HIGH-PERFORMANCE AND PORTABLE ML INFERENCE USING SOFIE

SANJIBAN SENGUPTA
Doctoral Student at CERN, University of Manchester

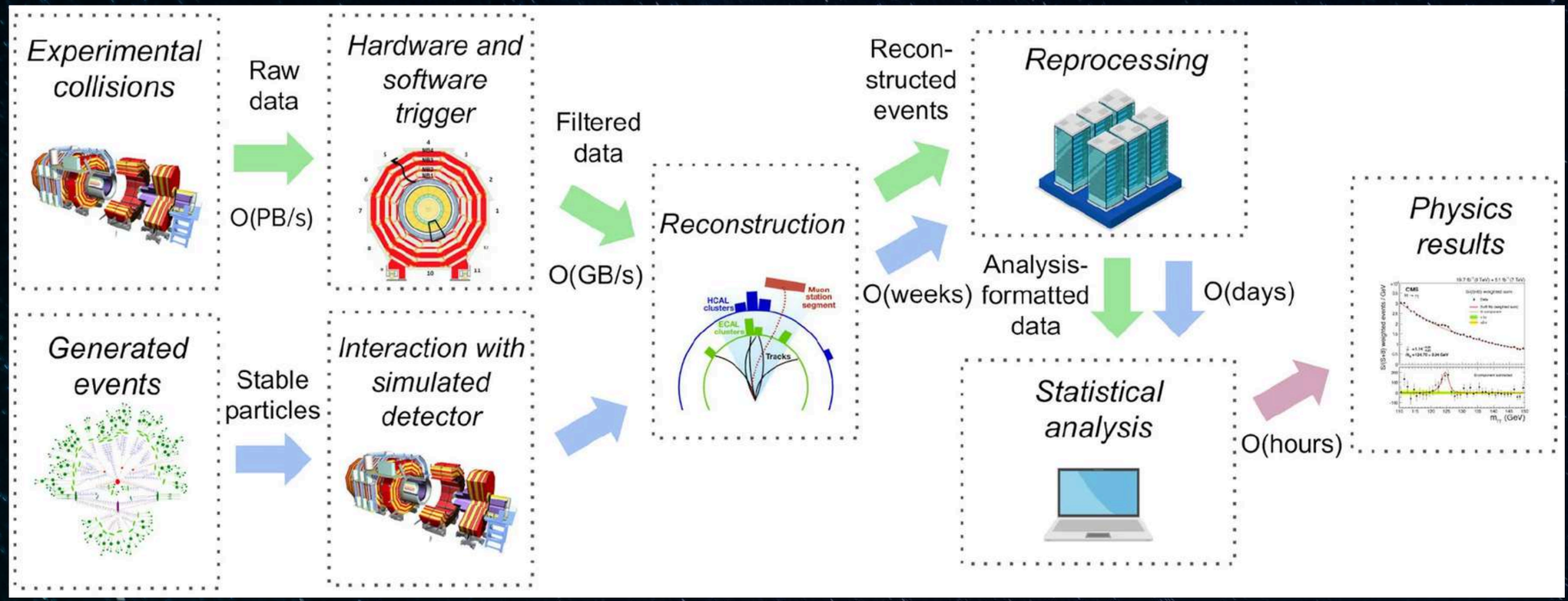
together with,
LORENZO MONETA, HARSH CHAUHAN, SHAUN LEE
ANTREAS IOANNOU, JONAS REMBSER

EP-SFT
Software Frameworks and Tools

NexTGen
Next Generation Triggers

MANCHESTER 1824
The University of Manchester

ML IN HEP

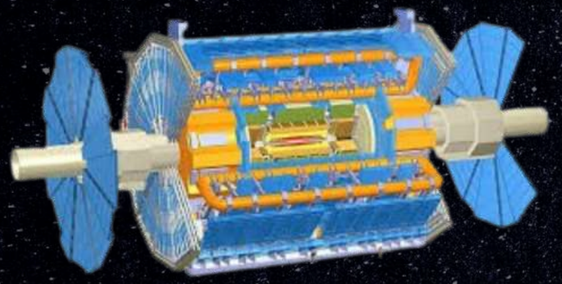


Šimko T, Heinrich LA, Lange C, Lintuluoto AE, MacDonell DM, Mečionis A, Rodríguez Rodríguez D, Shandilya P and Vidal García M (2021) Scalable Declarative HEP Analysis Workflows for Containerised Compute Clouds. Front. Big Data 4:661501. doi: 10.3389/fdata.2021.661501

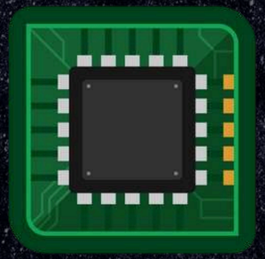
TRIGGERS

dealing with new collision data ~40 million times a second

Detector



L1 Trigger



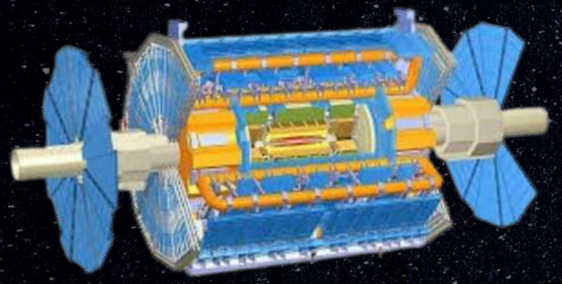
HL Trigger

Offline analysis

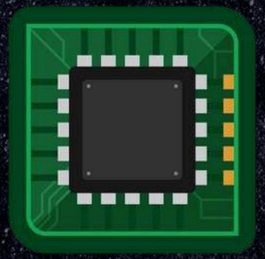
TRIGGERS

dealing with new collision data ~40 million times a second

Detector



L1 Trigger



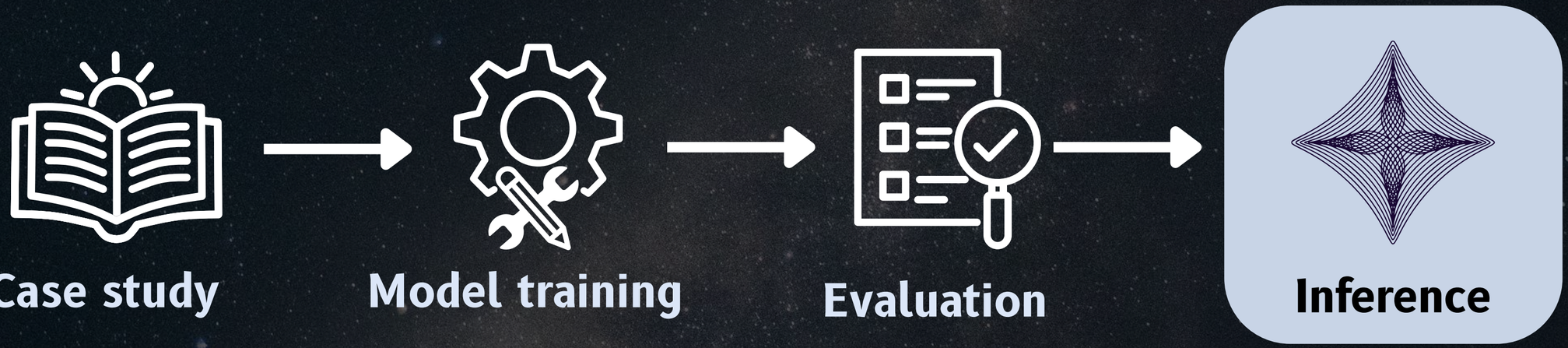
HL Trigger

Offline analysis

ML models are used in the HLTs!

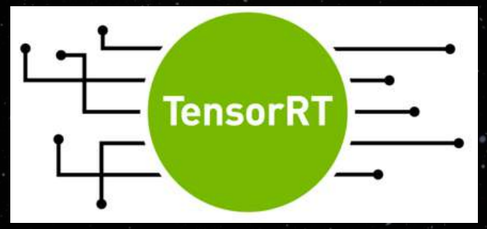
ML PARADIGM

ML PARADIGM



ML INFERENCE

Libraries that allow inferring trained models.



