1: Unlearning Feasibility. Table 1 shows that projection-only CCSP preserves near-original utility while maintaining low zero-mask residual sensitivity across all datasets. Its practical per-sensor runtime, including on FaceDetection with 144 channels, demonstrates the feasibility of training-free intervention without parameter re-optimization. RQ2 further validates this result using real-trajectory evaluations beyond the masking-based diagnostic.

RQ2: Intervention Effectiveness. Table 2 shows that CCSP better balances utility preservation and residual dependence than input-level interventions. Zero masking yields a zero influence gap by construction under the same intervention but can substantially degrade utility, while noise masking produces more variable residual dependence. Intervention location also matters: pre-graph CCSP achieves 94.67% utility with $|G| = 0.10$, versus 93.10% and $|G| = 1.15$ post-graph (Appendix B.5), supporting attenuation before target information propagates through message passing.

Real-trajectory evaluations provide stronger evidence than zero masking. Under target-only replacement, CCSP reduces mean intervention leakage from 0.0813 to 0.0480 while retaining 94.76% accuracy, with a mean sensor-wise reduction of 40.98% compared to 28.21% for LEACE and 17.78% for INLP-1 (Table 3). Full INLP reaches chance-level probe performance only after removing all 32 target-latent dimensions, reducing accuracy to 92.37%. In a five-permutation analysis, CCSP likewise reduces target PDR from 9.60% to 8.08% while preserving retained-channel sensitivity (9.60% vs. 9.87%; see Table 14 in Appendix B.8). These results support selective suppression rather than complete information erasure or equivalence to retraining.

The effect extends beyond single-channel classification: CCSP reduces set-level dependence in 98.6%, 100%, and 100% of two-, three-, and four-channel HAR runs, respectively (Appendix B.9). On C-MAPSS FD001 RUL regression, it reduces target-induced prediction variation across all 14 sensors by 31.41% on average, with comparable RMSE (11.6963 vs. 11.7600; Appendix D).

Table 4: Forgetting–utility–efficiency across target sensors. Accuracy and influence are percentages; peak GPU memory is in MB. Negative utility drop indicates improvement.

Dataset	Method	Retained Utility ↑	$\lvert G_{\mathbf{k}}^{\mathrm{unl}}\rvert$ ↓	Channel UA	Δ_{util} ↓	Peak GPU Mem. (MB) ↓
HAR	Zeroed-input Retraining	93.42 ± 1.20	**0.00 ± 0.00**	**100.00 ± 0.00**	1.32 ± 1.20	202.05 ± 0.00
	Zeroed-input Fine-tuning	94.46 ± 1.19	**0.00 ± 0.00**	**100.00 ± 0.00**	0.28 ± 1.19	202.05 ± 0.00
	SalUn	93.18 ± 3.62	1.87 ± 1.10	55.44 ± 25.82	1.56 ± 3.62	370.37 ± 0.00
	CCSP (ours)	**94.76 ± 0.36**	0.19 ± 0.30	99.93 ± 0.15	**0.07 ± 0.14**	**76.14 ± 0.00**
NATOPS	Zeroed-input Retraining	81.76 ± 2.74	0.67 ± 0.65	39.81 ± 65.20	1.02 ± 2.74	342.81 ± 0.00
	Zeroed-input Fine-tuning	83.01 ± 1.21	1.04 ± 0.82	2.37 ± 72.35	-0.23 ± 1.21	342.81 ± 0.00
	SalUn	**83.63 ± 0.73**	2.11 ± 1.15	-97.57 ± 132.64	**-0.86 ± 0.73**	379.29 ± 0.00
	CCSP (ours)	83.08 ± 0.79	**0.19 ± 0.27**	**78.72 ± 40.64**	-0.30 ± 0.79	**23.38 ± 0.00**
FaceDetection	Zeroed-input Retraining	64.38 ± 1.10	0.19 ± 0.14	-34.30 ± 164.02	2.96 ± 1.10	283.72 ± 0.00
	Zeroed-input Fine-tuning	66.84 ± 0.52	0.22 ± 0.17	-19.95 ± 158.33	0.50 ± 0.52	283.72 ± 0.00
	SalUn	67.06 ± 0.20	0.16 ± 0.14	3.87 ± 165.79	0.28 ± 0.20	536.88 ± 0.00
	CCSP (ours)	**67.18 ± 0.20**	**0.05 ± 0.10**	**85.93 ± 31.70**	**0.16 ± 0.20**	**99.15 ± 0.00**

Table 5: Mean end-to-end runtime (s) across target sensors; lower is better.

Dataset	Fine-tuning	SalUn	Retraining	CCSP (ours)
UCI-HAR	15.31	31.12	54.65	**1.48**
NATOPS	5.06	6.25	22.69	**1.98**
FaceDetection	79.04	89.46	595.46	**12.20**

RQ3: Forgetting–Utility–Efficiency Trade-off. Table 4 shows that CCSP consistently achieves low residual sensitivity, near-original utility, and low overhead across all three datasets without parameter re-optimization. In contrast, zeroed-input retraining, fine-tuning, and SalUn require iterative optimization. Table 5 and Fig. 2 quantify this advantage: CCSP runs in 1.48, 1.98, and 12.20 s on UCI-HAR, NATOPS, and FaceDetection, respectively, yielding 2.56∼48.80× speedups over optimization-based baselines.

RQ4: Sensor-wise Unlearning Difficulty. We characterize sensor-wise trade-offs using residual sensitivity $|G_i^{\mathrm{unl}}|$ and utility drop $\Delta_{\mathrm{util}}(s_i)$, summarized as $\mathrm{FD}(s_i) = |G_i^{\mathrm{unl}}| + \max(0, \Delta_{\mathrm{util}}(s_i))$. High FD, thus, reflects persistent sensitivity, utility loss, or both; utility gains do not offset residual sensitivity. Table 6 reveals high variation across sensors, motivating sensor-specific intervention strengths. FD remains a CCSP-specific summary under zero masking, rather than a general measure of feature importance or representation entanglement.

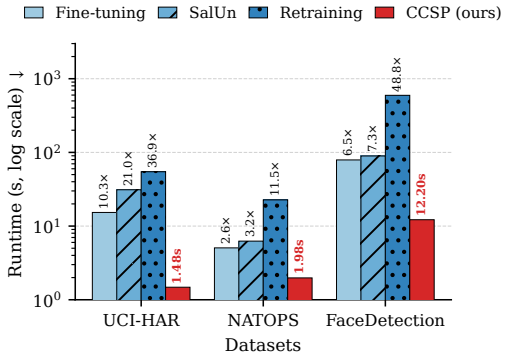

Figure 2: Runtime (log scale; lower is better), averaged across target sensors. Labels indicate CCSP speedups over baselines (2.56∼48.80×).

