Classification of KM3NeT online events with ONNX C++ API

Emmanuel Le Guirriec (CPPM)

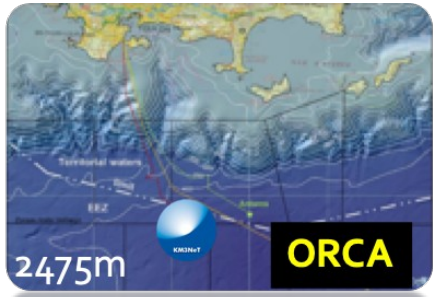
Feifei Huang, Damien Dornic

IN2P3/IRFU Machine Learning Workshop

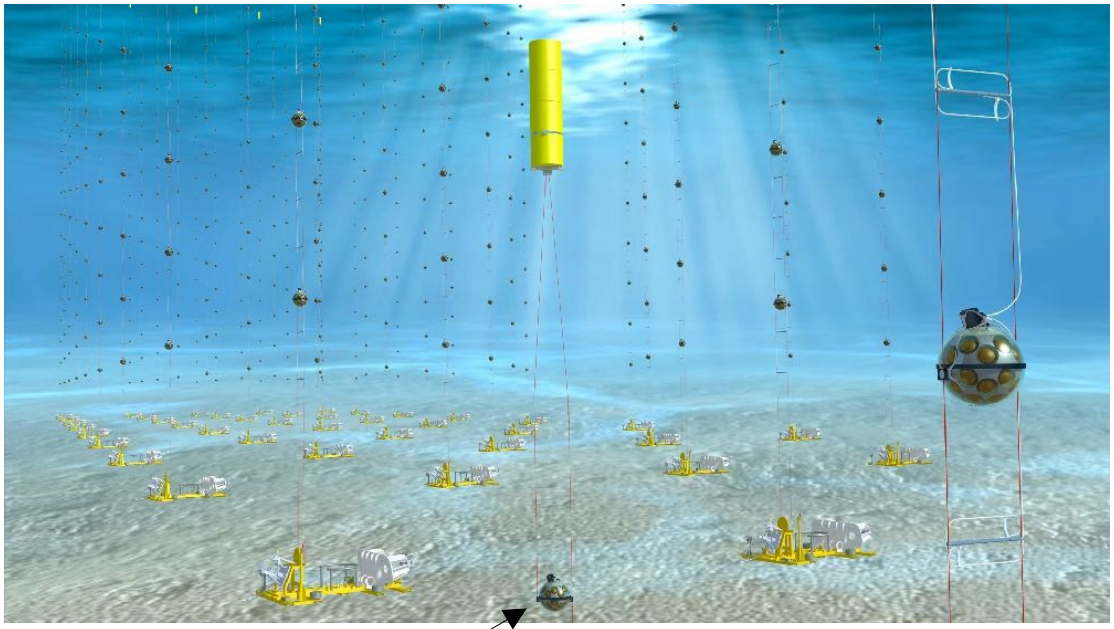
16-17 March 2021

KM3NeT: the neutrino research infrastructure in the deep Mediterranean Sea

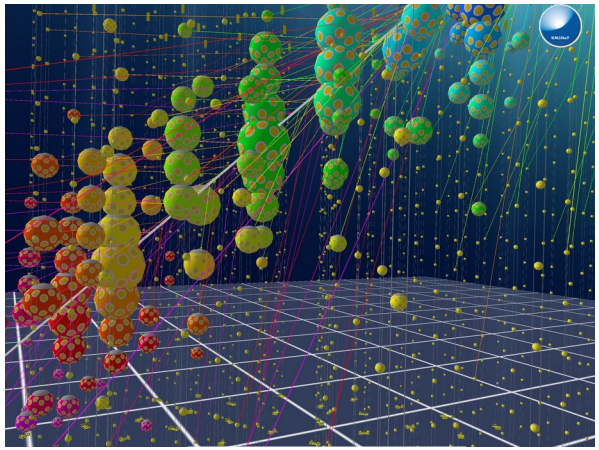
Oscillation **R**esearch with **C**osmics In the **A**byss



