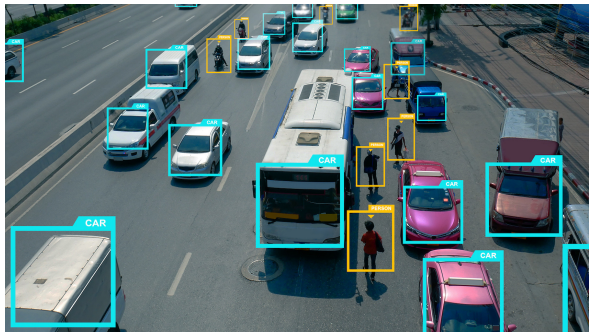
BoostIID: Fault-agnostic Online Detection of WCET Changes in Autonomous Driving

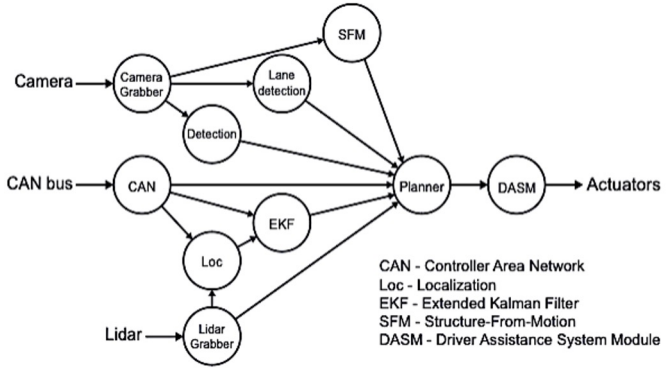
Saehanseul Yi[*], Nikil Dutt[*]

[*]University of California, Irvine, USA

Motivation

- **Autonomous Driving (AD)**
  - **Real-time (RT) Systems**: stringent timing guarantees → deterministic schedule
  - **Safety over *design-time* analyses:** Worst-Case Exec. Time (WCET)
  - **Tight margin**: demanding workloads in both computation and data volume
  - Small errors may lead to catastrophic results

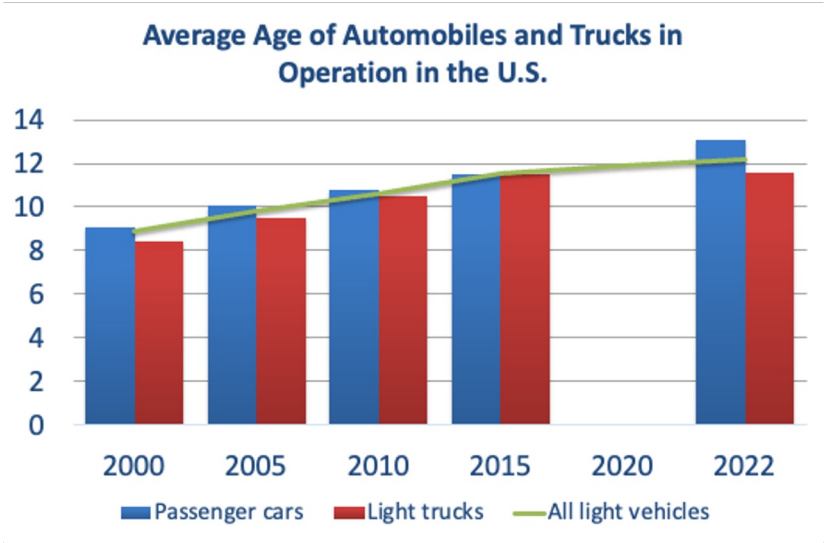




<Example AD Workloads>

Image courtesy of https://www.7wdata.be/digital-transformation/self-driving-cars-whos-winning-and-why

Motivation



- Passenger cars lasts 13.1 years
- Electric vehicles are expected to have longer lifespan than combustion engine vehicles
- ***Will WCETs stay the same during vehicle's lifetime?***

