AI-ENABLED FPGA TRIGGER FOR AUTONOMOUS RADIO DETECTION OF COSMIC RAYS

ARENA 2026 – KARLSRUHE

11TH JUNE 2026 | VESSELIN DIMITROV

Member of the Helmholtz Association
Universität Siegen
CPPS Center for Particle Physics Siegen
color meets flavor
JÜLICH Forschungszentrum
Integrated Computing Architectures
ICA

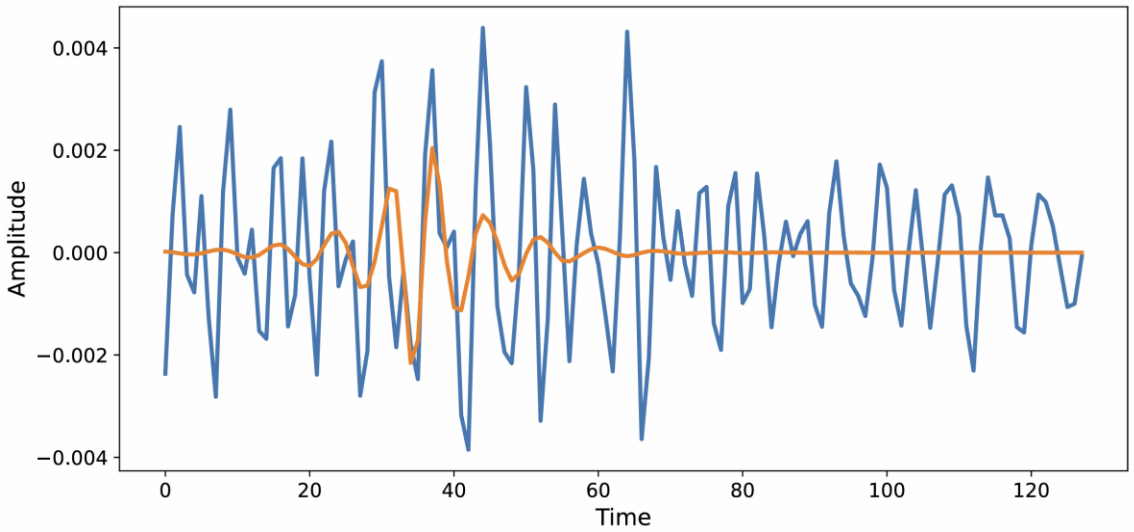
MOTIVATION

- **High** signal-to-noise ratio (SNR)
  - →Captured by **amplitude threshold**
- Near-threshold pulses **indistinguishable from noise**
  - →High false positive rate or efficiency loss

AI-based detection strategy:

- Learn **waveform structure** instead of **amplitude** only
- Pipeline:
  - **Denoiser → Classifier**
  - **FPGA** implementation
- Requirements:
  - Low latency, resource efficient, high accuracy

Near-threshold pulse



Signal trace buried in noise

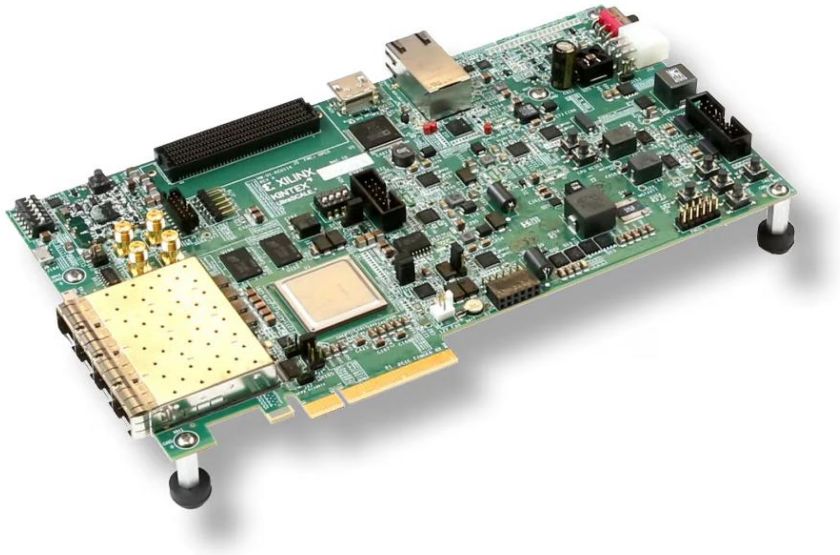
source: Q.Dorosti 2025 *JINST* **20** P10010

FIELD-PROGRAMMABLE GATE ARRAYS (FPGA)

- **Reconfigurable logic**
- Consists of dedicated resources
  - **Block RAM** (BRAM) → on-chip memory
  - **Digital Signal Processors** (DSPs) → arithmetic units
  - **Flip-Flops** (FFs) → sequential storage
  - **Look-Up Tables** (LUTs) → truth tables (NAND, XOR, …)
- True hardware-level **parallelism** (unlike CPUs/GPUs)