5.3 Lessons

CCSP shows that targeted pre-graph intervention can reduce sensor dependence while preserving near-original utility without updating pretrained parameters. Real-trajectory, multi-channel, and RUL evaluations broaden this evidence base; Transformer-MTS results remain preliminary (Appendix B.6).

Four lessons emerge: (i) *intervention location matters*: acting before cross-channel mixing limits the spread of target influence; (ii) *accuracy alone is insufficient*: representation-level detectability and prediction disagreement reveal sensitivity hidden by aggregate accuracy; (iii) *suppression must be sensor-specific*: channels exhibit different dependence–utility trade-offs, motivating validation-based intervention strength; and (iv) *functional suppression does not imply equivalence to retraining*: CCSP reduces measured target sensitivity without reproducing retrained representations.

Table 6: Sensor-level forgetting difficulty (FD). We report the top-3 easiest and hardest sensors using $\mathrm{FD}(s_i) = |G_i^{\mathrm{unl}}| + \max(0, \Delta_{\mathrm{util}}(s_i))$; higher FD indicates greater forgetting difficulty.

Dataset	Sensor	Type	Retained Utility ↑	$\lvert\mathbf{G_i^{unl}}\rvert$ ↓	$\boldsymbol{\Delta}_{\mathrm{util}}$ ↓	FD Score ↓	Best Config.
HAR	2	Easiest	94.74	0.00	0.00	0.00	$r = 2, \lambda = 0.25$
	4	Easiest	94.74	0.00	0.00	0.00	$r = 1, \lambda = 0.25$
	6	Easiest	94.77	0.03	-0.03	0.03	$r = 1, \lambda = 0.25$
	0	Hardest	94.40	0.34	0.34	0.68	$r = 1, \lambda = 0.25$
	3	Hardest	94.50	0.24	0.24	0.48	$r = 2, \lambda = 0.25$
	1	Hardest	94.67	0.07	0.07	0.14	$r = 4, \lambda = 0.25$
NATOPS	0	Easiest	82.78	0.00	0.00	0.00	$r = 1, \lambda = 0.25$
	1	Easiest	82.78	0.00	0.00	0.00	$r = 2, \lambda = 1.00$
	2	Easiest	84.44	0.00	-1.67	0.00	$r = 4, \lambda = 1.00$
	3	Hardest	81.67	0.56	1.11	1.67	$r = 3, \lambda = 1.00$
	4	Hardest	81.67	0.56	1.11	1.67	$r = 4, \lambda = 0.50$
	22	Hardest	82.22	0.56	0.56	1.11	$r = 4, \lambda = 0.50$
FaceDetection	7	Easiest	67.42	0.00	-0.09	0.00	$r = 4, \lambda = 1.00$
	25	Easiest	67.37	0.00	-0.03	0.00	$r = 1, \lambda = 0.25$
	28	Easiest	67.40	0.00	-0.06	0.00	$r = 2, \lambda = 0.75$
	104	Hardest	66.26	0.06	1.08	1.14	$r = 4, \lambda = 1.00$
	17	Hardest	67.42	0.71	-0.09	0.71	$r = 4, \lambda = 1.00$
	122	Hardest	66.66	0.03	0.68	0.71	$r = 3, \lambda = 1.00$

6 Limitations and Broader Impact

CCSP provides approximate dependence reduction without certified information erasure, formal privacy guarantees, or representation-level equivalence to retraining. Its effectiveness depends on the estimated target subspace and the representativeness of $\mathcal{D}_{\mathrm{ref}}$. For $\lambda < 1$, the latent transformation is invertible, and the encoder still processes the target trajectory; these settings therefore support attenuation rather than irreversible erasure.

Zeroed-input retraining and fine-tuning serve as adaptation references: training on a constant channel alone does not ensure invariance to arbitrary nonzero trajectories. Real-trajectory replacement avoids all-zero inputs but may disrupt cross-channel relationships. Likewise, our representation probe measures intervention detectability rather than direct target recoverability.

The evidence primarily covers graph-based MTS classification, supplemented by exploratory results from Transformer-MTS and C-MAPSS FD001 RUL regression. Generalization to forecasting, anomaly detection, and state estimation requires further validation. CCSP enables post-training control of sensor influence but could obscure signals of failures, safety risks, or bias. Deployment should cover application-specific utility checks, access controls, auditable intervention records, and downstream monitoring.

7 Conclusion and Future Work

We introduced CCSP, a training-free method for approximate channel-level unlearning in graph-based MTS models. By attenuating target-associated latent directions before graph propagation, CCSP reduces measured downstream target sensitivity while preserving near-original predictive utility. Across three classification benchmarks, it provides an efficient forgetting–utility trade-off without updating pretrained parameters. Complementary real-trajectory evaluations show reduced intervention detectability and prediction disagreement. Simultaneous multi-channel experiments and RUL regression extend the evaluation beyond individual-channel classification, while sensor-wise results reveal heterogeneous suppression–utility trade-offs. These findings support efficient post-training dependence reduction without establishing certified information erasure or equivalence to retraining.

Future work will extend CCSP to broader MTS tasks, including forecasting, anomaly detection, and state estimation. It will investigate stronger adaptive and sequential-unlearning strategies, together with more principled theoretical characterizations of forgetting in entangled temporal representations.

AI USE STATEMENT

Generative AI tools were used solely to assist with drafting and editing the manuscript, including grammar, language clarity, and readability. The authors reviewed and verified all AI-assisted edits and take full responsibility for the final content of the paper.

ETHICS STATEMENT

This work uses publicly available benchmark datasets and involves no new data collection or interaction with human participants. UCI-HAR contains sensor recordings originally collected from human participants; we use only the publicly released benchmark data. CCSP is intended to reduce the downstream influence of selected sensor channels, but it does not provide formal privacy guarantees or certified information erasure. Sensor-level interventions may also obscure safety-, failure-, or bias-related signals; practical deployment should therefore include appropriate auditability, access control, and external validation.

REPRODUCIBILITY STATEMENT

We provide an anonymized implementation of the core CCSP framework, including projection construction, hyperparameter selection, and evaluation scripts. Experimental protocols, baseline settings, and additional results are documented in the appendix. An anonymized implementation with configuration files and scripts for training, intervention, and evaluation is provided with the submission.

REFERENCES

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, Jorge Luis Reyes-Ortiz, et al. A public domain dataset for human activity recognition using smartphones. In *Esann*, volume 3, pp. 3–4, 2013.

Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

Thomas Baumhauer, Pascal Schöttle, and Matthias Zeppelzauer. Machine unlearning: Linear filtration for logit-based classifiers. *Machine Learning*, 111(9):3203–3226, 2022.

Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. Leace: Perfect linear concept erasure in closed form. *Advances in Neural Information Processing Systems*, 36:66044–66063, 2023.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pp. 463–480. IEEE, 2015.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. *arXiv preprint arXiv:2310.12508*, 2023.

Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast machine unlearning without retraining through selective synaptic dampening. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 12043–12051, 2024.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems*, 32, 2019.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9301–9309. IEEE, 2020.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, pp. 141–159. IEEE, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. Patching open-vocabulary models by interpolating weights. *Advances in Neural Information Processing Systems*, 35:29262–29277, 2022.

Ming Jin, Huan Yee Koh, Qingsong Wen, Daniele Zambon, Cesare Alippi, Geoffrey I Webb, Irwin King, and Shirui Pan. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *IEEE transactions on pattern analysis and machine intelligence*, 46(12):10466–10485, 2024.

Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International journal of forecasting*, 37(4): 1748–1764, 2021.

Yijing Liu, Qinxian Liu, Jian-Wei Zhang, Haozhe Feng, Zhongwei Wang, Zihan Zhou, and Wei Chen. Multivariate time-series forecasting with temporal polynomial graph neural networks. *Advances in neural information processing systems*, 35:19414–19426, 2022.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*, pp. 931–962. PMLR, 2021.

Thanh Tam Nguyen, Thanh Trung Huynh, Zhao Ren, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–46, 2025.

Clifton Poth, Hannah Sterz, Indraneil Paul, Sukannya Purkayastha, Leon Engländer, Timo Imhof, Ivan Vulić, Sebastian Ruder, Iryna Gurevych, and Jonas Pfeiffer. Adapters: A unified library for parameter-efficient and modular transfer learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 149–160, 2023.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 7237–7256, 2020.

Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pp. 1–9. IEEE, 2008.

Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021.

Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan Kankanhalli. Fast yet effective machine unlearning. *IEEE transactions on neural networks and learning systems*, 35(9):13046–13055, 2023.

Sana Tonekaboni, Shalmali Joshi, Kieran Campbell, David K Duvenaud, and Anna Goldenberg. What went wrong and when? instance-wise feature importance for time-series black-box models. *Advances in Neural Information Processing Systems*, 33:799–809, 2020.

Yucheng Wang, Yuecong Xu, Jianfei Yang, Min Wu, Xiaoli Li, Lihua Xie, and Zhenghua Chen. Fully-connected spatial-temporal graph for multivariate time-series data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 15715–15724, 2024.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 753–763, 2020.

Jie Xu, Zihan Wu, Cong Wang, and Xiaohua Jia. Machine unlearning: Solutions and challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 8(3):2150–2168, 2024.

Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. Graph-guided network for irregularly sampled multivariate time series. *arXiv preprint arXiv:2110.05357*, 2021.

Jianping Zhou, Junhao Li, Guanjie Zheng, Xinbing Wang, and Chenghu Zhou. Mtsci: A conditional diffusion model for multivariate time series consistent imputation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 3474–3483, 2024.

A Additional Methodological Details

A.1 Optional LPPR Repair and Adaptive Policy

Projection-only CCSP is the core method used for the paper's primary results. It collects reference activations, estimates an SVD-based target subspace, and applies the selected projection without any gradient-based update to the pretrained model parameters. We therefore reserve the term *training-free* for this projection-only variant.

When utility recovery is desired after projection, we additionally consider ***CCSP+LPPR***, an optional trainable extension. LPPR updates only a small parameter subset θ_p (e.g., the classifier head) while keeping the remaining pretrained parameters frozen:

$$\theta_p^\star = \arg\min_{\theta_p} \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(X,y)\in\mathcal{D}_{\text{train}}} \mathcal{L}_{\text{cls}}\left(f^{(k,r,\lambda)}_{\theta\setminus\theta_p,\theta_p}(X), y\right). \tag{9}$$

LPPR is not part of the core forgetting operation and is evaluated separately in Appendix C.

For the optional adaptive variant, we use a sensor-aware policy based on the original intervention gap. For target sensor s_k, let $|G_k^{\text{orig}}|$ denote the original model's sensitivity under the specified masking intervention. The policy is

$$a(s_k) = \begin{cases} \text{CCSP}, & |G_k^{\text{orig}}| < \delta, \\ \text{CCSP+LPPR}, & |G_k^{\text{orig}}| \geq \delta, \end{cases}$$

where

$$\delta = \text{median}_{s_k}\left(|G_k^{\text{orig}}|\right).$$

Because this policy may invoke LPPR, *Adaptive CCSP is a hybrid policy and is not unconditionally training-free*. The median threshold is a preliminary validation rule rather than a theoretically optimal decision criterion; the aggregate and sensor-wise results in Appendix C show that repair is not uniformly beneficial.

A.2 Computational Complexity

We analyze the overhead of CCSP for a target sensor s_k. Let $\mathcal{D}_{\text{ref}}$ be the reference set with $N_{\text{ref}} = |\mathcal{D}_{\text{ref}}|$ samples. Since each sample has T time steps and the pre-graph latent dimension is d, CCSP forms the target-sensor activation matrix

$$M_k \in \mathbb{R}^{N_{\text{ref}}T\times d}.$$

Computing the top-r target-associated directions with truncated SVD costs

$$O(N_{\text{ref}}Tdr).$$

At inference time, the projection can be applied without explicitly forming the full $d \times d$ matrix:

$$\tilde{z}_{t,k} = z_{t,k} - \lambda V_k^{(r)}\left((V_k^{(r)})^\top z_{t,k}\right).$$

This adds $O(dr)$ operations per time step, or $O(Tdr)$ per input sample. If projections are precomputed for all S possible target sensors, the total offline cost becomes

$$O(SN_{\text{ref}}Tdr).$$

Thus, CCSP scales linearly with the number of channels during offline projection construction, while a single-channel forget request introduces only $O(Tdr)$ additional inference overhead.

A.3 Theoretical analysis

We analyze how much CCSP changes the pre-graph latent representation. Let $Z \in \mathbb{R}^{T\times S\times d}$ be the pre-graph latent representation, where T is the number of time steps, S is the number of sensors, and d is the latent dimension. For a target sensor s_k, CCSP applies the projection

$$P_k = I_d - \lambda V_k^{(r)}(V_k^{(r)})^\top$$

only to the target-sensor representation $Z[:, k, :]$. Therefore, for every retained sensor $s \neq k$, the latent representation is unchanged:

$$\tilde{Z}[:, s, :] = Z[:, s, :], \qquad \forall s \neq k.$$

This means that CCSP introduces no direct perturbation to the retained sensors before graph propagation.

