Machine Learning meets Temporal Logic: an Integrated Framework and its Applications

Andrea Brunello Nicola Saccomanno

UNIVERSITÀ DEGLI STUDI DI UDINE
HIC SUNT FUTURA

MEDITERRANEAN MACHINE LEARNING SUMMER SCHOOL

Symbolic VS Sub-Symbolic AI

Symbolic AI approaches

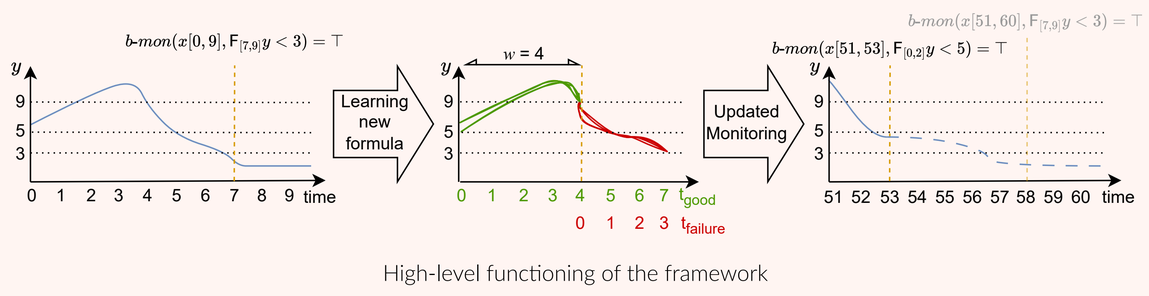
- Reduced dependency on the data
- Often inherently interpretable
- Explicit knowledge representation makes them reliable in critical and controlled scenarios
- Needs human-designed models and properties
- Difficult to apply to multi-modal and/or high-dimensional data
- Typically lacks generalization capabilities

Machine and Deep Learning

- Learns automatically from the data
- Leverages (and needs) high volume of data to learn complex modalities (signals, text, video)
- May generalize on unseen data
- Lacks explainability
- Black-box models are difficult to apply to real-world critical scenarios
- Risk of catastrophic forgetting

Can we leverage the best of both worlds, i.e., learning interpretable properties (logic formulas) from the data to solve complex tasks?

TL; DR;



High-level functioning of the framework

Monitoring (a lightweight run-time verification technique), evaluates whether an incoming system-generated trace results in a **failure** (an event that badly terminates the system execution), by continuously verifying a pool of **temporal logic formulas**. The **formulas are iteratively learned**, in a contrastive fashion, using an **Evolutionary Algorithm** (EA) to solve a genetic programming task. The EA is triggered by either the occurrence of a failure event or the satisfaction of a formula already present in the monitoring pool. In essence, we propose a framework for **highly interpretable failure detection** that also provides a performance comparable to the state-of-the-art approaches.

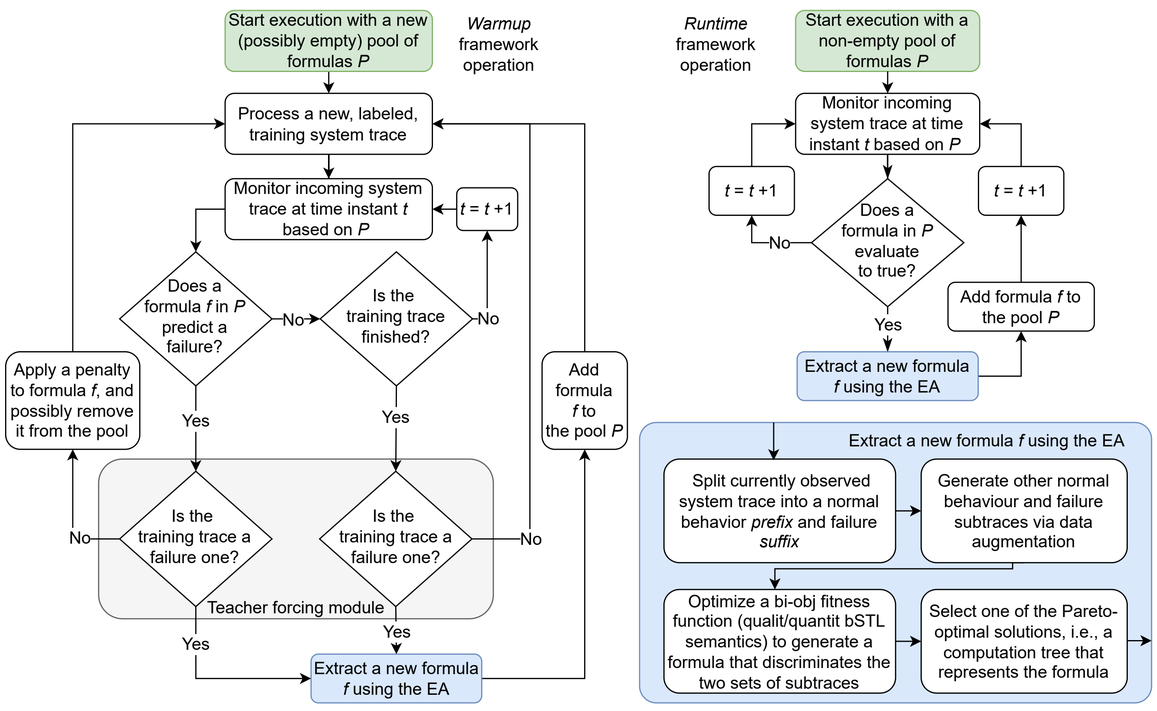
Monitoring of (bounded Signal) Temporal Logic