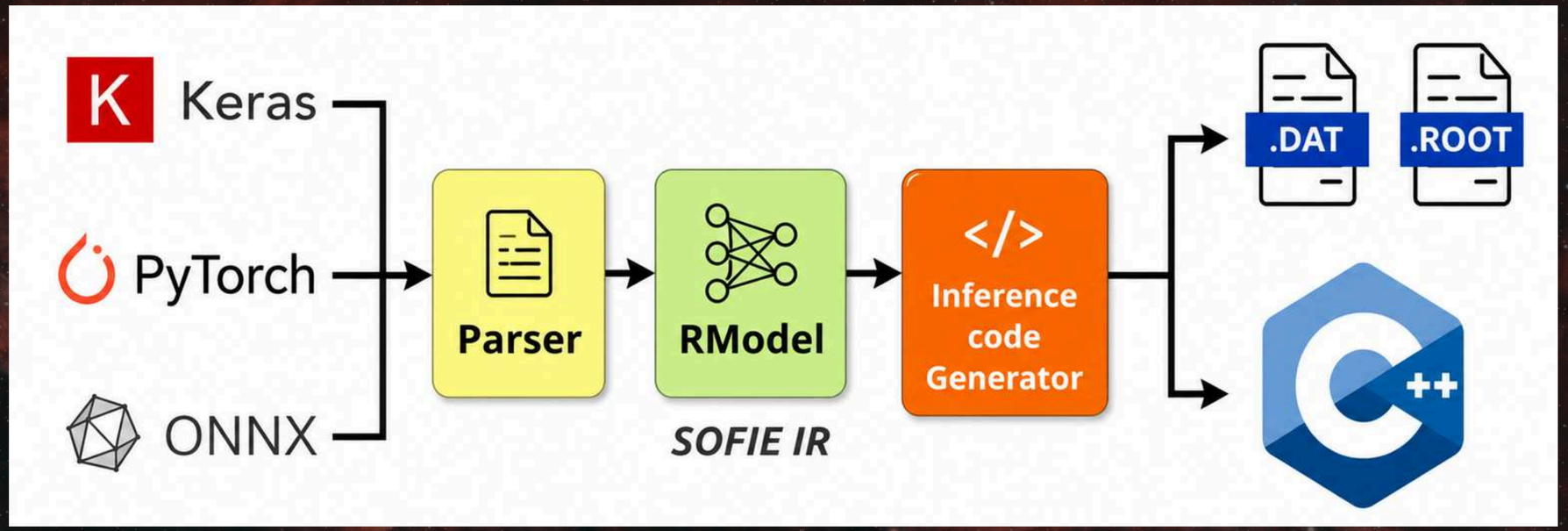
AMD
ROCm

inference through code generation.

SOFIE

System for Optimized Fast Inference code Emit

SOFIE architecture



- Parses a trained model in ONNX, Keras or PyTorch format to its IR (based on ONNX standard)
- Generates inference code in the form of C++ functions (only dependent on BLAS).
- Supports several ONNX operators, capable of inferring Transformers and GNNs.
- Optimizations for inference on CPU and GPU showed significant performance improvements.

SOFIE user interface

for code generation

```
SOFIE::RModelParser_ONNX parser;
SOFIE::RModel model = parser.Parse("Linear_16.onnx");
model.Generate();
model.OutputGenerated("Linear_16_FromONNX.hxx");
```

SOFIE user interface

inference

```
#include "Linear_16_FromONNX.hxx"

int main() {
    std::vector<float> input(1600);
    std::fill_n(input.data(), input.size(), 1.0f);
    TMVA_SOFIE_Linear_16::Session s("Linear_16_FromONNX.dat");
    std::vector<float> output = s.infer(input.data());
    return 0;
}
```

Include generated header file

model inputs

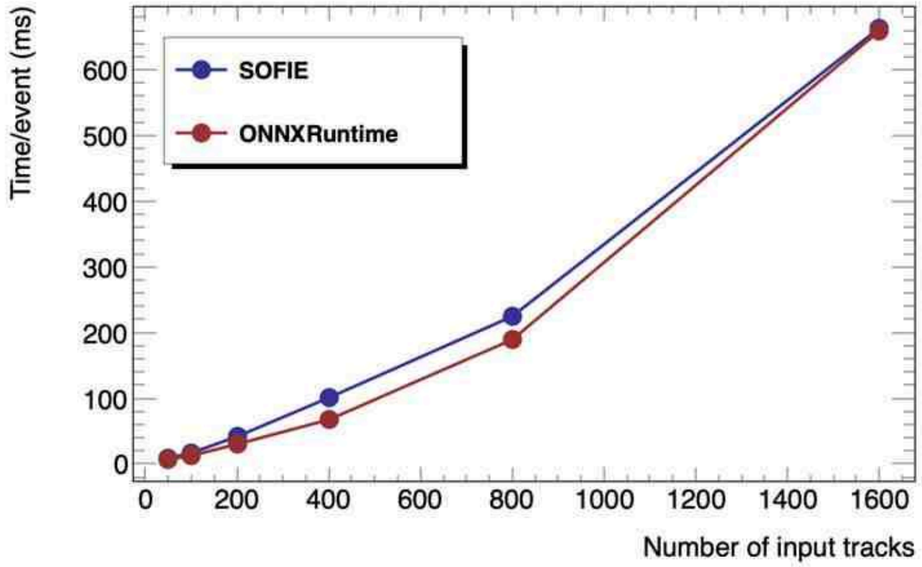
Instantiate session object that loads weights, run validations, etc.

