LogiC-Diff: Embedding Security Properties Into AI-Enabled Cyber-Physical Systems

Ziyan An[1], **John Stankovic**[2], **Meiyi Ma**[1]
[1]Department of Computer Science, Vanderbilt University, Nashville, TN
[2]Department of Computer Science, University of Virginia, Charlottesville, VA
{ziyan.an,meiyi.ma}@vanderbilt.edu
jas9f@virginia.edu

Abstract

AI-enabled Cyber-Physical Systems (CPS) are highly vulnerable to adversarial and anomalous inputs, where small perturbations can induce cascading errors and unsafe control actions. Existing approaches, such as rule-based filtering, training-time regularization, or diffusion-based reconstruction, either operate outside the model or lack mechanisms to incorporate formal security specifications into the prediction process. In this paper, we take the first step toward embedding security properties directly into AI-enabled CPS, enabling predictive models to enforce system-level constraints during inference rather than relying on external defenses. We introduce a logic-conditioned bi-stage diffusion framework that integrates Signal Temporal Logic (STL) specifications into forecasting. STL serves as a first-class conditioning signal that guides both an input repair stage and an output refinement stage, allowing the model to jointly mitigate adversarial perturbations and enforce desired temporal behaviors to satisfy security-critical properties. We evaluate our approach on two real-world multivariate CPS forecasting datasets under a diverse set of physical sensor and cyber attacks. Across sensor faults, gradient-based attacks, adaptive attacks, and varying attack strengths, our method consistently improves robustness and specification compliance, degrades more gracefully as attack strength increases, and generalizes better to unseen attacks. Ablation studies on specification coverage and quality further show that embedding logical security properties yields gains unattainable by reconstruction-based methods alone, highlighting a new direction for integrating formal methods with generative models in secure CPS.

1 Introduction

AI-enabled cyber-physical systems (CPS), such as smart healthcare, smart cities, and autonomous mobility, rely heavily on deep learning-based forecasting models to support real-time monitoring and control [56, 51, 44, 37]. However, these systems operate in high-stakes environments where failures can lead to unsafe or even catastrophic outcomes. A key vulnerability arises from data-level adversarial or anomalous inputs: even small ℓ_2-bounded perturbations can distort sensor signals, propagate through forecasting pipelines, and ultimately trigger unsafe downstream control actions [28, 53, 17]. As a result, ensuring the robustness of forecasting models is critical to the safe deployment of AI-enabled CPS.

At the core of this challenge is a fundamental limitation of current learning-based forecasting models. These models are largely data-driven and unconstrained, with no built-in guarantees of safety, temporal consistency, or adherence to system-level requirements. When inputs are corrupted, prediction errors can propagate and amplify across the pipeline, leading to cascading failures. This exposes a key gap: current forecasting models lack mechanisms to enforce system-level security properties during prediction. In this paper, we ask a central question: *can system-critical security*

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

properties be embedded into learning-enabled CPS by design, so that models remain robust under adversarial attacks?

Existing defenses for CPS forecasting fall into two broad categories. The first focuses on improving robustness through learning-based reconstruction methods [5, 41, 52]. Early approaches such as Defense-GAN [42] detect or repair adversarial inputs by exploiting distributional discrepancies, while more recent diffusion-based models restore samples via learned reverse processes [46, 20]. Although these methods achieve strong empirical robustness, they rely solely on distributional reconstruction and lack an explicit notion of what constitutes safe or acceptable system behavior. As a result, they cannot enforce temporal or logical constraints on predictions.

A complementary line of work incorporates rule-based properties to improve alignment with system requirements under nominal conditions [31, 21, 1]. These approaches encode semantic constraints—such as temporal ordering, persistence, and cross-feature consistency—using formal languages like Linear Temporal Logic [40], Metric Temporal Logic [25], and Signal Temporal Logic (STL) [34]. In practice, such properties are applied as training-time regularizers or post-hoc monitoring and filtering mechanisms [48, 14, 4, 55]. However, these approaches are not designed as security mechanisms: they operate outside the predictive process, are reactive rather than preventative, and do not mitigate adversarial perturbations during inference. This limitation is exacerbated under structured perturbations, such as gradient-based attacks that shift inputs across fragile decision boundaries: once temporal dynamics are distorted before prediction, the model may apply a qualitatively different mapping to the corrupted data [19, 50]. Indeed, post-hoc correction can enforce certain output constraints, but it cannot reliably recover information lost upstream, and the resulting forecast may still fail to reflect the true system dynamics.

In this work, we take a step toward embedding security properties directly into the forecasting process. We propose a logic-conditioned bi-stage diffusion framework that integrates formal specifications into model inference. Specifically, we use Signal Temporal Logic (STL) to express system-critical properties and incorporate them as conditioning signals within a diffusion-based repair pipeline. This enables a new paradigm of specification-aware prediction, where forecasts are actively shaped to satisfy security constraints while maintaining robustness.

Our work investigates repositioning prior secure system specifications, expressed as temporal logic, into two stages of the diffusion-based repair process, thereby enabling both pre-hoc and post-hoc defense. Specifically, we condition the diffusion-based repair model on a logic-informed signal obtained by projecting the data manifold, at different stages of the framework, onto the ℓ_1 minimum-distance φ-feasible set. We then modulate the diffusion prediction with a learnable gating coefficient that balances the diffusion reconstruction and the projected signal in the final output. The framework is trained end-to-end under loss terms that cover each of our three defense desideratum.

We evaluate our method on two real-world multivariate time-series CPS forecasting datasets, including a smart city dataset spanning Caltrans [8] and air quality and a UAV dataset [24]. The framework is evaluated against a comprehensive set of black-box sensor attacks and faults, gradient-based white-box attacks, adaptive and query-based attacks, sparse indirect attacks, and varying attack strengths. We further study cross-attack generalization, cross-domain transfer, and sensitivity to specification coverage and quality. Compared with a broad set of baselines spanning adversarial training, adversarial purification, randomized smoothing, and logic-regularized learning, our approach consistently reduces forecasting error and improves specification compliance and generalization across a range of attack strengths. Our ablation results further show that these gains depend strongly on the guidance provided by the embedded logic specifications.

To our knowledge, this is the first work to embed formal security properties directly into CPS forecasting via logic-conditioned generative modeling. **Contributions.** (1) We formulate secure CPS forecasting as embedding STL-based specifications into learning-based prediction. (2) We propose a logic-conditioned bi-stage diffusion framework for input repair and output refinement. (3) We develop an STL-guided conditioning mechanism that enables specification-aware generative reconstruction under adversarial settings. (4) We conduct a comprehensive evaluation across two CPS domains and a diverse set of threat models, including physical sensor attacks, gradient-based and adaptive cyber attacks, varying attack strengths, sparse indirect attacks, and unseen attack settings. The results show that LOGIC-DIFF consistently improves robustness and specification compliance while demonstrating stronger generalization across attacks and domains.

2 Background and Problem Statement

We introduce the key preliminaries underlying our approach, including DDPM models [20] for generative repair, Signal Temporal Logic (STL) [34] for formal specification, and the CPS security setting considered in this work.

Background: Diffusion Models for Data Repair. Diffusion models [46, 47] learn a data distribution $p(x)$ through gradual noising and denoising. In the forward process, clean data x_0 are corrupted by Gaussian noise over N steps until $x_{\mathrm{N}} \sim \mathcal{N}(0, I)$. We build on DDPMs [20] as the basis of our input repair mechanism, where a reverse diffusion network $\epsilon_\theta(x_n, n)$ predicts the injected noise and denoises corrupted inputs back toward the clean data manifold.

In forecasting settings with adversarial purification, a potentially corrupted input $x^* = x + \delta$ is treated as a noisy observation of an underlying clean sample x and is first injected with Gaussian noise during the forward diffusion step. In the reversing step, a denoiser ϵ_θ parameterized by θ guides the learned reverse process to produce the reconstructed data $\tilde{x} = \Phi_\theta(x^*)$, where Φ_θ denotes the forward-reverse denosing process. In the subsequent CPS forecasting step, a forecasting neural network F_ω is applied to $\tilde{x}$, with the goal that its prediction approximates the clean-input forecast $F_\omega(x)$.

Background: Signal Temporal Logic. STL is a real-valued, time-bounded formalism for specifying temporal properties over continuous signals. Given time-series data and a CPS specification, STL supports monitoring through qualitative semantics that describe Boolean satisfaction and quantitative semantics that measure the degree of satisfaction or violation [16]. In this work, we use STL to formalize "secure" system behaviors, including temporal requirements such as event ordering and persistence, as well as relationships between multiple signal traces over specified time intervals.

Consider an $[a, b]$-bounded time interval and an atomic predicate of the form $\mu \equiv f(x_t) \geq c$, where $f : \mathbb{R}^d \to \mathbb{R}$ is a valuation function and $c \in \mathbb{R}$ is a threshold. A formula φ is defined inductively as $\varphi := \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \mid \Box_{[a,b]}\varphi \mid \Diamond_{[a,b]}\varphi$. Here, the temporal operators include the "always" operator $\Box_{[a,b]}$, which requires that a time series satisfy φ at everywhere within the bounded interval; the "eventually" operator $\Diamond_{[a,b]}$, which requires that φ hold at least once within the bounded interval; and the "until" operator $\mathcal{U}_{[a,b]}$, which requires that φ_1 hold until φ_2 is true within the bounded interval. The STL quantitative semantics function $\rho_\varphi(x, t) \in \mathbb{R}$ measures the signed margin by which a signal satisfies or violates φ at time t. We refer the reader to App. B for the full definition of STL semantics and additional examples of logic rules.

Threat Model. We consider a comprehensive set of runtime attacks for CPS spanning two attack families, both of them aim to corrupt the forecasting model's output and adversely affect downstream system performance. The first family consists of black-box CPS *physical* system attacks, which we refer to as sensor attacks [57]. In this setting, the attacker has access to the physical or communication layer and can compromise sensor data by freezing, replaying, or spoofing sensor readings, as well as by applying randomized mixtures of these attack types [17].

The second family consists of gradient-based white-box attacks to the *cyber* system, bounded by an ℓ_2 budget $\|\delta\|_2 \leq \varepsilon$, where δ is the added perturbation. These attacks leverage model gradients to generate the worst-case adversarial input within the allowed budget, which is typically formulated as $x^* \in \arg\max_{x'} \mathcal{L}(x', y; \omega)$ s.t.$\|x' - x\|_2 \leq \varepsilon$, where $\mathcal{L}$ is the model's training loss, (x, y) is a clean input-output pair, and ω denotes the model parameters [18]. We consider specific attacks including PGD [32], MIM [15], CW [7], and AutoAttack [12].