Monitoring establishes satisfaction or violation of a property by analyzing a finite prefix of a single behavior (*trace/run*) of a system, and then issuing an irrevocable verdict. It is a lightweight technique (no system model required), but the gain in efficiency is paid in terms of expressivity: monitorable properties are a subset of those expressible in temporal logics commonly used for automated verification.

Signal Temporal Logic (STL) extends propositional logic with future modalities that allow one to express temporal properties over linear structures. It can be directly applied to time series data characterized by continuous values. Bounded STL (i.e., bSTL) considers only finite intervals of linear structures. A bSTL formula can be evaluated considering a qualitative (true/false) as well as quantitative (robustness) semantics. The latter provides a measure of how much a linear structure satisfies or violates a formula.

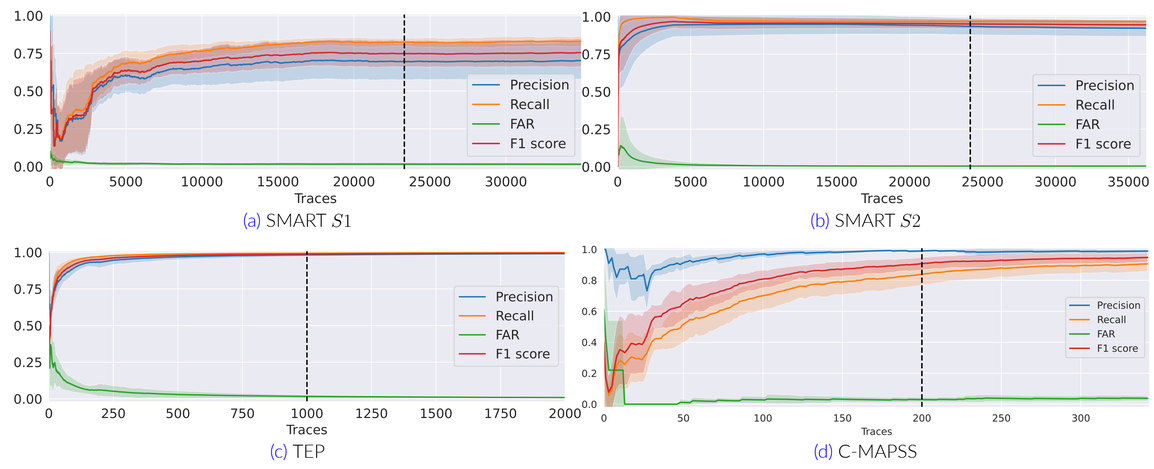
Other than classical boolean connectives ($\vee$, $\wedge$, $\neg$), several temporal operators can be used in bSTL (next, until, eventually, globally, ...). For example, bSTL formula $\phi = \mathsf{F}_{[7,9]} y < 3$ states that, given a trace y, there is a time instant, between instants 7 and 9, in which the value of y is less than 3. Formula bSTL formula $\phi = \mathsf{G}_{[7,9]} y < 3$ states that the same must be true at all time points between 7 and 9.

