TOWARDS RESOURCE EFFICIENT PARALLEL CONCEPT DRIFT DETECTION IN DATA STREAMS

Doctoral Showcase

Elias Werner[1 2 3] | *main advisor:* Prof. Dr. Wolfgang E. Nagel[1 2 3]
[1] CIDS, TU Dresden, [2] ScaDS.AI Dresden/Leipzig, , [3] ZIH, TU Dresden, Germany

INTRODUCTION

- Resource-efficient and sustainable AI demands computational performance engineering to use parallel resources efficiently
- This PhD project bridges between the domains: it advances computational performance engineering in data science using concept drift detection as a case study
- Concept drift, changes in data distributions after the deployment of a machine learning (ML) model, must be detected for robust ML systems
- The computational performance of concept drift detectors (DDs) is underexplored, despite its importance for real-time, resource-constrained, and sustainable AI
- [WerDATA24] presents a framework for performance in drift detection including 1) computational complexities 2) benchmarking 3) performance analysis

APPROACH

COMPUTATIONAL COMPLEXITIES

- Complexities important for analyzing a DD's behavior concerning runtime and memory
- Highlight factors that influence time/space complexity behavior
- Determined for different DDs and published in [WerDATA24]

BENCHMARKING

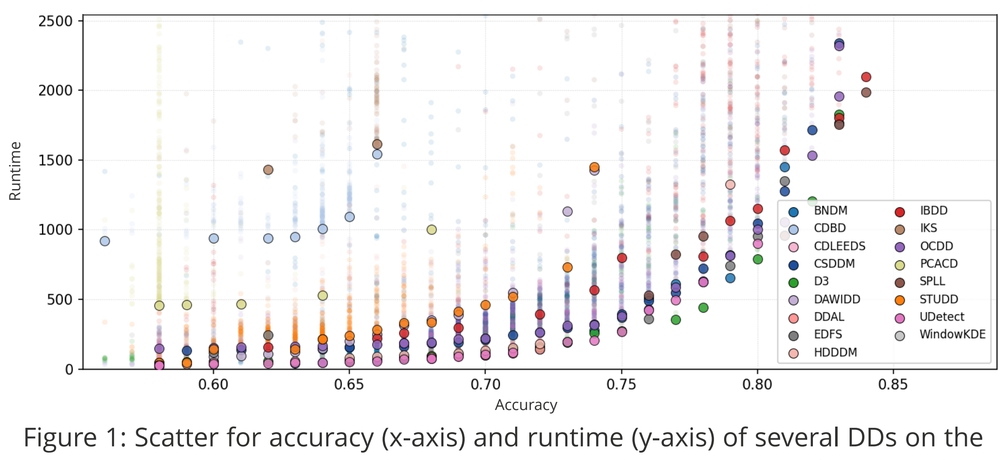


Figure 1: Scatter for accuracy (x-axis) and runtime (y-axis) of several DDs on the ForestCovertype dataset. Each point corresponds to one specific DD with a hyperparameter configuration. Bold points are Pareto-optimal points per DD.

DOCTORAL PROJECT'S MAIN SOFTWARE ARTIFACTS AND PAPERS

Benchmark website [WerJDSA26]

MoPEDDs @ Github [WerSCA27]

JUmPER @ Github [WerSC24]

- Benchmarking computational performance in data science is inherently a multi-objective optimization problem (e.g. runtime vs accuracy)
- [WerJDSA26] uses Bayesian optimization to identify trade-off configurations between runtime and accuracy (Pareto-optimal configurations of DDs)
- The developed benchmark leverages 9 diverse real-world datasets and 20 DDs, producing the following main outcomes:
  - Global trend: higher accuracy needs higher runtime
  - DDs can improve accuracy and runtime of ML-model based systems in data streams
  - Various DDs provide good accuracy but no single DD is superior to all others
  - Parallel resources are not used efficiently by SOTA DD implementation

PERFORMANCE ANALYSIS

- Instrumentation with Score-P and analysis with Vampir revealed that underlying libraries, e.g. BLAS, OpenMP are inefficiently invoked in DD streaming scenarios
- [WerSCA27] develops the **Mo**dular **P**arallel **E**nsemble for **D**rift Detector**s** (MoPEDDs) to utilize parallel resources and improve drift detection
- MoPEDDs shows promising results in both directions:
  - Single ensemble member decisions combined yielding higher accuracy
  - Achieved 13.2× speedup on 16 threads (82% efficiency) under realistic workload conditions on an AMD EPYC 7702