The total pre-graph perturbation is caused only by the target sensor:

$$\|\tilde{Z} - Z\|_F = \|\tilde{Z}[:, k, :] - Z[:, k, :]\|_F.$$

Using the CCSP projection, this becomes

$$\|\tilde{Z} - Z\|_F = \lambda \left\| Z[:, k, :] V_k^{(r)} (V_k^{(r)})^\top \right\|_F.$$

Thus, the size of the perturbation is controlled by the projection strength λ, the selected rank r, and the amount of target-sensor energy lying in the selected subspace. A larger λ or larger r attenuates a larger portion of the selected target-associated subspace, but may also cause a larger change to the model input representation.

If the downstream module Ψ_{θ_g} is L-Lipschitz with respect to the Frobenius norm, then the change in the model output is bounded by

$$\|f_\theta(X) - f'_\theta(X)\|_2 = \|\Psi_{\theta_g}(\tilde{Z}) - \Psi_{\theta_g}(Z)\|_2 \leq L \|\tilde{Z} - Z\|_F = L\lambda \left\| Z_k V_k^{(r)} (V_k^{(r)})^\top \right\|_F.$$

As we do not estimate L for the trained backbone, this bound is *qualitative* rather than a numerical guarantee: it indicates that the maximum possible output change grows at most linearly with λ and with the target-subspace energy $\|Z_k V_k^{(r)} (V_k^{(r)})^\top\|_F$. The bound also reveals that CCSP has a localized, controlled effect: for a single-channel forget request, the direct perturbation is confined to the target sensor's latent slice and does *not* scale with the total number of sensors S.

B Additional Experiments

B.1 Evaluation Protocol and Scope

Our primary experiments use projection-only CCSP with the pretrained model frozen. The projection rank and strength are selected on validation data from $r \in \{1, 2, 3, 4\}$ and $\lambda \in \{0.25, 0.5, 0.75, 1.0\}$. The masking-based influence gap $|G_k^{\text{unl}}|$ is interpreted as *intervention-specific functional dependence*; a small gap under zero masking is not treated as proof of complete information erasure. We therefore complement it with target-only real-trajectory representation leakage and Prediction Disagreement Rate (PDR) in Appendix B.8.

We use the following terminology consistently. ***CCSP*** denotes projection only and is training-free; ***CCSP+LPPR*** denotes projection followed by optional trainable repair; and ***Adaptive CCSP*** is a validation-based hybrid policy that may choose either variant. Aggregate and sensor-wise comparisons of these variants are reported in Appendix C.

B.2 Sensitivity to r and λ

We analyze the impact of projection rank and strength on forgetting and utility. Table 7 illustrates the trade-off controlled by the projection rank r and strength λ. Increasing λ or r generally improves forgetting by reducing $|G_k^{\text{unl}}|$ and increasing Channel UA, but it also increases Δ_{util} and lowers retained utility. For instance, at $r = 4$, increasing λ from 0.25 to 1.00 improves Channel UA from 23.41 to 82.68, but retained utility drops from 94.64 to 92.71. Similarly, at $\lambda = 1.00$, increasing r from 1 to 4 reduces $|G_k^{\text{unl}}|$ from 4.21 to 2.39, while the utility drop increases from 0.83 to 2.03. These results show that stronger projections attenuate more target-associated directions but may also suppress useful latent components. Therefore, (r, λ) should be selected using a validation-based forgetting–utility criterion.

Table 7: Sensitivity of CCSP to projection rank r and strength λ on HAR. Averaged over target sensors. Lower residual influence indicates stronger forgetting; lower utility drop indicates better utility.

r	λ	**Retained Utility** ↑	$\lvert\mathbf{G}_{\mathbf{k}}^{\mathrm{unl}}\rvert$ ↓	**Channel UA** ↑	$\boldsymbol{\Delta}_{\mathrm{util}}$ ↓
1	0.25	94.68 ± 0.18	7.34 ± 9.69	14.32 ± 18.54	0.06 ± 0.18
1	0.50	94.56 ± 0.40	6.21 ± 8.82	26.30 ± 34.16	0.18 ± 0.40
1	0.75	94.16 ± 1.15	5.11 ± 7.30	34.11 ± 38.26	0.58 ± 1.15
1	1.00	93.91 ± 1.05	4.21 ± 6.51	47.99 ± 23.65	0.83 ± 1.05
2	0.25	94.65 ± 0.23	7.10 ± 9.66	19.43 ± 19.69	0.09 ± 0.23
2	0.50	94.40 ± 0.59	5.69 ± 8.57	36.35 ± 35.40	0.34 ± 0.59
2	0.75	93.89 ± 1.01	4.45 ± 7.30	49.59 ± 34.53	0.85 ± 1.01
2	1.00	93.33 ± 1.11	3.11 ± 6.28	71.34 ± 20.78	1.41 ± 1.11
3	0.25	94.64 ± 0.30	6.98 ± 9.85	22.25 ± 18.60	0.10 ± 0.30
3	0.50	94.30 ± 0.63	5.60 ± 8.98	40.87 ± 33.69	0.44 ± 0.63
3	0.75	93.68 ± 1.25	4.26 ± 7.66	55.27 ± 32.42	1.06 ± 1.25
3	1.00	93.15 ± 1.24	3.08 ± 7.58	76.34 ± 22.22	1.59 ± 1.24
4	0.25	94.64 ± 0.32	6.75 ± 9.57	23.41 ± 21.43	0.10 ± 0.32
4	0.50	94.26 ± 0.74	5.00 ± 8.36	46.45 ± 35.23	0.48 ± 0.74
4	0.75	93.47 ± 1.39	3.52 ± 6.84	62.45 ± 37.21	1.27 ± 1.39
4	1.00	92.71 ± 1.53	2.39 ± 6.20	82.68 ± 21.72	2.03 ± 1.53

Table 8: Sensitivity to reference-set size $\mathcal{D}_{\mathrm{ref}}$ for estimating the CCSP projection on HAR. We fix sensor s_0, $r = 2$, and $\lambda = 0.75$.

$\mathcal{D}_{\mathrm{ref}}$ **size**	**Retained Utility** ↑	$\boldsymbol{\Delta}_{\mathrm{util}}$ ↓	$\mathbf{t}_{\mathrm{run}}$ **(s)** ↓
10%	93.04	1.70	1.86
25%	92.84	1.90	2.99
50%	92.87	1.87	3.99
100%	92.84	1.90	3.89

Table 9: Effect of removing fractions of the target-associated singular spectrum on HAR (λ=0.75). Averaged over target sensors.

