Fast Simulation of LHCb Time Sensor Using Machine Learning

Zhihua Liang

First Showcase of INFN Personnel Activities in Spoke 2

July 8, 2024

Executive Summary

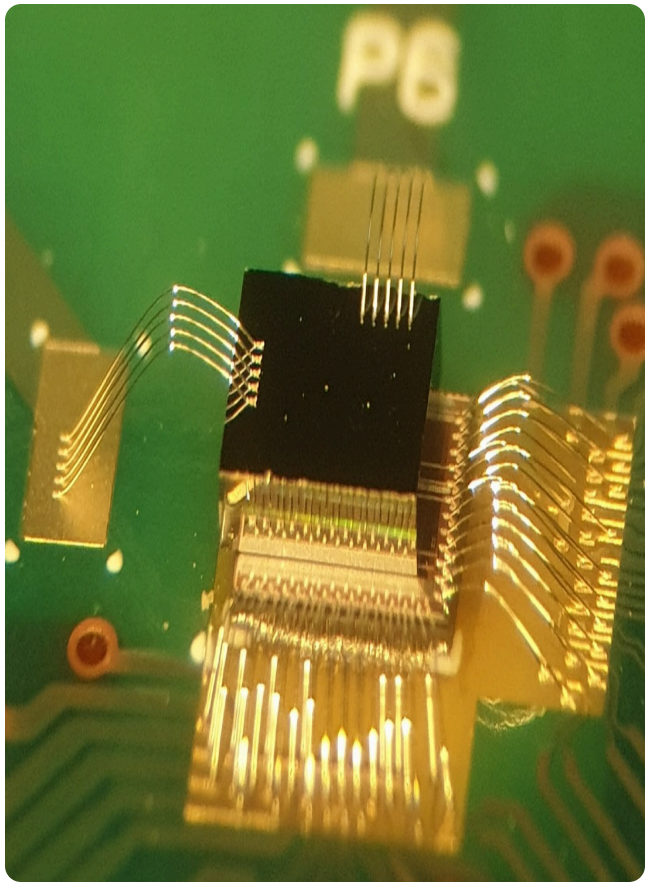
This presentation outlines our innovative approach to fast simulation of the LHCb time sensor using machine learning techniques. We have developed and integrating ML models, particularly gradient boosting decision trees(XGBoost), Multiple Layer Perceptrons (MLPs) and Edge-Activated Adaptive Function Networks (EAAFNs), to predict charge and center of gravity in the LHCb detector with high accuracy. Our approach significantly reduces computation time from seconds to microseconds per event, potentially revolutionizing LHCb simulation efficiency.

Outline

- Introduction
  - LHCb Time Sensor Simulation: Current Challenges
  - Motivation for ML-based Fast Simulation
- Methodology
  - Data Preparation and Preprocessing
  - Machine Learning Approach
- Results
  - Model Performance Comparison
  - Visualization and Analysis
- Integration and Applications
  - Integration with Gauss Framework
  - Potential Applications to Other Detector Components
- Conclusion
  - Challenges and Future Directions
  - Impact and Outlook