→Deterministic **latency**

→**High throughput** at **low power** consumption



source: AMD Kintex™ UltraScale+™ FPGA KCU116 Evaluierungskit

FRAMEWORKS

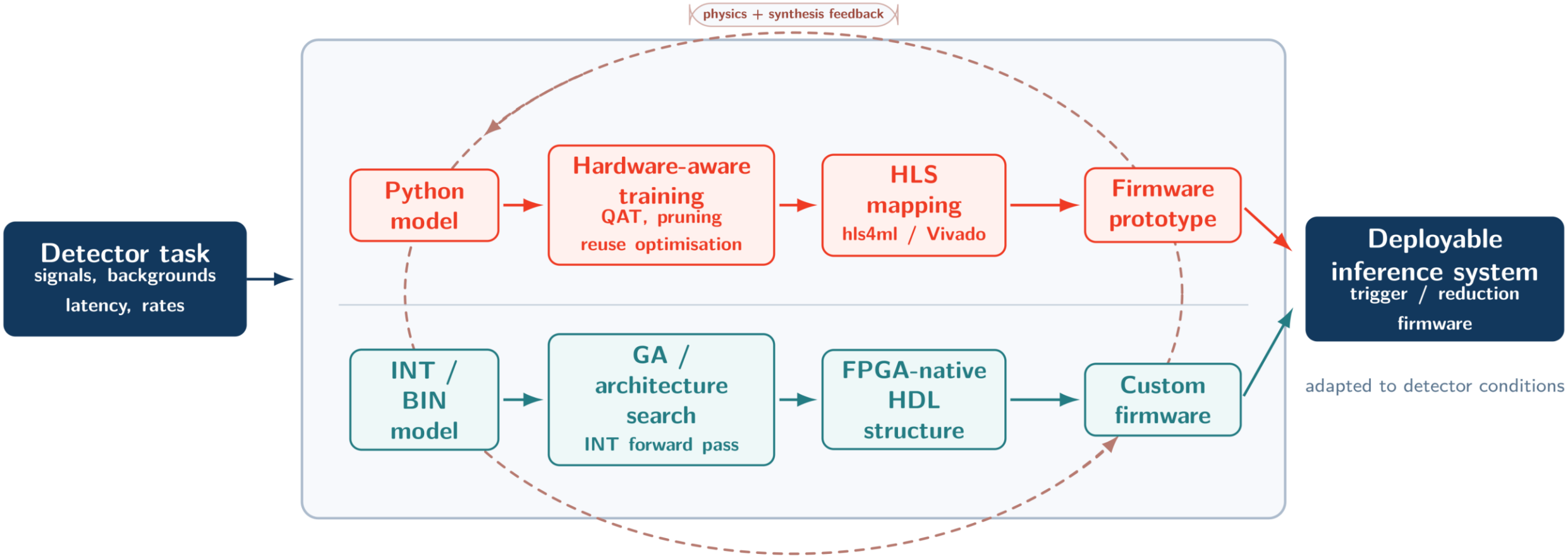
Model development (Keras)

- High-level Python framework for defining and training neural networks
- Quantization-Aware Training
  - Incorporates fixed-point constraints during training
  - Preserves accuracy and reduces model size

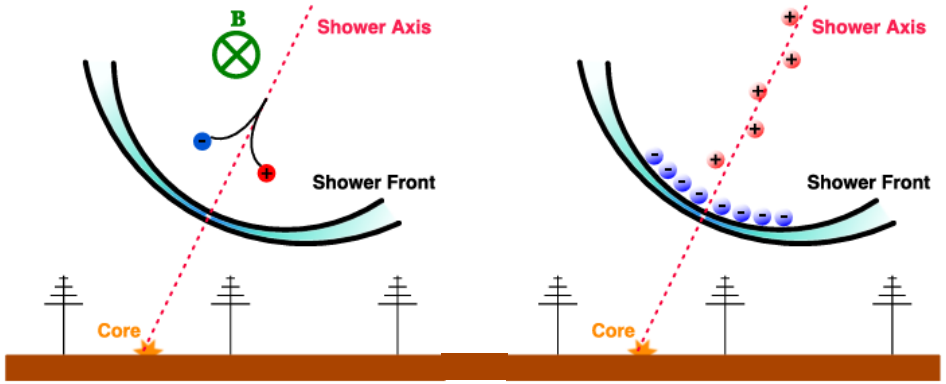
Firmware Generation (hls4ml)

- Create firmware implementations of AI models for FPGA deployment

WORKFLOW


physics + synthesis feedback
Detector task
signals, backgrounds
latency, rates
Python model
Hardware-aware training
QAT, pruning
reuse optimisation
HLS mapping
hls4ml / Vivado
Firmware prototype
INT / BIN model
GA / architecture search
INT forward pass
FPGA-native HDL structure
Custom firmware
Deployable inference system
trigger / reduction
firmware
adapted to detector conditions

DATASETS



Main mechanisms for radio production. (left) Geomagnetic emission, (right) Charge excess emission

T.Huege: https://arxiv.org/pdf/1601.07426v1

- Trace format:
  - 180 MHz – 250 MHz sampling rate
  - **128 samples** per trace
- Traces classes:
  - ***background***: Measured background traces
  - **pure pulse**: Simulated CORSIKA cosmic-ray pulse
  - **signal**: ***background*** injected with **pure pulse** [1]