Defense Goal. Given a potentially attacked input trajectory x^*, our goal is to produce a repaired trajectory $\tilde{x}$ and subsequently a final forecast $\tilde{y}$ that are both accurate and semantically valid. Specifically, the defense should produce: (i) a repaired input trajectory $\tilde{x}$ that remains close to the clean input x and satisfies an input-side STL specification φ, and (ii) a final forecast $\tilde{y}$ that approximates the clean-input forecast $F_\omega(x)$ and satisfies an output-side STL specification φ'. Formally, we have:

$$F_\omega(\tilde{x}) \approx F_\omega(x), \quad \tilde{x} \in \arg\max_{\bar{x} \in \mathcal{N}(x^*)} \rho_\varphi(\bar{x}, t), \quad \tilde{y} \in \arg\max_{\bar{y} \in \mathcal{N}(F_\omega(\tilde{x}))} \rho_{\varphi'}(\bar{y}, t),$$

where $\mathcal{N}(\cdot)$ denotes a projected neighborhood of the given point. Together, these objectives require a defense that produces outputs both likely under the clean data distribution and consistent with system-

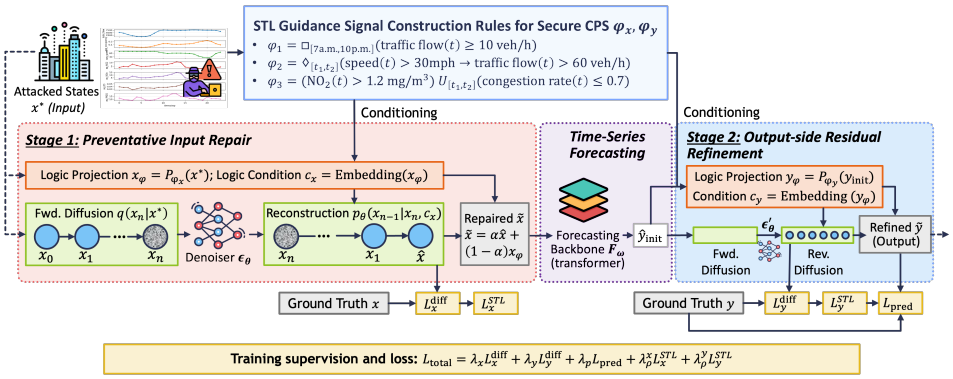

Figure 1: Overview of LogiC-Diff. Given a potentially attacked multivariate CPS time-series input, LogiC-Diff first constructs STL-guided signals from CPS security specifications. The framework then performs a repair and a refinement stage on input-side and an output-side, respectively. In both stages, the diffusion estimate is blended with the projection to encourage both data-plausible reconstruction and specification-consistent behavior. The framework is trained end-to-end.

level security specifications. We describe how our bi-stage diffusion framework operationalizes these goals in the next section.

3 Method: LogiC-Diff

We present our logic-guided bi-stage diffusion defense for secure CPS forecasting. At a high level, the method first projects STL specifications into a logic-consistent guidance signal, and then uses this signal to nudge the reconstruction process at two repair stages: an input-stage diffusion that preventatively reconstructs the attacked trajectory before forecasting, and an output-stage diffusion that refines the forecast residual afterward. We refer to this pipeline as LogiC-Diff, short for STL-conditioned diffusion reconstruction, and use Φ_θ to denote the full end-to-end pipeline. Figure 1 depicts the overall workflow of our framework.

3.1 STL Guidance Signal

In this section, we formalize how STL rules support defense against security attacks. We introduce CPS-oriented spatiotemporal logic formulas that encode safe global behavior and are also relevant to downstream control, then describe how an attacked signal is projected into a guidance signal. This signal is later used as the basis for an embedding to condition the diffusion model and as a reference trajectory in the final interpolation step, encouraging specification-compliant repair.

Security Guidance: STL Rules. In Section 2, we introduced the formal definition of STL and its temporal operators. In our setting, an STL rule takes a form such as $\varphi := \square_{[t_1,t_2]}(x_t \geq c)$, where the temporal operator $\square_{[t_1,t_2]}$ may be replaced by $\Diamond_{[t_1,t_2]}$ depending on the desired requirement, and $\mu := (x_t \geq c)$ denotes the atomic predicate that the temporal requirement is based on.

Specifically, we define the following set of spatiotemporal logic formulas to support the security requirements of CPS operation. Let $x_t^{(j)}$ denote the value of the j-th feature at time t in the forecasting dataset. We then apply the following coarse categories of logic rules, summarized in Table 1.

More specific examples specifications are listed under App. F. In practice, multiple STL rules can be combined through logical conjunction. We use φ to denote the set of rules applied for input reconstruction, and φ' to denote the set of rules applied for output-side residual repair and reconstruction. In many applications, these rules can be specified directly by system engineers; alternatively, several existing tools like TeLEx and Quivr can infer or construct such rules from sample data or query specifications [22, 35].

Guidance Signal Construction. We have established several categories of rules that define secure system behavior. We now describe how a logic specification is grounded into a guidance signal for

Table 1: STL-based CPS specifications. Each rule is parameterized by features and a time interval $[t_1, t_2]$ and encodes a physically meaningful constraint on feature trajectories.

STL-based Guidance Signal Construction Rules for Secure CPS	
$\varphi_1 := \square_{[t_1,t_2]}\big((x_t^{(j)} \geq c_1) \wedge \neg(x_t^{(j)} \geq c_2)\big)$	`Stability`$(x^{(j)}, [t_1, t_2])$ enforces physically valid ranges and prevents false high or low values outside feasible limits over time.
$\varphi_2 := \square_{[t_1,t_2]}\Big(\neg\big((x_t^{(j)} \geq c_1) \wedge \neg(x_t^{(j+1)} \geq c_2)\big)\Big)$	`Associativity`$(x^{(j)}, x^{(j+1)}, [t_1, t_2])$ encapsulates the correlational associativity across features over time.
$\varphi_3 := \neg(x_t^{(j)} \geq c_1)\,\mathcal{U}_{[t_1,t_2]}\,(x_t^{(j+1)} \geq c_2)$	`Propagation`$(x^{(j)}, x^{(j+1)}, [t_1, t_2])$ strengthens upstream and downstream system behavior between two sensor locations or across temporally related events.
$\varphi_4 := \square_{[t_1,t_2]}\Big(\neg\big(\lvert\bar{x}^{(j)}_{[t-w_1,t]} - \bar{x}^{(j)}_{[t-w_1-w_2,t-w_1]}\rvert \geq c_1\big)\Big)$	`Smoothness`$(x^{(j)}, [t_1, t_2], w_1, w_2)$ bounds the change in mean between adjacent windows of widths w_1 and w_2, detecting persistent changes in feature behavior.
$\varphi_5 := \square_{[t_1,t_2]}\Big(\neg\big((x_t^{(j)} \geq c_1) \wedge \neg\Diamond_{[d_1,d_2]}(x_t^{(j+1)} \geq c_2)\big)\Big)$	`Latency`$(x^{(j)}, x^{(j+1)}, [t_1, t_2], [d_1, d_2])$ enforces that whenever feature $x^{(j)}$ reaches level c_1, feature $x^{(j+1)}$ reaches level c_2 at some time within the future window $[d_1, d_2]$.
$\varphi_6 := \Diamond_{[t_1,t_2]}\,\square_{[0,d]}\big((x_t^{(j)} \geq c_1) \wedge \neg(x_t^{(j)} \geq c_2)\big)$	`Recovery`$(x^{(j)}, [t_1, t_2], d)$ requires the feature to eventually enter and remain within a safe operational band for a duration of at least d time units.

diffusion-based reconstruction. Following prior work [31], we project the attacked signal x^* onto the φ-feasible set, yielding a logic-consistent reference trajectory. We define the projection operator as $P_\varphi(x)$, such that it yields the closest satisfying trace in ℓ_1 distance.

The projection treats logical and temporal operators compositionally through the quantitative STL semantics. For an atomic predicate $\mu = (x_t \geq c)$, the projection clamps each violating time point upward to the threshold, $x'_t = \max(x^*_t, c)$, while the satisfying points are unchanged. For $\square_{[a,b]}\mu$ operator, which requires μ to hold at every $t \in [a, b]$, we apply this clamping at every time point in the interval, $x'_t = \max(x^*_t, c)$ for all $t \in [a, b]$. For the $\Diamond_{[a,b]}\mu$ operator, which requires μ to hold at some $t \in [a, b]$, we identify the time point already closest to satisfaction, $\hat{t} = \arg\min_{t\in[a,b]} \mid x^*_t - c \mid$, and clamp only that point, $x'_{\hat{t}} = \max(x^*_{\hat{t}}, c)$. For the $\mu_1\mathcal{U}_{[a,b]}\mu_2$ operator, which requires μ_1 to hold up to some time $\hat{t} \in [a, b]$ at which μ_2 holds, we select $\hat{t}$ to minimize the joint projection distance, then enforce μ_1 on $[a, \hat{t}]$ and μ_2 at $\hat{t}$ via the projections above.

For Boolean operators, the conjunction $\varphi_1 \wedge \varphi_2$ requires both subformulas to hold and is projected by sequentially applying the projection of each conjunct, $P_{\varphi_1\wedge\varphi_2}(x^*) = P_{\varphi_2}(P_{\varphi_1}(x^*))$; the disjunction $\varphi_1 \vee \varphi_2$ requires either subformula to hold and is projected toward the cheaper of the two, $P_{\varphi_1\vee\varphi_2}(x^*) = \arg\min_{i\in\{1,2\}} \|P_{\varphi_i}(x^*) - x^*\|_1$. We obtain the final guidance signal $x_{\text{logic}} = P_\varphi(x^*)$ by iteratively performing operator-level projections over the syntax tree of φ.

3.2 Logic-Conditioned Diffusion Reconstruction

Once the specification-faithful projection of the corrupted input time-series signal x^* is obtained, we employ a diffusion reconstruction process to recover $\tilde{x}$. We now describe this process under the discrete-time DDPM formulation.

Forward Diffusion. We start from the attacked input $x_0 = x^*$ and gradually add Gaussian noise according to a β schedule over N steps, where $x_1, \ldots, x_N$ denote the intermediate latent variables. The forward diffusion process is then formulated as $q(x_n \mid x_{n-1}) = \mathcal{N}(x_n; \sqrt{1-\beta_n}\,x_{n-1},\, \beta_n\mathbf{I})$. This is equivalent to the closed-form expression $q(x_n \mid x_0) = \mathcal{N}(x_n; \sqrt{\bar{\alpha}_n}\,x_0,\, (1-\bar{\alpha}_n)\mathbf{I})$, where $\alpha_n = 1 - \beta_n$, and $\bar{\alpha}_n$ is the cumulative product of noise retention at step n, $\bar{\alpha}_n = \prod_{i=1}^{n} \alpha_i$.