LHCb Time Sensor Simulation: Current Challenges

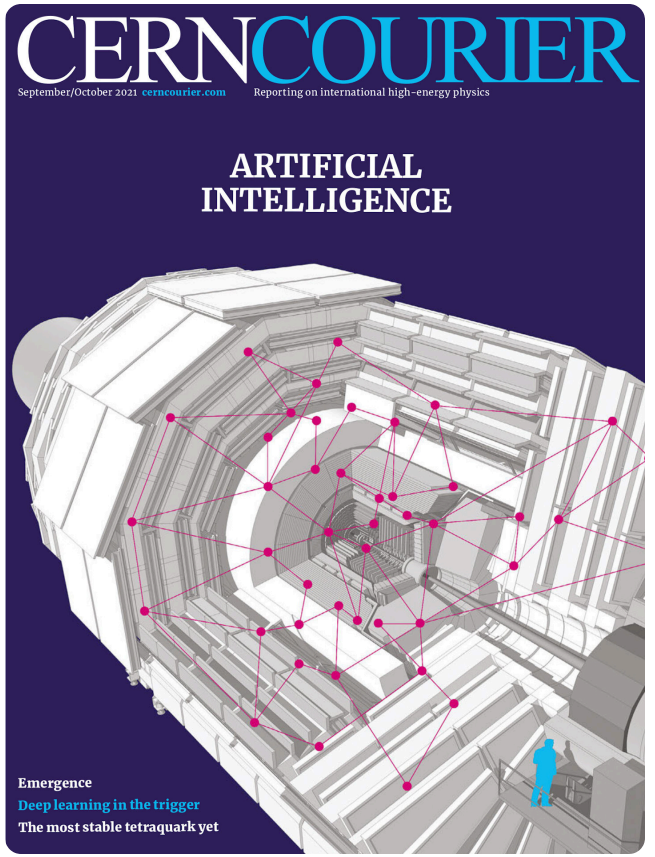
- Critical component for particle identification and timing measurements
- Current simulation uses computationally intensive TCoDe
  - Simulates charge deposition and signal formation in sensors
  - Crucial for accurate event reconstruction and physics analysis
- Key limitations:
  - Computationally expensive, limiting simulation throughput
  - Difficulty scaling to meet increasing physics analysis demands



source: Angelo's slides

Motivation for ML-based Fast Simulation

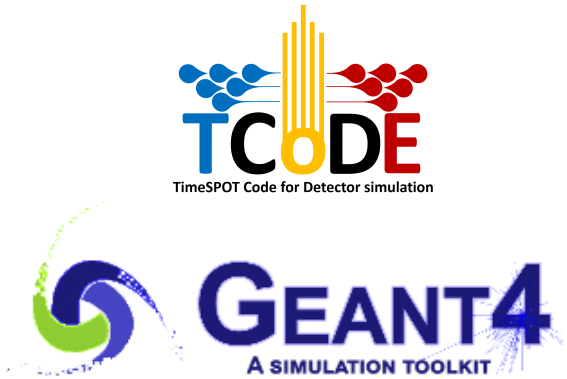
- TCoDe simulations are computationally expensive
  - Need for faster simulations to increase statistics in physics analyses
- ML models can provide rapid, accurate predictions
  - Potential for significant speedup in overall LHCb simulation pipeline
- Successful applications of ML in other HEP simulation tasks
  - Calorimeter showers, track reconstruction, etc.



CERN Courier Sep/Oct 2021

Data Preparation and Preprocessing

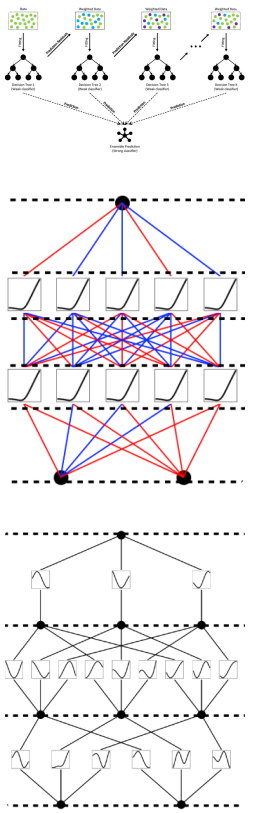
- Dataset: text files "*.dat" from Geant4 and TCoDe simulations
- 400 k events in total
- Converted to HDF5 format for efficient handling
- Features: particle impact/exit points, energy, energy loss
- Targets:
  - Charge: total electric charge collected by the sensor
  - Center of gravity: weighted average position of the collected charge
- Custom LHCbDataset class and data loader for preprocessing




```
3 anti_proton 48.027199 -0.000000 99.152000 34.712502 55.000000 119.399002 1.684758e+05 3.520860e+03 0.000000e+00 4.773386e-16 1.011375e+02
4 mu- 49.841202 55.000000 104.147003 0.000000 28.659401 57.173000 3.208703e+02 1.277989e+04 1.831811e+03 1.907071e-15 9.030624e+01
5 mu+ 47.970600 55.000000 32.078701 55.000000 47.590401 44.773302 3.536970e+01 9.820913e+03 7.507029e+03 1.513011e-15 1.378082e+02
6 e- 25.938601 0.000000 84.687599 53.507401 55.000000 30.514099 1.646600e+02 7.409782e+03 2.333340e+03 1.085129e-15 1.064865e+02
7 e+ 0.000000 12.929400 77.505203 55.000000 18.331900 60.944302 2.027806e+03 5.550394e+03 9.038860e+02 8.994872e-16 8.307108e+01
```