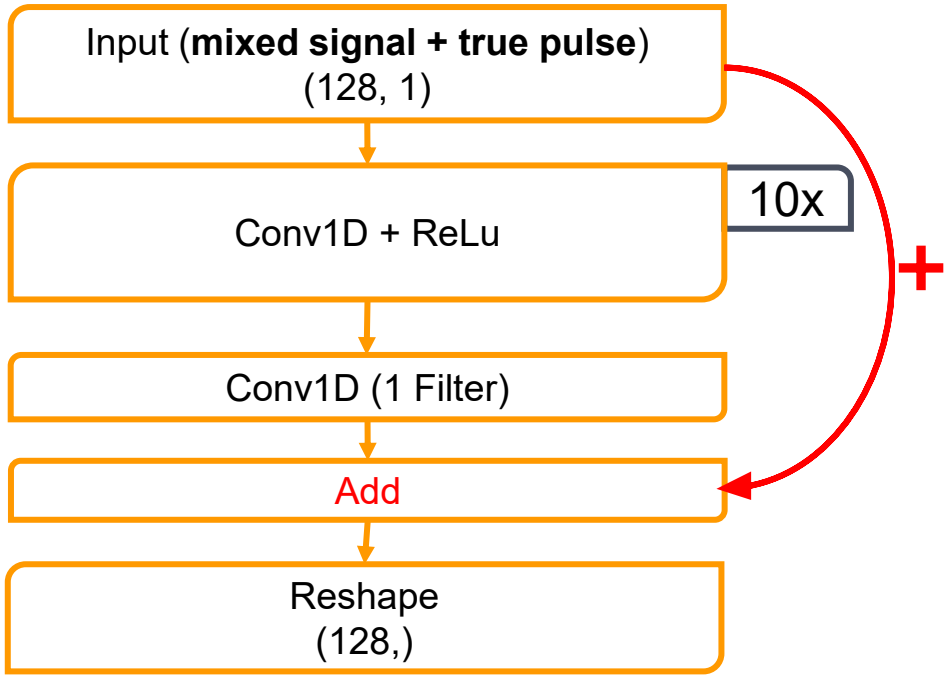
Butterfly antenna used for capturing the noise on the campus in Siegen

[1] Pierre Auger collaboration, T. Huege et al.: https://pos.sissa.it/501/292

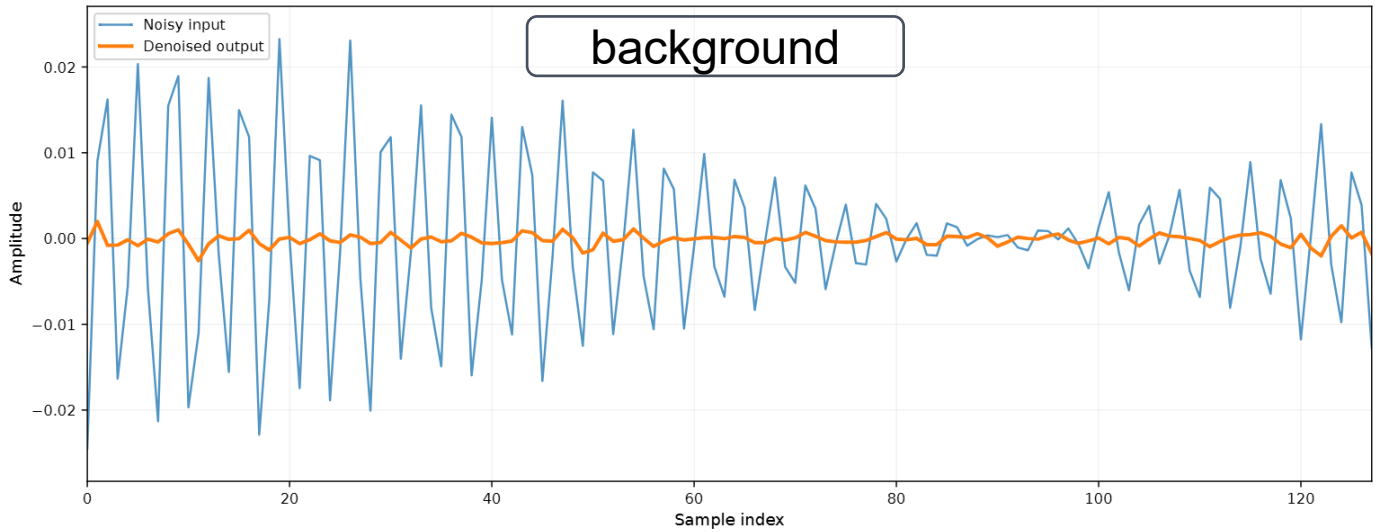
MODEL ARCHITECTURE

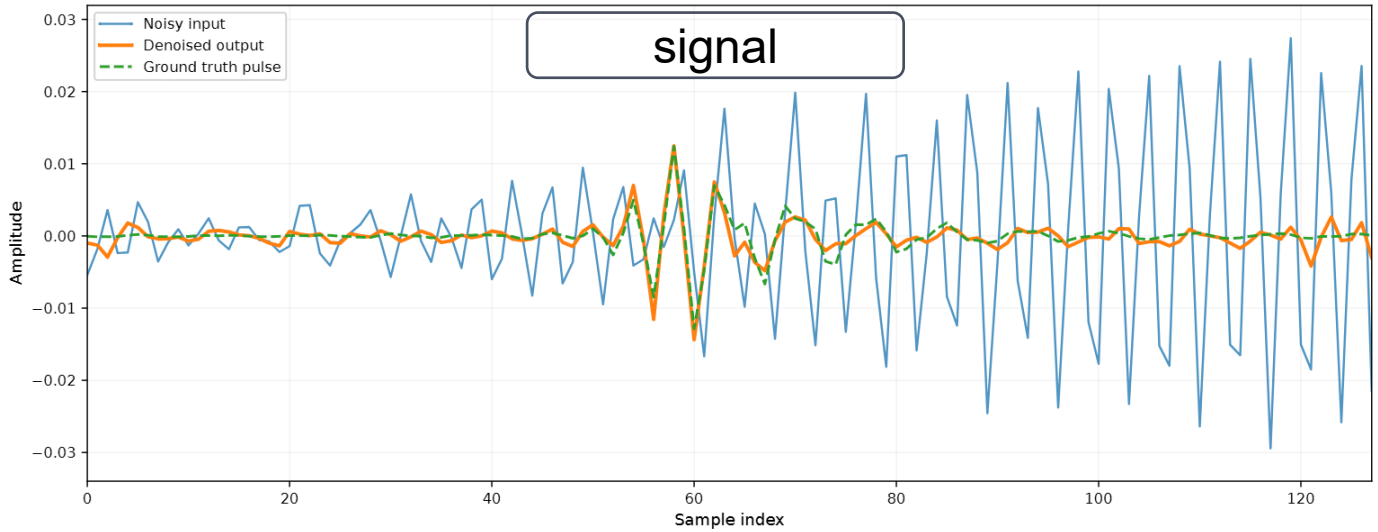
A.Aksoy et al.: http://arxiv.org/abs/2605.28332