Method for inference

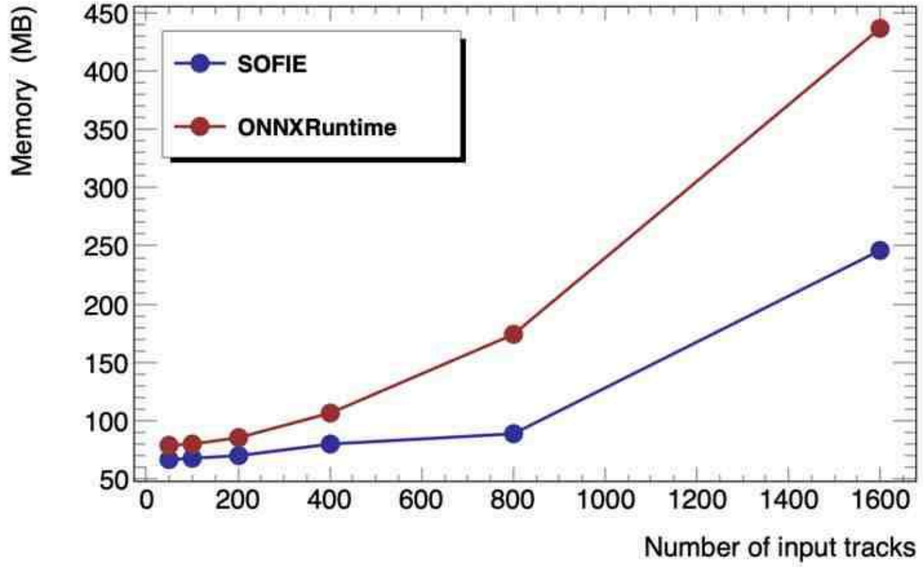
SOFIE Benchmarking on CPUs

ATLAS Jet Tagger GN2

Latency



Memory

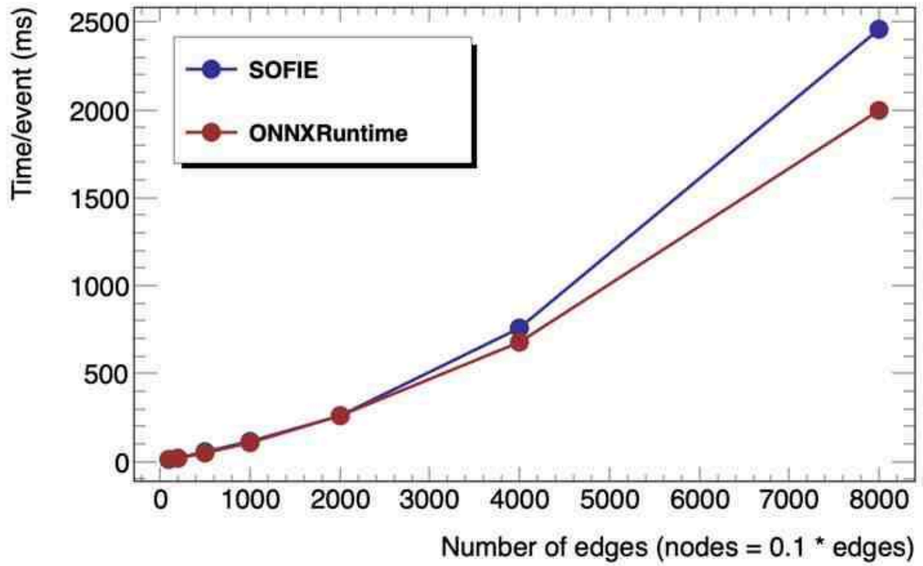


Tests run on
Intel Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz
1 single thread

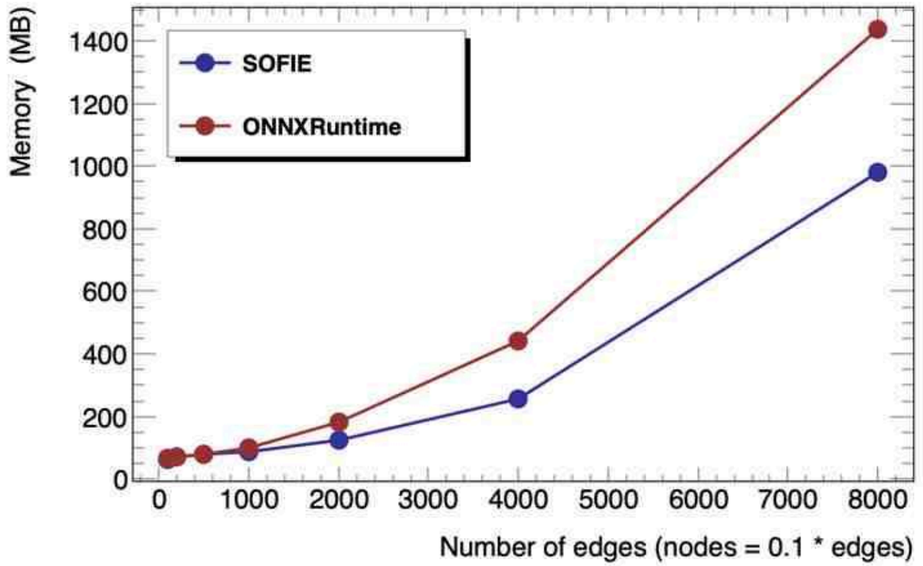
SOFIE Benchmarking on CPUs

CMS Jet Tagger ParticleNet

Latency



Memory

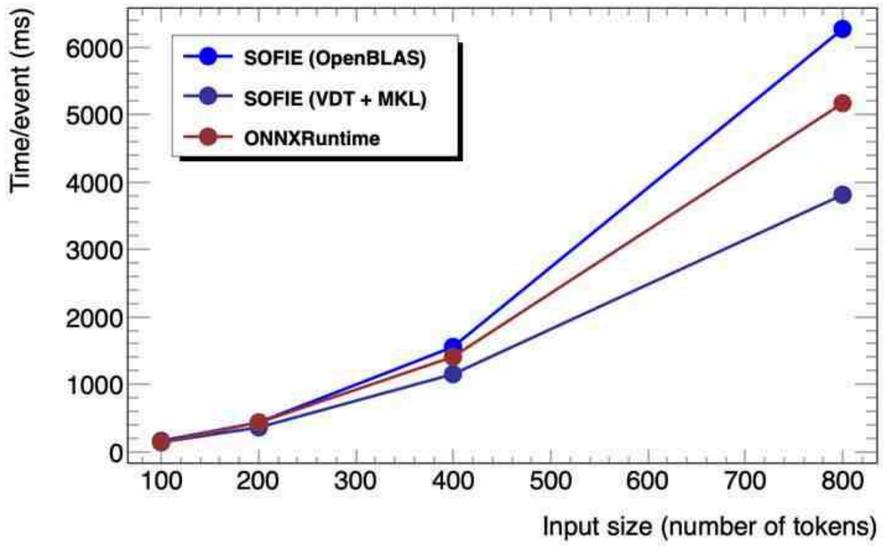


Tests run on
Intel Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz
1 single thread

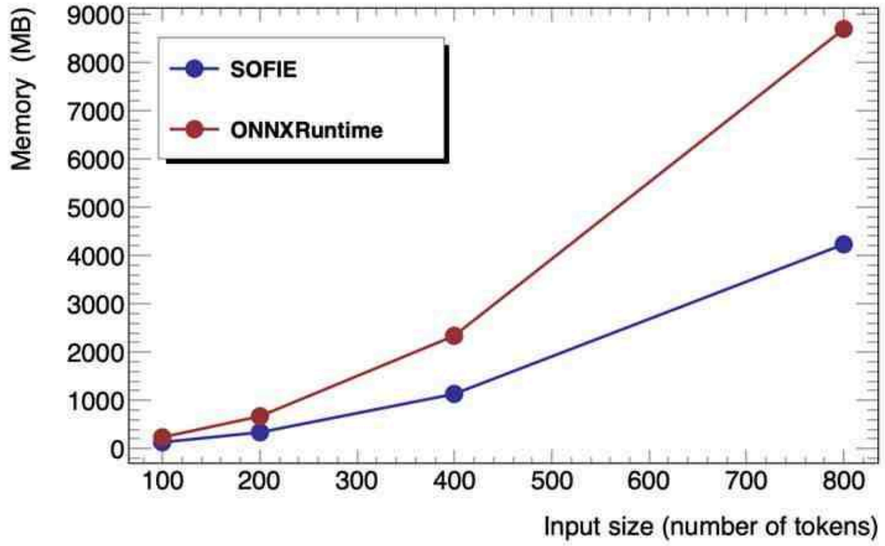
SOFIE Benchmarking on CPUs

Simple Transformer model – Attention on 8 heads

Latency



Memory



Tests run on
Intel Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz
1 single thread

SOFIE Benchmarking on CPUs

ATLAS GNN4ITK

Inference backend	SOFIE [s]	ONNX [s]
Graph construction	12.8	12.8
GNN 128/fp32	8.8	19.5
GNN 32/fp32	2.2	4.1
Track building	0.02	0.02

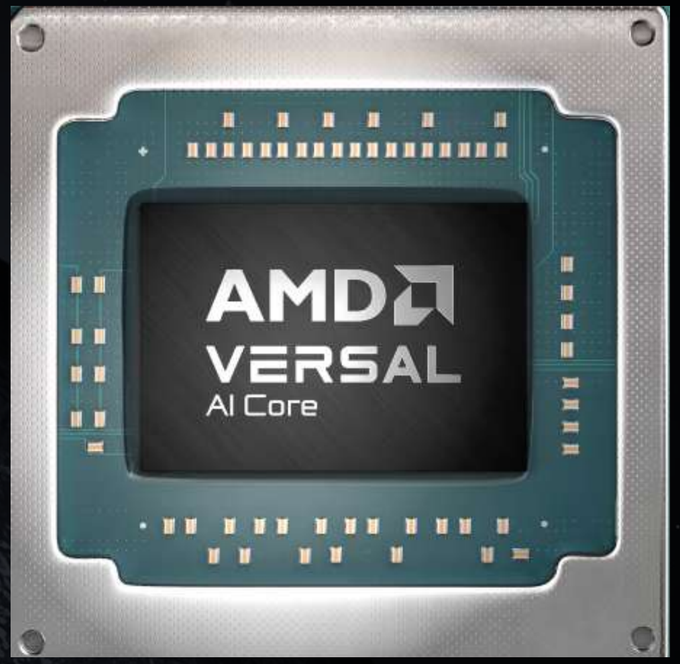
Benchmarking ATLAS model for track-finding (in collaboration with the ATLAS teams of NGT)