Removed Spectrum / Rank	**Energy Removed**	**Retained Utility** ↑	$\lvert\mathbf{G}_{\mathbf{k}}^{\mathrm{unl}}\rvert$ ↓	**Channel UA** ↑	$\boldsymbol{\Delta}_{\mathrm{util}}$ ↓
Top-1 direction	0.465	94.16	5.11	34.11	0.58
Top-2 directions	0.668	93.89	4.45	49.59	0.85
Top-3 directions	0.793	93.68	4.26	55.27	1.06
Top-4 directions	0.851	93.47	3.52	62.45	1.27

B.3 Reference Set Sensitivity

We evaluate robustness to the size and composition of $\mathcal{D}_{\mathrm{ref}}$. Table 8 shows that CCSP is robust to the size of the reference set $\mathcal{D}_{\mathrm{ref}}$. Retained utility changes only slightly across reference-set sizes, ranging from 92.84 to 93.04, while Δ_{util} remains between 1.70 and 1.90. This indicates that a small subset of reference samples is sufficient to estimate a useful target-associated subspace. Runtime increases with larger reference sets, from 1.86s at 10% to around 4s at 50–100%, but this additional cost does not lead to a substantial utility gain. Therefore, smaller reference sets can be used to reduce overhead while preserving the effectiveness of CCSP.

B.4 Singular Spectrum Analysis

We study how singular value decomposition relates to forgetting-utility trade-offs. Table 9 shows that removing more target-associated singular directions improves forgetting but gradually reduces utility. As the removed energy increases from 0.465 to 0.851, the residual influence gap decreases from 5.11 to 3.52, and Channel UA improves from 34.11 to 62.45. At the same time, retained

Table 10: Pre-graph versus post-graph CCSP on HAR. Results are reported as mean ± standard deviation across target sensors.

Intervention	Retained Utility ↑	$\lvert \mathbf{G}_{\mathbf{k}}^{\mathrm{unl}} \rvert \downarrow$	Channel UA ↑	$\boldsymbol{\Delta}_{\mathrm{util}} \downarrow$	$\mathbf{t}_{\mathrm{run}}$ (s) ↓
Pre-Graph CCSP	**94.67 ± 0.14**	**0.10 ± 0.12**	**99.93 ± 0.15**	**0.07 ± 0.14**	**1.48 ± 0.06**
Post-Graph CCSP	93.10 ± 1.86	1.15 ± 0.91	71.23 ± 33.00	1.64 ± 1.86	3.76 ± 0.59

Table 11: Exploratory additional-backbone evaluation on HAR. CCSP is applied to FC-STGNN and a Transformer-based MTS classifier using pre-graph or pre-classifier representations.

Backbone	Method	Retained Utility ↑	$\lvert \mathbf{G}_{\mathbf{k}}^{\mathrm{unl}} \rvert \downarrow$	$\boldsymbol{\Delta}_{\mathrm{util}} \downarrow$
FC-STGNN	Original	94.74	8.41	0.00
FC-STGNN	CCSP	94.67	0.10	0.07
Transformer-MTS	Original	18.76	0.73	0.00
Transformer-MTS	CCSP	20.19	0.37	-1.43

utility drops from 94.16 to 93.47, and Δ_{util} increases from 0.58 to 1.27. These results suggest that larger removed subspaces suppress more of the target-sensor influence but may also remove useful predictive information, motivating validation-based rank selection.

B.5 Ablation Study: Pre-Graph vs Post-Graph Projection

CCSP is designed to intervene before graph propagation, so that target-sensor information is suppressed before it is mixed with neighboring sensors' information. To test this design choice, we compare the proposed pre-graph CCSP with a post-graph variant that applies the same SVD-based projection after the message-passing layers. In the post-graph variant, the target sensor has already participated in graph propagation before the projection is applied.

Table 10 shows that pre-graph projection provides a substantially better forgetting–utility trade-off than post-graph projection. Pre-graph CCSP achieves higher retained utility, lower residual influence, lower utility drop, and lower runtime. In contrast, post-graph projection leaves a much larger residual influence gap because the target sensor has already affected neighboring representations through message passing. This supports our central design choice that channel-level unlearning should occur before graph entanglement.

B.6 Exploratory Backbone Generality

Table 11 shows that CCSP can be applied to backbones beyond graph-based architectures. For FC-STGNN, CCSP reduces the residual influence gap from 8.41 to 0.10 while preserving utility, which supports the main graph-based setting of the paper. For Transformer-MTS, applying CCSP to the pre-classifier representation also reduces the influence gap from 0.73 to 0.37 and improves retained utility from 18.76 to 20.19. However, because the Transformer-MTS backbone has substantially lower original utility in this setup, these results are used only as preliminary evidence of architectural applicability, not as a primary backbone comparison.

B.7 Baseline Optimization Details

Zeroed-input retraining and fine-tuning. Both baselines preserve the original FC-STGNN input topology while replacing the complete target-sensor trajectory with zeros. For retraining, we initialize FC-STGNN from scratch and train for 41 epochs using cross-entropy loss and Adam ($\mathrm{lr} = 10^{-3}$, weight decay $= 0$, batch size $= 100$). For fine-tuning, we initialize from the pretrained checkpoint and update all model parameters for 10 epochs using Adam ($\mathrm{lr} = 10^{-4}$, weight decay $= 0$, batch size $= 100$). Both methods select the checkpoint with the highest validation accuracy.

Interpretation of the retraining reference. These baselines represent adaptation under fixed-topology, zeroed-input channel removal. Since optimization uses a constant target channel without

Table 12: Sensor-wise representation-leakage reduction on HAR under the controlled probe. Values are relative reductions with respect to the Original model.

Sensor	LEACE (%)	INLP-1 (%)	CCSP (%)
0	28.60	11.72	37.46
1	18.95	35.20	45.01
2	26.80	0.16	42.39
3	31.87	15.83	36.22
4	24.80	5.81	26.40
5	21.94	4.98	30.48
6	-6.56	19.25	36.29
7	58.34	35.76	86.28
8	49.16	31.33	28.31
Mean	28.21	17.78	40.98

explicit invariance regularization, it does not guarantee invariance to an arbitrary nonzero target trajectory. We therefore interpret these results alongside real-trajectory probing, prediction disagreement, and representation alignment. Stronger perturbation-trained retraining remains an avenue for future work.

B.8 Representation-Level Target Leakage

To assess target-sensor dependence beyond zero masking, we employ *real-trajectory replacement*. For each sensor s_i, we construct $X^{\mathrm{perm}(i)}$ by replacing only its trajectory with that from a different real test sample. All retained channels remain unchanged; self-pairings are excluded, and identical replacement pairs are used across methods. We extract retained post-graph representations and train a linear logistic probe to distinguish original from intervened inputs.

Using the intervention leakage L_i defined in Section 5.1, we compute the sensor-wise relative leakage reduction for method m:

$$R_i^m = 100 \left(1 - \frac{L_i^m}{L_i^{\mathrm{Orig}}} \right). \tag{10}$$

Lower leakage indicates weaker linear detectability of the target-only trajectory replacement in retained representations. It does not imply the complete erasure of the target information or an equivalence to retraining.