Logic-Conditioned Reverse Diffusion. For the reverse diffusion process, standard DDPM defines $p_\theta(x_{n-1} \mid x_n) = \mathcal{N}(x_{n-1}; \mu_\theta(x_n, n), \sigma^2_\theta(x_n, n)\mathbf{I})$. Following this formulation, we incorporate the conditioning signal c_x and obtain $p_\theta(x_{n-1} \mid x_n, c_x) = \mathcal{N}(x_{n-1}; \mu_\theta(x_n, n, c_x), \sigma^2_\theta(x_n, n)\mathbf{I})$. Here, the mean function μ_θ is parameterized through a noise predictor $\epsilon_\theta(x_n, n, c_x)$ and the variance follows the standard DDPM schedule. Rather than iterating this transition over all N steps, we

leverage the one-shot x_0-estimate implicit in the noise-prediction parameterization:

$$\hat{x} = \frac{x_n - \sqrt{1-\bar{\alpha}_n}\,\epsilon_\theta(x_n, n, c_x)}{\sqrt{\bar{\alpha}_n}} = x^* + \frac{\sqrt{1-\bar{\alpha}_n}}{\sqrt{\bar{\alpha}_n}}(\epsilon - \epsilon_\theta).$$

Here, $\hat{x}$ approximates the conditional mean $\mathbb{E}[x_0 \mid x_n, c_x]$. To further encourage the repaired sequence to be both data-plausible and specification-consistent, we form the input-side repair as $\tilde{x} = \alpha\,\hat{x} + (1-\alpha)\,x_{\text{logic}}$, where $\alpha \in (0,1)$ is a learnable weight.

Logic-Conditioned Diffusion Refinement on Outputs. The repaired input $\tilde{x}$ is passed to the forecasting module F_ω to produce an initial forecast $\hat{y}_{\text{init}} = F_\omega(\tilde{x})$. To refine $\hat{y}_{\text{init}}$ and account for residual errors, we then apply a second logic-conditioned diffusion on $\hat{y}_{\text{init}}$.

Mirroring the first diffusion behavior, we project $\hat{y}_{\text{init}}$ onto the φ'-feasible set to obtain y_{logic}, and form the conditioning signal c_y. The forward process noises $\hat{y}_{\text{init}}$ over M steps following the same DDPM schedule as above. We use a separate noise predictor ϵ'_θ and conditioning encoder for the output stage. The conditioned reverse process yields a one-shot estimate of the underlying ground-truth forecast,

$$\hat{y} = \frac{y_m - \sqrt{1-\bar{\alpha}_m}\,\epsilon'_\theta(y_m, m, c_y)}{\sqrt{\bar{\alpha}_m}}.$$

We $\hat{y}$ combine with y_{logic} to produce the final refined forecast $\tilde{y} = \alpha'\,\hat{y} + (1-\alpha')\,y_{\text{logic}}$, where $\alpha' \in (0,1)$ is a learnable weight.

3.3 End-to-End Training and Inference

We train the bi-stage pipeline end-to-end with a joint objective that directly instantiates the defense goal in Section 2. At the input stage, the objective encourages the repaired trajectory $\tilde{x}$ to remain close to the clean signal x while achieving high robustness with respect to the input-side STL specification φ. At the output stage, it encourages the final forecast $\tilde{y}$ to remain accurate relative to the target forecast and to achieve high robustness with respect to the output-side STL specification φ'.

Each desideratum is realized through explicit loss terms. We supervise the induced clean estimates $\hat{x}$ and $\hat{y}$ via the reparameterization trick in [20], which maps predicted noise to the corresponding x_0-estimate. In practice, the reverse models are still parameterized through noise predictors ϵ_θ and $\epsilon_{\theta'}$. On the input side, a reconstruction loss $\mathcal{L}^x_{\text{diff}} = \mathbb{E}\|x - \hat{x}\|_2^2$ pulls the diffusion estimate toward the clean signal, while a smoothed STL penalty $\mathcal{L}^x_{\text{stl}} = \mathbb{E}\left[\text{softplus}(-\tilde{\rho}_\varphi(\hat{x}, 0))\right]$ discourages residual specification violations. On the output side, a reconstruction loss $\mathcal{L}^y_{\text{diff}} = \mathbb{E}\|y - \hat{y}\|_2^2$ aligns the diffusion estimate with the ground-truth forecast, a smoothed STL penalty $\mathcal{L}^y_{\text{stl}} = \mathbb{E}\left[\text{softplus}(-\tilde{\rho}_{\varphi'}(\hat{y}, 0))\right]$ encourages output-side feasibility, and a forecasting loss $\mathcal{L}_{\text{pred}} = \mathbb{E}\|y - \tilde{y}\|_2^2$ ensures that the final blended forecast remains accurate. Here, $\tilde{\rho}$ denotes the smooth surrogate of STL robustness [26], and the softplus function provides stable gradients. The full training objective is given by $\mathcal{L}_{\text{total}} = \lambda_x \mathcal{L}^x_{\text{diff}} + \lambda_y \mathcal{L}^y_{\text{diff}} + \lambda_p \mathcal{L}_{\text{pred}} + \lambda^x_\rho \mathcal{L}^x_{\text{stl}} + \lambda^y_\rho \mathcal{L}^y_{\text{stl}}$, where the coefficients $\lambda_{(\cdot)}$ balance reconstruction quality, forecasting accuracy, and specification compliance across the two stages.

4 Evaluation

4.1 Experimental Setup

Datasets. We evaluate LOGIC-DIFF on two CPS domains. Our primary dataset is a *smart city* dataset constructed from the Caltrans PeMS system [8] for the Los Angeles region. It contains hourly traffic features, including total flow, average occupancy, and average speed, augmented with hourly air-quality and emissions measurements (NO_x, NO_2, ozone, PMx) from the California Air Resources Board [6]. The data span 2024-2025 and contain d=7 features. Each sample uses m=24 hours of history to forecast the next H=6 hours. We also consider the San Francisco region of PeMS to compare with the benchmark used by [30] for sparse indirect attacks. Our second domain is *UAV flight telemetry* from the ALFA dataset [24], which contains 47 fixed-wing flights with real mid-flight engine and actuator faults. We use seven channels that capture the closed-loop dynamics.

Table 2: *Robust* performance under *physical* attacks on the smart city dataset. MSE and STL Sat% reported as $\text{mean}_{\pm 95\%\ \text{CI}}$.

	Dropout		FDI		Replay		Mixed	
Method	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑
Default	$0.469_{\pm.017}$	$77.3_{\pm 3.7}$	$0.197_{\pm.008}$	$84.2_{\pm 2.2}$	$0.538_{\pm.021}$	$80.5_{\pm 2.8}$	$0.283_{\pm.017}$	$84.5_{\pm 2.2}$
Inherent (STL)	$0.470_{\pm.017}$	$77.2_{\pm 3.5}$	$0.207_{\pm.009}$	$84.3_{\pm 2.2}$	$0.551_{\pm.023}$	$79.6_{\pm 2.6}$	$0.295_{\pm.019}$	$84.4_{\pm 2.0}$
AT (Madry)	$0.484_{\pm.015}$	$78.2_{\pm 3.5}$	$0.175_{\pm.007}$	$86.1_{\pm 2.4}$	$0.459_{\pm.018}$	$80.7_{\pm 2.9}$	$0.281_{\pm.016}$	$84.6_{\pm 2.5}$
AT (TRADES)	$0.489_{\pm.016}$	$78.7_{\pm 3.7}$	$0.205_{\pm.008}$	$85.1_{\pm 2.6}$	$0.438_{\pm.016}$	$80.8_{\pm 3.2}$	$0.287_{\pm.013}$	$83.7_{\pm 2.7}$
AP (DiffPure)	$0.473_{\pm.017}$	$77.4_{\pm 3.7}$	$0.186_{\pm.008}$	$84.5_{\pm 2.2}$	$0.525_{\pm.021}$	$80.5_{\pm 2.8}$	$0.276_{\pm.017}$	$84.3_{\pm 2.3}$
RS (Cohen)	$0.468_{\pm.017}$	$77.4_{\pm 3.7}$	$0.201_{\pm.008}$	$84.3_{\pm 2.3}$	$0.529_{\pm.021}$	$80.9_{\pm 2.8}$	$0.290_{\pm.018}$	$84.7_{\pm 2.2}$
LogiC-Diff	$\mathbf{0.246_{\pm.010}}$	$\mathbf{84.8_{\pm 2.9}}$	$\mathbf{0.162_{\pm.007}}$	$\mathbf{87.9_{\pm 2.2}}$	$\mathbf{0.207_{\pm.009}}$	$\mathbf{88.5_{\pm 2.3}}$	$\mathbf{0.187_{\pm.008}}$	$\mathbf{87.6_{\pm 2.2}}$

Table 3: *Robust* performance under gradient-based white-box *cyber* attacks (ℓ_2, ε=1.0) on the smart city dataset. MSE reported as $\text{mean}_{\pm 95\%\ \text{CI}}$; lower is better. Columns right of the rule apply the STL projection P_φ at both stages, alone or after LOGIC-DIFF.

Attack	Default	Inherent (STL)	AT (Madry)	AT (TRADES)	AP (DiffPure)	RS (Cohen)	**LogiC-Diff**	Pre-hoc & Post-hoc	**LogiC-Diff -Post-hoc**
PGD	$0.324_{\pm.011}$	$0.316_{\pm.012}$	$0.226_{\pm.009}$	$0.244_{\pm.009}$	$0.301_{\pm.011}$	$0.311_{\pm.011}$	$\mathbf{0.197_{\pm.008}}$	$0.221_{\pm.011}$	$\mathbf{0.173_{\pm.008}}$
MIM	$0.320_{\pm.011}$	$0.312_{\pm.012}$	$0.225_{\pm.009}$	$0.244_{\pm.009}$	$0.297_{\pm.011}$	$0.307_{\pm.011}$	$\mathbf{0.197_{\pm.008}}$	$0.218_{\pm.011}$	$\mathbf{0.173_{\pm.008}}$
CW	$0.290_{\pm.012}$	$0.283_{\pm.013}$	$0.186_{\pm.009}$	$0.210_{\pm.008}$	$0.265_{\pm.012}$	$0.278_{\pm.012}$	$\mathbf{0.179_{\pm.008}}$	$0.200_{\pm.011}$	$\mathbf{0.162_{\pm.008}}$
APGD	$0.325_{\pm.011}$	$0.317_{\pm.012}$	$0.226_{\pm.009}$	$0.244_{\pm.009}$	$0.300_{\pm.011}$	$0.311_{\pm.011}$	$\mathbf{0.197_{\pm.009}}$	$0.224_{\pm.011}$	$\mathbf{0.174_{\pm.008}}$

Forecaster and metrics. We use a multivariate time-series Transformer [27] as the forecasting backbone. We report two performance metrics, including mean squared error (MSE) over the forecast horizon and Sat%, the average percentage of logic specifications satisfied by each prediction. The Sat% formula and the complete specification set are provided in App. B.