Chart from Bureau of Transportation Statistics 2022

Motivation

- Various aging and permanent faults
  - **DRAM**: Bit Error Rate (BER) increases → re-execution
  - **Cache (SRAM)**: p_{fail} increases → re-partition
  - **SSD, NVMe**: cell wear-out → fail-slow
- Aging may unpredictably affect WCETs that were used in the system design
- Safety-critical systems must be safe when designed, and continue to be safe as conditions change

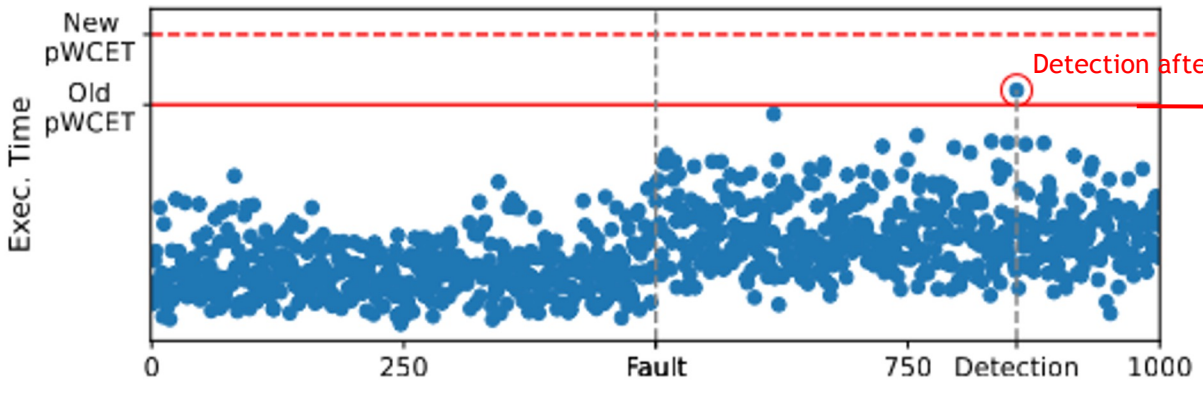
First step: how to proactively detects these safety threats?

Related Work

- **Design-time fault tolerance [… citation]**
  - Add *safety margin* to WCET
  - (-) Fault-specific analyses to measure potential impact
  - (-) Safety margin incurs energy inefficiency during normal operation
  - (+) Does not require recovery actions

- **Run-time fault tolerance [… citations]**
  - Control-Flow Checking (CFC): monitors a critical region of a program
  - (-) Heavy overhead; typically requires hardware support
  - (-) Detects after actual violation
  - (+) Does not require safety margin

