Verifying STL Properties of Sliding-Window Neural Classifiers under Adversarial Perturbations

Xaver Fink[1,2], Borja Fernandez Adiego[1], Joost-Pieter Katoen[2]

[1] CERN, Geneva, Switzerland
[2] RWTH Aachen University, Aachen, Germany

Abstract. Sliding-window neural classifiers process time-series data by producing class-score vectors for successive signal segments, yielding a *classification trace* over time. However, downstream decisions typically depend on *temporal patterns* in this trace. As a result, standard local adversarial robustness is insufficient: *small perturbations to the input signal can induce temporally coherent changes in the classification trace and lead to violations of decision-relevant temporal patterns.* We study robustness with respect to Signal Temporal Logic (STL) properties defined over the resulting classification trace under bounded perturbations of the original time series. Our contributions are twofold. First, we propose an input-space *falsification method* that backpropagates through both the sliding-window classifier and a differentiable quantitative STL semantics, and uses Projected Gradient Descent (PGD) to find perturbations that maximally violate the property. Second, we outline an *abstraction-refinement verification method* that combines neural network output bounds with interval-based STL checking and iteratively refines the trace abstraction to reduce over-approximation.

1 Introduction

In recent years, time series classification via deep neural networks (DNNs) has been successfully applied in domains such as medicine [11], finance [10], or control [3]. At CERN, one such application is a neural-network classifier used in an equipment-parameter-optimization workflow [12,20]. The classifier runs on overlapping windows of a time series and induces a *trace of class scores*. The operational decision is then not based on a single window classification, but on temporal patterns in this trace. This decision process should be automatic in order to optimize CERN resources [1]. For this, safety guarantees are needed.

Prior work has shown that the currently used classifier at CERN is vulnerable to local adversarial perturbations on individual windows [9]. However, this is not a complete analysis: the operational decision depends on the trace over many consecutive window outputs. In that work, we introduced *adversarial sequences*, i.e., perturbations of the underlying time series that remain consistent across overlapping windows. As a first step towards trace-level reasoning, we considered simple smoothness constraints between consecutive class-score vectors, for

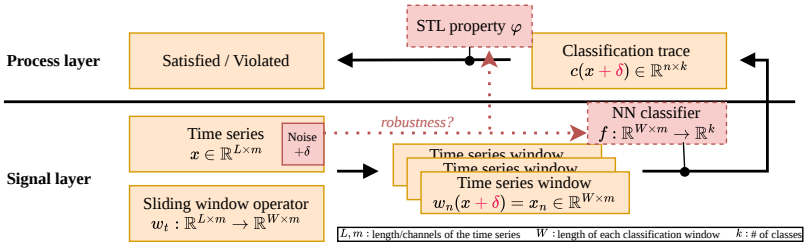

Fig. 1: Pipeline and threat model: time series x is perturbed by $\|\delta\|_p \leq \epsilon$, inducing correlated changes across overlapping windows $w_t(x+\delta)$, and thus on the classification trace $c(x+\delta)$, impacting robustness of the STL property φ.

example requiring that the classifier output does not change abruptly from one window to the next.

The sequence-level property considered there was still local in time: it only constrained adjacent classifier outputs, for example by requiring consecutive class-score vectors to change smoothly. Such pairwise constraints are useful to rule out implausible traces, but they do not express richer decision-level requirements, such as *"eventually detect a sustained high confidence for class a"* or *"never report class a before a minimum confidence pattern has appeared"*. These requirements are naturally temporal and are better expressed as formulas over the whole induced classification trace. For this, Signal Temporal Logic (STL) [16] is a natural formalism. Using quantitative STL semantics [7], we can measure how robustly an induced classification trace satisfies a temporal specification, and we can search for worst-case perturbations of the upstream signal by minimizing STL robustness. Beyond falsification, we consider formal verification of STL properties under bounded noise by tailoring an abstraction-refinement framework [4] to our purpose. In this extended abstract, we formalize our problem setting and give the intuition behind falsification and verification algorithms.