Baselines. We compare LOGIC-DIFF with the undefended forecaster (**Default**) [49], logic-regularized **Inherent (STL)** [31], adversarial training (**AT-Madry** [33] and **AT-TRADES** [58]), adversarial purification (**AP-DiffPure**) [39], and **randomized smoothing (RS)** [11, 29]. Three *projection-only* variants apply P_φ before inference, after inference, or at both stages, without diffusion-based repair or logic conditioning.

Threat model and attacks. At the physical layer, we consider attacks that corrupt sensor streams without access to the forecasting model. We consider **Dropout** (ε=5), **False Data Injection** (FDI, ε=1), **Replay** (ε=0.5), and **Mixed** (ε=2), with attack-specific scalar budgets. At the cyber layer, we evaluate PGD [33], MIM [15], CW [7], and APGD [12] under an ℓ_2 budget of ε=1.0, using MSE as the objective and BPDA [3] through the projection P_φ. We further consider defense-aware **Proj.-PGD** and **BPDA/EOT**, the query-only **Square Attack** [2], and sparse indirect attacks [30] that perturb correlated series instead of the target series. Attack definitions, hyperparameters, and query accounting are provided in App. D.

Clean performance. As a reference, we first compare clean forecasting performance. Default achieves $0.154_{\pm.006}$ MSE and $87.5_{\pm 2.1}$% Sat; Inherent (STL) achieves $0.164_{\pm.006}$ / $88.1_{\pm 1.9}$; AT (Madry) $0.172_{\pm.006}$ / $87.1_{\pm 2.4}$; AT (TRADES) $0.200_{\pm.007}$ / $85.3_{\pm 2.6}$; AP (DiffPure) $0.157_{\pm.006}$ / $87.3_{\pm 2.1}$; RS $0.153_{\pm.006}$ / $87.7_{\pm 2.1}$; and LOGIC-DIFF $0.155_{\pm.006}$ / $88.1_{\pm.2}$. Thus, LOGIC-DIFF preserves clean forecasting accuracy while tying for the highest clean Sat%. In contrast, both adversarial-training baselines incur some loss in clean forecasting accuracy. We next evaluate how these methods behave under attack.

4.2 Robustness Across Various CPS Attack Models

Physical-layer attacks. We first evaluate robustness to four sensor-level attacks in Table 2. LOGIC-DIFF achieves the lowest MSE and highest Sat% across all four settings. Compared with adversarial training, our method reduces MSE from 0.484 to 0.246 under Dropout and from 0.438 to 0.207 under Replay. It also achieves lower MSE under FDI, where adversarial training already performs well. The strength of our approach also persists for Mixed attacks, where the corruption type varies across

Table 4: *Robust* MSE ($\text{mean}_{\pm 95\%\ \text{CI}}$) under adaptive (Proj.-PGD, BPDA/EOT), query-based (Square Attack, 1000 proposal steps), and sparse indirect [30] *cyber* attacks (ℓ_2, ε=1.0) on the smart city dataset. Sparse columns perturb κ correlated series and report MSE on the target series.

Method	Proj.-PGD	BPDA/EOT	Square@1000	Sparse [30] (κ=2)	Sparse [30] (κ=4)
Default	$0.3218_{\pm.011}$	$0.3551_{\pm.025}$	$0.2613_{\pm.0102}$	$0.2523_{\pm.016}$	$0.2757_{\pm.017}$
AT (Madry)	$0.2393_{\pm.009}$	$0.2913_{\pm.019}$	$0.2103_{\pm.0084}$	$0.1738_{\pm.012}$	$0.1830_{\pm.012}$
AT (TRADES)	$0.2304_{\pm.009}$	$0.2817_{\pm.018}$	$0.2092_{\pm.0082}$	$0.1737_{\pm.011}$	$0.1795_{\pm.011}$
AP (DiffPure)	$0.2975_{\pm.011}$	$0.3388_{\pm.024}$	$0.2232_{\pm.0095}$	$0.2341_{\pm.016}$	$0.2516_{\pm.017}$
RS (Cohen)	$0.3092_{\pm.011}$	$0.3468_{\pm.024}$	$0.2342_{\pm.0094}$	$0.2381_{\pm.016}$	$0.2575_{\pm.016}$
LogiC-Diff	$\mathbf{0.2005_{\pm.008}}$	$\mathbf{0.2375_{\pm.017}}$	$\mathbf{0.1833_{\pm.0079}}$	$\mathbf{0.1217_{\pm.008}}$	$\mathbf{0.1287_{\pm.008}}$

samples. This is particularly helpful for temporally structured attacks, where individual sensor values may appear plausible while the overall trajectory violates expected temporal relationships.

Cyber-layer attacks. We evaluate ℓ_2-bounded white-box attacks at ε=1.0 in Table 3. The main result is that LogiC-Diff consistently improves robustness over adversarial training across different gradient-based attack objectives, while logic regularization or diffusion purification alone provides less robustness. Compared with the strongest baseline, AT-Madry, LogiC-Diff reduces MSE from 0.225-0.226 to 0.197 under PGD, MIM, and APGD, and from 0.186 to 0.179 under CW. LogiC-Diff also maintains overall higher specification satisfaction than AT-Madry (86.9% vs. 85.3% under PGD) and reaches 100% Sat with post-hoc projection (Table 7 in Appendix).

Projection alone is less effective. Table 3 also shows that directly enforcing STL feasibility is not sufficient to obtain the robustness of LogiC-Diff. Under PGD, applying P_φ before inference, after inference, or at both stages yields 0.277, 0.225, and 0.221 MSE, respectively (Table 7). While post-hoc projection can improve over the undefended model, its MSE remains above LogiC-Diff under all four gradient-based attacks. Applying the same post-hoc projection after LogiC-Diff performs best, with a 0.173 MSE under PGD.

Adaptive, query-based, and sparse indirect attacks. We further evaluate stronger attackers that explicitly account for the defense (Table 4). LogiC-Diff remains the strongest method under **Proj.-PGD**, achieving 0.2005 MSE compared with 0.2304 for the best baseline. LogiC-Diff also retains the lowest MSE under BPDA/EOT and Square Attack. Under Square Attack, it remains best across all query budgets. For sparse indirect attacks [30], LogiC-Diff achieves the lowest target error in all settings, across both deterministic and probabilistic variants and all sparsity levels, improving over the strongest baseline by 28-30% on our dataset and by 11-12% on the original benchmark.

4.3 Generalizability Across Attack Severity, Access, and CPS Domains

Increasing attack strength. To assess how stronger attacks affect prediction error and how well our approach mitigates their impact, we increase the perturbation budget while keeping the attack type and dataset fixed. Figure 2 shows PGD with $\varepsilon \in \{1.5, 1.75, 2, 2.25\}$ and Mixed physical attacks with $\varepsilon \in \{1.5, 1.75, 2.25, 2.5\}$. LogiC-Diff achieves the lowest median MSE in all eight settings, outperforming the strongest baseline by 7.3%-28.8%. More importantly, our advantage persists as the attack becomes stronger. We observe that the margin becomes 14.0% at the largest PGD budget and increases from 14.3% to 28.8% across the Mixed attacks. The CDFs show the same pattern across the test distribution, indicating that the improvement is not driven by a small subset of samples.

Unseen attacks. We test cyber models trained on PGD or MIM with ℓ_∞ perturbations against the other attack with ℓ_2 perturbations, while physical models transfer between Dropout and Replay. LogiC-Diff reduces MSE by 52.7% under Replay and 44.7% under Dropout relative to the best AT baseline, whereas AT on the mismatched physical attack performs worse than the undefended model.
Cross-domain transfer. On ALFA UAV, we use the same pipeline and rule families, with thresholds re-synthesized from clean flight data (Table 5b). Despite changes in signals, dynamics, and data scale, LogiC-Diff achieves the lowest MSE in six of eight attacked settings. The largest gains occur under temporally structured corruption, with Replay MSE decreasing from 1.139 to 0.332.

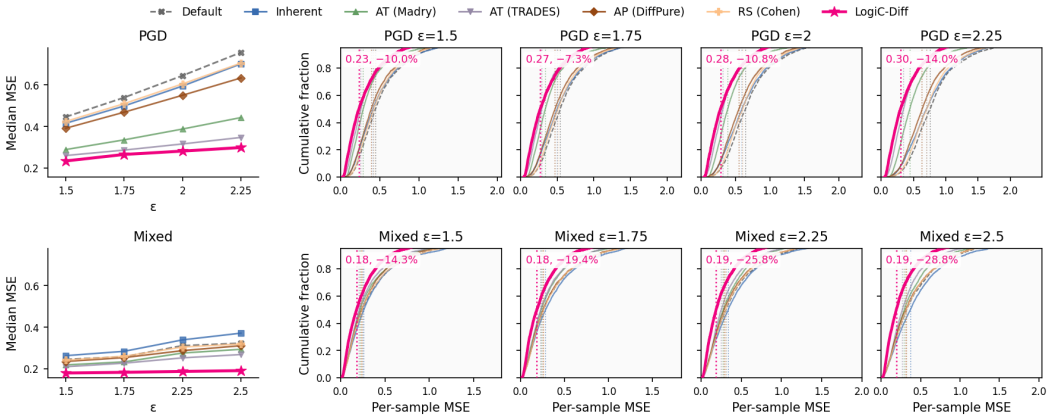

Figure 2: LogiC-Diff achieves the lowest median MSE across all attack budgets and types on the smart city dataset. *Left*: median per-sample MSE versus attack budget ε for PGD and Mixed attacks. *Right*: cumulative distributions of per-sample MSE at four budgets per attack family. Annotations show LogiC-Diff's median MSE and the percentage reduction relative to the best-performing baseline at each budget. Vertical dashed lines mark per-method medians.

Table 5: Generalization across attacks and domains.

(a) *Cross-attack generalization* on the Caltrans PeMS smart-city dataset. Each method is adversarially trained on the opposite attack within the same category. For cyber attacks, the training budget is ℓ_∞ with $\varepsilon = 0.1$; for physical attacks, we use the same budgets as in Tables 2 and 3.