Controlled representation-editing baselines. LEACE (Belrose et al., 2023), INLP (Ravfogel et al., 2020), and CCSP intervene at the same pre-graph target-sensor representation. LEACE uses the raw target-channel patch as the continuous concept variable and is fitted only to the training split. Because INLP requires discrete concept labels, we discretize the same target-patch variable into eight clusters using K-means fitted exclusively on the training split; all INLP classifiers and null-space projections are also fitted using training data only. We report INLP-1, which removes rank $8/32$, and the four-iteration INLP endpoint, which removes all $32/32$ target-latent dimensions. CCSP uses its originally selected $r \in \{1, 2, 3, 4\}$ configurations and is not retuned based on probe performance, LEACE/INLP outcomes, or the retraining reference.

Sensor-wise leakage reductions are reported in Table 12, complementing the aggregate results in the main text. CCSP achieves lower leakage than LEACE on $8/9$ sensors and lower leakage than INLP-1 on $8/9$ sensors. Full INLP reaches chance-level probe performance only after removing all 32 target-latent dimensions, with mean test accuracy decreasing to 92.37%. In contrast, CCSP attenuates only 1–4 of 32 directions and retains 94.76% mean test accuracy. This comparison supports a selective leakage–utility trade-off rather than a claim of complete null-space elimination.

Zeroed-input retraining reference and representation alignment. Under the target-neutralized probing protocol, retraining yields $\mathrm{AUC} = 0.5000$ because the target input remains zeroed during representation extraction. This serves as a protocol-defined reference rather than evidence of learned

Table 13: Prediction disagreement under target-only real-trajectory replacement on HAR. Reduction is computed relative to Original for each sensor; the Mean row averages the nine sensor-wise relative reductions. Results use a single fixed random trajectory-replacement permutation (seed $= 1$)

Sensor	Orig. PDR (%) ↓	CCSP PDR (%) ↓	Reduction (%) ↑
0	13.91	10.35	25.61
1	2.00	1.56	22.03
2	2.75	2.41	12.35
3	10.59	9.67	8.65
4	6.72	6.48	3.54
5	5.67	5.02	11.38
6	26.20	25.55	2.46
7	13.95	8.99	35.52
8	4.11	3.05	25.62
Mean	9.54	8.12	16.35

invariance to arbitrary nonzero target trajectories. We use this as a reference endpoint, not as evidence that independently trained networks share the same internal geometry. To test the stronger representation-equivalence hypothesis directly, we additionally compute alignment with the retrained reference. We further assess representation similarity to retraining using CKA and prediction agreement. Original–Retraining and CCSP–Retraining CKA are 0.8784 and 0.8776, respectively, while prediction agreement increases marginally from 94.74% to 94.84%. Thus, CCSP reduces measured intervention detectability without reconstructing the representations learned through retraining.

B.8.1 Prediction Disagreement Rate (PDR)

Aggregate accuracy can conceal sample-level behavioral changes because identical accuracies may arise from different prediction sets. Under the same target-only real-trajectory intervention, we therefore define

$$\mathrm{PDR}_k = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[\hat{y}(X_i) \neq \hat{y}\left(X_i^{\mathrm{perm}(k)}\right)\right]. \tag{11}$$

A lower PDR indicates that replacing only the target trajectory changes fewer individual predictions.

CCSP reduces PDR for all $9/9$ target sensors. Mean PDR decreases by 1.42 percentage points, from 9.54% to 8.12%, as shown in Table 13. Table 13 uses a single fixed random trajectory-replacement permutation (seed $= 1$). This corresponds to an approximately 14.9% reduction when computed from the two means; the 16.35% value in the bottom row of Table 13 instead averages the nine sensor-wise relative reductions. The two figures differ because the reduction computed from the means is influenced more by high-PDR sensors, notably sensor 6 (26.20% original PDR, 2.46% reduction), whereas the sensor-wise average weights all sensors equally. The closely related 8.08% figure in Table 14 comes from a separate five-permutation analysis that averages results over five random trajectory-replacement permutations rather than a single fixed permutation. This explains the slight numerical difference between the two tables; both use the same frozen CCSP configurations. Because all non-intervened raw channels remain unchanged within each pair, PDR provides a sample-level measure of residual target sensitivity that complements aggregate accuracy and representation leakage.

B.8.2 Retained-Channel Sensitivity Control

To distinguish selective suppression from general model insensitivity, we evaluate target- and retained-channel sensitivity on HAR. For each forgotten sensor, we independently replace its trajectory and the trajectories of each retained sensor with those from other real test samples. All non-intervened channels remain unchanged. Original and CCSP use identical replacement pairs across five individual random derangements, with previously selected CCSP configurations fixed.

We measure Prediction Disagreement Rate (PDR) and probability total variation (TV), defined as half the L1 distance between predicted class probability vectors. Results are averaged over all nine forgetting targets. Relative reductions are computed per target and then averaged.

Table 14: Target and retained-channel sensitivity on HAR, averaged over five random trajectory-replacement permutations. Results are averaged over five random trajectory-replacement permutations.

Intervention	Metric	Original	CCSP	Reduction (%)
Target	PDR (%)	9.60	8.08	17.87
Retained	PDR (%)	9.60	9.87	-2.79
Target	Prob. TV	0.0998	0.0842	17.48
Retained	Prob. TV	0.0998	0.1027	-2.97

Table 15: Exhaustive simultaneous multi-channel removal on HAR. Results are averaged over all target subsets and twelve projection settings for each subset. Lower residual influence and higher test accuracy are better.

$\lvert\mathcal{S}_f\rvert$	Subsets	Runs	Orig. $\lvert G_{\mathcal{S}}\rvert$	CCSP $\lvert G_{\mathcal{S}}\rvert \downarrow$	Test Acc. (%) $\uparrow$
2	36	432	21.68	**11.55**	93.43
3	84	1008	37.09	**20.14**	92.70
4	126	1512	50.20	**30.34**	91.85

Table 14 shows that CCSP reduces target sensitivity while largely preserving aggregate retained-channel responsiveness. Original and mean CCSP test accuracies are 94.74% and 94.76%, respectively. The five donor repetitions show consistent aggregate behavior, although individual sensor responses vary. These findings support intervention-specific selectivity, without establishing invariance of every retained channel or complete information erasure.

B.9 Simultaneous Multi-Channel Removal

We extend CCSP to simultaneous multi-channel forget requests. For a target set $\mathcal{S}_f$, sensor-specific projections are applied jointly to their corresponding pre-graph latent slices, while all retained slices remain unmodified at the intervention point. We measure residual set-level dependence using $|G^g_{\mathcal{S}_f}|$, where $G^g_{\mathcal{S}_f} = A^g_{\text{test}} - A^g_{\text{masked}(\mathcal{S}_f)}$.