Our Idea

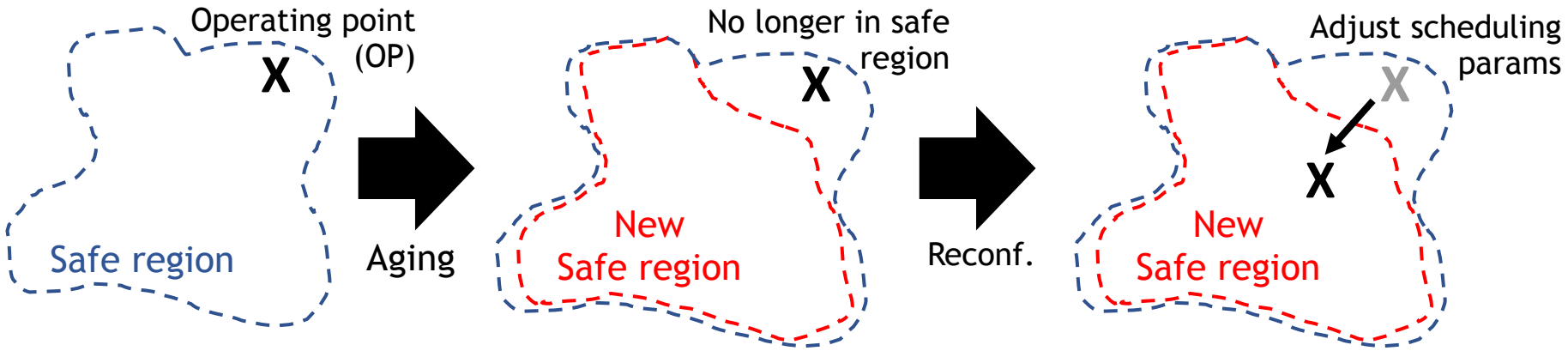


<Example fault detection scenario>

- View faults as a statistical distribution change of execution times
- ***(+) Proactively* detect distribution change using *independent & identical distribution (i.i.d) test***
- (+) No need for fault-specific analyses
- (+) Less pessimistic WCET estimation
- (+) Run-time method with low overhead
- (-) Does not recover from fault

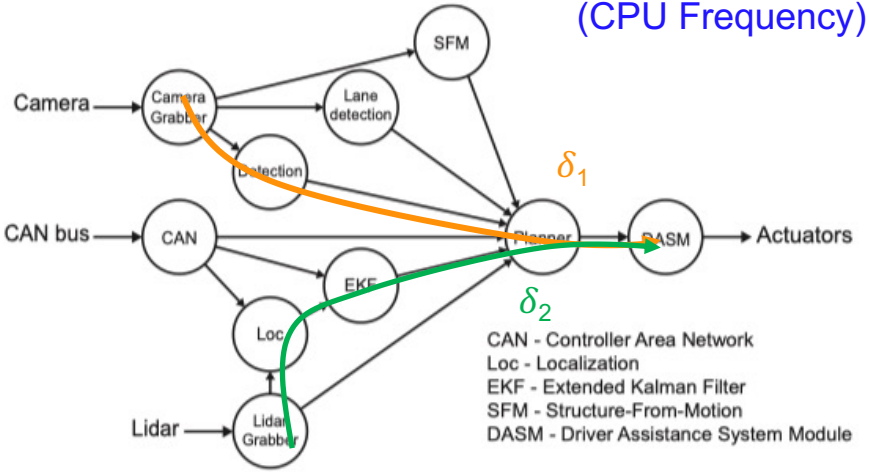
Why is Proactive important?

- ❑ Enables continuous optimization & reconfiguration of the system (future work)
- ❑ Operation point(OP) results from optimization
- ❑ BoostIID can detect the fault + collect data for further runtime optimization



Background: Example RT System Design

- Non-blocking periodic tasks
  - Async buffer between tasks
  - Decision variables: p_i : task period
    s_i : per-task speed factor (CPU Frequency)



A DAG of tasks from Bosch WATERS Industrial Challenge 2019

- Constraints (for each mode)
  - **Schedulability** (assuming EDF)

    Can we schedule tasks without violating periods?

$$U = \sum_i \frac{e_i}{p_i s_i} \leq 100\%$$

e_i: WCET at s_i=1
U : system utilization

  - **End-to-end deadline**

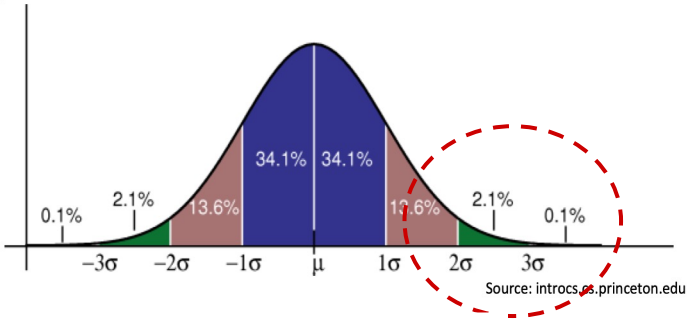
    Timing constraint from sensors to actuators

$$\sum_{i \in \delta} 2p_i \approx \sum_{i \in \delta} (p_i + r_i) \leq d$$

d : end-to-end deadline
r_i: worst-case delay

Background: Probabilistic WCET (pWCET)