	MIM-ℓ_2		PGD-ℓ_2		Replay		Dropout	
Method	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑
AT (Madry)	$0.245_{\pm.009}$	$84.4_{\pm2.6}$	$0.249_{\pm.009}$	$85.0_{\pm2.5}$	$0.582_{\pm.021}$	$77.0_{\pm3.1}$	$0.499_{\pm.019}$	$76.9_{\pm4.0}$
AT (TRADES)	$0.232_{\pm.009}$	$85.1_{\pm2.5}$	–	–	$0.646_{\pm.027}$	$78.6_{\pm2.8}$	$0.550_{\pm.020}$	$76.5_{\pm3.8}$
LogiC-Diff	$\mathbf{0.213}_{\pm.009}$	$\mathbf{87.3}_{\pm2.3}$	$\mathbf{0.205}_{\pm.009}$	$\mathbf{86.5}_{\pm2.4}$	$\mathbf{0.275}_{\pm.014}$	$\mathbf{87.2}_{\pm2.2}$	$\mathbf{0.276}_{\pm.014}$	$\mathbf{85.4}_{\pm2.9}$

(b) *Cross-domain transfer*. *Robust* MSE on the ALFA UAV dataset across physical and cyber attacks, reported as mean ± 95% CI; lower is better.

	Physical Attacks				*Cyber Attacks*			
Method	**Dropout**	**FDI**	**Replay**	**Mixed**	**PGD**	**MIM**	**CW**	**APGD**
Default	$0.3036_{\pm.0271}$	$0.2351_{\pm.0136}$	$1.2283_{\pm.0909}$	$0.6246_{\pm.0594}$	$0.5036_{\pm.0219}$	$0.4876_{\pm.0221}$	$0.3476_{\pm.0138}$	$0.5350_{\pm.0217}$
Inherent (STL)	$0.2958_{\pm.0309}$	$0.2523_{\pm.0137}$	$1.2774_{\pm.0916}$	$0.7305_{\pm.0698}$	$0.5086_{\pm.0202}$	$0.4867_{\pm.0203}$	$0.3496_{\pm.0137}$	$0.5362_{\pm.0196}$
AT (Madry)	$0.3663_{\pm.0334}$	$\mathbf{0.1549}_{\pm.0107}$	$1.2051_{\pm.0871}$	$0.5780_{\pm.0590}$	$0.2554_{\pm.0147}$	$0.2534_{\pm.0147}$	$\mathbf{0.2040}_{\pm.0142}$	$0.2560_{\pm.0147}$
AT (TRADES)	$0.4304_{\pm.0312}$	$0.1727_{\pm.0143}$	$1.1387_{\pm.0822}$	$0.5542_{\pm.0562}$	$0.2637_{\pm.0172}$	$0.2627_{\pm.0172}$	$0.2213_{\pm.0175}$	$0.2639_{\pm.0172}$
AP (DiffPure)	$0.3457_{\pm.0308}$	$0.2140_{\pm.0132}$	$1.2045_{\pm.0886}$	$0.5890_{\pm.0550}$	$0.4358_{\pm.0202}$	$0.4203_{\pm.0205}$	$0.3111_{\pm.0152}$	$0.4525_{\pm.0203}$
RS (Cohen)	$0.3276_{\pm.0298}$	$0.2387_{\pm.0142}$	$1.2135_{\pm.0898}$	$0.6595_{\pm.0681}$	$0.4899_{\pm.0206}$	$0.4740_{\pm.0212}$	$0.3462_{\pm.0139}$	$0.5149_{\pm.0206}$
LogiC-Diff	$\mathbf{0.1795}_{\pm.0175}$	$0.1613_{\pm.0165}$	$\mathbf{0.3324}_{\pm.0266}$	$\mathbf{0.2197}_{\pm.0187}$	$\mathbf{0.2248}_{\pm.0191}$	$\mathbf{0.2252}_{\pm.0191}$	$0.2063_{\pm.0187}$	$\mathbf{0.2243}_{\pm.0191}$

4.4 Understanding the Sources of Robustness

We examine which parts of LogiC-Diff contribute most to its robustness and how sensitive the framework is to specification quality. Table 6 reports the ablations on the smart city dataset.

Pipeline components. Input-side repair plays a significant role in the overall model robustness. We find that removing it increases Dropout MSE from 0.246 to 0.496, Mixed MSE by 25%, and PGD MSE by 27%. Output-stage rule conditioning also provides essential and consistent improvements, while the STL loss improves forecasting accuracy and specification satisfaction. **Specification families.** Removing `Stability` degrades performance across all attacks, while removing relational rules such as `Associativity` impacts PGD and CW. Single-variable constraints help detect implausible individual trajectories, while relational constraints help when multiple signals are perturbed jointly or indirectly. **Specification quality.** Widening bounds or removing a moderate fraction of rules has less effect, whereas overly tight, mismatched, or contradictory specifications will noticeably

Table 6: Component and logic-rule ablation of LogiC-Diff under Dropout, Mixed, PGD, and CW attacks on the smart-city dataset. Each variant removes one mechanism or rule family from the full pipeline; metrics are reported as mean ± 95% CI.

	Dropout		**Mixed**		**PGD**		**CW**	
Variant	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑
-Rule (Input)	$0.496_{\pm.020}$	$79.7_{\pm3.2}$	$0.234_{\pm.011}$	$86.5_{\pm2.2}$	$0.250_{\pm.010}$	$85.8_{\pm2.6}$	$0.210_{\pm.010}$	$87.3_{\pm2.4}$
-Rule (Output)	$0.250_{\pm.010}$	$84.9_{\pm3.0}$	$0.194_{\pm.008}$	$87.2_{\pm2.2}$	$0.203_{\pm.008}$	$86.3_{\pm2.5}$	$0.185_{\pm.008}$	$87.0_{\pm2.4}$
-Rule Loss $\mathcal{L}_{\text{stl}}$	$0.260_{\pm.010}$	$84.6_{\pm3.1}$	$0.203_{\pm.008}$	$87.2_{\pm2.3}$	$0.219_{\pm.008}$	$86.4_{\pm2.5}$	$0.197_{\pm.008}$	$87.3_{\pm2.4}$
-`Stability`	$0.269_{\pm.012}$	$85.1_{\pm3.0}$	$0.205_{\pm.008}$	$87.7_{\pm2.2}$	$0.224_{\pm.009}$	$87.1_{\pm2.3}$	$0.201_{\pm.009}$	$87.8_{\pm2.2}$
-`Associativity`	$0.248_{\pm.010}$	$86.0_{\pm3.1}$	$0.193_{\pm.008}$	$88.0_{\pm2.3}$	$0.229_{\pm.008}$	$87.2_{\pm2.6}$	$0.211_{\pm.008}$	$87.7_{\pm2.5}$
Full LogiC-Diff	$\mathbf{0.246_{\pm.010}}$	$84.8_{\pm2.9}$	$\mathbf{0.187_{\pm.008}}$	$87.6_{\pm2.2}$	$\mathbf{0.197_{\pm.008}}$	$86.9_{\pm2.4}$	$\mathbf{0.179_{\pm.008}}$	$87.6_{\pm2.3}$

degrade performance. Limited calibration data also hurts, but performance improves quickly with more samples. Thus, we find missing guidance is less harmful than misleading guidance.

Limitations. First, our evaluation is limited to two important attack families. However, additional threat models, such as training time data poisoning or model extraction would provide more comprehensive security guarantees. Second, STL specifications used by our framework are learned from clean training data and reflect domain assumptions. However, miscalibrated or incomplete specifications could systematically bias predictions in ways that propagate to downstream control.

5 Related Work

Adversarial Robustness for Forecasting and CPS. Prior research has shown that attacks can occur across the CPS sensing, communication, and learning pipeline [30, 53, 45, 57], including sensor tampering [10], surrogate-based attacks [23, 30], and white-box adaptive attacks [7]. Existing defenses span CPS-level methods such as detection, resilient estimation, secure control, and fault-tolerant control [13], as well as ML defenses such as adversarial training [33, 58] and randomized smoothing [11]. However, CPS deployments may face mixtures of physical and cyber attacks, making single-family defenses insufficient [17]. Our framework handles physical-layer sensor attacks and cyber-layer gradient attacks under a unified defense. **Diffusion-Based Adversarial Defenses and Logic-Aware ML.** Generative defenses repair corrupted inputs by moving them toward the clean data distribution, with diffusion models [46, 20] becoming especially effective. Methods such as Denoised Smoothing [41], DiffPure [39], AudioPure [52], and AAOpt [50] use diffusion priors or denoising to improve adversarial robustness, but rely mainly on distributional plausibility under $p(x)$ rather than CPS semantics. Separately, logic-aware methods incorporate prior knowledge through training regularization, output filtering, or verification [43, 54, 31, 36], and in security settings are often used to constrain training, verify outputs, or filter predictions [38, 9]. Our work bridges these directions by using logic as semantic guidance within diffusion-based repair, steering reconstruction toward trajectories that are both data-plausible and consistent with CPS security specifications.

6 Conclusion

We propose a new bi-stage, logic-guided, diffusion-based framework for preventatively enhancing the security of CPS forecasting by embedding formal specifications directly into the prediction pipeline. We identify a broad set of security properties and show how they can be incorporated both before forecasting, through input-side repair, and after forecasting, through output-side refinement. Across more than one hundred specifications in our evaluation, we demonstrate the benefits of the proposed method from multiple perspectives, including forecasting accuracy, specification compliance, robustness under stronger attacks, component and rule ablations, and cross-attack generalization. Overall, the results suggest that the embedded rules provide three main benefits: they improve overall robustness and predictive performance, help the model degrade more gracefully as attack strength increases, and transfer better to unseen attacks. In future work, additional security properties can be formally specified and incorporated within the same framework. Therefore, this work represents an important step toward embedding security properties into CPS forecasting models and suggests that logic-guided repair has strong potential to extend to a broader range of CPS domains.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Science Foundation under grants 2443803 and 2427711. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Ziyan An, Taylor T Johnson, and Meiyi Ma. Formal logic enabled personalized federated learning through property inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10882–10890, 2024.

[2] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *Computer Vision – ECCV 2020*, pages 484–501. Springer, 2020.

[3] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pages 274–283. PMLR, 2018.

[4] Ezio Bartocci, Jyotirmoy Deshmukh, Alexandre Donzé, Georgios Fainekos, Oded Maler, Dejan Ničković, and Sriram Sankaranarayanan. Specification-based monitoring of cyber-physical systems: a survey on theory, tools and applications. In *Lectures on Runtime Verification: Introductory and Advanced Topics*, pages 135–175. Springer, 2018.

[5] Zhipeng Cai, Zuobin Xiong, Honghui Xu, Peng Wang, Wei Li, and Yi Pan. Generative adversarial networks: A survey toward private and secure applications. *ACM Computing Surveys (CSUR)*, 54(6):1–38, 2021.

[6] California Air Resources Board. Air quality and emissions data, n.d. URL `https://ww2.arb.ca.gov/capp/cst/rdi/air-quality-and-emissions-data`. Accessed: 2026-04-24.

