SOFIE: C++ Code Generation from ROOT/TMVA for Fast Deep Learning Inference

Sitong An, Lorenzo Moneta, Sanjiban Sengupta, Ahmat Hamdan, Federico Sossai, Aaradhya Saxena

ROOT Data Analysis Framework

ROOT 6.24 Experimental

Code Generation for Inference

- Accept ONNX, Keras, PyTorch & ROOT models
- Emit C++ code that can be easily included and invoked for fast inference of model
- Minimal dependency (BLAS/Eigen only)
- Modular. Users can easily add custom operators
- Thread-safe

Code Generation

```
using namespace TMVA::Experimental;
SOFIE::RModelParser_ONNX parser;
SOFIE::RModel model = parser.Parse("model.onnx");
model.Generate();  // generate output header and weights
model.OutputGenerated();
```

Inference

```
#include "model.hxx"
TMVA_SOFIE_model::Session s;
std::vector<float> x = {...};
auto y = s.infer(x.data());
```

Multithreaded Inference with RDataFrame

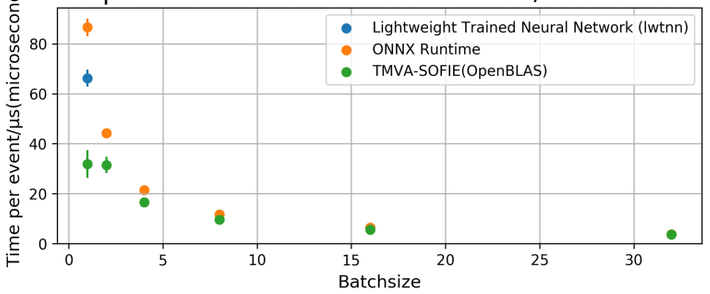
```
ROOT::RDataFrame df(treeName, fileName);
SofieFunctor<TMVA_SOFIE_model::Session> functor(n_threads);
// functor calls Session.infer() for each thread, see QR code for details
auto hist1 = df.DefineSlot("y",functor, col_names).Histo1D("y");
hist1->DrawClone();
```

- Supported Operators:
  - ✓ Gemm (linear layers)
  - ✓ Conv, Pool
  - ✓ RNN, GRU, LSTM
  - ✓ BatchNorm, InstanceNorm
  - ✓ Relu, Selu, Sigmoid…

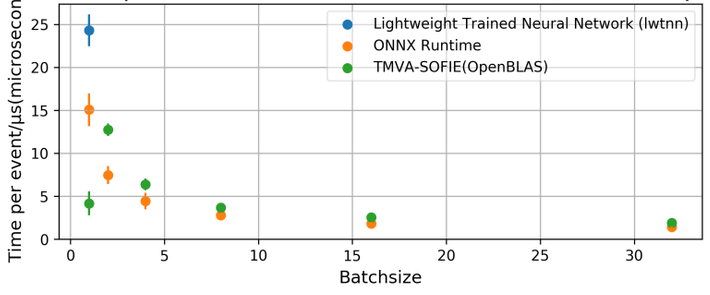
Benchmark Results

- Linear network runtime

Time per event for different batch size, cache flushed



Time per event for different batch size, cache kept



- RDataFrame runtime

SOFIE/µs	ONNXRuntime/µs	LWTNN/µs
3.2	8.0	8.1

- Convolutional network runtime

Model	SOFIE/ms	ONNXRuntime/ms
1xConv, Batch=1	**0.05**	0.08
14xConv, Batch=1	126	**100**
14xConv, Batch=32	50	**49**
Resnet18	44	**34**

Future Plan

- Further inference speed optimisation
- Expand operator support from users' demand
- Improve interoperability with RDataFrame

repo

rootbench

Benchmarked Models (see above QR code for onnx models)
Linear: 10x[Linear(50, bias=True) + ReLu]
RDataFrame: 5x[Linear(200, bias=True) + ReLu]
1xConv: Conv2d(1,2, (5,5)) + ReLu; (input_H, input_W) = (100, 100)
14xConv: 14x[Conv2d(N_channel, (5,5)) + ReLu]; N_channel 1->128->1

Acknowledgement
S. An. gratefully acknowledges the support of the Marie Sklodowska-Curie Innovative Training Network Fellowship of the European Commission Horizon 2020 Programme, under contract number 765710 INSIGHTS.
F. Sossai is a 2021 CERN summer student. S. Sengupta, A. Hamdan and A.Saxena are 2021 Google Summer of Code students with CERN-HSF.