Machine Learning Approach: Overview

- Explored three ML algorithms:
  - XGBoost: Ensemble of decision trees
  - Multi-Layer Perceptron (MLP): Feedforward neural network
  - Edge-Activated Adaptive Function Network (EAAFN): Novel adaptive architecture
- Extensive experimentation with architectures and hyperparameters
- Training objectives: minimize MSE, maximize R2 score
- Cross-validation for robust performance estimation



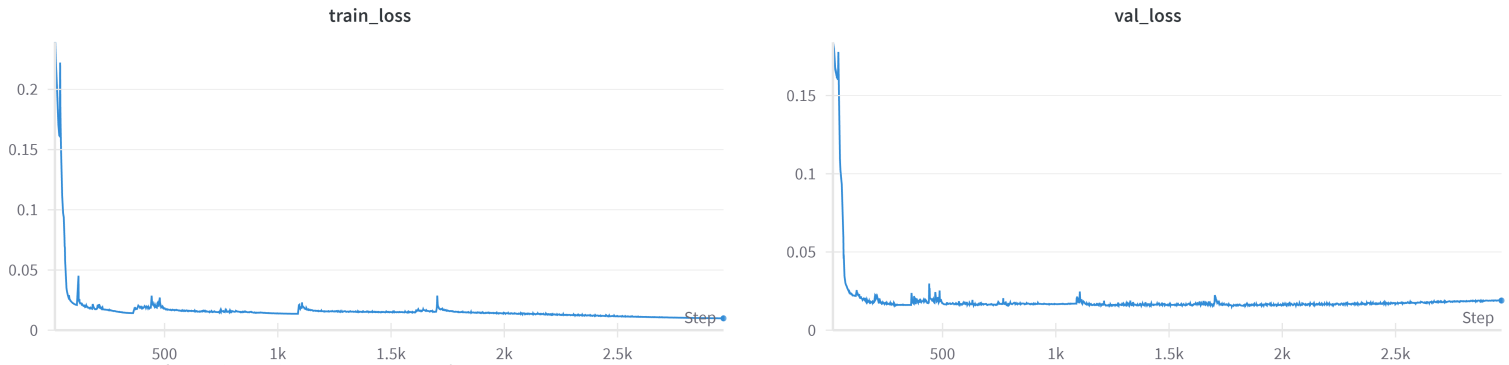
from top to bottom: XGBoost, MLP, EAAFN

Machine Learning Approach: Algorithm Details

- XGBoost (Extreme Gradient Boosting):
  - Ensemble method combining multiple decision trees
  - Trees are trained sequentially to correct errors of previous trees
  - Advantages: Handles missing data, robust to outliers, provides feature importance
- Multi-Layer Perceptron (MLP):
  - Feedforward neural network with input, hidden, and output layers
  - Information flows through the network, transformed by weights and activation functions
  - Advantages: Approximates complex functions, scales well with large datasets
- Edge-Activated Adaptive Function Network (EAAFN):
  - Novel architecture adaptively learning both connectivity and activation functions
  - Edges and nodes are dynamically pruned or added during training
  - Advantages: Discovers optimal structure, provides interpretable feature interactions

Model Training Process

- Training infrastructure:
  - INFN Cagliari computation server with NVIDIA A100 GPU
- Training parameters:
  - Batch size: 1024
  - Optimizer: Adam (lr = 1e-4)
  - Epochs: 10,000
  - LR scheduler: Cosine annealing with warm restarts
- Data: 400k events, 3% validation split