2 Problem Formulation

We consider sliding-window time-series classification, where a neural classifier f induces a discrete-time trace of class scores over overlapping windows. Let $x \in \mathbb{R}^{L \times m}$ be a time series of length L with m channels (e.g., beam loss monitor signals [9]). For $t \in \{0, \ldots, L-W\}$, let

$$w_t(x) := (x_t, x_{t+1}, \ldots, x_{t+W-1}) \in \mathbb{R}^{W \times m}$$

denote the window of length W starting at time t. A DNN classifier $f : \mathbb{R}^{W \times m} \to [0,1]^k$ maps each window to a vector of class scores, typically softmax outputs. This induces the classification trace

$$c(x) = (c_0(x), \ldots, c_{L-W}(x)), \qquad c_t(x) = f(w_t(x)).$$

For each class $j \in \{1, \ldots, k\}$, we write $y_j(t) := c_t(x)[j]$ for the real-valued signal giving the classifier's score for class j at window index t. STL formulas are interpreted over these real-valued score signals: an atomic predicate $y_j(t) \geq \theta$ states that the score of class j exceeds the decision threshold θ at time t. We do not require the classifier scores to be probabilities, and the STL formulas are not probabilistic temporal-logic formulas. The requirement meaning of such predicates comes from the application semantics of the classes and the downstream decision rule. For instance, if class a denotes a target condition, then $\mathbf{F}_{[0,T]}\mathbf{G}_{[0,h]}(y_a \geq \theta)$ expresses eventual sustained high-confidence detection of that condition.

The core question we consider is: *How does noise on the time series x impact the robustness of the DNN classifier f and thus satisfaction of STL formula φ over the downstream classification trace $c(x)$?*

Threat model. As illustrated in Fig. 1, we model noise as an additive perturbation δ on the signal $x : \hat{x} := x+\delta$ such that $\|x-\hat{x}\|_p \leq \epsilon$. This induces a (generally infinite) set of perturbed traces $c(x+\delta) = (f(w_0(x+\delta)), \ldots, f(w_{L-W}(x+\delta)))$. We consider an STL formula φ over the trace components (e.g., sustained high confidence for a target class). The satisfaction problem we study is a combination of the local adversarial robustness problem on neural networks and the STL satisfaction problem.

Definition 1 (ϵ-robust STL satisfaction). *Let $x \in \mathbb{R}^{L\times m}$ be a time series, let $c(x)$ denote the induced classification trace, and let φ be an STL formula over such traces. We say that x is* ϵ-robustly satisfying *φ (with respect to the L_p-norm $\|\cdot\|_p$) if*

$$\forall \hat{x} \in \mathbb{R}^{L\times m}, \quad \|\hat{x} - x\|_p \leq \epsilon \implies c(\hat{x}) \models \varphi.$$

Using quantitative STL semantics $\rho_\varphi(\cdot, t)$ which assigns a real-valued number of quantitative satisfaction (lower is more violating) for trace $\cdot$ on formula φ (evaluated at time step t of the trace), we can express the satisfaction as the search for maximally violating perturbations:

$$R_\varphi(x, \epsilon) = \min_{\|\hat{x}-x\|_p \leq \epsilon} \rho_\varphi(c(\hat{x}), 0).$$

If $R_\varphi(x, \epsilon) \geq 0$, then all admissible perturbations preserve φ; if < 0, a violating perturbation exists. Importantly, we optimize over perturbations of the original time series, not over independent perturbations of the individual windows. Since sliding windows overlap, independent window perturbations may assign different perturbed values to the same original sample. Such a collection of perturbed windows need not be realizable by any single perturbed signal $\hat{x} = x + \delta$. A global perturbation δ, in contrast, induces window perturbations $w_t(\delta)$ that agree on all overlapping samples. By optimizing over one signal-level perturbation δ, overlap-induced consistency is enforced by construction. In the following section, we seek methods to efficiently compute or approximate R_φ via gradient-based optimization (falsification) or abstraction-refinement (verification).

Algorithm 1: Computation of counterexamples via modified PGD

Input: time series x, initial $\delta^{(0)} = 0$, step size $\alpha > 0$, K iterations, bound ϵ, STL formula φ

for $k \leftarrow 1$ **to** K **do**

$\hat{x}^{(k)} \leftarrow x + \delta^{(k-1)}$

Compute windows, $c(\hat{x}^{(k)})$, and robustness $\rho \leftarrow \rho_\varphi(c(\hat{x}^{(k)}), 0)$

$\delta^{(k)} \leftarrow \delta^{(k-1)} - \alpha \cdot (\nabla_\delta \rho)$, and project to domain $\|\delta^{(k)}\|_p \leq \epsilon$

end

Output: Counterexample candidate $\hat{x}^{(k)}$

3 Proposed Approach

3.1 Input-space gradient-based falsification

For falsification of R_φ, we propose an incomplete, gradient-based procedure commonly used in related work ([15,17]) on differentiable STL satisfaction of DNN-controlled systems. In contrast, in our setting we optimize over perturbations of the base time series x, not over direct trace perturbations. This modification is straightforward, since gradients can be propagated through both the robustness term and the windowing operator. This enables (incomplete) counterexample search via projected gradient descent (PGD) as illustrated in Algorithm 1.