- Tested within ACTS.
- Single-event Inference, embedded in the full tracking chain.
- 20 events from the EF-tracking ttbar PUP200 sample run for each configuration.
- Prototype code can be found here: *github.com/sanjibansg/sofie-atlas-tracking*

HETEROGENEOUS ARCHITECTURES

GPUs

FPGAs



AI Engines

HETEROGENEOUS ARCHITECTURES

Pros

- Parallelism
- High-throughput
- Specialized computations

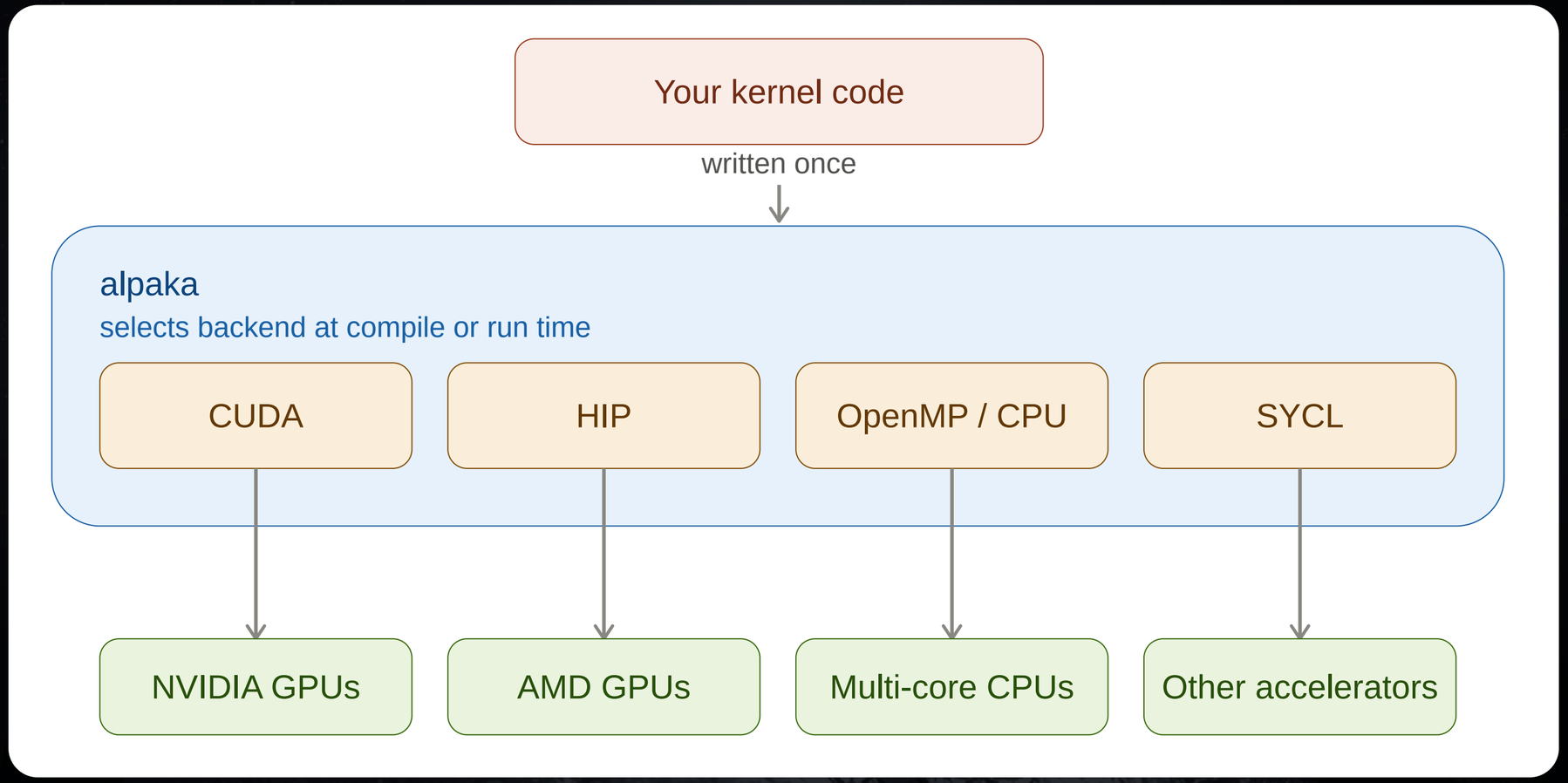
Cons

- Specialized code
- Portability
- Configurations

HETEROGENEOUS ARCHITECTURES

Header-only abstract interface for programming heterogeneous architectures

HETEROGENEOUS ARCHITECTURES



HETEROGENEOUS INFERENCE WITH SOFIE

using alpaka (v2.2)

- Accepts and returns abstract buffers.
- No intermediate layout transformation required
- Abstract Inference!
  - user has the control of running it on Intel, NVIDIA, or AMD GPUs

SOFIE alpaka user interface

Same code can be called for inference on different architectures just by changing this device tag

```
// Instantiate session to load weights
SOFIE_Linear_16::Session<alpaka::TagGpuCudaRt> session;


// Host device setup
alpaka::PlatformCpu hostPlatform{};
auto host = alpaka::getDevByIdx(hostPlatform, 0u);


// Select GPU device + queue
alpaka::PlatformCudaRt platform{};
alpaka::DevCudaRt device = alpaka::getDevByIdx(platform, 0u);
alpaka::Queue<...> queue{device};


// Allocate buffers & transfer data
auto A_d = alpaka::allocBuf<float, Idx>(device, ...);
alpaka::memcpy(queue, A_d, A);


// Inference execution
auto result = session.infer(A_d);
```

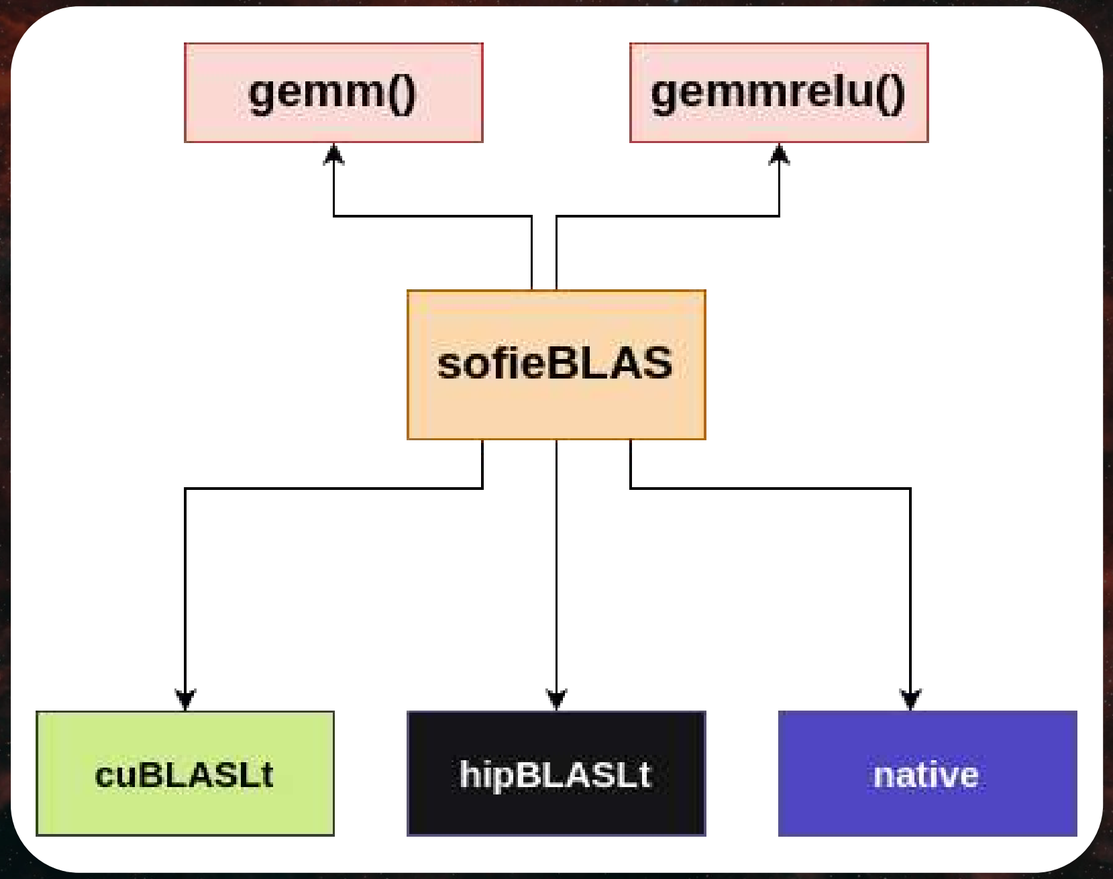
Instantiate session object that loads model weights, runs validations, and prepares the model for inference