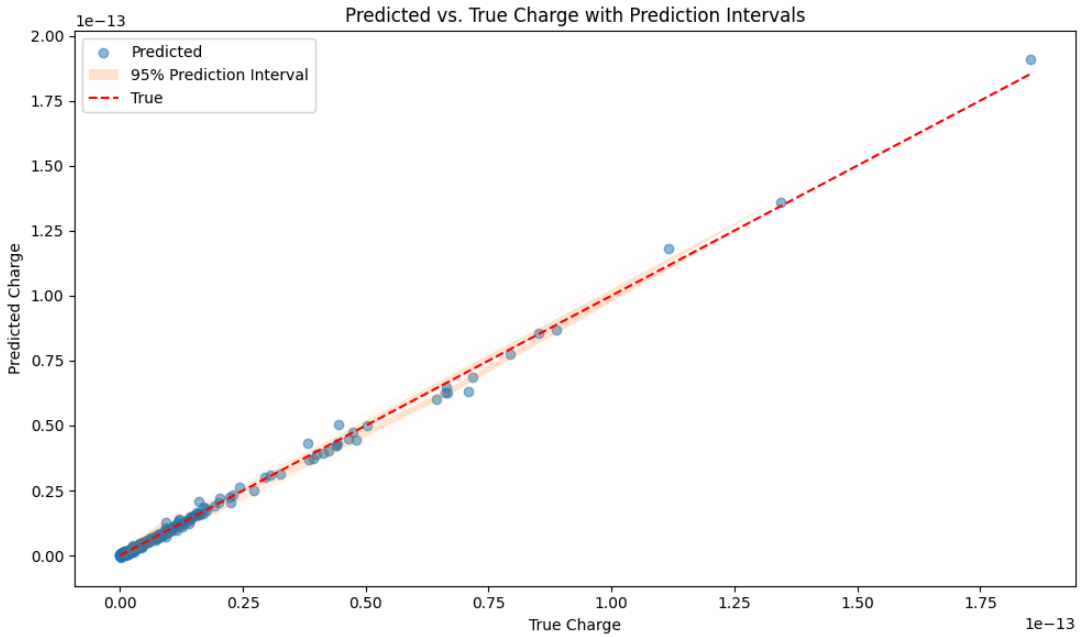
Results: Model Performance Comparison

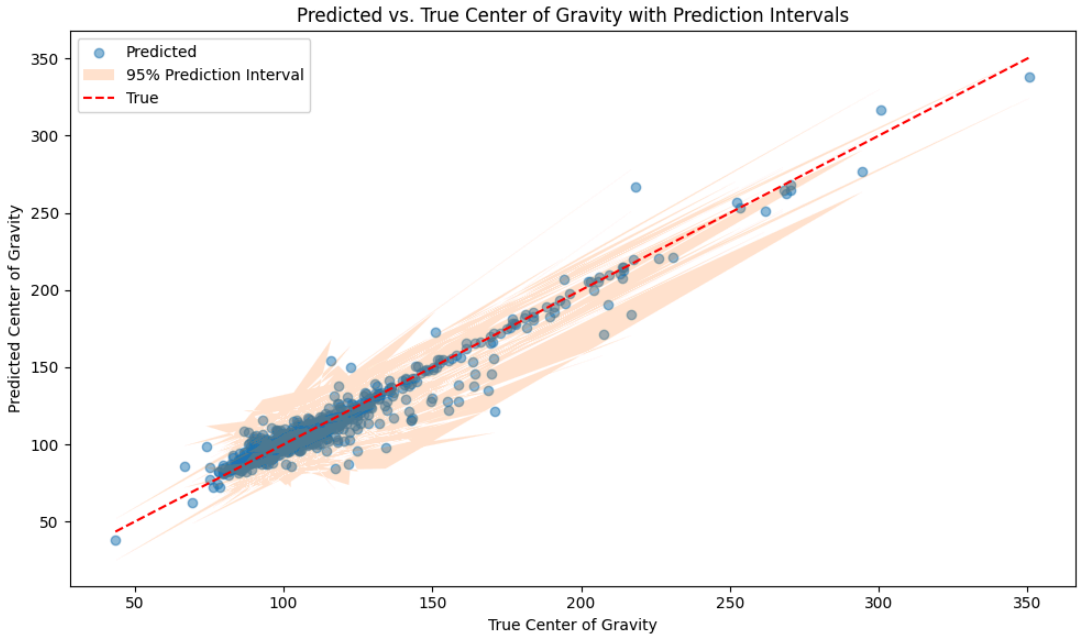
Model	Charge Prediction			Center of Gravity			Time per Event
	R2 Score	MAE	MSE	R2 Score	MAE	MSE	(s)
XGBoost	0.9742	0.0167	0.0258	0.8982	0.0513	0.1018	2.945E-05
Deep MLP	0.9890	0.0219	0.0110	0.9205	0.0417	0.0049	2.125E-05
Deep EAAFN	0.9944	0.0125	0.0056	0.9368	0.0357	0.0063	3.475E-05