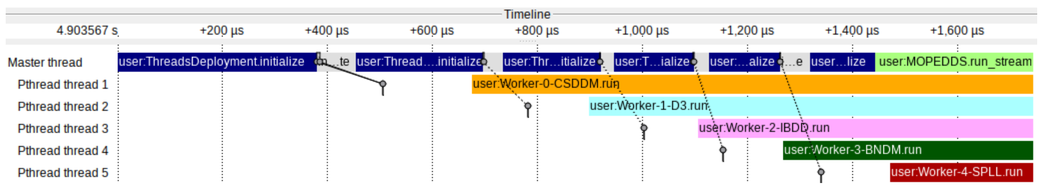


Figure 2: Startup of MoPEDDs with five ensemble members. X-axis shows the time and y-axis the spawned Pthreads for running the single DDs (orange=CSDDM, cyan=D3, purple=IBDD, green=BNDM, red=SPLL) over the stream. The Master thread initializes the single DDs (blue) and handles the main communication and ensemble results aggregation (lime).

THE BIGGER PICTURE: PERFORMANCE ENGINEERING IN DATA SCIENCE

- Computational performance often secondary objective in data science and HPC performance tools often not well embedded in data science tools, e.g. Jupyter
- **JU**pyter **m**eets **PER**formance (JUmPER) integrates computational performance perspectives in Jupyter via interactive performance plots for the user's code [WerSC24]
- JUmPER's main features include:
  - Support of any programming language and pre-existing Jupyter kernel
  - Automatic and configurable collection of system metrics (e.g. CPU util, GPU util, I/O, ...) with low monitoring overhead (up to 4Hz, negligible runtime overhead)
  - Score-P instrumentation for Python kernels (7% overhead for reference application)

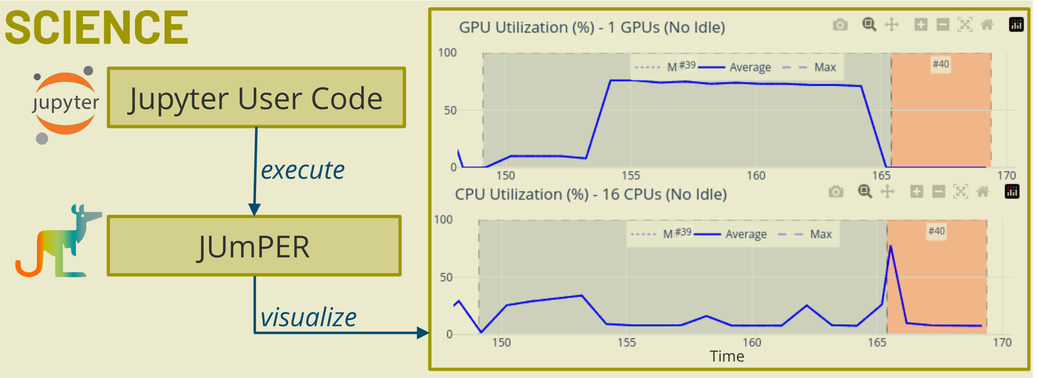


Figure 3: JUmPER provides interactive performance plots for Jupyter user code. Different colors in the performance plots represent different code segments (cells) in a Jupyter data science application.

RÉSUMÉ AND OUTLOOK

- Computational performance investigations are manifold and require awareness and accessibility of respective tools and methods
- This PhD project conducts such analysis for concept drift detection with three main contributions: 1) a multi-objective benchmark, 2) a modular parallel ensemble for drift detection, 3) a universal Jupyter-based performance analysis tool for data science that encompasses the broader vision of the thesis
- **Future directions:** real-world big data use cases for parallel DDs, benchmark-driven DD configuration, user studies on performance engineering in data science

[WerDATA24] Werner et al., DATA 2024. DOI:10.5220/0012758600003756 | [WerJDSA26] Werner et al., Int J Data Sci Anal, 2026. DOI:10.1007/s41060-026-01260-0 | [WerSCA27] Werner et al., SCA27 (in-review) | [WerSC24] Werner et al., SC24-W, 2024. DOI:10.1109/SCW63240.2024.00250

https://scads.ai/

CONTACT
Elias Werner
elias.werner@tu-dresden.de

SC26 Chicago, IL | hpc unites.

IN COOPERATION WITH

UNIVERSITÄT LEIPZIG

Technische Universität Dresden

FUNDED BY

Bundesministerium für Forschung, Technologie und Raumfahrt

SACHSEN

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.