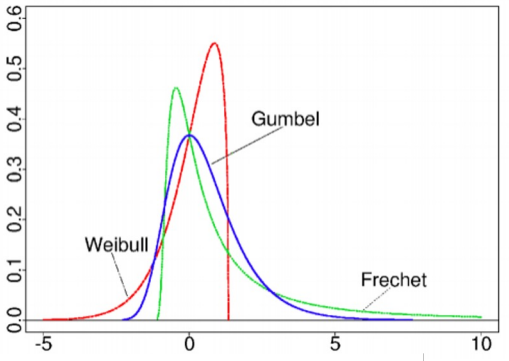
- Central Limit Theorem (CLT): good for estimating mean (center)
- **Extreme Value Theory (EVT)**: extreme value (worst case) distribution follows one of the three forms: *Gumbel*, the *Weibull*, or the *Frechet*



$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

μ: mean
σ: std dev

<Normal Dist.>

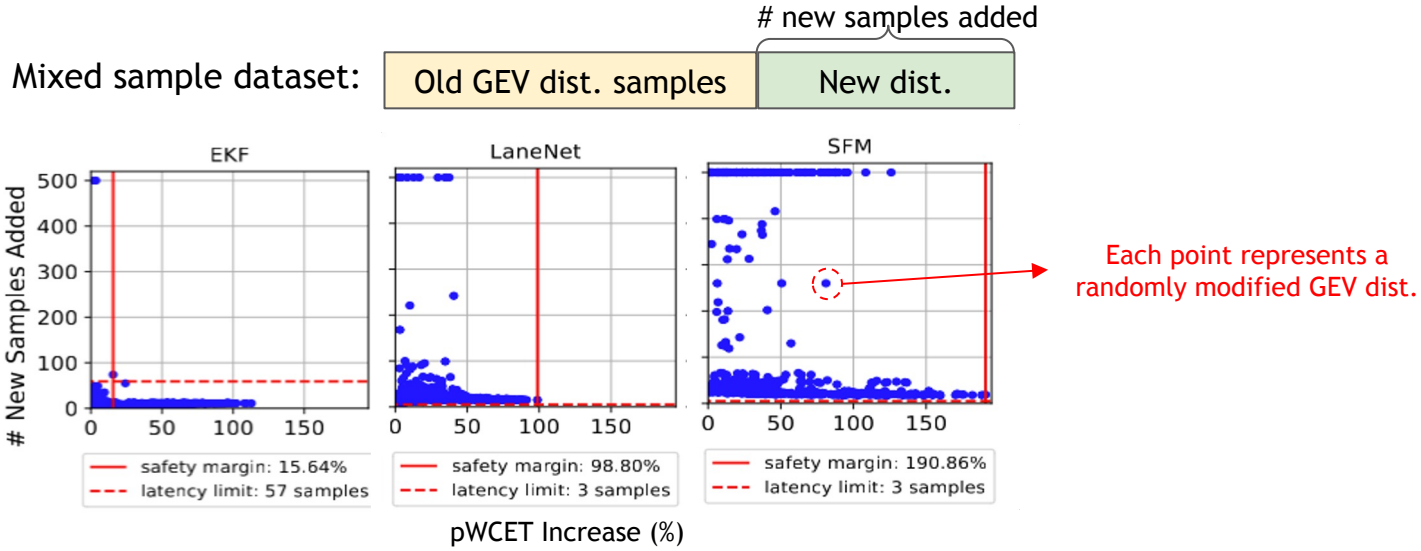


$$G(x) = \begin{cases} e^{-e^{\frac{x-\mu}{\sigma}}} & \xi = 0 \\ e^{-[1+\xi(\frac{x-\mu}{\sigma})]^{-1/\xi}} & \xi \neq 0. \end{cases}$$