Preparing model inputs

Method to run inference

SOFIE BLAS INTERFACE

- Unified Interface
- Common C++ API over multiple BLAS backends.
- Heterogeneous Support
  - CPU: OpenBLAS, MKL
  - GPU: cuBLASlt, hipBLASlt, native
- Dynamic operations support via LRU caching

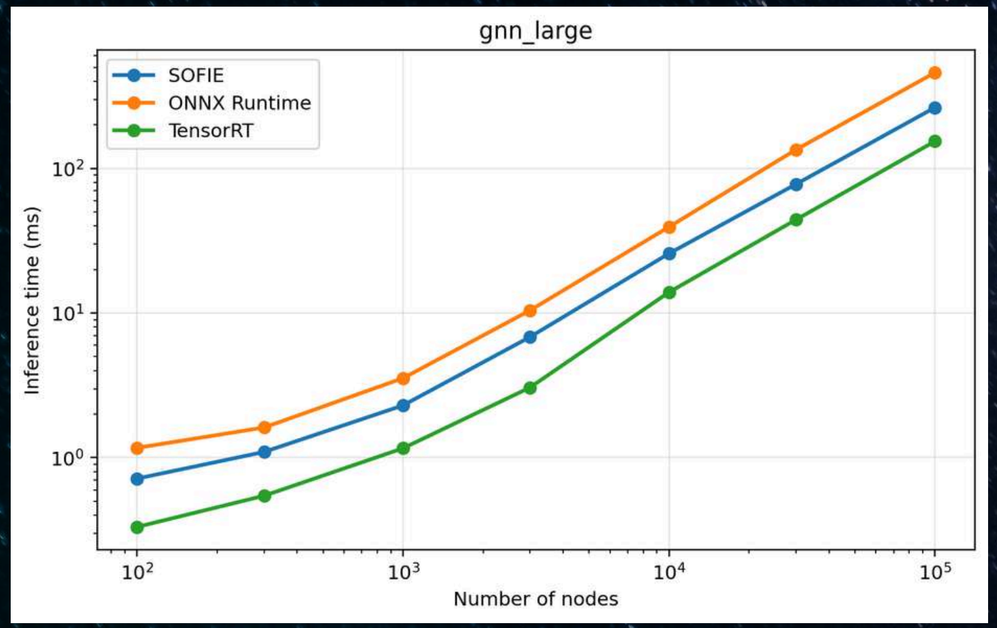


Thanks to Andrea Bocci (CMS) for the initial development support of sofieBLAS.

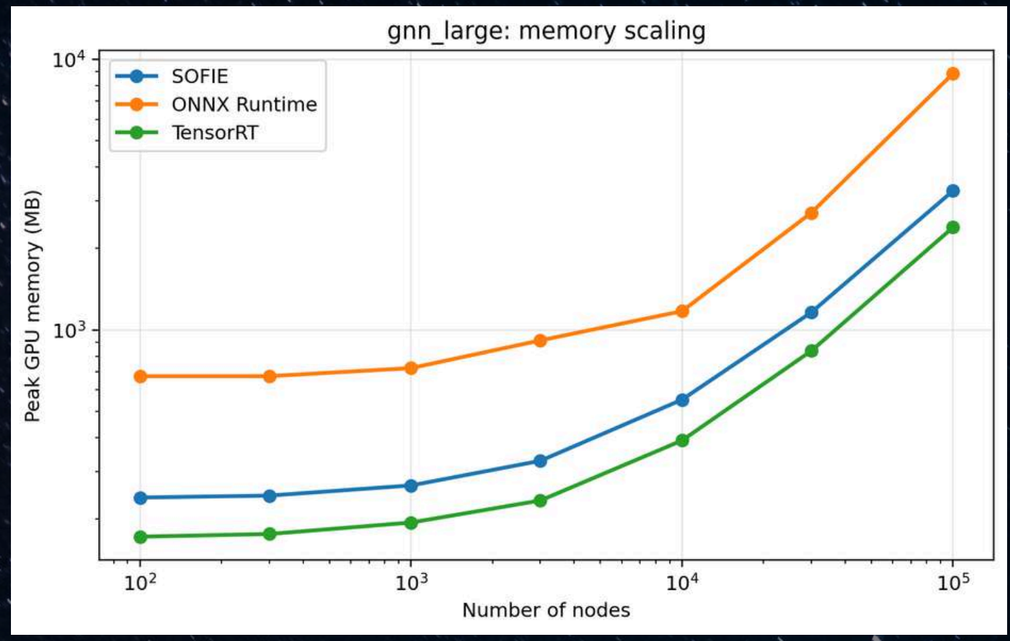
SOFIE Benchmarking on GPUs

Experimented performance on several ML models used in experiments against TensorRT and ONNXRuntime.

