TMVA SOFIE

Developing the Machine Learning Inference Engine

Sanjiban Sengupta
EP-SFT

Supervisor: Lorenzo Moneta

/sanjibansg /sanjiban-sengupta

ROOT TMVA

- Toolkit for Multivariate Analysis
- Provides a Machine Learning environment for training, testing and evaluation of multivariate methods.

SOFIE

SOFIE

SOFIE

System for

SOFIE

System for Optimized

SOFIE

System for Optimized Fast

SOFIE

System for Optimized Fast Inference

SOFIE

System for Optimized Fast Inference code Emit

SOFIE

System for **O**ptimized **F**ast **I**nference code **E**mit

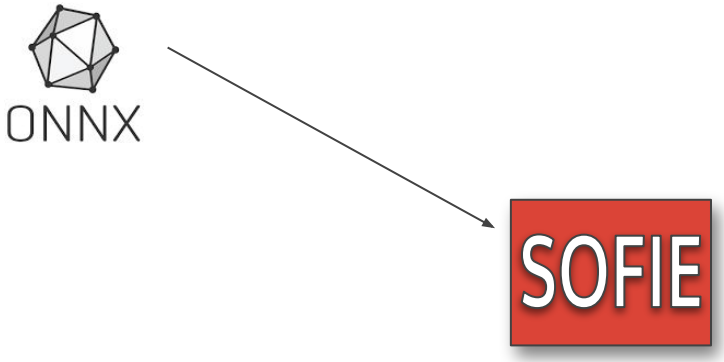
inference code, fast to operate, with least dependencies

TMVA SOFIE

- Intermediate representation following ONNX standards.
- Inference code generation with least latency and minimal dependency

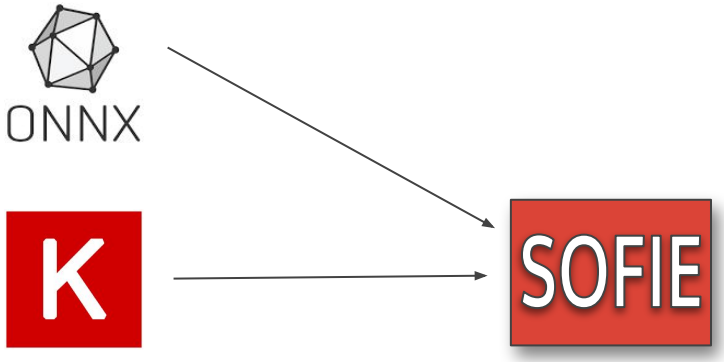
TMVA SOFIE

- Intermediate representation following ONNX standards.
- Inference code generation with least latency and minimal dependency



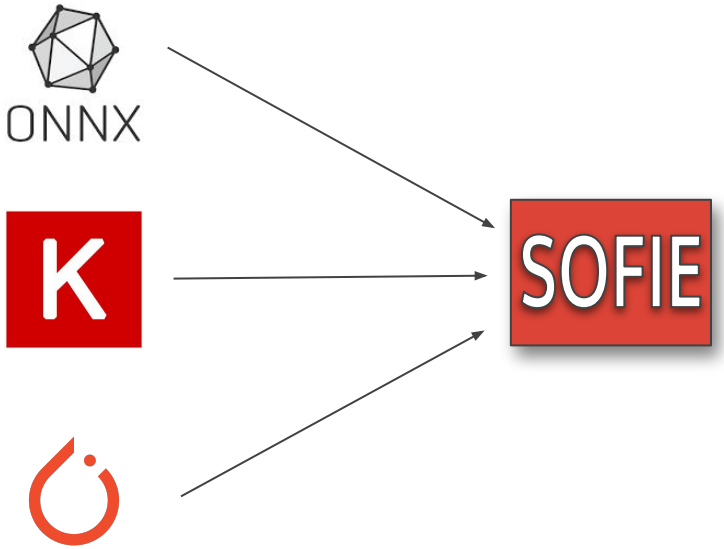
TMVA SOFIE

- Intermediate representation following ONNX standards.
- Inference code generation with least latency and minimal dependency