Denoiser



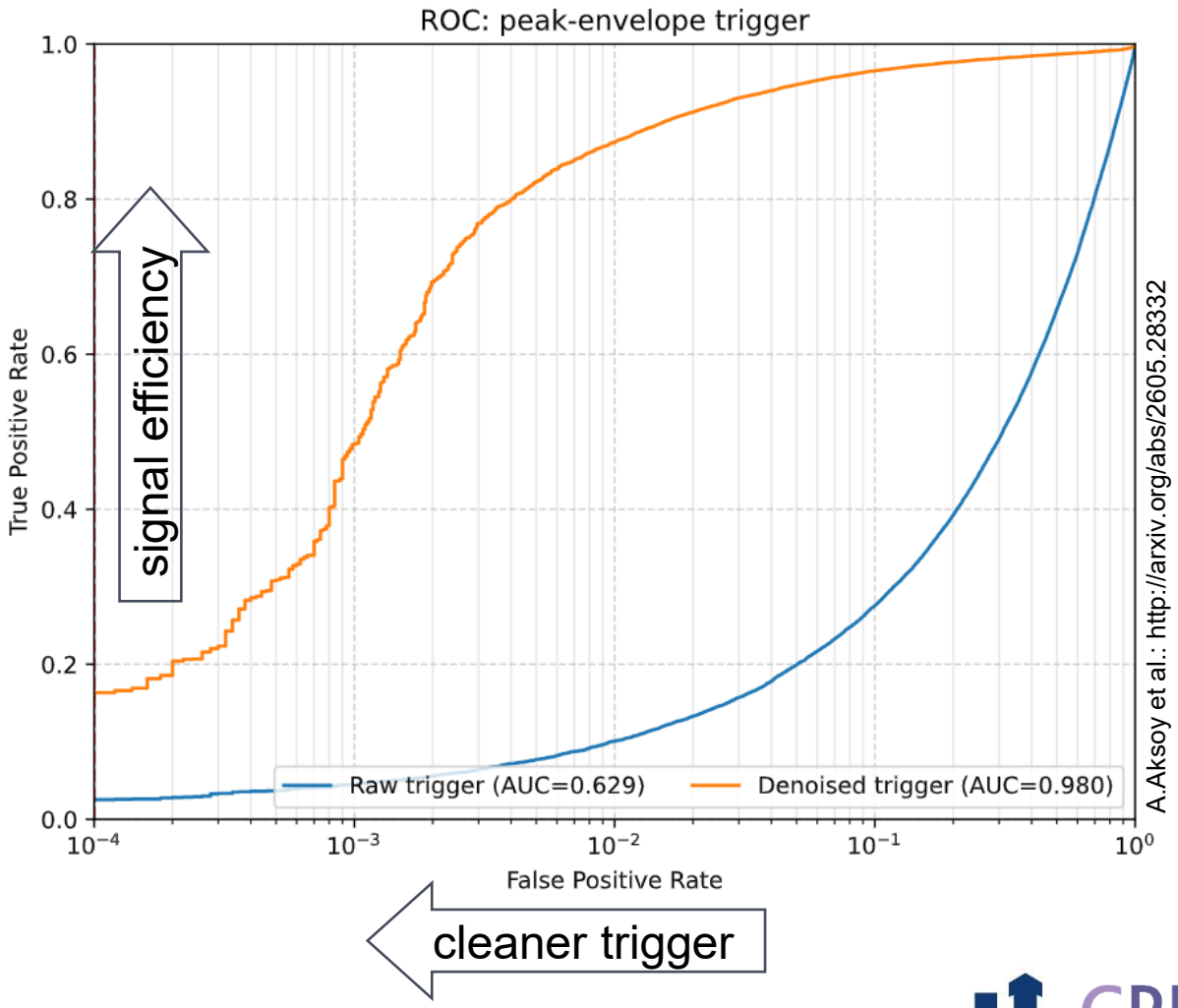
→Suppress noise while preserving signal structure