3.2 Verification via abstraction-refinement

Our verification goal is to verify that every possible noisy time series results in a trace that satisfies the STL formula. Using the quantitative STL satisfaction semantics this reduces to proving $R_\varphi(x, \epsilon) = \min_{\|\hat{x}-x\|_p \leq \epsilon} \rho_\varphi(c(\hat{x}), 0) \geq 0$.

Gradient-based methods are not suitable for this as they are incomplete and approximate. Instead, we propose an abstraction-refinement loop combining existing methods for DNN reachability and interval-based STL checking, illustrated in Fig. 2. The core principle is the following. First, we construct an over-approximation of all possible perturbed traces $c(\hat{x})$ *(abstraction)*. Concretely, for each window $w_t(x)$, a neural-network output-bound method computes intervals for the relevant class-score signals $y_j(t)$. These intervals form an *interval trace*, on which we evaluate the STL formula using interval robustness semantics. The required bound engine can be instantiated at different levels of precision: a lightweight implementation may use auto-LiRPA-style bound propagation, while tighter but more expensive variants may use CROWN-style optimizations or branch-and-bound. The abstraction-refinement loop is independent of this choice.

Second, if interval checking returns a counterexample candidate, we try to determine whether it is a genuine counterexample or an artifact of the abstraction. Such a violation may be spurious for two reasons: the neural-network bounds may be loose, and the interval trace treats different time points independently even though neighboring windows share most input samples. Hence the abstract

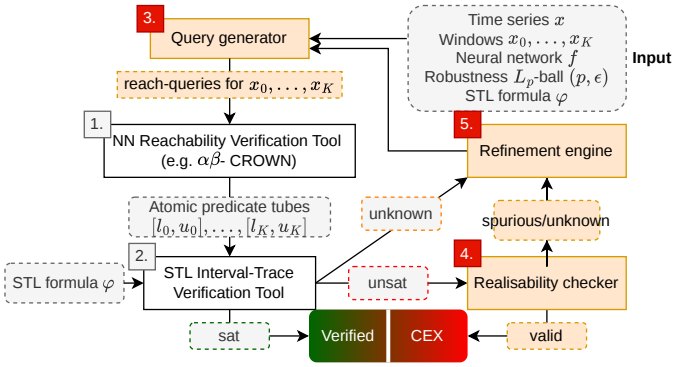

Fig. 2: Verification algorithm, our algorithmic contributions in orange.

trace may combine class-score values that would require inconsistent perturbations on overlapping windows and therefore cannot arise from one shared signal perturbation $\hat{x} = x + \delta$. If no concrete realization is found, the result is inconclusive and the over-approximation is made more precise *(refinement)* by tightening the relevant neural-network bounds or by reintroducing dependencies between time points in the region responsible for the abstract violation.

Counterexample analysis and refinement. If interval STL checking returns an abstract violation, we first ask whether this violation is concretely realizable:

$$\exists \delta : \ \|\delta\|_p \leq \epsilon \quad \text{and} \quad c(x+\delta) \not\models \varphi.$$

This check can be performed incompletely by the PGD falsifier from Section 3.1, using the time region and predicates identified by the interval STL checker to guide the search. If such a perturbation is found, the violation is a genuine counterexample. If no perturbation is found, the result is inconclusive: the violation may be spurious, or the falsifier may simply have failed. In this case, the abstraction is refined.