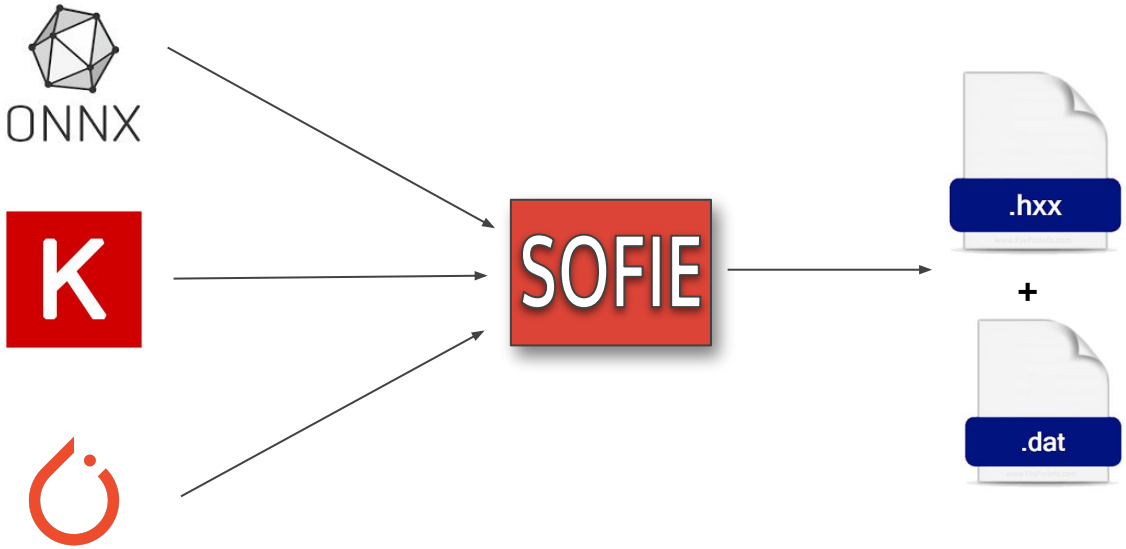
TMVA SOFIE

- Intermediate representation following ONNX standards.
- Inference code generation with least latency and minimal dependency



TMVA SOFIE

- Intermediate representation following ONNX standards.
- Inference code generation with least latency and minimal dependency



TMVA SOFIE

Parser for translating an ONNX model to SOFIE's IR

```
using namespace TMVA::Experimental::SOFIE;
RModelParser_ONNX parser;
RModel model = parser.Parse("model.onnx");
```

TMVA SOFIE

Parser for translating an ONNX model to SOFIE's IR

```
using namespace TMVA::Experimental::SOFIE;
RModelParser_ONNX parser;
RModel model = parser.Parse("model.onnx");
```

Parser for translating PyTorch (`.pt`) and Keras (`.h5`) models

```
SOFIE::RModel model = SOFIE::PyTorch::Parse("PyTorchModel.pt");
SOFIE::RModel model = SOFIE::PyKeras::Parse("KerasModel.h5");
```

TMVA SOFIE

Parser for translating an ONNX model to SOFIE's IR

```
using namespace TMVA::Experimental::SOFIE;
RModelParser_ONNX parser;
RModel model = parser.Parse("model.onnx");
```

Parser for translating PyTorch (`.pt`) and Keras (`.h5`) models

```
SOFIE::RModel model = SOFIE::PyTorch::Parse("PyTorchModel.pt");
SOFIE::RModel model = SOFIE::PyKeras::Parse("KerasModel.h5");
```

Inference code generation

```
// generate text code internally (with some options)
model.Generate();
// write output header file and data weight file
model.OutputGenerated();
```

TMVA SOFIE

```
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){
————————————.
//——— Gemm
    BLAS::sgemm_(&op_0_transB, &op_0_transA, &op_0_n, &op_0_m, &op_0_k, &op_0_alpha,
tensor_0weight, &op_0_ldb, tensor_input1, &op_0_lda, &op_0_beta, tensor_21, &op_0_n);

//------ RELU
   for (int id = 0; id < 50 ; id++){
      tensor_22[id] = ((tensor_21[id] > 0 )? tensor_21[id] : 0);
   }
————————————..
   BLAS::sgemm_(&op_18_transB, &op_18_transA, &op_18_n, &op_18_m, &op_18_k, &op_18_alpha,
tensor_18weight, &op_18_ldb, tensor_38, &op_18_lda, &op_18_beta, tensor_39, &op_18_n);

// return output
   std::vector<float> ret (tensor_39, tensor_39 + 10);
   return ret;
}
};
}
```