DENOISER

Performance



Peak-envelope trigger

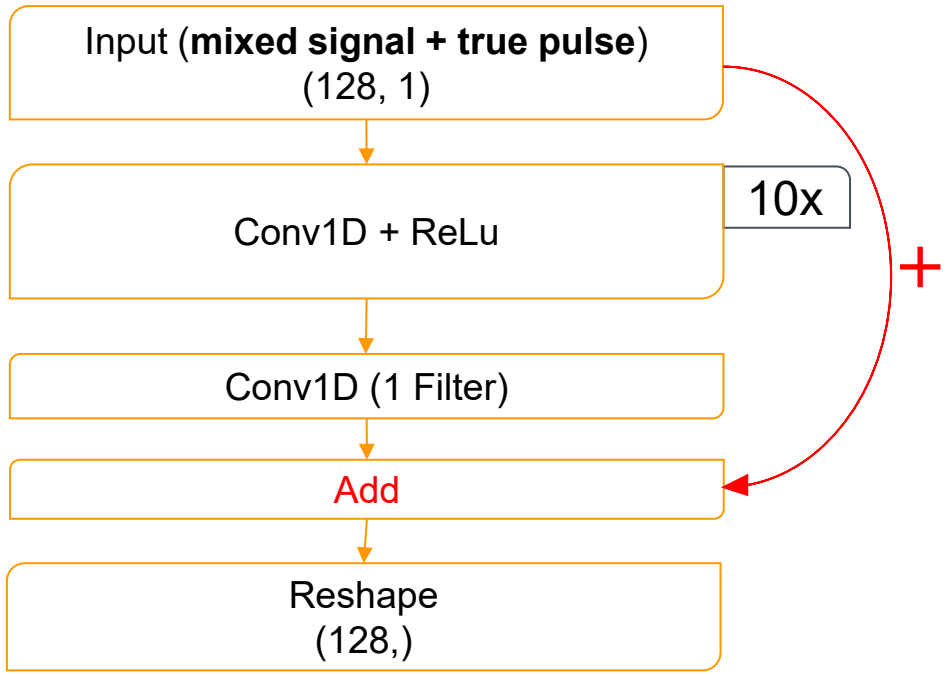
- Determine peak envelope amplitude A_{peak}
- Select threshold θ based on target false positive rate
- Trigger if $A_{peak} > \theta$

FPGA	Latency	BRAM	DSP	FF	LUT
Zynq-7020	7.68 µs	168 60%	45 20%	38,710 36%	34,733 65%
Kintex U040	3.03 µs	168 14%	45 2%	17,042 3%	30,520 12%
Alveo U250	2.30 µs	121 2%	45 ~0%	16,564 ~0%	33,826 1%