ATLAS GNN4ITk



Latency



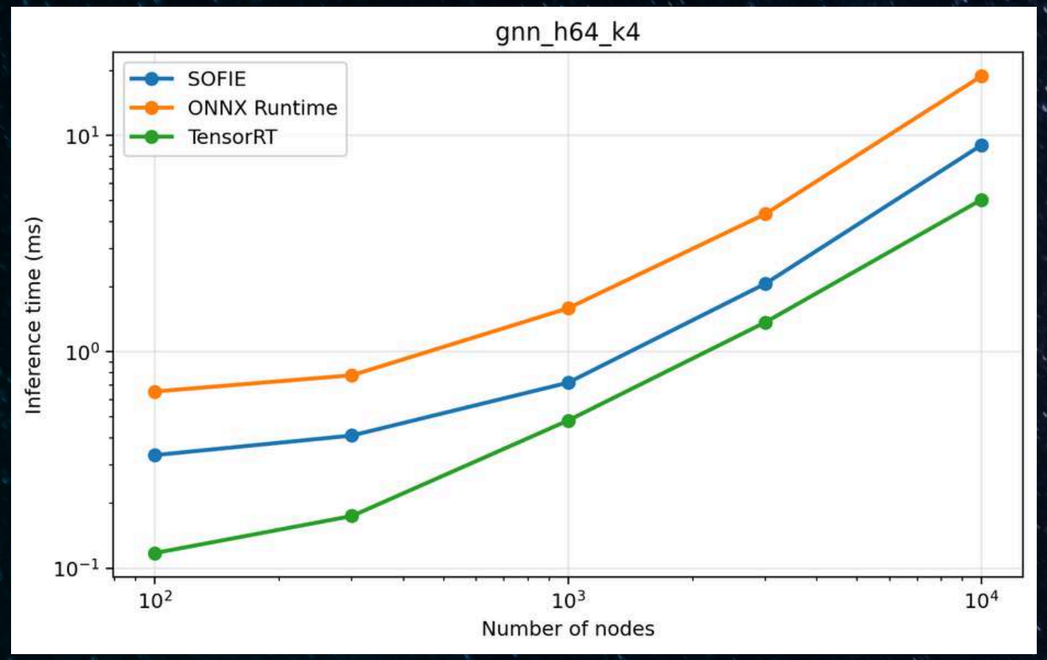
Memory

Measured on NVIDIA GPU GeForce RTX 5060 Ti

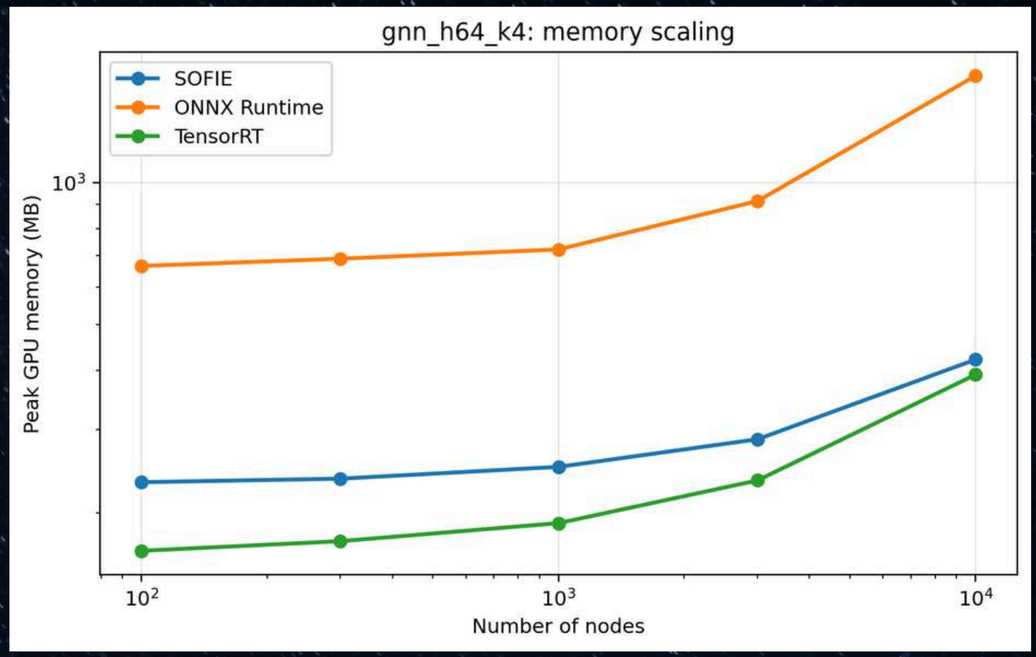
SOFIE Benchmarking on GPUs

Experimented performance on several ML models used in experiments against TensorRT and ONNXRuntime.

Reconstruction models currently in development in CMS



Latency

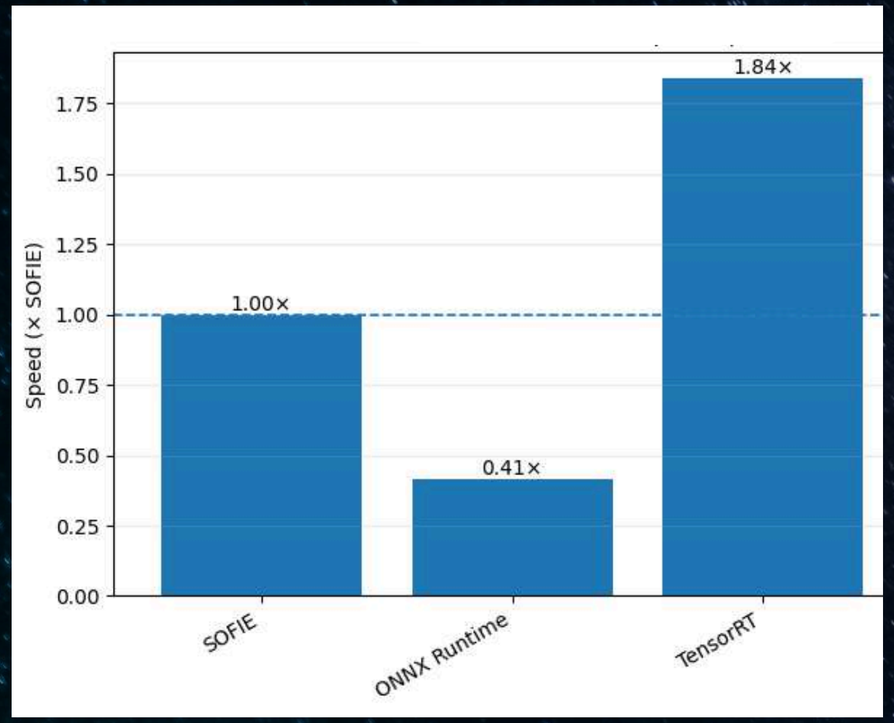


Memory

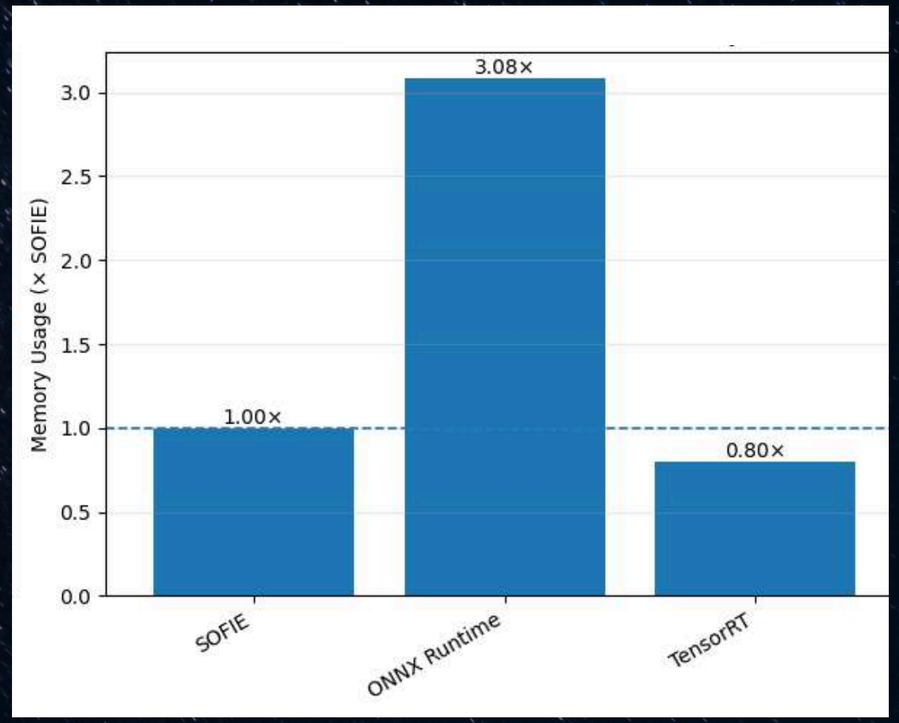
Measured on NVIDIA GPU GeForce RTX 5060 Ti

SOFIE Benchmarking on GPUs

Comparing SOFIE with TensorRT and ORT, averaged over 47 Models comprising of Transformers and GNNs for purposes like tracking.



