ROOT
Data Analysis Framework
https://root.cern

CERN

Fast Inference of Machine Learning Models with SOFIE

Lorenzo Moneta, Ioanna Panagou, Sanjiban Sengupta

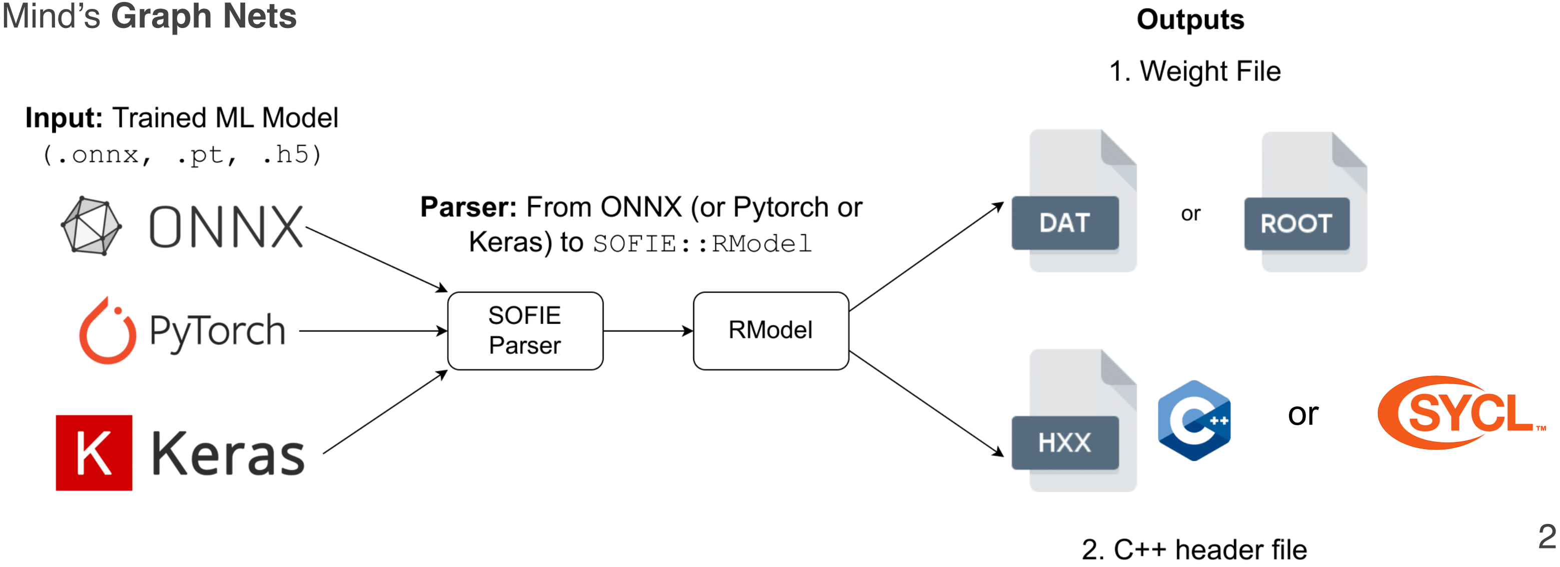
EUROPEAN AI FOR FUNDAMENTAL PHYSICS CONFERENCE
EuCAIFCon 2024

Machine Learning Inference in ROOT

SOFIE : System for Optimised Fast Inference code Emit

- **Input**: trained ML model file
  - **ONNX**: Common standard for ML models
  - **Tensorflow/Keras** and **PyTorch** models (with reduced support than ONNX)
  - Since 6.32 support message passing **GNNs** from DeepMind's **Graph Nets**

- **Output**: **generated C++ code**
  - Easily **invokable directly** from C++ (plug-and-use)
  - **Minimal dependency** (on BLAS only)
  - Can be **compiled at run time** using ROOT Cling JIT and can be **used in Python.**



Outputs

1. Weight File

Input: Trained ML Model `(.onnx, .pt, .h5)`

Parser: From ONNX (or Pytorch or Keras) to `SOFIE::RModel`

2. C++ header file

GPU Extension of SOFIE

- Extended SOFIE functionality to produce **GPU** code using **SYCL**

```
// generate SYCL code internally
model.GenerateGPU();
// write output header and data weight file
model.OutputGeneratedGPU();
```