We consider two complementary refinement mechanisms. First, *input-set splitting* divides the perturbation set $B_\epsilon(x) := \{\hat{x} \mid \|\hat{x} - x\|_p \leq \epsilon\}$ into smaller regions $\mathcal{B}_r$, computes one interval trace per region, and checks all regions separately. This reduces the overapproximation of the neural-network bounds while preserving soundness. Second, *dependency-aware trace refinement* uses the structure of the abstract violation to refine only the relevant part of the trace abstraction. The initial abstraction stores independent intervals

$$y_j(t) \in [\ell_{t,j}, u_{t,j}],$$

and therefore forgets dependencies between values at different time points. When the interval STL checker returns a candidate violation, it also identifies the subformula, time interval, and predicates responsible for the negative robustness

value. Suppose this critical part is a subformula ψ evaluated over $I = [t, t+q]$. We can then issue a refined reachability query for the corresponding windows and predicates. Instead of immediately projecting the result back to independent intervals, we retain dependency information for the relevant values, for example as a relational over-approximation

$$\mathbf{y}_I := (y_{j_1}(t_1), \ldots, y_{j_r}(t_r)) \in \mathcal{R}_I,$$

where $\mathcal{R}_I$ soundly overapproximates the jointly realizable values under admissible perturbations. Unlike the initial interval check, reasoning over $\mathcal{R}_I$ is no longer simple interval monitoring: the local subproblem may require solving a small optimization or constraint query, for example an MILP or SMT query depending on the representation of $\mathcal{R}_I$ and the operators in ψ. The key point is that this expensive relational reasoning is applied only to the critical subformula and time interval. Outside I, the cheaper interval abstraction is left unchanged.

Practicality. The falsification algorithm is directly implementable for differentiable classifiers and differentiable STL robustness surrogates. Its cost is comparable to PGD on the unfolded sliding-window computation graph. The verification loop is more expensive: the initial abstraction requires output-bound queries for the windows in the STL horizon, and refinement increases this cost through input splitting or local relational constraints. For this reason, we view the verification method as a sound but potentially incomplete abstraction-refinement scheme. A systematic empirical scalability study is left for future work.

Soundness (sketch). If each predicate interval over-approximates, the predicate's value over all admissible perturbations at each t, then tube-level STL satisfaction implies concrete satisfaction for all perturbations (by structural induction over φ).

Related work. We relate to (i) differentiable quantitative semantics for temporal logics (STL/MTL) used in gradient-based synthesis and trajectory optimization [19,13], (ii) robustness (quantitative) semantics of MTL/STL over real-valued signals and their use in monitoring [8,7], and (iii) adversarial attacks on neural networks for time-series tasks [2,14,18,5,21,6]. Our setting differs in (a) that our goal goes beyond gradient-based methods towards a sound verification algorithm, and (b) that STL is evaluated on a *downstream classification trace*, while noise is injected on the *upstream signal trace* which enables unique refinement heuristics.

References

1. Benedikt, M., Bartmann, W., Burnet, J.P., Carli, C., Chance, A., Craievich, P., Giovannozzi, M., Grojean, C., Gutleber, J., Hanke, K., Henriques, A., Janot, P., Lourenco, C., Mangano, M., Otto, T., Poole, J.H., Rajagopalan, S., Raubenheimer,