μ: location
σ: scale
ξ: **shape**

<Generalized EV (GEV) Dist.>

Detection with a single i.i.d test



- KPSS i.i.d test as a boolean classifier
  - Are the samples from the same distribution? *True*/*False*
- Latency limit (dashed horizontal): within *k* new samples, the WCET change is detected (configurable)
- Safety margin (solid vertical): detector's blind spot; incorporate it to WCET
- KPSS performs well for some tasks

Detection with other i.i.d tests (Sensitivity)

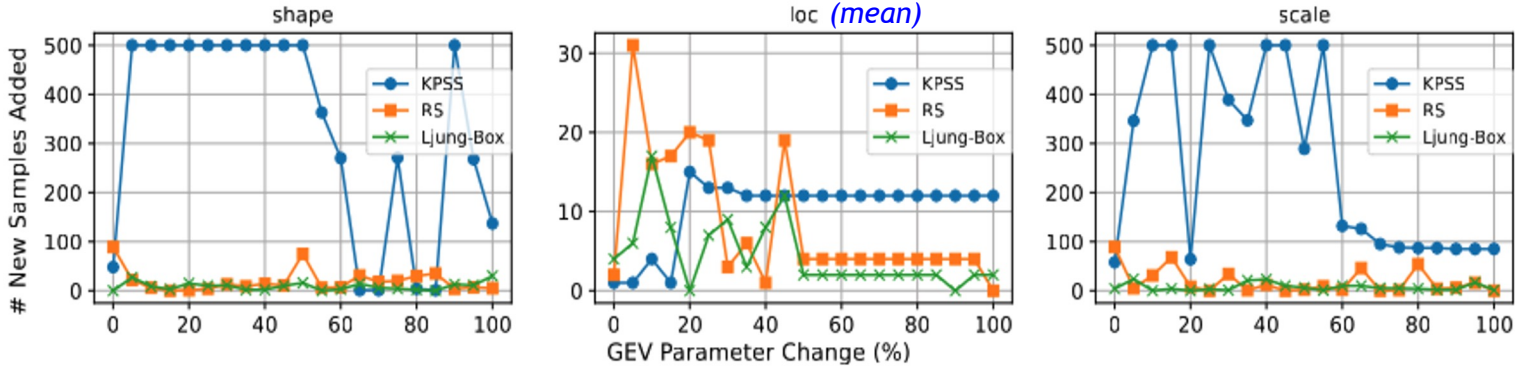
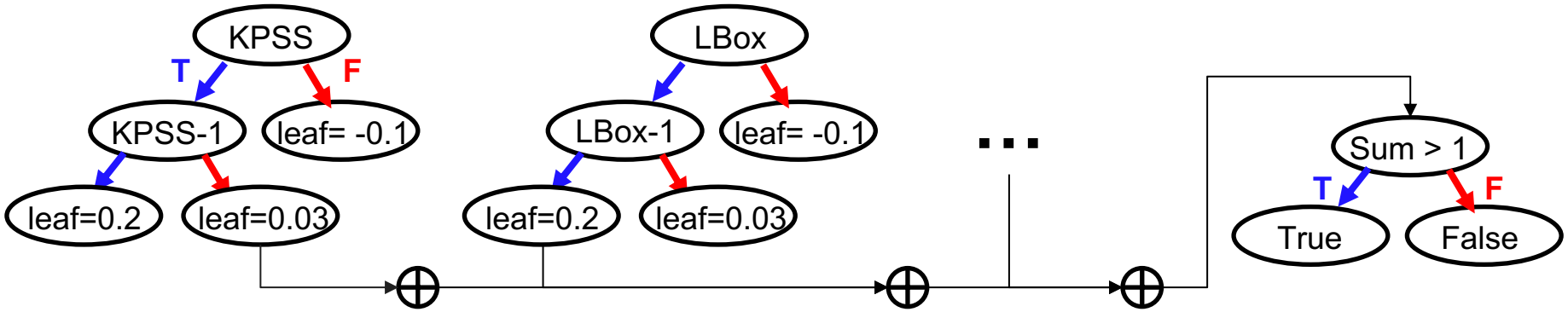