- **Minimise overhead of data transfers** between host and device
- **Manage buffers efficiently, declaring them at the beginning**
- Use libraries for **GPU Offloading**: GPU BLAS from Intel one API and PortBLAS for other GPUs
- **Fuse operators** when possible in a single kernel
- **Replace conditional check** with relational functions

model.hxx

```
namespace TMVA_SOFIE_Linear_event{

struct Session {

Session(std::string filename ="") {
   if (filename.empty()) filename =
"Linear_event.dat";
   std::ifstream f;
   f.open(filename);
   // read weight data file
   ……………………..
}
std::vector<float> infer(float*
tensor_input1){
```

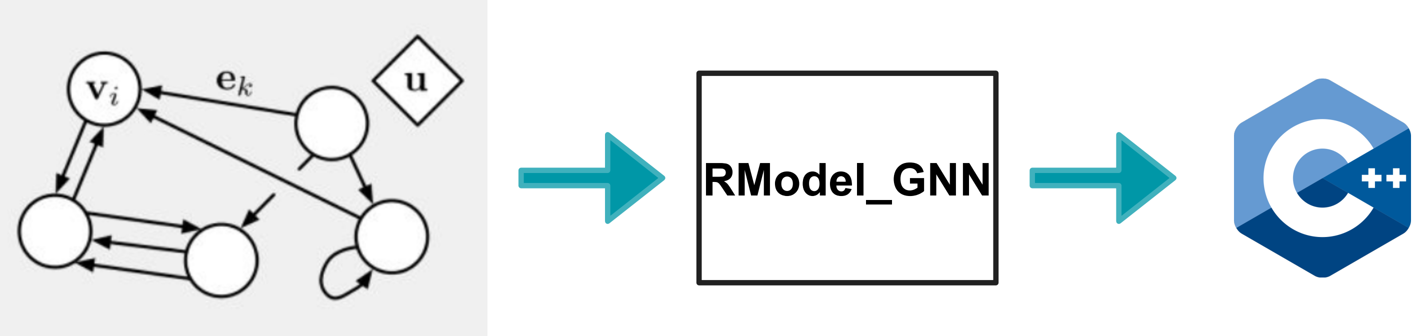
with SYCL code

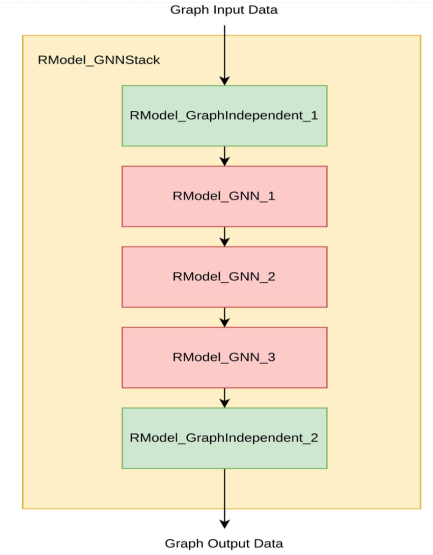
```
#include “Model.hxx”

// create session class

TMVA_SOFIE_Model::Session
ses(“model_weights.dat”);

//—- event loop

for (ievt = 0; ievt < N; ievt++) {

   // evaluate model: input is a C float array

   float * input = event[ievt].GetData();

   auto result = ses.infer(input);
```

Inference code needs to be linked against oneAPI MKL libraries and compiled using SYCL compiler

SOFIE GNN Support

- Since ROOT version 6.32 support inference of **GNNs**
  - parsing available for GNNs built from DeepMind’s Graph Net library
  - supporting a LHCb model for full event interpretation (arXiv:2304.08610)