On HAR, we exhaustively evaluate all the target subsets with $|\mathcal{S}_f| \in \{2, 3, 4\}$ using three projection ranks and four strengths (12 configurations per subset). This yields 36, 84, and 126 subsets, corresponding to 432, 1008, and 1512 evaluated runs, respectively.

Table 15 shows that CCSP reduces set-level dependence in 98.6%, 100%, and 100% of the evaluated two-, three-, and four-channel runs, respectively. Mean residual influence decreases from 21.68 to 11.55, 37.09 to 20.14, and 50.20 to 30.34, while test accuracy remains 93.43%, 92.70%, and 91.85%. These results demonstrate simultaneous dependence reduction, although residual influence and utility loss increase as more channels are targeted.

Sequential removal and reintroduction. For fixed, independently estimated sensor-specific projections, each Π_i modifies only latent slice i. Consequently, projections for distinct sensors commute:

$$\Pi_i(\Pi_j(Z)) = \Pi_j(\Pi_i(Z)), \qquad i \neq j. \tag{12}$$

Thus, applying the same fixed projections sequentially or jointly produces identical pre-graph representations, provided that neither the projections nor the pretrained model is updated between requests.

This establishes mathematical order independence for the fixed projection-only CCSP; our experiments directly evaluate simultaneous, rather than adaptive, sequential removal. Re-estimating subspaces or applying LPPR between requests may introduce order-dependent effects and is outside the scope of the present evaluation. Since the pretrained parameters remain unchanged, a sensor can be reintroduced by disabling its projection while retaining other active projections.

Table 16: Representative sensor-wise forgetting–utility outcomes across datasets. FD summarizes residual dependence and positive utility degradation; the two constituent terms remain the primary quantities.

Dataset	Sensor	$\lvert G_i^{\mathrm{unl}}\rvert$	Δ_{util}	FD	Main pattern
HAR	2	0.00	0.00	0.00	Low difficulty
	0	0.34	0.34	0.68	Mixed
	3	0.24	0.24	0.48	Mixed
NATOPS	0	0.00	0.00	0.00	Low difficulty
	3	0.56	1.11	1.67	Utility-dominated
	22	0.56	0.56	1.11	Mixed
FaceDetection	17	0.71	-0.09	0.71	Residual-dominated
	104	0.06	1.08	1.14	Utility-dominated
	122	0.03	0.68	0.71	Utility-dominated

B.10 Sensor-wise Forgetting–Utility Heterogeneity

For sensor s_i, we treat post-unlearning residual dependence $|G_i^{\mathrm{unl}}|$ and utility degradation $\Delta_{\mathrm{util}}(s_i)$ as the primary sensor-wise quantities. For compact summarization, we define

$$\mathrm{FD}(s_i) = |G_i^{\mathrm{unl}}| + \max(0, \Delta_{\mathrm{util}}(s_i)) \,. \tag{13}$$

A low FD therefore indicates that the target sensor can be suppressed with little residual dependence and little or no utility degradation, whereas a larger FD may arise from residual dependence, utility degradation, or both. FD is a CCSP-specific operational summary of this trade-off; it is not a model-independent measure of feature importance or representation entanglement and should not be compared across unrelated methods, architectures, or tasks.

Table 16 illustrates this heterogeneity using representative sensors from all three classification datasets. On HAR, sensor 2 can be suppressed with essentially no residual dependence or utility loss, whereas sensor 0 incurs both residual dependence and utility degradation. NATOPS exhibits an even wider range, from FD $= 0$ for sensor 0 to FD $= 1.67$ for sensor 3. The FaceDetection examples further show why the constituent terms should be inspected separately: sensor 17 is difficult primarily because residual dependence remains, whereas sensors 104 and 122 exhibit low residual dependence but larger utility degradation. HAR sensor 6 further illustrates the protocol-specific nature of FD: despite its low zero-mask FD, it retains relatively high post-CCSP PDR under real-trajectory replacement. This discrepancy is not contradictory, since the two metrics measure target sensitivity under different interventions. Thus, similar FD values need not correspond to the same type of sensor-specific unlearning difficulty.

C Adaptive CCSP Analysis

This section analyzes the optional CCSP+LPPR extension and the adaptive selection policy. Projection-only CCSP remains the core training-free method; CCSP+LPPR is trainable, and Adaptive CCSP is a hybrid policy that may invoke repair. The purpose of this section is to quantify when repair is selected and whether it consistently improves the forgetting–utility trade-off.

C.1 Aggregate Variant Comparison

Table 17 shows that projection-only CCSP provides the most consistent aggregate forgetting–utility trade-off in these experiments. It retains higher utility than CCSP+LPPR across all three datasets, matches or improves residual dependence, and is approximately 1.9–$4.3\times$ faster. Adaptive CCSP reduces unnecessary repairs compared with applying LPPR broadly, but it still does not consistently outperform projection-only CCSP. We therefore treat LPPR as an optional repair mechanism rather than part of the core CCSP method.

Table 17: CCSP variant comparison. CCSP uses projection only; CCSP+LPPR adds optional trainable repair; and Adaptive CCSP selects between the two variants per sensor. Results are averaged over target sensors.

Dataset	Strategy	Retained Utility ↑	$\lvert \mathbf{G}_{\mathbf{k}}^{\mathrm{unl}} \rvert \downarrow$	$\boldsymbol{\Delta}_{\mathrm{util}} \downarrow$	$\mathbf{t}_{\mathrm{run}}$ (s) ↓	Peak GPU Mem. (MB) ↓
HAR	Plain CCSP	**94.76 ± 0.36**	**0.19 ± 0.30**	**0.07 ± 0.14**	**1.48 ± 0.06**	**76.14 ± 0.00**
	CCSP+LPPR	92.20 ± 3.34	1.92 ± 4.82	2.54 ± 3.34	6.38 ± 0.51	85.10 ± 0.00
	Adaptive CCSP	92.59 ± 3.63	1.98 ± 4.80	2.15 ± 3.63	4.32 ± 2.59	81.12 ± 4.73
NATOPS	Plain CCSP	**83.08 ± 0.79**	**0.19 ± 0.27**	-0.30 ± 0.79	**1.98 ± 0.10**	**23.38 ± 0.00**
	CCSP+LPPR	82.89 ± 0.75	0.25 ± 0.54	-0.12 ± 0.75	3.90 ± 0.21	32.74 ± 0.00
	Adaptive CCSP	82.43 ± 0.82	0.95 ± 0.64	0.35 ± 0.82	3.10 ± 0.93	29.23 ± 4.63
FaceDetection	Plain CCSP	**67.18 ± 0.20**	0.05 ± 0.10	**0.16 ± 0.20**	**12.20 ± 4.42**	**99.15 ± 0.00**
	CCSP+LPPR	66.88 ± 0.17	**0.05 ± 0.09**	0.46 ± 0.17	23.49 ± 4.75	110.47 ± 0.00
	Adaptive CCSP	66.88 ± 0.30	0.26 ± 0.23	0.45 ± 0.30	18.24 ± 7.57	104.97 ± 5.68

Table 18: Adaptive CCSP selection summary. Sensor counts for CCSP and CCSP+LPPR.