[7] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017.

[8] Chao Chen. *Freeway performance measurement system (PeMS)*. University of California, Berkeley, 2002.

[9] Xuan Chen, Shiwei Feng, Zikang Xiong, Shengwei An, Yunshu Mao, Lu Yan, Guanhong Tao, Wenbo Guo, and Xiangyu Zhang. Temporal logic-based multi-vehicle backdoor attacks against offline rl agents in end-to-end autonomous driving. *arXiv preprint arXiv:2509.16950*, 2025.

[10] Zirong Chen, Isaac Li, Haoxiang Zhang, Sarah Preum, John A. Stankovic, and Meiyi Ma. Cityspec with shield: A secure intelligent assistant for requirement formalization. *Pervasive and Mobile Computing*, 92:101802, 2023. ISSN 1574-1192. doi: https://doi.org/10.1016/j.pmcj.2023.101802. URL `https://www.sciencedirect.com/science/article/pii/S1574119223000603`.

[11] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR, 2019.

[12] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR, 2020.

[13] Seyed Mehran Dibaji, Mohammad Pirani, David Bezalel Flamholz, Anuradha M Annaswamy, Karl Henrik Johansson, and Aranya Chakrabortty. A systems and control perspective of cps security. *Annual reviews in control*, 47:394–411, 2019.

[14] Namiot Dmitry, Ilyushin Eugene, and Chizhov Ivan. On a formal verification of machine learning systems. *International Journal of Open Information Technologies*, 10(5):30–34, 2022.

[15] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[16] Alexandre Donzé and Oded Maler. Robust satisfaction of temporal logic over real-valued signals. In *Formal Modeling and Analysis of Timed Systems (FORMATS)*, pages 92–106. Springer, 2010.

[17] Wenli Duo, MengChu Zhou, and Abdullah Abusorrah. A survey of cyber attacks on cyber physical systems: Recent advances and challenges. *IEEE/CAA Journal of Automatica Sinica*, 9 (5):784–800, 2022.

[18] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[19] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. *Advances in neural information processing systems*, 30, 2017.

[20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[21] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420, 2016.

[22] Susmit Jha, Ashish Tiwari, Sanjit A Seshia, Tuhin Sahai, and Natarajan Shankar. Telex: learning signal temporal logic from positive examples using tightness metric. *Formal Methods in System Design*, 54(3):364–387, 2019.

[23] Fazle Karim, Somshubra Majumdar, and Houshang Darabi. Adversarial attacks on time series. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3309–3320, 2020.

[24] Azarakhsh Keipour, Mohammadreza Mousaei, and Sebastian Scherer. Alfa: A dataset for uav fault and anomaly detection. *The International Journal of Robotics Research*, 40(2-3):515–520, 2021.

[25] Ron Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990. doi: 10.1007/BF01995674.

[26] Karen Leung, Nikos Aréchiga, and Marco Pavone. Backpropagation through signal temporal logic specifications: Infusing logical structure into gradient-based methods. *The International Journal of Robotics Research*, 42(6):356–370, 2023.

[27] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32, 2019.

[28] Fan Liu, Hao Liu, and Wenzhao Jiang. Practical adversarial attacks on spatiotemporal traffic forecasting models. *Advances in Neural Information Processing Systems*, 35:19035–19047, 2022.

[29] Linbo Liu, Youngsuk Park, Trong Nghia Hoang, Hilaf Hasson, and Jun Huan. Robust multivariate time-series forecasting: Adversarial attacks and defense mechanisms. *arXiv preprint arXiv:2207.09572*, 2022.

[30] Linbo Liu, Youngsuk Park, Trong Nghia Hoang, Hilaf Hasson, and Luke Huan. Robust multivariate time-series forecasting: Adversarial attacks and defense mechanisms. In *The Eleventh International Conference on Learning Representations*, 2023.

[31] Meiyi Ma, Ji Gao, Lu Feng, and John Stankovic. Stlnet: Signal temporal logic enforced multivariate recurrent neural networks. *Advances in Neural Information Processing Systems*, 33:14604–14614, 2020.

[32] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[33] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[34] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems (FORMATS)*, pages 152–166. Springer, 2004.

[35] Stephen Mell, Favyen Bastani, Steve Zdancewic, and Osbert Bastani. Synthesizing trajectory queries from examples. In *International Conference on Computer Aided Verification*, pages 459–484. Springer, 2023.

[36] Mark Huasong Meng, Guangdong Bai, Sin Gee Teo, Zhe Hou, Yan Xiao, Yun Lin, and Jin Song Dong. Adversarial robustness of deep neural networks: A survey from a formal verification perspective. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[37] Hanan Abdullah Mengash, Lubna A Alharbi, Saud S Alotaibi, Sarab AlMuhaideb, Nadhem Nemri, Mrim M Alnfiai, Radwa Marzouk, Ahmed S Salama, and Mesfer Al Duhayyim. Deep learning enabled intelligent healthcare management system in smart cities environment. *Computers, Materials & Continua*, 74(2), 2023.

[38] Dung Thuy Nguyen, Ziyan An, Taylor T Johnson, Meiyi Ma, and Kevin Leach. Formal logic-guided robust federated learning against poisoning attacks. *arXiv preprint arXiv:2411.03231*, 2024.

[39] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022.

[40] Amir Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57. IEEE, 1977. doi: 10.1109/SFCS.1977.32.

[41] Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Denoised smoothing: A provable defense for pretrained classifiers. *Advances in Neural Information Processing Systems*, 33:21945–21957, 2020.

[42] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[43] Joseph Scott, Maysum Panju, and Vijay Ganesh. Lgml: logic guided machine learning (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13909–13910, 2020.

[44] Seyed Salar Sefati, Bahman Arasteh, Simona Halunga, Octavian Fratu, and Asgarali Bouyer. Meet user's service requirements in smart cities using recurrent neural networks and optimization algorithm. *IEEE Internet of Things Journal*, 10(24):22256–22269, 2023.

[45] Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Benchmarking adversarial attacks and defenses for time-series data. In *International Conference on Neural Information Processing*, pages 544–554. Springer, 2020.

[46] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[47] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[48] Anand Swaroop, Abhishek Singh, Girish Chandra, Shiv Prakash, Sohan Kumar Yadav, Tiansheng Yang, and Rajkumar Singh Rathore. A comprehensive overview of formal methods and deep learning for verification and optimization. In *2024 International Conference on Decision Aid Sciences and Applications (DASA)*, pages 1–6. IEEE, 2024.

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

[50] Daniel Wesego and Pedram Rooshenas. Adversary aware optimization for robust defense. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[51] Shaofei Wu. Spatiotemporal dynamic forecasting and analysis of regional traffic flow in urban road networks using deep learning convolutional neural network. *IEEE transactions on intelligent transportation systems*, 23(2):1607–1615, 2021.

[52] Shutong Wu, Jiongxiao Wang, Wei Ping, Weili Nie, and Chaowei Xiao. Defending against adversarial audio via diffusion model. *arXiv preprint arXiv:2303.01507*, 2023.

[53] Tao Wu, Xuechun Wang, Shaojie Qiao, Xingping Xian, Yanbing Liu, and Liang Zhang. Small perturbations are enough: Adversarial attacks on time series prediction. *Information Sciences*, 587:794–812, 2022.

[54] Yuan Yang, James C Kerce, and Faramarz Fekri. Logicdef: An interpretable defense framework against adversarial examples via inductive scene graph reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8840–8848, 2022.

[55] Zhuolin Yang, Zhikuan Zhao, Boxin Wang, Jiawei Zhang, Linyi Li, Hengzhi Pei, Bojan Karlaš, Ji Liu, Heng Guo, Ce Zhang, et al. Improving certified robustness via statistical learning with logical reasoning. *Advances in Neural Information Processing Systems*, 35:34859–34873, 2022.

[56] Huaxiu Yao, Yiding Liu, Ying Wei, Xianfeng Tang, and Zhenhui Li. Learning from multiple cities: A meta-learning approach for spatial-temporal prediction. In *The world wide web conference*, pages 2181–2191, 2019.

[57] Zhenhua Yu, Hongxia Gao, Xuya Cong, Naiqi Wu, and Houbing Herbert Song. A survey on cyber–physical systems security. *IEEE Internet of Things Journal*, 10(24):21670–21686, 2023.

[58] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR, 2019.

A General Information

Computation resources. All experiments were conducted on a machine equipped with 128 GB of RAM, an AMD Ryzen Threadripper Pro 7975WX CPU, and an NVIDIA RTX 6000 Ada GPU.

Broader Impact. This paper develops methods for improving the security of forecasting models deployed in safety-critical CPS, with direct applications in transportation infrastructure, air-quality monitoring, and autonomous mobility, domains where prediction errors propagate to physical control actions and carry real-world consequences. By embedding formal Signal Temporal Logic specifications directly into a diffusion-based repair pipeline, LOGIC-DIFF reduces the risk of unsafe downstream behavior under both physical sensor faults (e.g., dropout, replay, false data injection) and adaptive cyber attacks, while degrading more gracefully than purely data-driven defenses as attack strength increases. The framework also generalizes substantially better than adversarial training to attacks not seen during training, which is operationally important for deployments where the threat landscape evolves faster than retraining cycles.

We do not foresee direct negative societal impact from this work, as our contributions are defensive in nature. Two indirect risks merit acknowledgment, however. First, like all logic-guided methods, the framework's behavior depends on the correctness and completeness of the specifications it enforces; miscalibrated or biased specifications could systematically distort predictions in ways that propagate to control decisions, especially in subpopulations underrepresented in the training data used to calibrate feasibility bounds. Second, no defense provides absolute guarantees, and over-reliance on any single robustness mechanism in safety-critical deployments is itself a hazard. We encourage operators to combine logic-guided repair with continued monitoring, redundant sensing, and conventional CPS safety mechanisms.

B Signal Temporal Logic Semantics

This section provides additional background on STL, expanding on the inductive definition introduced in Section 2. We give the full qualitative (Boolean) and quantitative (robustness) semantics [16], which together support both monitoring and gradient-based optimization over STL formulas. Throughout this section, x denotes a signal trace, μ an atomic predicate of the form $f(x_t) \geq c$, and φ, φ_1, φ_2 denote STL formulas.

The quantitative semantics yield a real-valued measure of *how strongly* a signal satisfies or violates a formula. As introduced in Section 2, the STL robustness function $\rho_\varphi(x,t) \in \mathbb{R}$ measures the signed margin by which x satisfies or violates φ at time t, with positive values denoting satisfaction and negative values denoting violation. Definition B.1 gives the inductive computation.