- Deep EAAFN achieves the highest R2 scores and lowest errors
- All ML models provide significant speedup compared to TCoDe (seconds per event)
  - 10^4 - 10^5 times faster, enabling much higher simulation throughput

Visualization: Prediction Intervals

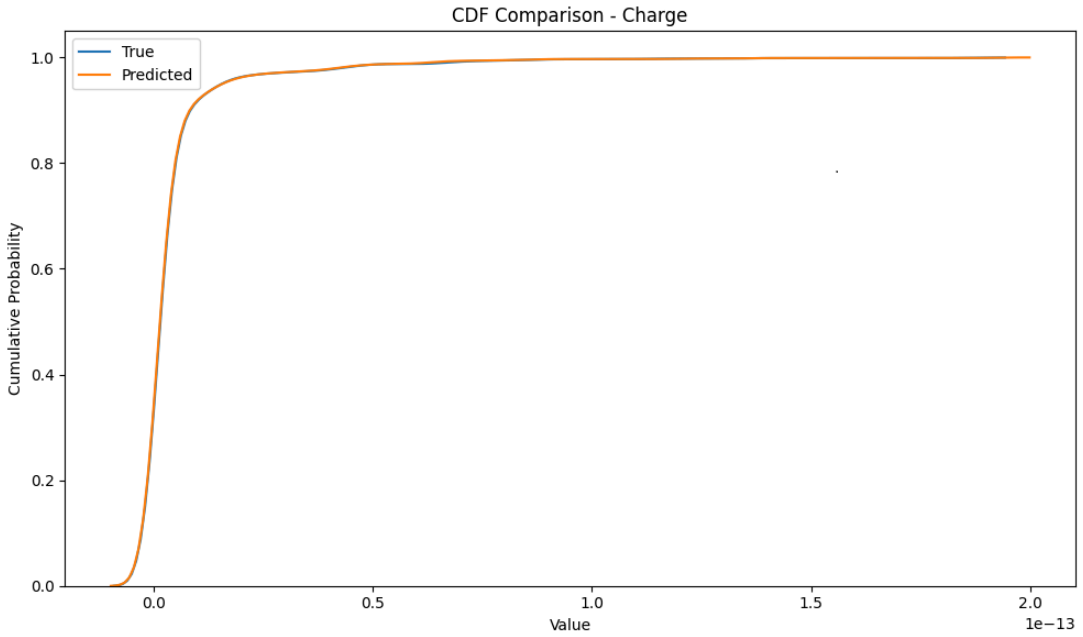
- 95% prediction intervals show the range where true values are expected to fall
- Narrower intervals indicate higher confidence in predictions
- Wider intervals suggest higher uncertainty and potential for improvement