Project Objectives

- Extending support of SOFIE Keras parser

Project Objectives

- Extending support of SOFIE Keras parser
- Implement SOFIE Custom operator support

Project Objectives

- Extending support of SOFIE Keras parser
- Implement SOFIE Custom operator support
- Implement support for parsing Graph Neural Networks in SOFIE

SOFIE Keras Parser

- No native support for ONNX translation

SOFIE Keras Parser

- No native support for ONNX translation
- `TF2ONNX` may convert a Keras .h5 model to ONNX

SOFIE Keras Parser

- No native support for ONNX translation
- TF2ONNX may convert a Keras .h5 model to ONNX
- SOFIE Keras Parser!
    - simpler to use
    - no need for input spec
    - built on latest opset

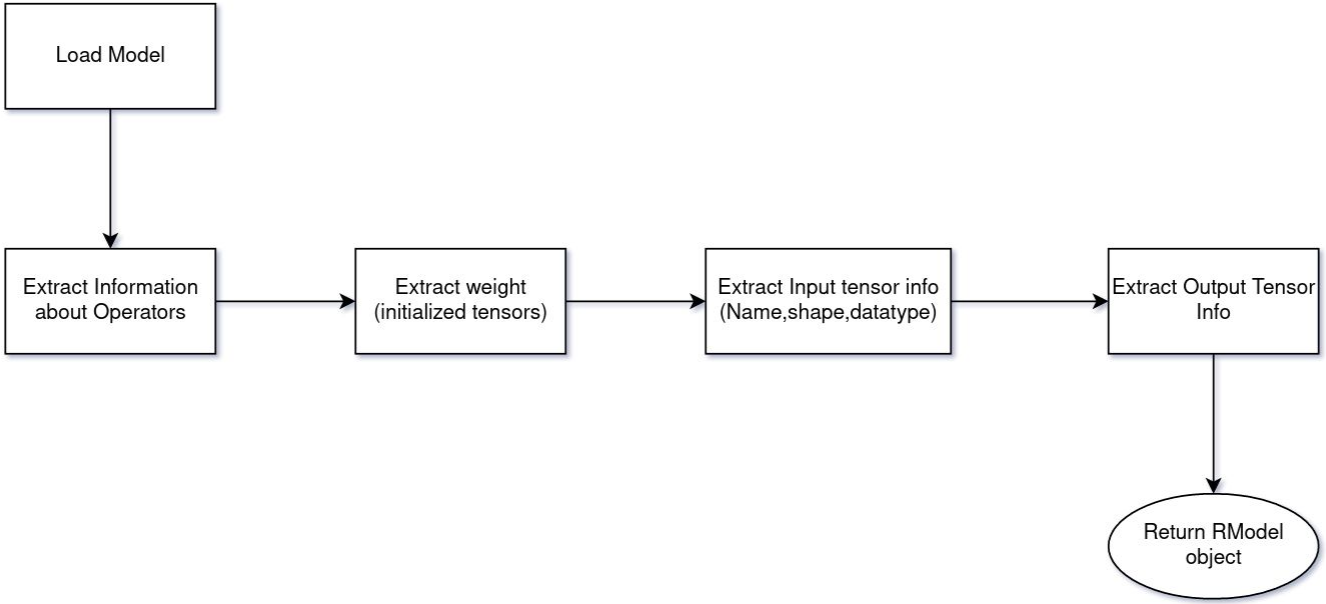
SOFIE Keras Parser

- No native support for ONNX translation
- TF2ONNX may convert a Keras .h5 model to ONNX
- SOFIE Keras Parser!
    - simpler to use
    - no need for input spec
    - built on latest opset

```
auto model = TMVA::Experimental::SOFIE::PyKeras::Parse("trained_model_dense.h5");
```

SOFIE Keras Parser

Algorithm for Parser



SOFIE Keras Parser

Current Support

Keras Layer	Status
Dense	Implemented & Integrated
Permute	Implemented & Integrated
ReLU, Selu, Sigmoid	Implemented & Integrated
Batch Normalization	PR Merged
Convolution (2D)	PR Merged
Basic Binary Operators: Add, Subtract, Multiply	PR Under Review
Reshape	PR Under Review
Activations: Softmax, LeakyRelu, Tanh	PR Drafted
Concat	PR Drafted

SOFIE Custom Operator

- ONNX standards specifies 183 operators currently.