Definition B.1 (STL Robustness ρ)**.**

$$
\begin{aligned}
\rho_\mu(x,t) &= f(x_t) - c \\
\rho_{\neg\varphi}(x,t) &= -\rho_\varphi(x,t) \\
\rho_{\varphi_1 \wedge \varphi_2}(x,t) &= \min\big\{\rho_{\varphi_1}(x,t),\ \rho_{\varphi_2}(x,t)\big\} \\
\rho_{\Diamond_{[a,b]}\varphi}(x,t) &= \max_{t' \in [t+a,\, t+b]} \rho_\varphi(x,t') \\
\rho_{\Box_{[a,b]}\varphi}(x,t) &= \min_{t' \in [t+a,\, t+b]} \rho_\varphi(x,t') \\
\rho_{\varphi_1\, \mathcal{U}_{[a,b]}\, \varphi_2}(x,t) &= \max_{t' \in [t+a,\, t+b]} \min\Big\{\rho_{\varphi_2}(x,t'),\ \min_{t'' \in [t,\, t']} \rho_{\varphi_1}(x,t'')\Big\}
\end{aligned}
$$

The robustness ρ_φ is non-differentiable due to the $\min$ and $\max$ operations. For gradient-based optimization, we use the smooth surrogate $\tilde{\rho}_\varphi$, which replaces these with differentiable approximations (e.g., soft-min and soft-max) and is the form used in the STL training losses $\mathcal{L}^x_{\text{stl}}$ and $\mathcal{L}^y_{\text{stl}}$.

C Dataset Preparation

We describe the construction of the forecasting windows used in Section 4.1. All three prepared datasets contain $d = 7$ channels, with an input length of $m = 24$, a forecast horizon of $H = 6$, and

a sliding-window stride of one. The saved arrays have shape (N_s, m, d) for inputs and (N_s, H, d) for targets; the evaluation code transposes them to channel-first tensors. Unless otherwise stated, forecasting errors and attack budgets are measured in standardized feature coordinates.

Smart-City Traffic and Air Quality. The smart-city data combine hourly Caltrans PeMS District 7 measurements with air-quality observations for the LA region over 2024–2025 [8, 6]. The feature order is total flow, average occupancy, average speed, NO_2, NO_x, ozone, and $\mathrm{PM}_{2.5}$. Traffic stations are matched to the nearest air-quality site within 50 km using geographic distance, and records are joined by site and timestamp. Missing traffic values are filled with station-specific hour-of-day means. Missing air-quality values are forward-filled and backward-filled within station, with station-specific hour-of-day means used for remaining missing entries.

UAV Flight Telemetry. The ALFA dataset [24] contains 47 fixed-wing flight sequences. We use GPS groundspeed, relative altitude, measured and commanded roll, measured and commanded pitch, and throttle PWM output. The asynchronous channels are aligned to a 4 Hz grid using the most recent observation at or before each grid time, with a maximum staleness of 1.5 s. Rows with an unavailable or stale channel are removed. Windows are formed from consecutive retained rows; their nominal input and output durations are 6 s and 1.5 s, respectively, although removing invalid rows can introduce temporal gaps. ALFA contains roughly three orders of magnitude fewer training windows than the smart-city dataset; for this setting, we use per-sample input-bound calibration.

Traffic Benchmark for Sparse Indirect Attacks. We use a seven-series adaptation of the traffic dataset used by [30], rather than reproducing their complete forecasting setup. The source contains 862 hourly lane-occupancy series, truncated to $14{,}036$ observations. The selected multivariate sequence is split chronologically into $70\%/15\%/15\%$ segments before windowing. The sparse attack targets the first selected series, `lane_039`, while perturbing a subset of the other six series.

D Attack Implementation and Settings

All attacks operate on input tensors of shape (B, d, m), where B denotes the batch size, d the number of features, and m the input window length. Each attack is parameterized by a scalar ε that controls attack strength; the precise interpretation of ε varies by attack family. All attacks preserve tensor shape and are applied independently per batch. We group the attacks into two families: (i) *black-box physical-layer attacks* that simulate sensor faults and do not require model gradients, and (ii) *white-box gradient-based attacks* that require gradient access to the evaluated forecasting pipeline. For a defended method, the attacked function includes its inference-time defense, with BPDA used for non-differentiable repair operations.

Physical-Layer Attacks. The main physical-attack comparison uses $\varepsilon{=}5$ for Dropout, $\varepsilon{=}1$ for FDI, $\varepsilon{=}0.5$ for Replay, and $\varepsilon{=}2$ for Mixed. At these settings, Dropout zeros a contiguous outage window for each sensor, FDI adds a signed bias of up to $\pm 1\sigma$ at each timestep, and Replay replaces the trailing half of the window with earlier genuine measurements. Mixed samples one of the three attacks for each input at its shared budget.

Gradient-Based Attacks. All four attacks below maximize the per-sample MSE between the forecaster's prediction and the ground-truth horizon, subject to a hard ℓ_p budget $\|\delta\|_p \le \varepsilon$. The target model is held in evaluation mode. Both ℓ_2 and ℓ_∞ variants are provided for PGD, MIM, and APGD; CW is implemented for ℓ_2 only. Unless stated otherwise, perturbations are initialized uniformly at random inside the ε-ball.

The full LogiC-Diff pipeline Φ_θ contains a non-differentiable STL projection operator P_φ. To attack the full pipeline rather than a differentiable surrogate, all gradient-based attacks above are wrapped in the BPDA approximation: the forward pass uses the true P_φ, while the backward pass treats it as the identity. This is the same protocol used to report the robust-MSE results in Table 3.

Projection-Aware PGD. Proj.-PGD alternates loss ascent with input-side logic repair and perturbation-budget projection. Let P_{φ_g} denote the shared attacker-side repair operator, distinct from the per-sample input bounds used by LogiC-Diff. Starting from a random point in the ℓ_2 ball, each

iteration computes

$$u_k = \delta_k + 0.25\varepsilon \frac{\nabla_\delta \mathcal{L}(G(x+\delta_k), y)}{\|\nabla_\delta \mathcal{L}(G(x+\delta_k), y)\|_2}, \qquad \delta_{k+1} = \Pi_\varepsilon\big(P_{\varphi_g}(x+u_k) - x\big).$$

We use $\varepsilon = 1.0$, 20 iterations, one random initialization, and the final iterate.

E Framework Implementation Notes

Diffusion U-Net. Both diffusion stages instantiate the same 1D U-Net (DDPM-style ϵ-predictor). When logic conditioning is enabled, the noisy latent and the conditioning signal c_x (or c_y) are concatenated along the channel dimension before the input projection.

The input-stage U-Net operates on temporal length $m = 24$; the output-stage U-Net operates on horizon length $H = 6$. Both follow the standard DDPM encoder–bottleneck–decoder structure with residual blocks and skip connections at every level. The input-stage U-Net (conditioned, $m = 24$) maps $(B, 14, 24)$ through an initial Conv1D ($14 \to 32$), two downsampling stages ($32 \to 32$ at length 12; $32 \to 64$ at length 6), a residual bottleneck at $(B, 64, 6)$, two upsampling stages with skip connections ($64{+}64 \to 32$; $32{+}32 \to 32$), and a final Conv1D ($32 \to 7$) producing $(B, 7, 24)$. The output-stage U-Net follows the same channel topology applied to the horizon-shaped tensor $(B, 14, 6) \to (B, 7, 6)$.

Diffusion Process. We use a discrete-time DDPM formulation:

- Number of diffusion steps: $N = 10$ (input stage), $M = 10$ (output stage)
- Linear β schedule: $\beta_n \in [10^{-4},\, 2 \times 10^{-2}]$
- Forward: $q(x_n \mid x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_n}\, x_0,\ (1 - \bar{\alpha}_n)\mathbf{I})$
- Inference: a single-step x_0-estimate at a fixed step $n^\star = N/2 = 5$.

The U-Nets are trained with the standard ϵ-prediction objective.

STL-Guided Logic Correction. At each diffusion call, the U-Net produces the one-shot estimate $\hat{x}$ (or $\hat{y}$ on the output side), which is passed through the STL projection operator P_φ (or $P_{\varphi'}$) to produce the logic-feasible signal x_{logic} (or y_{logic}). These projected signals serve as both the conditioning input to the diffusion stage and as the reference trajectory for the final blend $\tilde{x} = \alpha\hat{x} + (1-\alpha)x_{\text{logic}}$. Since P_φ is non-differentiable, gradients are propagated through it via straight-through estimation during adversarial-training inner loops, following the BPDA convention [3].

Sequence-to-Sequence Forecaster. The forecaster F_ω is an encoder–decoder transformer that consumes the repaired input $\tilde{x} \in \mathbb{R}^{B\times m\times d}$ and produces an initial forecast $\hat{y}_{\text{init}} \in \mathbb{R}^{B\times H\times d}$.

- Model dimension: $d_{\text{model}} = 128$
- Encoder layers: 3
- Decoder layers: 3
- Attention heads: 4
- Feed-forward dimension: 512
- Dropout: 0.1

F Security-Oriented Logic Formulas

We formalize security-relevant attack patterns and system consistency constraints using STL, expressed in the same notation as the main-paper templates in Table 1. All formulas are defined over bounded temporal intervals $[t_1, t_2]$ and use the atomic predicate form $x_t^{(j)} \geq c$ from Section 2, where $x_t^{(j)}$ denotes the value of the j-th feature at time t. We use the implication shorthand $p \to q \equiv \neg(p \wedge \neg q)$ for readability; all formulas are equivalent to compositions of operators in

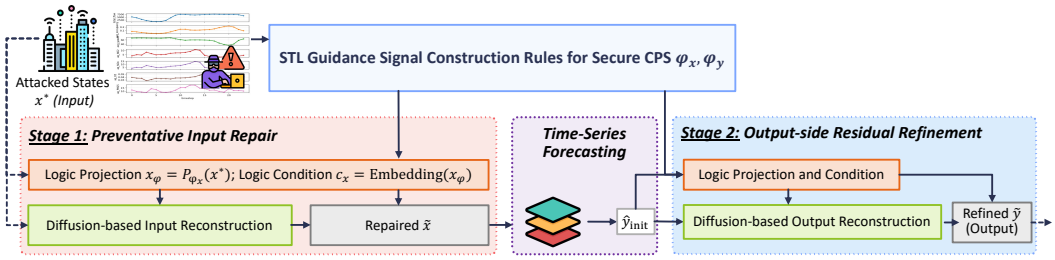

Figure 3: LogiC-Diff at inference time.

our defined grammar. The Sat% metric reported in Section 4.1 is the average percentage of logic specifications satisfied by each prediction: $\text{Sat}\% = \frac{1}{N_s}\sum_{i=1}^{N_s} \mathbf{1}[\rho_i \geq 0] \times 100$, where N_s is the number of specifications and ρ_i is the robustness value of the i-th specification.