→Denoising improves trigger discrimination

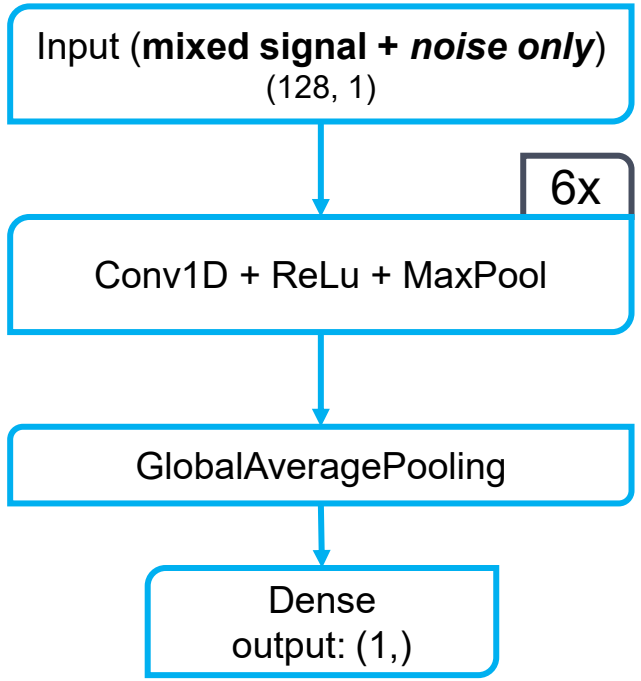
MODEL ARCHITECTURES

Denoiser



→Suppress noise while preserving signal structure

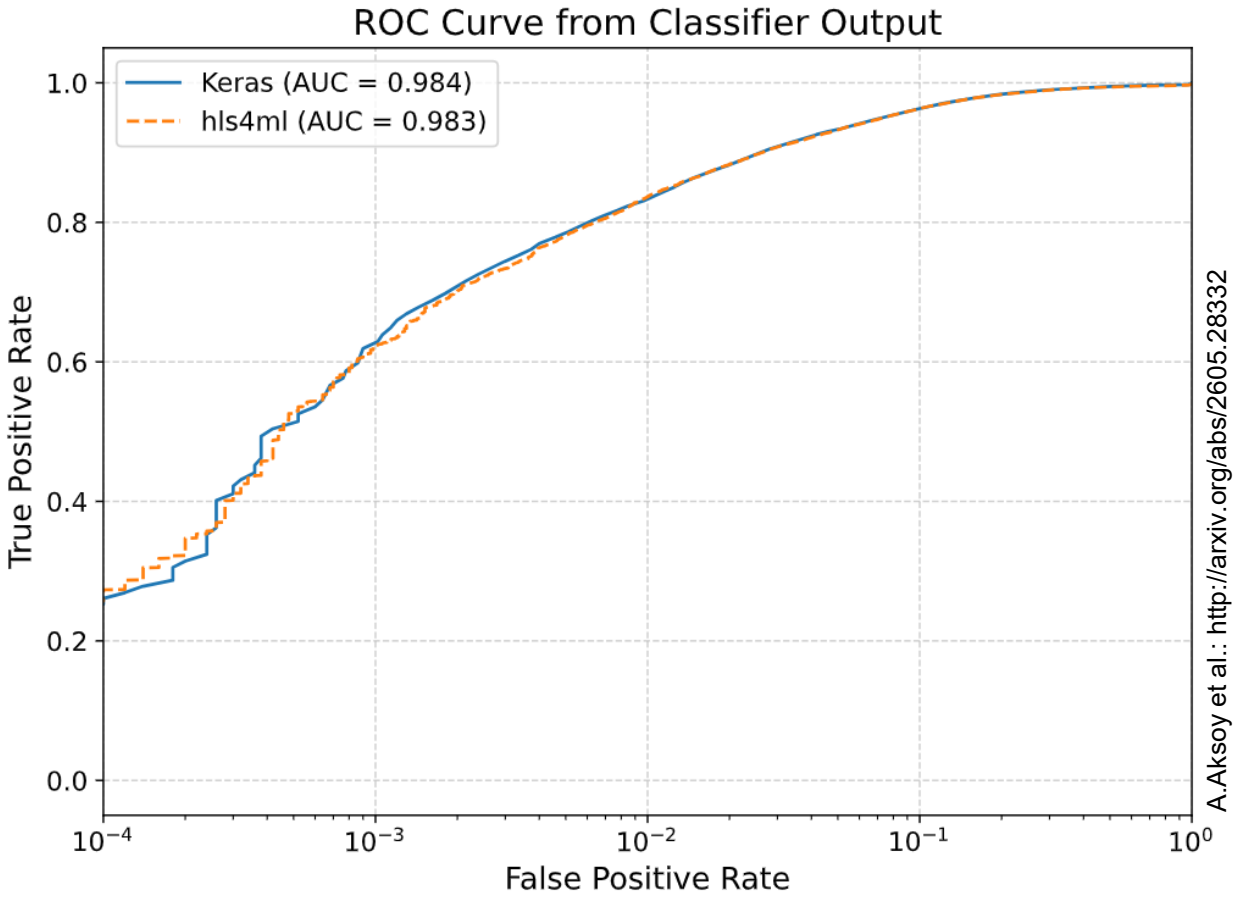
Classifier



→Binary classification: signal vs. noise

CLASSIFIER

Performance



FPGA	Latency	BRAM	DSP	FF	LUT
Zynq-7020	5.87 µs	29 10%	1 ~0%	17,592 16%	21,721 40%
Kintex U040	2.63 µs	29 2%	1 ~0%	10,553 2%	20,188 8%
Alveo U250	1.34 µs	21 ~0%	2 ~0%	8,848 ~0%	20,154 1%

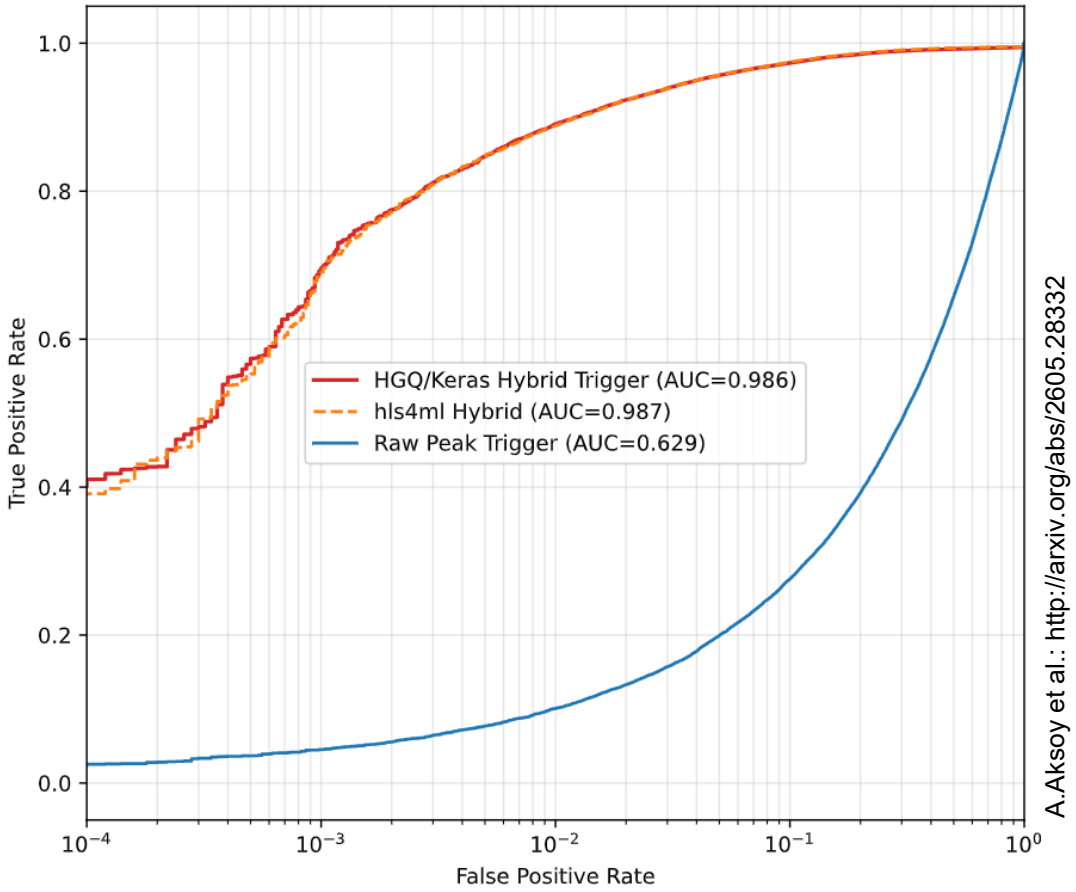
Input for training and evaluation:

- Raw traces (no denoising yet)

→Achieves high accuracy (high AUC – Area Under Curve) with low resource usage

HYBRID (DENOISER + CLASSIFIER)