Framework algorithm intuition



Results for preemptive failure detection

Dataset	Approach	Prec.	Recall	FAR	F1	Dataset	Approach	Prec.	Recall	FAR	F1
SMART S1	Lu et al. '20	0.87	0.41	0.00	0.55	TEP	Hajihosseini et al. '18	1.00	1.00	-	1.00
	Huang '17	0.51	0.54	0.00	0.52		Onel et al. '19	1.00	1.00	0.00	1.00
	our	0.54	0.60	0.00	0.56		our	1.00	1.00	0.00	1.00
SMART S2	Lu et al. '20	0.98	0.87	0.01	0.92	C-MAPSS	Kim and Sohn '20	0.71	1.00	-	0.83
	Su and Li '19	0.91	0.94	0.05	0.93		our	0.96	0.77	0.01	0.86
	our	0.89	0.97	0.00	0.93						



(a) SMART $S1$ (b) SMART $S2$ (c) TEP (d) C-MAPSS

Figure 1. Evaluation metrics (avg and std). The vertical line represents the transition from warmup to runtime phase.

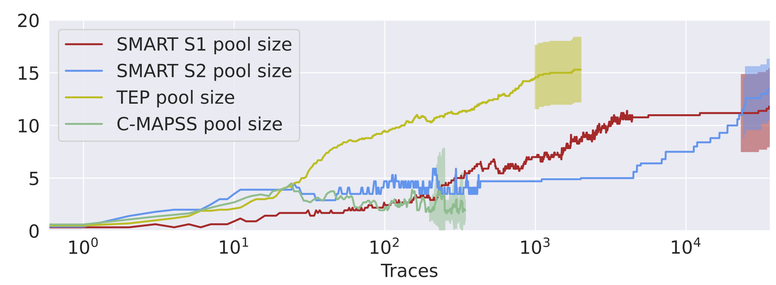


Figure 2. Pool size (avg and std) in both warmup (transparent) and runtime (opaque) phases.

Example of bSTL formulas extracted and used within the framework

- $(\mathsf{G}_{[0,2]} SMART_{194} > 45.6) \wedge (\mathsf{F}_{[2,3]} SMART_{198} > 0.32)$. It signals a bad behavior, which would ultimately lead to a failure, where the hard drive maintains a temperature exceeding 45.6 °C in the first 3 days, and then, in the following 2 days, its *uncorrectable sector count* becomes greater than 0.
- $f_1 = \mathsf{F}_{[0,19]} SMART_{198} > 2.59$ encodes a bad behaviour where the threshold 2.59 of *uncorrectable sector count* is exceeded. Later, a failure prediction is issued thanks to the triggering of f_1. As a consequence, $f_2 = \mathsf{F}_{[1,16]} SMART_{189} > 8.28$ is extracted, meaning that a certain number (8.28) of *unsafe fly height conditions* is reached before the critical number of sector read/write errors is exceeded. The pattern describes a scenario in which the disk head is operating at an unsafe height, ultimately damaging a disk sector and consequently causing read and write errors. Formula f_2 can effectively act as an early warning for the critical situation described by formula f_1.

Results comparable with SOTA but intepretable.

Next theoretical and applied research topics to address

Alignment with Formal Theory of Monitoring and Logic

Following the theory of monitoring in runtime verification, we know that (in the unbounded case) not all formulas are monitorable. Moreover, considering different logical formalisms and fragments leads to different level of expressiveness.

- What is the link between these theoretical aspects and the learning of logic formulas?
- What are the practical implications (faster convergence? better performance? succinctness?) of focusing on different (often simpler, yet powerful enough) fragments?

Extension towards Neuro-Symbolic integration and Self-Supervised Learning

The framework now works in a strictly supervised manner: failure/termination events are always recognizable during training, following a teacher-forcing approach. Working on transient, non-terminating anomalies and interval-based phenomena is more complex, yet there are numerous application domains (e.g., sleep apnea detection or seizure prediction in the medical field). Possible solution: find events through self-supervised neural networks, then extract and monitor formulas.

- Can the framework match the performance of the network used for event extraction?
- Are different logical formalisms needed for this task (e.g., interval temporal logic)?
- How do properties/limitations of neural networks transfer to monitoring and vice-versa (e.g., adversarial robustness, catastrophic forgetting)?

Investigate Alignment between Learned and Human-based Properties

In some contexts, e.g., sleep apnea detection, domain rules are defined to characterize events. Still, doctors who evaluate bio signals using often provide subjective and non-consistent outcomes.

- Which interpretable properties does our framework generate, based on such inconsistent labels?
- Can they be used to detected biases in the physicians' evaluations?
- How does a domain expert judge such properties compared to the gold standard?
- Do the learned properties lead to a better consistency and performance if used by the doctors?

Additional info and acknowledgments

Linktree: `https://linktr.ee/ml_logic`

Work done at the University of Udine, in collaboration with Dario Della Monica, Luca Geatti, Andrea Urgolo, Angelo Montanari. Funding (in part) from Silicon Austria Labs, National Research Programme 2021-2027, iNEST PNRR Spoke 3, and INdAM GNCS.