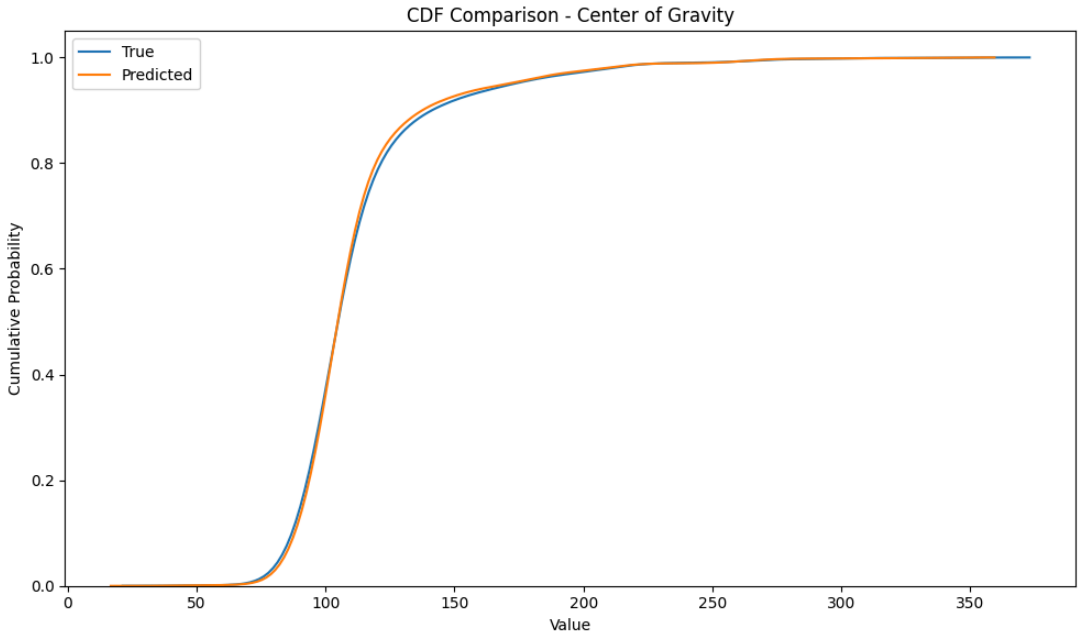




Visualization: CDF Comparison

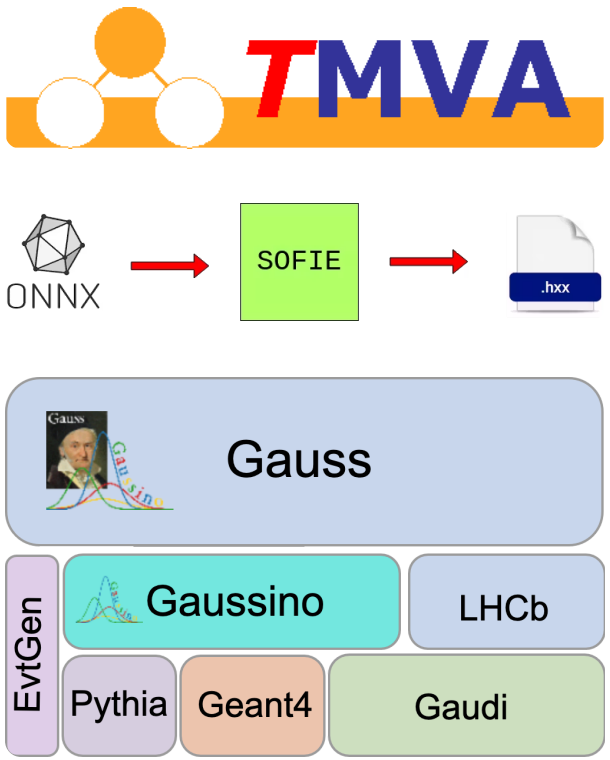
- Quantile-Quantile (QQ) plots assess normality of residuals (predicted - true)
- Points along the diagonal line indicate normally distributed residuals
- Normally distributed residuals suggest a well-fitted model





Integration with Gauss Framework

- Implemented TimeSensorFastSim class in GaussFastSim module
- Use of ROOT's TMVA/SOFIE for efficient model inference
  - Load trained ONNX model
  - Generate optimized C++ code for inference
- Modification of ProcessHits method for ML-based predictions
- Flexible configuration options for model path and activation