Inference speedup

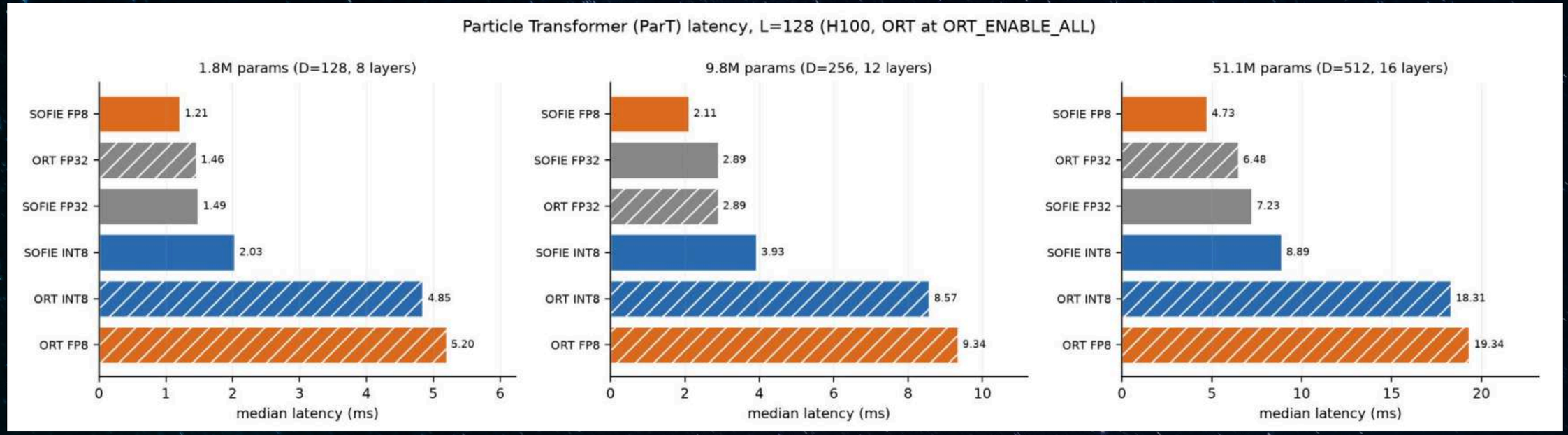


Memory

Measured on NVIDIA GPU GeForce RTX 5060 Ti

SOFIE Benchmarking on Quantized models

Experimented against ONNXRuntime for ParT in three sizes, comparing FP32, FP8 and INT8.
SOFIE FP8 latency improvements grew from 20% to 4x in comparison with FP32.

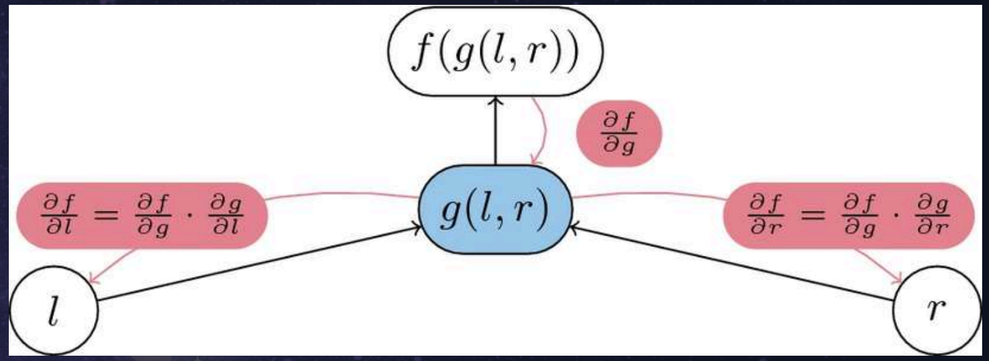


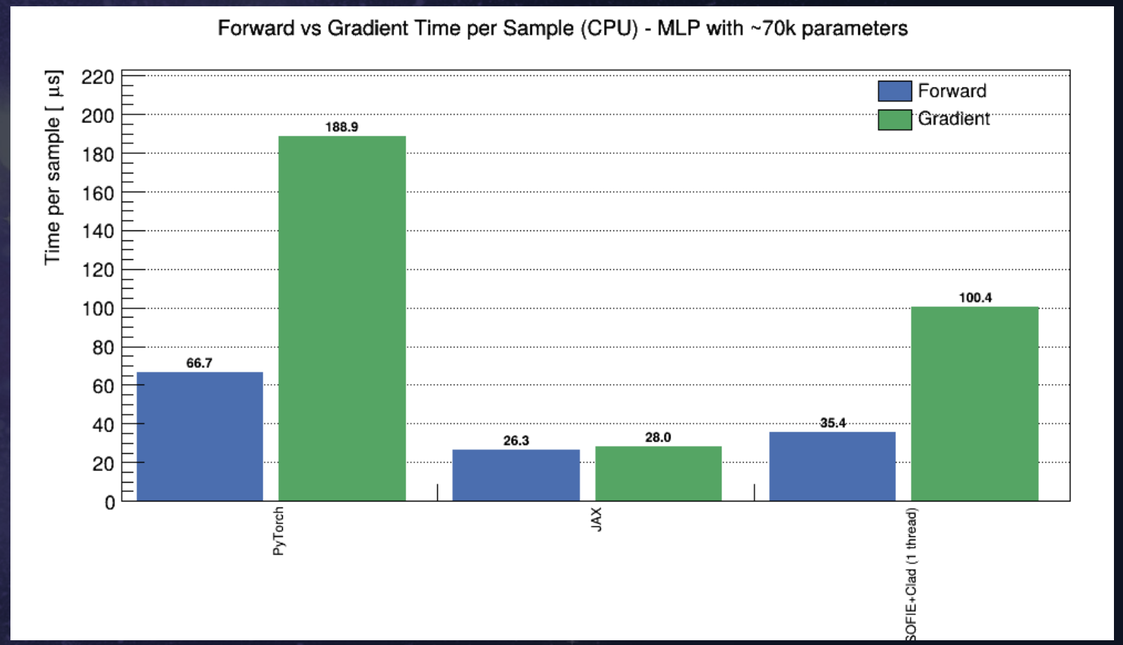
Measured on a single H100, with batch size 32 throughout

SOFIE USAGE

INTEGRATION IN ROOFIT

- Leverage Neural Simulation Based Inference to increase sensitivity
  - Construct probability density functions and likelihood functions using ML models
- Support for NSBI is available now in RooFit using SOFIE
  - With SOFIE we can leverage the Automatic Differentiation capabilities available in CLAD/ROOT