SOFIE Custom Operator

- ONNX standards specifies 183 operators currently.
- Need a custom user operator specification

SOFIE Custom Operator

- ONNX standards specifies 183 operators currently.
- Need a custom user operator specification
  - simple to define

SOFIE Custom Operator

- ONNX standards specifies 183 operators currently.
- Need a custom user operator specification
  - simple to define
  - easy to test, debug, and evaluate

SOFIE Custom Operator

- ONNX standards specifies 183 operators currently.
- Need a custom user operator specification
  - simple to define
  - easy to test, debug, and evaluate
  - few overheads and dependencies

SOFIE Custom Operator

Definition

- Define custom operator with the required attributes
  - Operator name
  - Input tensor names
  - Output tensor names
  - Output tensor shapes
  - Header file name

SOFIE Custom Operator

Definition

- Define custom operator with the required attributes
  - Operator name
  - Input tensor names
  - Output tensor names
  - Output tensor shapes
  - Header file name
- Define Compute function in Header file

SOFIE Custom Operator

Definition

- Define custom operator with the required attributes
  - Operator name
  - Input tensor names
  - Output tensor names
  - Output tensor shapes
  - Header file name
- Define Compute function in Header file

SOFIE Custom Operator

Definition

- Define custom operator with the required attributes
  - Operator name
  - Input tensor names
  - Output tensor names
  - Output tensor shapes
  - Header file name
- Define Compute function in Header file

Interface

```
std::unique_ptr<SOFIE::ROperator> op;
op.reset(new SOFIE::ROperator_Custom<float>("Exp", {"denseBiasAdd0"}, {"exp_out"}, {{1,4}},
"exp_compute.hxx"));
```

SOFIE GNN Support

- High demands of Graph Neural Networks in High energy physics research

SOFIE GNN Support

- High demands of Graph Neural Networks in High energy physics research
- CMS

SOFIE GNN Support

- High demands of Graph Neural Networks in High energy physics research
- CMS
  - uses Particlenet; graph neural network supporting graph convolution, i.e. edge convolution and dynamic graph CNN methods

SOFIE GNN Support

- High demands of Graph Neural Networks in High energy physics research
- CMS
  - uses Particlenet; graph neural network supporting graph convolution, i.e. edge convolution and dynamic graph CNN methods
  - applications in heavy flavour jet tagging, jet mass regression, etc.

SOFIE GNN Support

- High demands of Graph Neural Networks in High energy physics research
- CMS
  - uses Particlenet; graph neural network supporting graph convolution, i.e. edge convolution and dynamic graph CNN methods
  - applications in heavy flavour jet tagging, jet mass regression, etc.
- LHCb

SOFIE GNN Support

- High demands of Graph Neural Networks in High energy physics research
- CMS
  - uses Particlenet; graph neural network supporting graph convolution, i.e. edge convolution and dynamic graph CNN methods
  - applications in heavy flavour jet tagging, jet mass regression, etc.
- LHCb
  - uses DeepMind's Graph Nets library; builds GNN on top of Tensorflow & Sonnet

Future Plan

- Finishing implementation & integration of Graph Neural Network support in SOFIE

Future Plan

- Finishing implementation & integration of Graph Neural Network support in SOFIE
- Integrating support for inference of BDT models in TMVA

Future Plan

- Finishing implementation & integration of Graph Neural Network support in SOFIE
- Integrating support for inference of BDT models in TMVA
    - Parser & Inference engine already built; requires a translation bridge.

Future Plan

- Finishing implementation & integration of Graph Neural Network support in SOFIE
- Integrating support for inference of BDT models in TMVA
    - Parser & Inference engine already built; requires a translation bridge.
- Implementing support for new operators in TMVA SOFIE

Conclusion

- Link to Forked Repository
  github.com/sanjibansg/root
- Link to SOFIE in current ROOT master
  github.com/root-project/root/tree/master/tmva/sofie
- Link to TMVA/SOFIE tutorials
  root.cern.ch/doc/master/group__tutorial__tmva.html
- Link to SOFIE notebooks
  github.com/lmoneta/tmva-tutorial/tree/master/sofie