Fig. 4. Characterization of i.i.d tests with GEV parameter sensitivity using Darknet

- Three i.i.d tests: KPSS, R/S, and Ljung-Box
- Different i.i.d tests respond differently to each GEV parameter change
- Difficult to define situations to prioritize a certain i.i.d test
- Accuracy is fluctuating

Boosting



<Boosting example>

- Treat each i.i.d test as a weak classifier
- **eXtreme Gradient Boosting (XGBoost)**: combine multiple *weak classifiers* with weights to create a *strong predictive model*
- XGBoost input: 3 i.i.d tests + their history (previous 5 predictions)

Experimental Setup

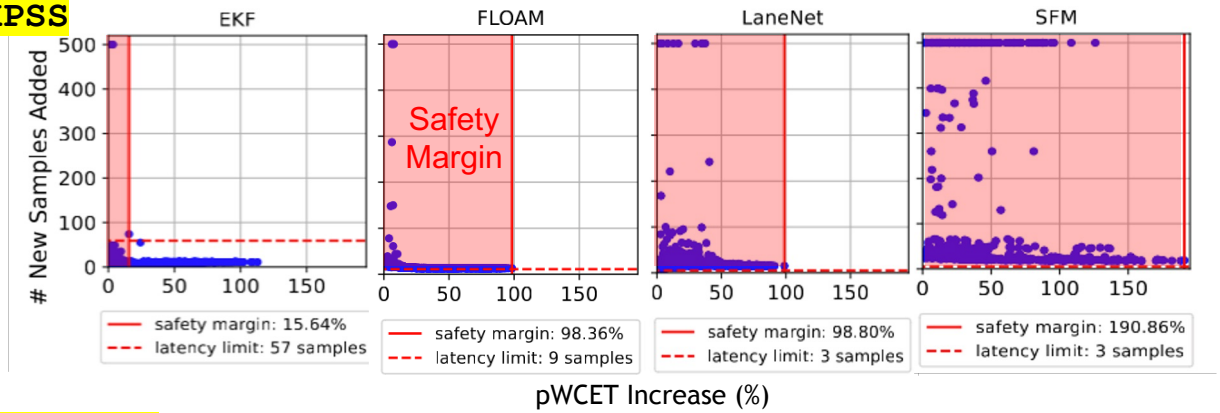
- NVIDIA Jetson TX2 platform
  - 6 task implementation from *Chauffeur*: autonomous driving benchmark suite
    EKF, Hybrid A*, FLOAM, LaneNet, Darknet, and SFM
  - i.i.d test implementation from *Chronovise*: a C++ framework for MBPTA

Measurement-Based
Probabilistic
Timing
Analysis

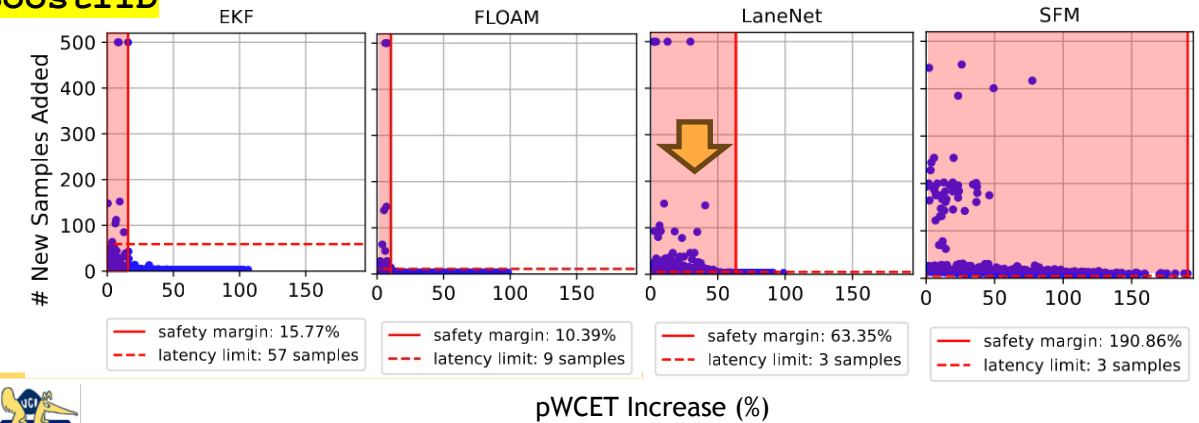
- pWCET
  - pWCET estimation using *Chronovise* with a prob. 10^{-4} & cross-checked with MATLAB's *gevcdf*
  - pWCET for KPSS / RS/ L-Box: 1.16 ms / 0.63 ms / 1.03 ms
- GEV parameter estimation
  - Maximum Likelihood Estimation (MLE) from *Chronovise* on 500 samples
- GEV parameter random modification
  - Each parameter 0~200%; uniform distribution
  - Generated by using MATLAB 2022b's *gevrnd*
- XGBoost python package with 100k detection dataset