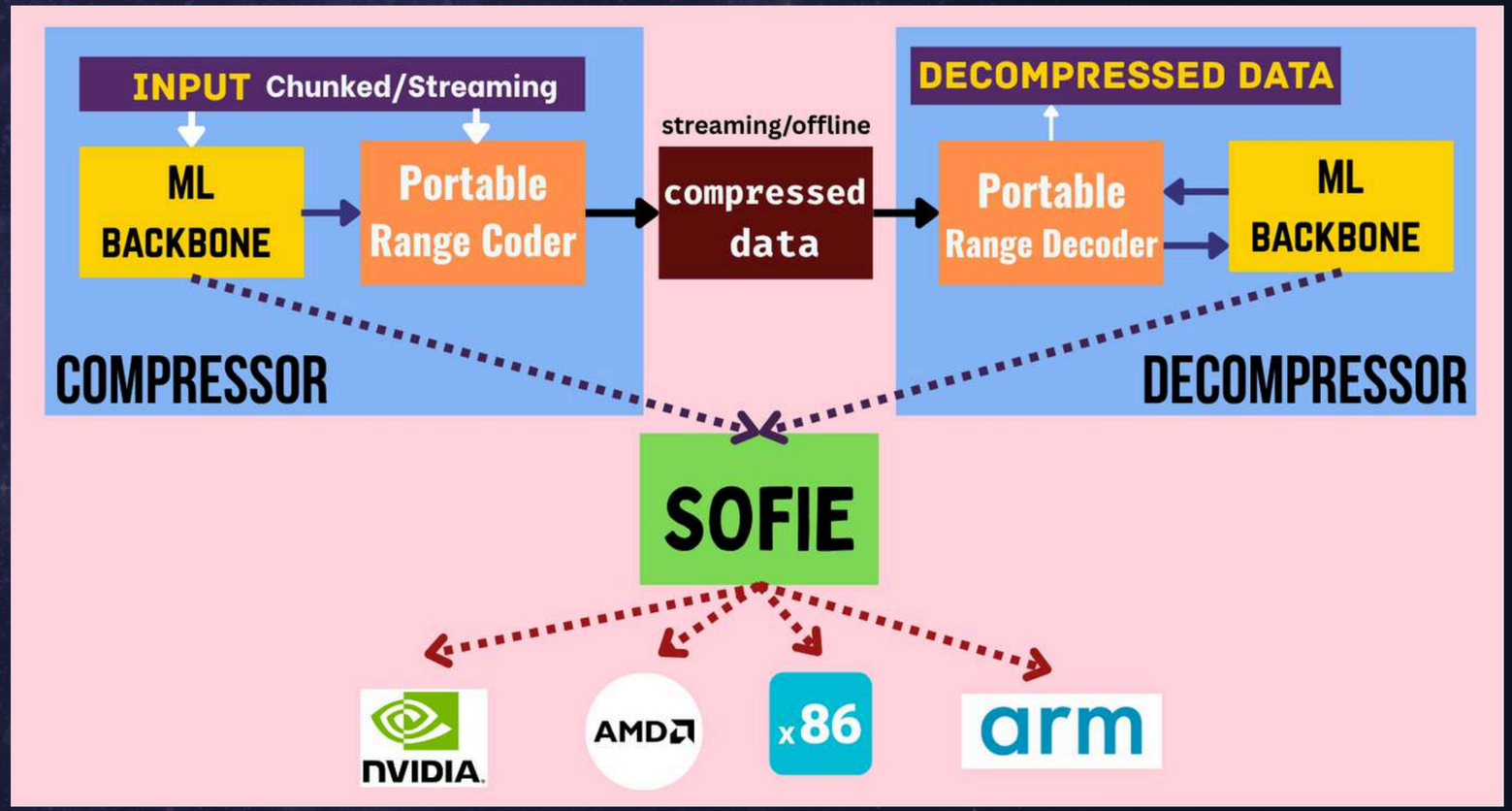




SOFIE USAGE

INFERENCE IN BOA THROUGH SOFIE

- BOA is a Mamba-based lossless neural compressor for scientific data.
  - It uses the MAMBA SSM model to capture patterns in a dataset and the Range Coder for the encoding/decoding process.
- SOFIE shall help in accelerating inference and hardware portability.



SOFIE CONCLUSION & FUTURE DIRECTIONS

- SOFIE is capable of performing optimized inference on CPU and GPU while only being dependent on BLAS libraries, on models used in HEP Experiments.
- R&D on further optimization of CPU and GPU Inference.
- Quantization and Low-rank factorizations
- Exploring application of SOFIE's heterogeneous implementation on other avenues.
- Welcoming collaboration with users for optimizing their ML Inference pipelines.

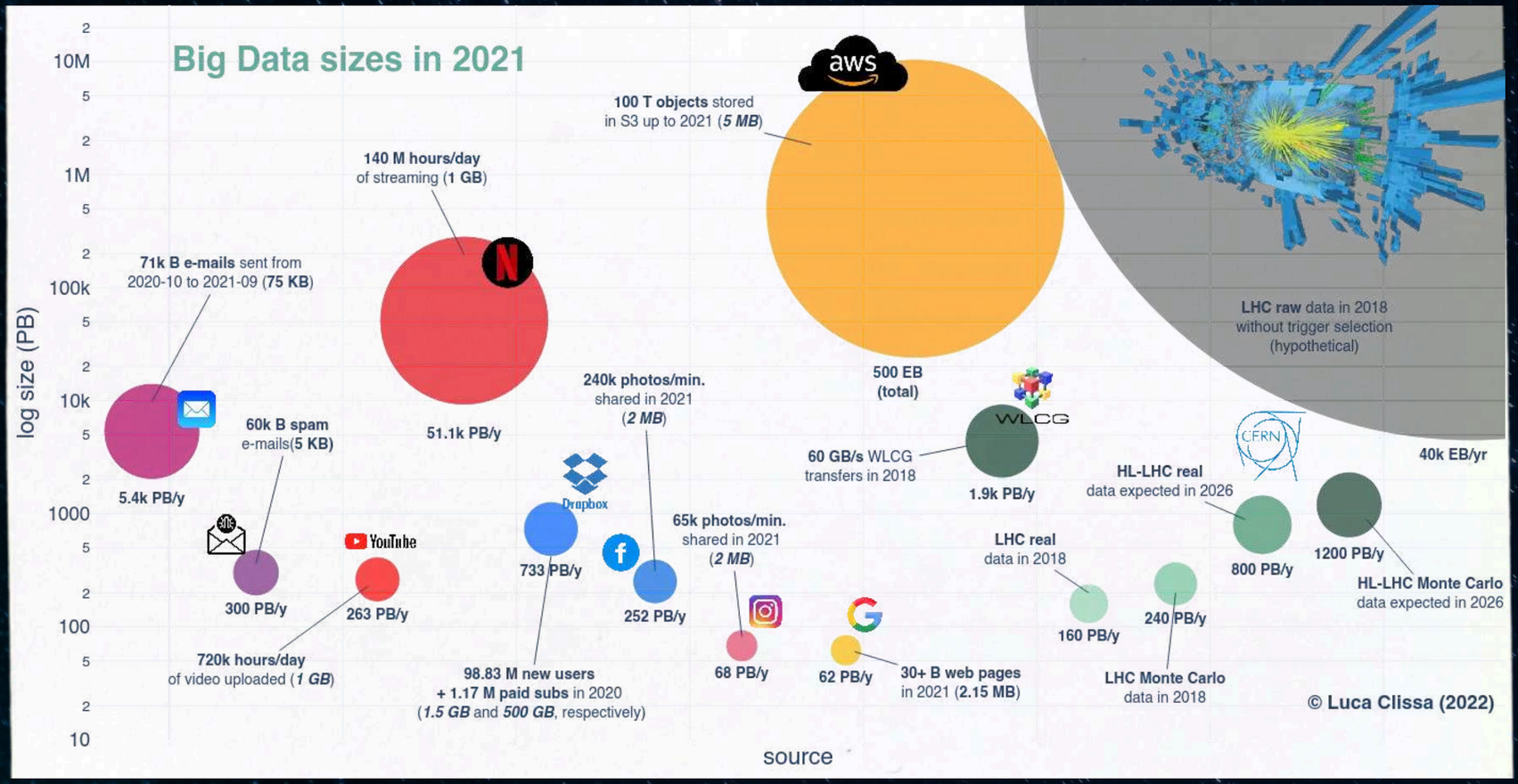
thanks, questions?

SANJIBAN SENGUPTA
<sanjiban.sengupta@cern.ch>

SOFIE for CPU Inference available in ROOT
github.com/root-project/root/tree/master/tmva/sofie

SOFIE for Inference on heterogeneous architectures
github.com/ml4ep/sofie

DATA IN HEP

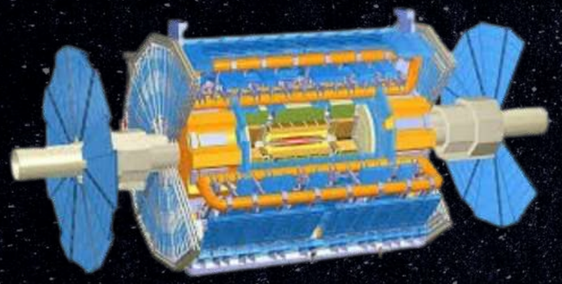


Clissa L, Lassnig M and Rinaldi L (2023) How big is Big Data? A comprehensive survey of data production, storage, and streaming in science and industry. Front. Big Data 6:1271639. doi: 10.3389/fdata.2023.1271639

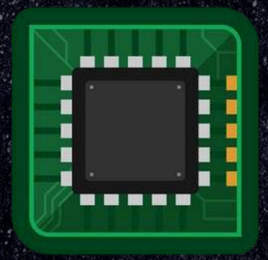
TRIGGERS

dealing with new collision data ~40 million times a second

Detector



L1 Trigger



HL Trigger

Offline analysis

Optimization techniques

- *Efficient operations*
- *Minimal data movements*
- *Data Compression*