Triggerless data acquisition pipeline for Machine Learning based statistical anomaly detection

Grosso Gaia, Lai Nicolò, Migliorini Matteo, Pazzini Jacopo, Triossi Andrea, Zanetti Marco, Zucchetta Alberto
University and INFN Padova

CHEP2023
26th International Conference on Computing in High Energy and Nuclear Physics
May 9, 2023 - Norfolk, Virginia, USA
Introduction and Outline

Triggerless data acquisition?

- Stream all data from detector without waiting for a trigger signal
- Why? Hardware triggers may be insufficient for the selection
  - Perform an online analysis on all the data, selection based on high level features
  - Well suited for anomaly detection applications

Introduce an example pipeline for collecting and processing triggerless data

- Local reconstruction using neural networks on FPGA and data transmission to a server memory
- Data quality monitoring (DQM) as an example of anomaly detection
  - New Physics Learning Machine (NPLM) technique to spot anomalies [1]
  - Run preprocessing and NPLM on a GPU for optimal performance
Detector: **miniDT**

Reduced area CMS Drift Tube (DT) muon detector

- First built for the test-beams of LEMMA project for muon collider
- Currently used as testbed for multiple applications
  - development and evaluation of new CMS phase-2 upgrade DT front end boards (OBDT) [2]

Composed of 4 layers of cells (tubes) filled with Ar-CO\textsubscript{2} gas mixture

- **Electron avalanche** produced by the passage of a muon
- Collected by a wire in the middle of each tube
- Uniform electric field provides constant drift velocity of the electrons

**Mean-Timer algorithm** allows to determine the muon passage time

- Find track parameters, slope and position
Readout and Backend

Signals produced by the electron avalanches are amplified, shaped, and discriminated by custom ASIC chips in the Front-End electronics of the chambers.

- Two evaluation boards Xilinx VC707/OBDTs used to perform Time-to-Digital conversion (TDC) in FW
- Send data to a backend board, a Xilinx KCU1500, mounted on the PCIe of a server

Inside the KCU, a reconstruction algorithm processes hits from miniDTs to identify the muon “stubs” based on neural networks[3]. DMA transfers from the FPGA to the server memory of the stubs + all hits.
Backend: **Reconstruction algorithm**

Neural networks adopted in two steps

- **Filtering**: hits produced by noise are removed, keeping only the 4 left in each layer by the muon
- **Disambiguation**: Identify if the muon passed on the left or right of wire

Once the laterality of the 4 hits is given, the *crossing time* $t_0$ can be found using a simple analytical relation

- Use it to find position inside each cell and fit the track

Neural network were trained using QKeras and HLS code of the models produced using the package **HLS4ML**

![Diagram of the reconstruction algorithm](image-url)
First steps of data processing

Hits and stubs are transferred to the memory of the backend server

- Reformatted and buffered temporarily on a ramdisk

First steps of the processing based on DataFrame-like operations

- Standard data manipulation, e.g. filter rows, aggregations and columnar operations
- Dask used as a scheduler to distribute the workload [4]
- Test a different approach?

⇒ GPGPU acceleration

- Using a NVIDIA A100 (40GB) GPU (thanks to NVIDIA academic hardware grant)
- Use it for pre-processing testing using CUDA-DataFrame (CuDF) and machine learning solutions for anomaly detection
Data preparation with cuDF

CuDF is a Python/C++ GPU DataFrame library built on top of Apache Arrow memory format

- Implements many standard DataFrame operations, e.g. aggregations, filters, joins, ...
  - I/O modules for standard formats such as Arrow and Parquet
- Can be extended by writing custom kernels using Numba/CuPy/CUDA

Data preparation for the anomaly detection application makes use of the following operations

- Aggregate hits in time with the muons stubs
  - Operations on individual “events”
- Filter-out hits not compatible outside the muon time window
- Columnar operations to manipulate hits features and prepare them for anomaly detection algorithm
NPLM in one slide

Method follows a classical Hypothesis Testing based on the Likelihood Ratio

- Model $f_w(x)$ used to define set of alternatives $p(x|H_w)$ to $p(x|0)$, with $w$ trainable parameters
- Model trained to minimize the logistic loss

⇒ Trained model approximates log-likelihood ratio between data and reference distributions
⇒ Can compute the test statistics $t(D)$

Calibration procedure
Train model using reference-distributed data samples
⇒ empirical distribution of the test statistics in validity of the reference hypothesis
⇒ follows the chi-squared distribution

Test data distribution
Train the model to obtain $t_{obs}$
⇒ compute p-value using the chi-squared approximation
⇒ one value per each data sample!
DQM as an Anomaly Detection problem

Create a reference dataset $R$ of data collected under **nominal conditions**

- Use it to perform the test statistics calibration “offline”

For every new batch of data $D$ run the training procedure against $R$ and obtain a $t_{\text{obs}}$

- Compute a $p$-value and determine if the batch contains **anomalies**

Model $f_w(x)$ used is based on (gaussian) kernels

- Implemented using the **Falkon library**[5][6], developed to run kernel methods at scale
- Designed to exploit GPU acceleration and parallelization over multiple GPUs
- Found to be much faster than ANN-based approaches
Monitoring miniDTs

Used low-level quantities for the monitoring:

- Collection of the hits’ drift times
  - 4 in total, one per layer
- Slope of the muon stub
- Other quantities could be used in principle, such as the hit rate, residuals of the track reconstruction etc.