Adversarial Behavior Models. The following formulas characterize the attack patterns in our threat model (Section 2.3). They are not enforced as constraints but used to describe and reason about adversarial signal manipulations.

- **Data suppression.** Periodic suppression of feature values over fixed durations: whenever the sampling phase aligns with the suppression schedule, the feature stays below the operational floor c_1 for at least τ time units, formalized as $\square_{[t_1,t_2]}\big((t \equiv 0 (\text{mod } \Delta)) \rightarrow \square_{[0,\tau]}(\neg(x_t^{(j)} \geq c_1))\big)$.
- **Replay attack.** Reuse of stale measurements following a triggering condition on feature $x^{(k)}$: when the trigger fires at time t, the feature value at t matches its value τ steps prior within tolerance c_2, formalized as $\square_{[t_1,t_2]}\big((x_t^{(k)} \geq c_1) \rightarrow \neg(|x_t^{(j)} - x_{t-\tau}^{(j)}| \geq c_2)\big)$.
- **Sensor spoofing.** Substitution of measurements with crafted values $s_t^{(j)}$ when detection confidence on feature $x^{(k)}$ is low: whenever confidence falls below c_1, the feature matches the spoofed signal within c_2, formalized as $\square_{[t_1,t_2]}\big(\neg(x_t^{(k)} \geq c_1) \rightarrow \neg(|x_t^{(j)} - s_t^{(j)}| \geq c_2)\big)$.
- **Backdoor poisoning.** Conditional manipulation activated by a trigger feature p_t: when the trigger fires, the input matches the manipulated signal $\tilde{x}_t^{(j)}$ within c_2, formalized as $\square_{[t_1,t_2]}\big((p_t \geq c_1) \rightarrow \neg(|x_t^{(j)} - \tilde{x}_t^{(j)}| \geq c_2)\big)$.

System Consistency Specifications. The following formulas instantiate the rule families from Table 1 for specific CPS domains and serve as guidance signals during logic-conditioned repair. Each is annotated with the rule family it belongs to.

- **Physical validity** (`Stability`). Feature values remain within feasible physical ranges $[c_1, c_2]$: $\square_{[t_1,t_2]}((x_t^{(j)} \geq c_1) \wedge \neg(x_t^{(j)} \geq c_2))$.
- **Capacity constraint** (`Stability`). Traffic flow cannot exceed physical capacity c_1: $\square_{[t_1,t_2]}(\neg(x_t^{(j)} \geq c_1))$.
- **Queue storage limit** (`Stability`). Queue length stays within storage bound c_1: $\square_{[t_1,t_2]}(\neg(x_t^{(j)} \geq c_1))$.
- **Zone-based speed regulation** (`Stability`, conditional). Speed in zone z respects limit c_2 whenever the vehicle is in-zone: $\square_{[t_1,t_2]}((x_t^{(k)} \geq c_1) \rightarrow \neg(x_t^{(j)} \geq c_2))$.
- **Travel-time progression** (`Smoothness`, first order). Travel time changes between adjacent samples stay within c_1: $\square_{[t_1,t_2]}(\neg(|x_{t+1}^{(j)} - x_t^{(j)}| \geq c_1))$.
- **Temporal smoothness (second order)** (`Smoothness`). Acceleration and trend changes remain bounded: $\square_{[t_1,t_2]}(\neg(|x_{t+1}^{(j)} - 2x_t^{(j)} + x_{t-1}^{(j)}| \geq c_1))$.
- **Cross-sensor consistency** (`Associativity`, value-level). Co-located sensors agree within tolerance c_1: $\square_{[t_1,t_2]}(\neg(|x_t^{(j)} - x_t^{(k)}| \geq c_1))$.

- **Flow conservation** (`Associativity`, value-level). Inflow and outflow stay balanced within c_1: $\square_{[t_1,t_2]}(\neg(|x_t^{(j)} - x_t^{(k)}| \geq c_1))$.
- **Upstream-downstream correlation** (`Associativity`, value-level). Connected segments remain correlated within c_1: $\square_{[t_1,t_2]}(\neg(|x_t^{(u)} - x_t^{(d)}| \geq c_1))$.
- **Post-disturbance recovery** (`Recovery`). After a transient excursion, the feature returns to and remains within a safe band $[c_1, c_2]$ for at least d time units: $\Diamond_{[t_1,t_2]} \square_{[0,d]}((x_t^{(j)} \geq c_1) \wedge \neg(x_t^{(j)} \geq c_2))$.

G Training Configurations and Hyperparameters

We document the training configuration for all evaluated models, covering optimization, schedules, adversarial-training protocols, and early stopping. The settings below are the script defaults used to reproduce all reported results.

Optimizer and Schedule. All models (LogiC-Diff and transformer baselines) are trained with AdamW, initial learning rate 2×10^{-4}, and weight decay 1×10^{-4}. The learning rate is held constant throughout training. **Batch Size and Epochs.** Default batch size is 128 on CUDA-enabled devices and 64 on CPU. The default training budget is 50 epochs on CUDA and 20 on CPU. More training epochs can be used by setting the corresponding training option. **Early Stopping.** We monitor validation MSE and stop training when no improvement is observed for 10 consecutive epochs. The checkpoint with the lowest validation MSE is used for all reported evaluations. **Adversarial Training.** The adversarial mode is selected from $\{\text{data-level}, \text{PGD-AT}, \text{MIM-AT}, \text{TRADES}\}$. Default attack injection probability per step is 1.0 for the gradient-based modes and 0.5 for the data-level mode. Default inner-loop budgets are $\varepsilon = 0.1$ (PGD/MIM, ℓ_∞), 7 inner steps with relative step size 0.3, and TRADES $\beta = 6.0$ with 10 steps. The data-level mode samples from the physical-attack suite described in Section 2 (FDI, replay, dropout, mixed) with the same ε schedule.

Table 7: Full version of Table 3: *robust* performance under gradient-based white-box *cyber* attacks (ℓ_2, ε=1.0). MSE and STL Sat% are reported as $\text{mean}_{\pm 95\%\ \text{CI}}$ for the main baselines; projection-based baselines added during the extended evaluation are reported using point estimates.

	PGD		MIM		CW		APGD	
Method	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑	MSE↓	Sat%↑
Default	$0.324_{\pm.011}$	$83.1_{\pm 2.4}$	$0.320_{\pm.011}$	$82.9_{\pm 2.4}$	$0.290_{\pm.012}$	$84.2_{\pm 2.3}$	$0.325_{\pm.011}$	$83.0_{\pm 2.4}$
Inherent (STL)	$0.316_{\pm.012}$	$83.4_{\pm 2.2}$	$0.312_{\pm.012}$	$83.5_{\pm 2.2}$	$0.283_{\pm.013}$	$84.9_{\pm 2.2}$	$0.317_{\pm.012}$	$83.4_{\pm 2.2}$
AT (Madry)	$0.226_{\pm.009}$	$85.3_{\pm 2.5}$	$0.225_{\pm.009}$	$85.3_{\pm 2.6}$	$0.186_{\pm.009}$	$86.6_{\pm 2.4}$	$0.226_{\pm.009}$	$85.3_{\pm 2.5}$
AT (TRADES)	$0.244_{\pm.009}$	$84.4_{\pm 2.7}$	$0.244_{\pm.009}$	$84.4_{\pm 2.7}$	$0.210_{\pm.008}$	$85.1_{\pm 2.6}$	$0.244_{\pm.009}$	$84.4_{\pm 2.7}$
AP (DiffPure)	$0.301_{\pm.011}$	$83.4_{\pm 2.4}$	$0.297_{\pm.011}$	$83.4_{\pm 2.4}$	$0.265_{\pm.012}$	$85.0_{\pm 2.3}$	$0.300_{\pm.011}$	$83.5_{\pm 2.4}$
RS (Cohen)	$0.311_{\pm.011}$	$83.5_{\pm 2.4}$	$0.307_{\pm.011}$	$83.5_{\pm 2.4}$	$0.278_{\pm.012}$	$84.7_{\pm 2.3}$	$0.311_{\pm.011}$	$83.6_{\pm 2.4}$
LogiC-Diff	$0.197_{\pm.008}$	$\mathbf{86.9_{\pm 2.4}}$	$0.197_{\pm.008}$	$\mathbf{86.9_{\pm 2.4}}$	$0.179_{\pm.008}$	$\mathbf{87.6_{\pm 2.3}}$	$0.197_{\pm.009}$	$\mathbf{86.9_{\pm 2.4}}$
Projection-based Baselines								
Pre-hoc Projection	$0.277_{\pm.010}$	$88.0_{\pm 2.4}$	$0.274_{\pm.010}$	$87.8_{\pm 2.4}$	$0.239_{\pm.010}$	$88.8_{\pm 2.3}$	$0.278_{\pm.010}$	$88.0_{\pm 2.5}$
Post-hoc Projection	$0.225_{\pm.011}$	**100.0**	$0.222_{\pm.011}$	**100.0**	$0.203_{\pm.011}$	**100.0**	$0.228_{\pm.011}$	**100.0**
Pre-hoc & Post-hoc	$0.221_{\pm.011}$	**100.0**	$0.218_{\pm.011}$	**100.0**	$0.200_{\pm.011}$	**100.0**	$0.224_{\pm.011}$	**100.0**
LogiC-Diff-Post-hoc	$\mathbf{0.173_{\pm.008}}$	**100.0**	$\mathbf{0.173_{\pm.008}}$	**100.0**	$\mathbf{0.162_{\pm.008}}$	**100.0**	$\mathbf{0.174_{\pm.008}}$	**100.0**

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: We confirm that the claims made in the abstract and introduction are supported by Sections 4 and 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: A detailed discussion of the limitations and future work is available in Section 4.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.

- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes].

   Justification: Assumptions are provided in Sections 2 and 4 and in Appendix 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: Please refer to Section 4 for the evaluation setup and parameter selection, and to Appendix 6 for additional information. The code is provided in the supplementary materials.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.

- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: All data used are open source and can be accessed through Caltrans PeMS and the California Air Resources Board.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes].

Justification: Please refer to Section 4 for the evaluation setup and parameter selection, and to Appendix 6 for additional information. The code is provided in the supplementary materials.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes].

Justification: All tables and figures in Section 4 report error bars and statistical significance.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes].

Justification: Information about computational resources is provided in the Appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: We confirm we follow NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: While we do not foresee any negative societal impacts, we provide a discussion of the potential broader impacts in the Appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: The paper poses no such risks on the potential misuse of pre-trained language models, image generators, or scraped datasets.

    Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: We confirm existing assets are properly credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A].

    Justification: We combine two existing assets, both of which are properly cited.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A].

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A].

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.