Performance



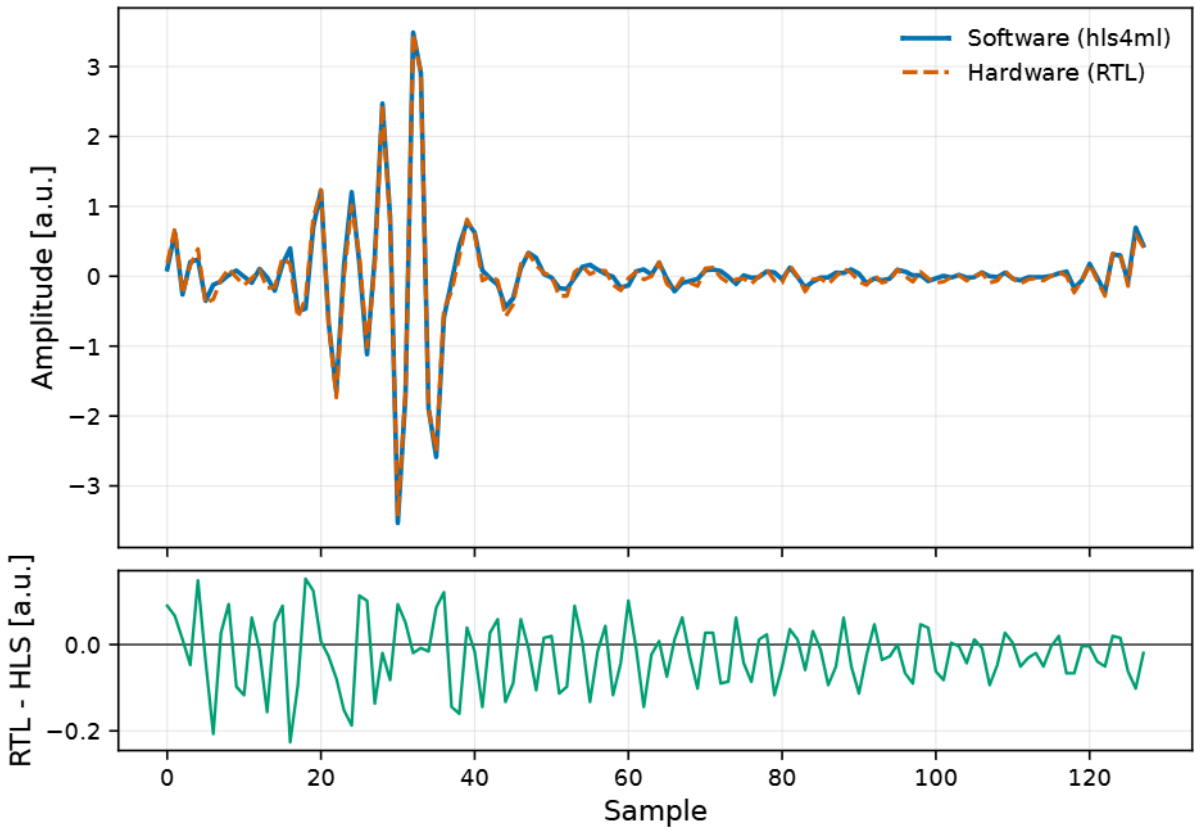
A.Aksoy et al.: http://arxiv.org/abs/2605.28332

FPGA	Power	Latency	BRAM	DSP	FF	LUT
Zynq-7020	0.562 W	13.55 µs	100 36%	42 19%	39,187 37%	22,906 43%
Kintex U040	0.356 W	5.66 µs	63 5%	68 4%	25,370 5%	22,624 9%
Alveo U250	0.270 W	3.63 µs	64 1%	70 ~0%	20,777 ~0%	22,287 1%

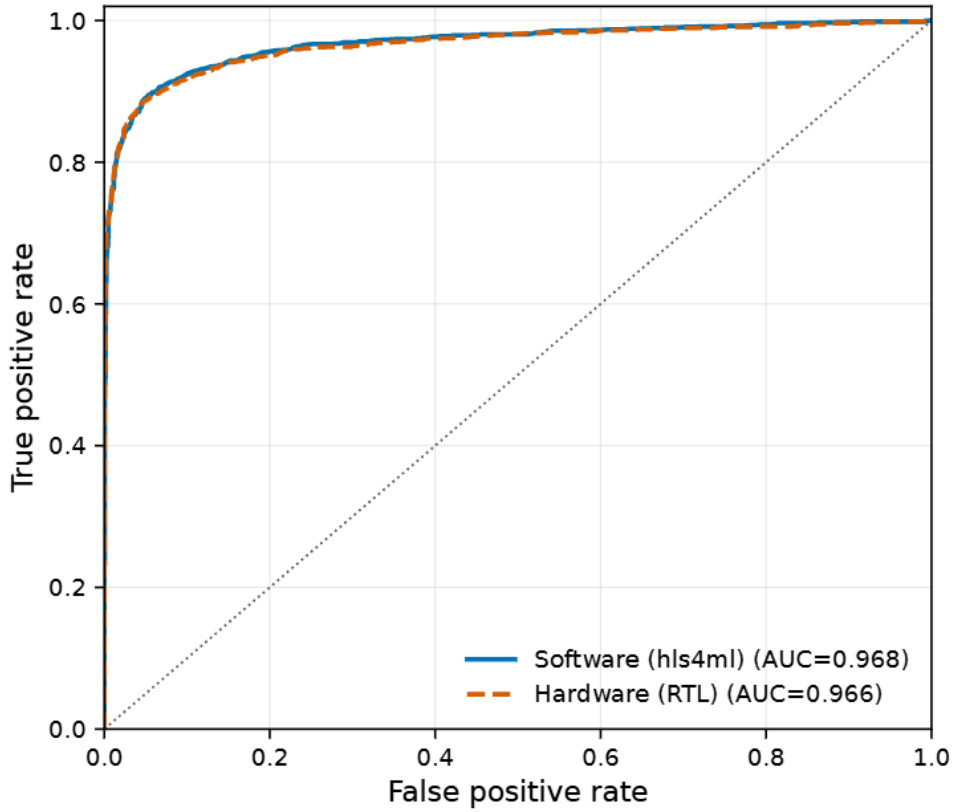
→Improved signal efficiency for very low FPR