T., Todesco, E., Ulrici, L., Watson, T.P., Wilkinson, G., Zimmermann, F.: Future Circular Collider Feasibility Study Report Volume 2: Accelerators, technical infrastructure and safety. Tech. rep., CERN Document Server (2025). https://doi.org/10.17181/CERN.EBAY.7W4X, http://cds.cern.ch/record/2928793
2. Carlini, N., Wagner, D.A.: Audio Adversarial Examples: Targeted Attacks on Speech-to-Text. In: IEEE Symposium on Security and Privacy Workshops. pp. 1–7. IEEE Computer Society (2018)
3. Churchill, R.M., Tobias, B., Zhu, Y., DIII-D team: Deep convolutional neural networks for multi-scale time-series classification and application to tokamak disruption prediction using raw, high temporal resolution diagnostic data. Physics of Plasmas **27**(6), 062510 (Jun 2020). https://doi.org/10.1063/1.5144458, https://pubs.aip.org/pop/article/27/6/062510/1025325/Deep-convolutional-neural-networks-for-multi-scale
4. Clarke, E.M., Grumberg, O., Jha, S., Lu, Y., Veith, H.: Counterexample-Guided Abstraction Refinement. In: CAV. Lecture Notes in Computer Science, vol. 1855, pp. 154–169. Springer (2000)
5. Ding, D., Zhang, M., Feng, F., Huang, Y., Jiang, E., Yang, M.: Black-Box Adversarial Attack on Time Series Classification. In: AAAI. pp. 7358–7368. AAAI Press (2023)
6. Dix, M., Manca, G., Okafor, K.C., Borrison, R., Kirchheim, K., Sharma, D., R, C.K., Maduskar, D., Ortmeier, F.: Measuring the Robustness of ML Models Against Data Quality Issues in Industrial Time Series Data. In: INDIN. pp. 1–8. IEEE (2023)
7. Donzé, A., Maler, O.: Robust Satisfaction of Temporal Logic over Real-Valued Signals. In: FORMATS. Lecture Notes in Computer Science, vol. 6246, pp. 92–106. Springer (2010)
8. Fainekos, G.E., Pappas, G.J.: Robustness of temporal logic specifications for continuous-time signals. Theor. Comput. Sci. **410**(42), 4262–4291 (2009)
9. Fink, X., Adiego, B.F., Mirarchi, D., Matheson, E., González, G., Ricci, G., Katoen, J.P.: Adversarial robustness of time-series classification for crystal collimator alignment. In: NFM (2026), to appear
10. Fischer, T., Krauss, C.: Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research **270**(2), 654–669 (Oct 2018). https://doi.org/10.1016/j.ejor.2017.11.054, https://linkinghub.elsevier.com/retrieve/pii/S0377221717310652
11. Hannun, A.Y., Rajpurkar, P., Haghpanahi, M., Tison, G.H., Bourn, C., Turakhia, M.P., Ng, A.Y.: Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. Nature Medicine **25**(1), 65–69 (Jan 2019). https://doi.org/10.1038/s41591-018-0268-3, https://www.nature.com/articles/s41591-018-0268-3
12. Holzer, E., Dehning, B., Effinger, E., Emery, J., Ferioli, G., Gonzalez, J., Gschwendtner, E., Guaglio, G., Hodgson, M., Kramer, D., Leitner, R., Ponce, L., Prieto, V., Stockner, M., Zamantzas, C.: Beam loss monitoring system for the LHC. In: IEEE Nuclear Science Symposium Conference Record, 2005. vol. 2, pp. 1052–1056 (Oct 2005). https://doi.org/10.1109/NSSMIC.2005.1596433, https://ieeexplore.ieee.org/document/1596433/
13. Leung, K., Aréchiga, N., Pavone, M.: Backpropagation through signal temporal logic specifications: Infusing logical structure into gradient-based methods. Int. J. Robotics Res. **42**(6), 356–370 (2023)
14. Li, H., Cui, Y., Wang, S., Liu, J., Qin, J., Yang, Y.: Multivariate Financial Time-Series Prediction With Certified Robustness. IEEE Access **8**, 109133–109143 (2020)

15. Liu, W., Mehdipour, N., Belta, C.: Recurrent Neural Network Controllers for Signal Temporal Logic Specifications Subject to Safety Constraints. IEEE Control. Syst. Lett. **6**, 91–96 (2022)
16. Maler, O., Nickovic, D.: Monitoring temporal properties of continuous signals. In: FORMATS/FTRTFT. Lecture Notes in Computer Science, vol. 3253, pp. 152–166. Springer (2004)
17. Meng, Y., Fan, C.: Signal Temporal Logic Neural Predictive Control. IEEE Robotics Autom. Lett. **8**(11), 7719–7726 (2023)
18. Mode, G.R., Hoque, K.A.: Adversarial Examples in Deep Learning for Multivariate Time Series Regression. In: AIPR. pp. 1–10. IEEE (2020)
19. Pant, Y.V., Abbas, H., Mangharam, R.: Smooth operator: Control using the smooth robustness of temporal logic. In: CCTA. pp. 1235–1240. IEEE (2017)
20. Ricci, G., D'Andrea, M., Di Castro, M., Matheson, E., Mirarchi, D., Mostacci, A., Redaelli, S.: Machine learning based crystal collimator alignment optimization. Physical Review Accelerators and Beams **27**(9), 093001 (Sep 2024). https://doi.org/10.1103/PhysRevAccelBeams.27.093001, https://link.aps.org/doi/10.1103/PhysRevAccelBeams.27.093001
21. Wu, T., Wang, X., Qiao, S., Xian, X., Liu, Y., Zhang, L.: Small perturbations are enough: Adversarial attacks on time series prediction. Inf. Sci. **587**, 794–812 (2022)