Integration: Key Code Snippet

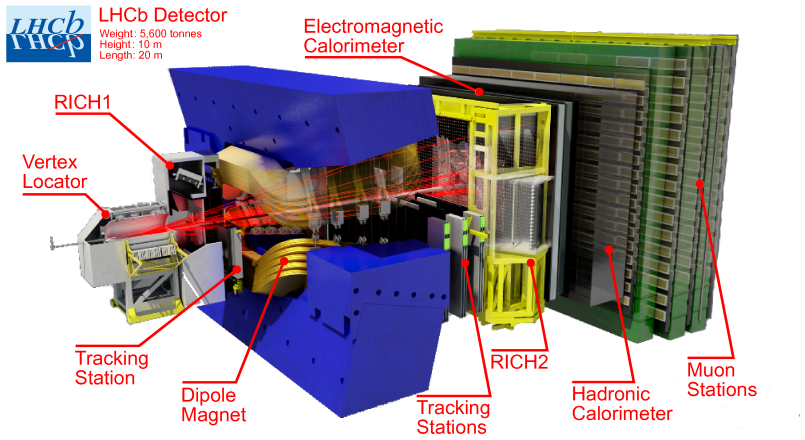
```
class TimeSensorFastSim : public GiGaVolumeBase {
private:
  std::unique_ptr<TMVA::Experimental::SOFIE::RModel> m_model;
  std::unique_ptr<TMVA::Experimental::SOFIE::Session> m_session;

public:
  StatusCode initialize() override {
    // Load ONNX model
    m_model = std::make_unique<TMVA::Experimental::SOFIE::RModel>(
      TMVA::Experimental::SOFIE::RModelParser_ONNX().Parse(m_modelPath)
    );
    m_model->Generate();
    m_session = std::make_unique<TMVA::Experimental::SOFIE::Session>(
      m_model->GetSessionFile()
    );
    return StatusCode::SUCCESS;
  }

  bool ProcessHits(G4Step* step, G4TouchableHistory*) {
    std::vector<float> input = extractFeatures(step);
    std::vector<float> output = m_session->infer(input.data());
    createHits(output, step);
```

Potential Applications to Other Detector Components

- RICH detectors: Cherenkov photon propagation and detection
- Calorimeters: Shower development and energy deposition
  - Potential for significant speedup in full calorimeter simulation
- Tracking systems: Charge cluster formation in silicon strips
- Potential for end-to-end detector response simulation
  - Enabling faster, more detailed physics studies



Next Steps

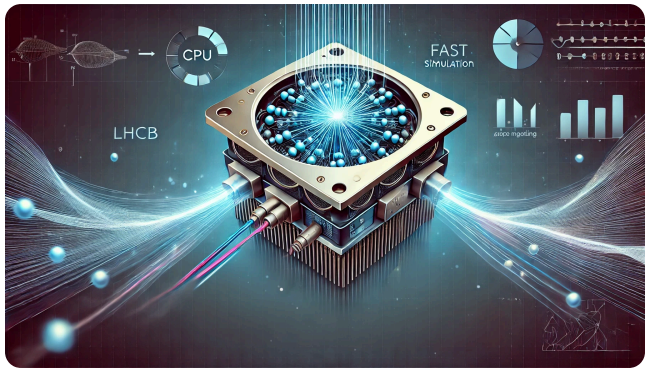
- Comprehensive validation of integrated ML models
  - Comparison with full simulation and real data
  - Ensure physics performance is accurately reproduced
- Optimization of ML models for time sensor simulation
  - Fine-tuning of hyperparameters and architectures
  - Exploration of additional input features
- Performance benchmarking within Gauss framework
  - Assess speedup and resource utilization
- Documentation and knowledge sharing
  - Detailed guide for using ML-based fast simulation in Gauss
  - Collaboration with LHCb simulation team for maintenance and updates

Impact and Outlook

- Success on ML-based fast simulation for LHCb time sensor
  - Significant speedup while maintaining high accuracy
- Paves the way for adoption in LHCb simulation production
  - Potential for major impact on simulation efficiency and physics reach
- Opens possibilities for further applications of ML in LHCb simulation
  - Building on experience and infrastructure from time sensor integration
- Faster simulations can accelerate physics discoveries
  - More efficient use of computing resources
  - Ability to generate larger simulated datasets for precision measurements

Conclusion

- Developed ML models for fast simulation of LHCb time sensor
  - XGBoost, MLP, and EAAFN architectures explored
  - Deep EAAFN provides best accuracy and speed
- Achieved 10^4 - 10^5 speedup compared to TCoDe simulation
  - Enables much higher simulation throughput
- Integrating ML models into Gauss framework
- Promising approach for accelerating LHCb simulation
  - Potential for broader application to other detectors and experiments



Thank You

Email：zhihua.liang@ca.infn.it