Dataset	No. Sensors	CCSP Selected	CCSP+LPPR Selected
HAR	9	4	5
NATOPS	24	9	15
FaceDetection	144	70	74

C.2 Selection Summary

We first summarize the number of sensors assigned to each strategy across datasets. Table 18 summarizes how often the adaptive policy selects projection-only CCSP versus CCSP+LPPR. The policy activates LPPR for a substantial fraction of sensors: 5 out of 9 sensors on HAR, 15 out of 24 on NATOPS, and 74 out of 144 on FaceDetection. This shows that many sensors are classified as high-influence under the adaptive criterion. However, selecting LPPR more frequently does not necessarily improve the aggregate forgetting–utility trade-off, as shown in Table 17. Instead, the selection summary indicates that the adaptive policy is conservative and often invokes sensor repair when the sensor's original influence exceeds the dataset-level threshold.

C.3 Sensor-wise Decisions

We further inspect the adaptive decisions at the sensor level on HAR, reporting both the projection-only and LPPR outcomes for each target sensor. Table 19 provides a sensor-level view of the adaptive decisions on HAR. The results show that LPPR does not consistently improve over projection-only CCSP. For example, sensors 3, 4, 5, and 6 are assigned to CCSP+LPPR, but for these sensors, LPPR reduces utility and increases residual influence compared with CCSP. Sensor 6 is especially notable: CCSP achieves 94.77% utility with residual influence 0.03, whereas LPPR reduces utility to 83.85% and increases residual influence to 14.76. This explains why Adaptive CCSP does not consistently outperform CCSP in the aggregate results.

At the same time, the table also shows why an adaptive mechanism may still be useful: some sensors, such as sensor 7, have higher residual influence under CCSP and lower residual influence after LPPR. Therefore, LPPR can be beneficial in selected cases, but the decision rule must be conservative. These sensor-wise results support the main conclusion that projection-only CCSP should be the default method, while LPPR should be treated as an optional repair step rather than a universally beneficial component.

D Beyond Classification: Remaining Useful Life Regression

To examine CCSP beyond classification, we evaluate continuous-valued remaining useful life (RUL) regression using FC-STGNN (Wang et al., 2024) on C-MAPSS FD001 (Saxena et al., 2008). This experiment evaluates whether pre-graph projection reduces target-sensor influence on continuous

Table 19: Sensor-wise adaptive CCSP decisions on HAR. Selections use validation scores; test-set utility and residual influence are reported.

Sensor	CCSP Utility ↑	|G| CCSP ↓	LPPR Utility ↑	|G| LPPR ↓	Selected Method	Adaptive Runtime (s) ↓
0	94.40	0.34	92.06	0.37	CCSP+LPPR	7.36
1	94.67	0.07	94.10	0.27	CCSP	1.55
2	94.74	0.00	94.06	0.10	CCSP	1.98
3	94.50	0.24	90.87	0.75	CCSP+LPPR	5.94
4	94.74	0.00	93.45	0.27	CCSP+LPPR	6.41
5	94.71	0.03	92.84	0.31	CCSP+LPPR	6.12
6	94.77	0.03	83.85	14.76	CCSP+LPPR	6.53
7	95.66	0.92	94.30	0.10	CCSP	1.38
8	94.67	0.07	93.96	0.61	CCSP	1.61

Table 20: Aggregate C-MAPSS FD001 results at the shared CCSP setting $r = 2$, $\lambda = 0.5$.

Metric	Result
Evaluated sensors	14
Original test RMSE	11.7600
Mean CCSP test RMSE	11.6963
Mean relative RMSE change	-0.54%
Mean sensor-wise influence reduction	31.41%
Median influence reduction	31.96%
Per-sensor reduction range	20.55%–39.33%
Sensors with positive influence reduction	14/14
Sensors within 5% test-RMSE budget	13/14

Table 21: Sensor-wise CCSP results on C-MAPSS FD001. D_i is the mean absolute RUL prediction shift under target-only real-trajectory replacement. Lower D_i is better.

Sensor	Original D_i	CCSP D_i ↓	Reduction (%) ↑	CCSP RMSE
0	7.47	5.12	31.44	11.98
1	2.89	1.81	37.32	11.49
2	8.18	5.52	32.47	11.50
3	4.37	3.40	22.14	11.38
4	5.71	3.71	35.05	11.87
5	3.62	2.64	27.22	11.69
6	5.42	3.29	39.33	12.36
7	4.35	3.04	30.08	11.96
8	5.16	3.27	36.55	11.58
9	3.99	3.08	22.88	11.57
10	3.58	2.47	30.81	11.79
11	3.93	2.43	38.19	11.40
12	5.00	3.98	20.55	11.51
13	3.59	2.31	35.74	11.66
Mean	4.81	3.29	31.41	11.6963

predictions. RUL regression differs from sequence-to-sequence forecasting and is therefore treated as an initial extension beyond classification.

For each of the 14 evaluated sensors, we replace only the complete target-sensor trajectory with the corresponding trajectory from a different real test sample while keeping all other retained sensors exactly unchanged. We exclude self-pairings, use the same replacement pairs for the Original model and CCSP, and verify that the maximum absolute change over all retained input channels is zero. For target sensor s_i, influence is the mean absolute change in predicted RUL,

$$D_i = \frac{1}{N} \sum_{n=1}^{N} \left| f(x_n) - f\left(x_n^{(i)}\right) \right|. \tag{14}$$

Unlike the classification masking gap, D_i directly measures the magnitude of target-sensor-induced variation in a continuous prediction.

RUL results and controllability. Tables 20 and 21 summarize the results using a shared configuration of $r = 2$ and $\lambda = 0.5$. CCSP reduces target-induced prediction variation for all 14 sensors, achieving a mean sensor-wise reduction of 31.41%. Mean test RMSE remains comparable to the original model (11.6963 versus 11.7600), with $13/14$ sensors satisfying the 5% test-RMSE budget.

We additionally evaluate controllability using a validation-only sweep over $r \in \{1, 2, 3, 4\}$ and $\lambda \in \{0.25, 0.5, 0.75, 1.0\}$ Of 224 sensor-specific configurations, 147 satisfy the 5% validation-RMSE budget, with at least one feasible setting for every sensor. All pretrained parameters remain frozen.

These findings provide initial evidence of reduced target-sensor sensitivity in continuous-output regression. Since the evaluation is limited to C-MAPSS FD001, the generality to forecasting, anomaly detection, and other MTS tasks remains unestablished.