Artificially injected real-life detector anomalies:

- Lowered cathodic strips voltage to 25% / 50% / 75% of the nominal levels
  ⇒ Electrical field not uniform inside the cell
- Reduced front-end threshold to 25% / 50% / 75% of the nominal levels
  ⇒ Higher noise producing more fake hits
Results with Falkon

Falkon-based NPLM is capable of identifying the anomalies

- Using 2000 events for the reference dataset
- Probing batches of 500 events every time
- Easier job if more informative features were used
  - Test the method under challenging conditions

Performance evaluation of Falkon for DQM applications is ongoing

- First tests with small batches dominated by Falkon overhead
- Size selected based on the cosmic muons rate
- Training time ~0.5s

Method is capable of handling millions of events efficiently[7]

- $O(10s)$ vs $O(10h)$ for neural networks
Outlook and future perspective

Example of an entire pipeline, from detector to anomaly detection

- System to collect and process a continuous stream of data
- DQM with low level features as an example application of NPLM
- Extend it to work with higher level quantities
  - Add more processing inside the GPU

Current work on the hardware side

- Substituting KCU1500 with a larger VCU118
  - Larger number of links
  - Accept external clock / signals
- ROCE to transfer data from the board to a server
  - FEROCE - FrontEndROCE project
- Based on the EMP firmware framework from CMS and ETH Scalable Network Stack for FPGAs [8]
  - Currently tested TCP/IP, moving to ROCEv2!
References

[1] Unbiased detection of data departures from expectations with machine learning
[2] Trigger-less readout and unbiased data quality monitoring of the CMS drift tubes muon detector
[3] Muon trigger with fast Neural Networks on FPGA, a demonstrator
[4] A horizontally scalable online processing system for trigger-less data acquisition
[5] Kernel methods through the roof: handling billions of points efficiently
[7] Learning new physics efficiently with nonparametric methods
[8] ETH FPGA Network Stack

Acknowledgments

This research was supported by grants from NVIDIA and utilized a NVIDIA A100 GPU
New Physics Learning Machine with Falkon

Algorithm: New Physics Learning Machine

input:
- Reference sample $\mathcal{R} \sim p(x|0)$.
- Data sample $\mathcal{D} \sim p(x|1)$.
- Set of reference-distributed data samples $\{\mathcal{R}_i \sim p(x|0)\}_{i=1}^N$.
- Binary classifier $f_\mathbf{w}$.

calibration:
- foreach $\mathcal{D}_\mathcal{R} \in \{\mathcal{R}_i \sim p(x|0)\}_{i=1}^N$ do
  - Train $f_\mathbf{w}$ using the reference sample $\mathcal{R}$ and the reference-distributed data sample $\mathcal{D}_\mathcal{R}$.
  - Compute the test statistics $t(\mathcal{D}_\mathcal{R}) = 2 \sum_{x \in \mathcal{D}_\mathcal{R}} f_\mathbf{w}(x)$.
  - Build the empirical distribution of test statistics in validity of the reference hypothesis $p(t | \mathcal{R})$.

training:
- Train $f_\mathbf{w}$ using the reference sample $\mathcal{R}$ and the data sample $\mathcal{D}$.
- Compute the test statistics $t(\mathcal{D}) = 2 \sum_{x \in \mathcal{D}} f_\mathbf{w}(x)$.

output:
- The $p$-value $p[t(\mathcal{D})] = \int_{t(\mathcal{D})}^{\infty} p(t' | \mathcal{R}) \, dt'$. 
MiniDT

Each miniDT is composed of 4 layers of cells (tubes) arranged with ½ cell staggering to allow an estimation of the muon track

- 16 (42x14 mm²) cells per layer
- A total of ~70x70 cm² active area per chamber
- Filled with an Ar-CO₂ (85/15%) gas mixture
- Uniform electric field inside the cell providing a constant drift velocity
Neural network reconstruction- hits collection and grouping

Hits collection and grouping

Filtering

hit_1, ..., hit_5
Neural network reconstruction: filtering and reconstruction

Hits collection

Filtering

Disambiguation

Reco