Graph Input Data

RModel_GNNStack

RModel_GraphIndependent_1

RModel_GNN_1

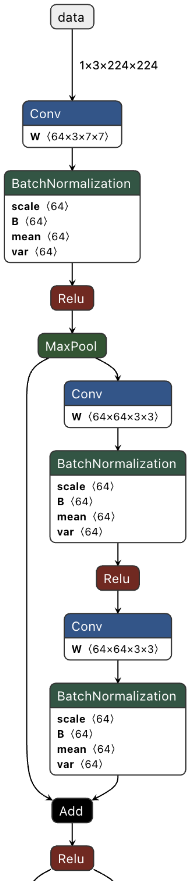
RModel_GNN_2

RModel_GNN_3

RModel_GraphIndependent_2

Graph Output Data

ONNX Supported Operators

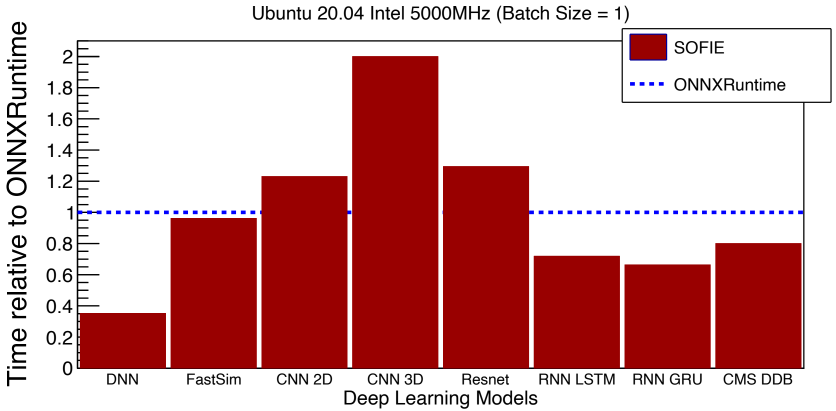


Operators implemented in ROOT	**CPU**	**GPU**
Perceptron: Gemm	✓	✓
Activations: Relu, Selu, Sigmoid, Softmax, Tanh, LeakyRelu, Swish	✓	✓
Convolution and **Deconvolution** (1D, 2D and 3D)	✓	✓
Pooling: MaxPool, AveragePool, GlobalAverage	✓	✓
Recurrent: RNN, GRU, LSTM	✓	✓
Layer Unary operators: Neg, Exp, Sqrt, Reciprocal, Identity	✓	✓
Layer Binary operators: Add, Sum, Mul, Div	✓	✓
Other Layer operators: Reshape, Flatten, Transpose, Squeeze, Unsqueeze, Slice, Concat, Reduce, Gather	✓	✓
BatchNormalization, LayerNormalization	✓	✓
Custom operator	✓	

- current CPU support available in **ROOT 6.30**
- GPU/SYCL is implemented in a ROOT PR

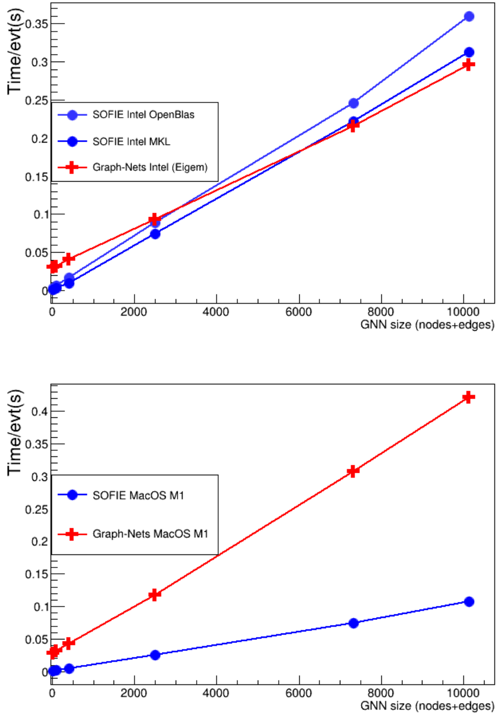
Benchmarking Time of Inference

CPU event performance of **SOFIE** vs **ONNXRuntime**



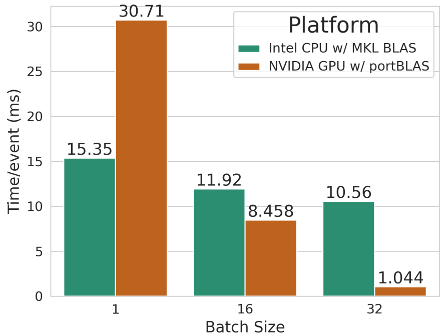
CPU time for **GNN inference**

- varying GNN size (node + edges)



GPU (SYCL) vs CPU performance

- using a Resnet model with varying batch size



Summary

- **SOFIE**, fast and easy-to-use inference engine for Deep Learning models, is available in ROOT
  - Can be easily integrated with other ROOT tools (*RDataFrame)* for ML inference in end-user analysis
  - Supporting several **ONNX** operators and also **GNN**s
  - A prototype implementation for **GPU** using **SYCL** has been developed
    - plan to extend to **CUDA** and/or **ALPAKA** following some interest by experiments to deploy in their GPU-based trigger system
- Future developments according to user needs and received feedback
  - aim to support the latest production model of experiments (GNN and transformers)
  - models used for fast simulations (GAN and VAE)

Useful Links

- **Examples and tutorials** are available in the tutorial/tmva directory
  - C++ (`TMVA_SOFIE_*.C`) and Python examples (`TMVA_SOFIE_*.py`)
- Link to **SOFIE code** in current ROOT master in GitHub
- Example **notebooks** on using SOFIE:
  - https://github.com/lmoneta/tmva-tutorial/tree/master/sofie
- Link to PR implementing SYCL code generation
- Link to benchmarks in *rootbench* repository