Experimental Results: Overview

KPSS

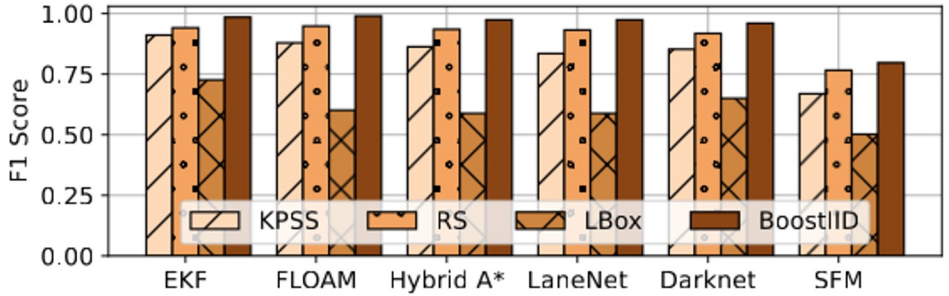


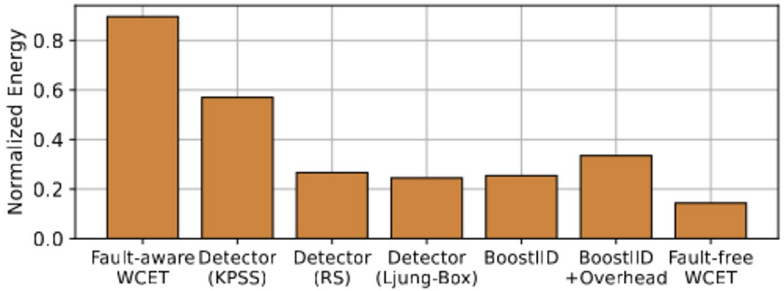
BoostIID



- Detection latency ↓ (# samples added)
  - → Safety margin ↓
  - → Energy efficiency ↑
- Huge improvement in FLAOM and LaneNet
- Poor results on EKF and SFM
  - EKF: already good enough with single i.i.d test
  - SFM: distribution is too difficult for the current set of i.i.d tests.

Experimental Results: F1 Score & Energy Efficiency





$$F_1 = \frac{2TP}{2TP + FP + FN}$$

False positive

False negative

- Except for SFM, the F1 score ranges from 0.96 to 0.99 on 100k datasets
- 62.6% energy reduction compared to classical Fault-aware WCET technique (=100% safety margin)
- BoostIID is fault-agnostic and improves energy efficiency by reducing the safety margin
- But cannot tolerate faults as the Fault-aware approach does

Conclusion

- Novel usage of i.i.d test for runtime detection of execution time change
- BoostIID alleviates pessimistic safety margin in WCET with previous fault-aware WCET methods
- As a result, BoostIID achieved 62.6% average dynamic power reduction in an example RT system with Autonomous Driving workloads
- Proactiveness of our method provides time for further runtime reconfiguration
- In the future, we plan to extend our work to recover from faults after detection using BoostIID