RTL VALIDATION

cocotb testbench



A.Aksoy et al.: http://arxiv.org/abs/2605.28332



→Excellent agreement between RTL simulation and software performance

SUMMARY AND OUTLOOK

- Two-stage architecture:
  - **Denoiser** for noise suppression
  - **Classifier** for signal identification
  - Models trained independently of each other
- **High classification** performance demonstrated (AUC=0.986, high TPR at low FPR)
- **Low resource usage** achieved (0.562 W on Zynq-7020)

- Further **reduction of resource usage** for an efficient deployment on low- to mid-range FPGAs will be investigated
- Implementation with **Binary Neural Networks**
- Full Hardware validation
  - Validation with **real, live antenna data**

BACKUP

HYPERPARAMETER OPTIMIZATION - HYPERBAND

- Training a model fully is expensive

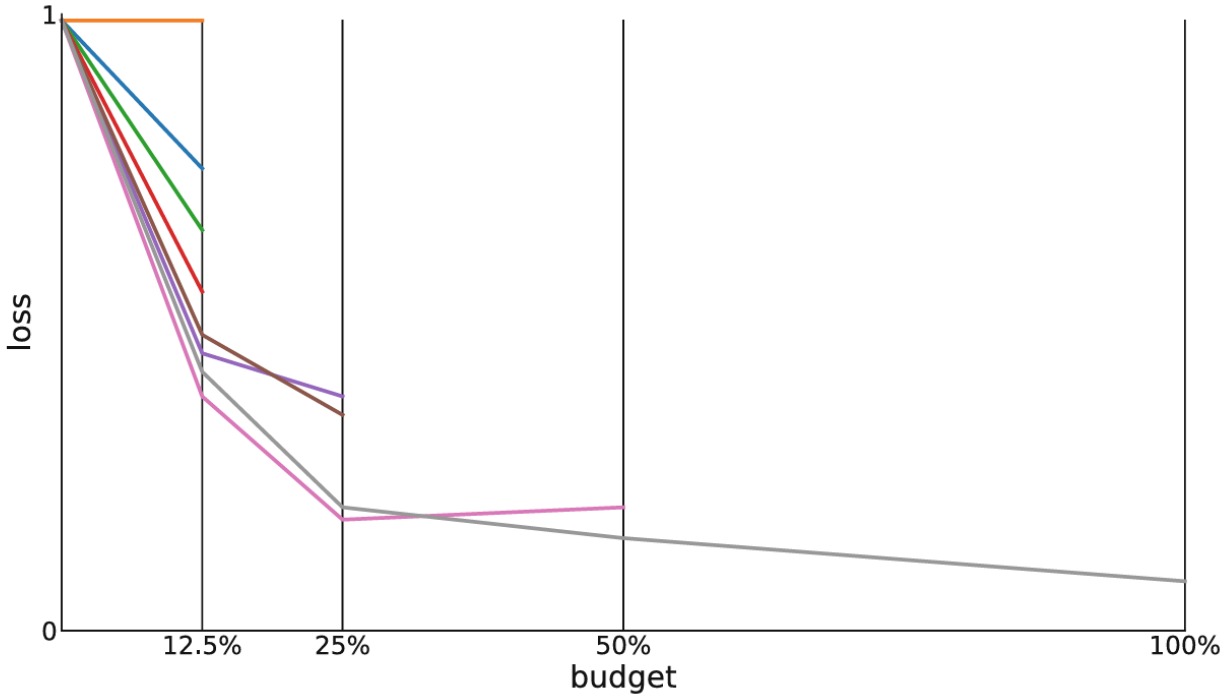
→Train **many** configurations a **little bit**

Successive Halving

- Start with **many hyperparameter configurations**
- Give each a **small budget** (e.g. few epochs)
- **Evaluate** them
- Keep the **best fraction**
- **Increase budget** for survivors
- **Repeat** until one candidate left

  → *Choose* **many configs/tiny budget** or **few configs/large budget**

HYPERBAND (HB)



HB runs **many schedules** with **different budgets**

Successive Halving for 8 different configurations

- Start with low budget ($^1/_8$)
- Drop half configs after evaluating
- Increase budget for the remaining by same factor

→Automatically optimize the model, without manual tuning

ENVELOPE THRESHOLD TRIGGER

Purpose and decision rule

- Classical amplitude-based trigger used as benchmark
- Compute peak envelope **amplitude** A_{peak}
- Select decision **threshold** θ based on target **FPR**
- Trigger if $\mathrm{A_{peak}} > \theta$

Evaluation

- ROC curve (TPR vs. FPR)

HARDWARE IMPLEMENTATION

- At sampling rate of ~180 – 250 MHz → trace length of ~711 – 512 ns
- Inference of trigger ~10 µs

Possible solutions:

1. Implement multiple models on an FPGA

   Run in parallel

   → Continuous operation at higher resource usage

2. Keep one model

   Start new inference with same model with randomly sampled trace

   → Dead time affected operation

BINARY NEURAL NETWORKS

- Neural networks with **binary weights and activations** (±1)
- **Typical methods:**
  - Binarization of weights and activations
  - Straight-Through Estimators (STE) for backpropagation
  - Specialized training strategies to recover accuracy

→**Much lower memory usage** at the cost of accuracy

→**Faster inference** compared to full-precision networks