Astroparticle **R**esearch with **C**osmics In the **A**byss



Detection unit with several optical modules



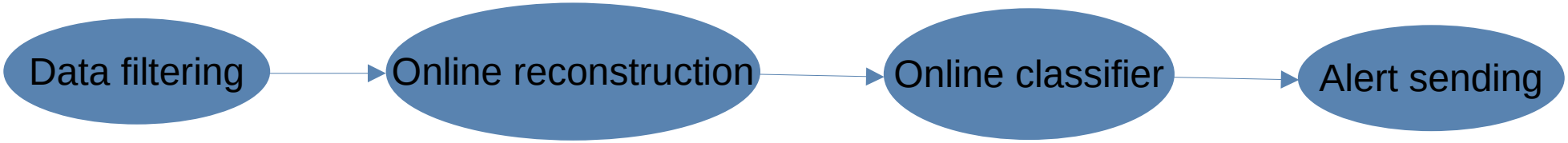
Simulated event

Significant events will trigger alerts that will be distributed publicly to the astro community within 10 seconds

Online event classifier

- Goal: select a high purity neutrino sample out of background
  - Signal: neutrinos
  - Background: atmospheric muons
- Algorithm developed by Feifei using LightGBM
  - Gradient boosting decision tree
  - Model dumped with joblib
- Computation done on a local server
  - Hyper-parameters tuning done with 48 CPUs in 5 hours (600000 events)
  - Classifier training done with 12 CPUs in 10 minutes (7.8 million events)

Why using a C++ API to infer the event type?



  - Online reconstruction in C++
  - The online event is classified in a separate Python job with LightGBM framework
- Goal: reduce the delay to send alert
  - To minimize I/O, the classification should be done within reconstruction job
  - Need a ML framework with C++ API

General question: how to deploy in C++ a machine learning model trained with your favorite framework?

Open Neural Network eXchange (ONNX) Runtime

- ONNX Runtime is an open source project that is designed to accelerate machine learning across a wide range of frameworks, operating systems, and hardware platforms
  - https://www.onnxruntime.ai
  - Standardization for ML model formats by Microsoft
  - Models trained on various frameworks can be converted to the ONNX format
  - Runtime engine for high-performance inferencing that provides hardware acceleration
  - The same model can be deployed across a variety of platforms and technology stacks

LightGBM model conversion and Python test

- onnxmltools can convert LightGBM model to ONNX
  - Trivial onnxmltools installation (pip install)
  - Python script to convert from LightGBM model file
    - Load the LightGBM model file (joblib)
    - Use onnxmltools.convert_lightgbm function
      - Trick to set unknown dimension (C++ needs it)
  - Check that the converted model is fine with onnx.checker.check_model
- Test with ONNX Python runtime
  - Very few lines to
    - Create the ONNX session using the converted model
    - Fill the input Numpy array
    - Run the ONNX session and retrieve the probability
  - Results are compatible with LightGBM

Implement the classifier in C++

- Install onnxruntime CXX API from source
  - Git repository: https://github.com/Microsoft/onnxruntime
    - Many dependencies
    - But share object should not be heavy… By default compilation in debug mode!!
- Implementation in C++ (not as easy as in Python)
  - Create the ONNX session using the converted model
  - Get model input layer (node names, types, shape etc.)
  - Create the input tensor for ONNX session
    - Use an intermediate std::vector and use CreateTensor
  - Run session and get back output tensor
    - Need the output node name (obtained from Python script)
  - Tricky to get the final probability (not well documented)
    - Also compatible results with Python version

ONNX experience feedback

- Easy to convert LightGBM model and test Python implementation
- Quite technical to do the C++ implementation
  - Some information from Python needed
  - I didn´t know anything about the original model and about ONNX
- Short survey of different supported ML toolkits
  - Tensorflow/Keras: onnxtools converter
  - scikit-learn (subset of models): onnxtools converter
  - pyTorch : torch export function
  - TMVA (ROOT): “some success at integrating ONNX Runtime into the analysis frameworks of some of the large experiments at CERN”

Conclusion

- ONNX is a good candidate to make predictions in C++ code with a model trained with another framework
- The new classification is in production since the beginning of the year
- This solution is allowing to welcome models that other KM3NeT groups will develop for their analysis.