Machine Learning Preservation in the CMS Experiment

Joshua C. Hiltbrand

Baylor University (US)

26 February 2025

Towards ML Model Preservation in CMS

Somewhat early days, but building momentum

§ An important piece of analysis preservation

- Analyses are using ML more and more at their core—*robustly preserving* those ML models is critical for reinterpretation
- In CMS, ML has been used at the level of defining signal/control regions, etc.

§ Pillars of support for ML usage in CMS

- Already a dedicated ML group within CMS is overseeing the publication of papers concerning analyses' advanced ML techniques
- ⚠ **Broader community (including CMS members) establishing guidelines on ML reusability**
- ✓ **A lot of investment in ML that should carry through to the preservation side !**

Key Considerations [1/2]

Some considerations for having a well-defined path towards ML preservation

1. ML models tend to be getting larger and more complicated in CMS

- Structures may not be supported by standard preservation frameworks
- Inputs may be more low-level/custom than what can be reasonably reused

⧖ **How to accommodate all models and able to preserve them ?**

2. Where to preserve ML models ?

- Some attractive solutions including quasi-industry-standards

The CMS ML group recommends several possible options with no single-solution answer

1. Preservation Longevity feat. Model Inputs

- **Higher-level ML model inputs are easier for reimplementing and reusing in reinterpretations**
  - Definitions of objects/variables can/should be easily provided with the model
- **Lower-level (detector-level) inputs are experiment-specific and not easily (impossible) to reproduce**
  - As alternative, provide detailed efficiency (and misidentification) maps
  - Perhaps can provide surrogate models that approximates original model

⚠ **Some tension with complex ML models utilizing lower-level quantities and preservation-friendliness**

2. Long-Term Place for Storing ML Models

§ A growing standard in industry is Hugging Face, which has a few examples from particle physics

- Essentially a GitHub-style framework including description of the model
- ✓ **Provides built-in tools for facilitating proper documentation about the model**

§ Other options available

- Zenodo (built and operated by CERN team)
- GitHub
- HEPData (a natural choice if already preserving analysis info, but how to accommodate for ML model type information ?)

No formal standard yet in current CMS ML recommendations

Key Considerations [2/2]

Some considerations for having a well-defined path towards ML publication/preservation

3. What format should preserved ML models be in ?

- ✓ Fairly strong format in ONNX
- ⚠ **Some missing compatibility for specific architectures, but could/will probably be resolved over time**

4. When a ML model is reused, how to ensure used/behaving properly ?

- Exemplary inputs and associated outputs i.e. histograms/plots
- **Precise definitions of all inputs, including ordering**
- **Feature importance can also help keep focus on variables that need to be modeled the best**

3. ML Model Preservation Format

§ Several formats available in the wild, depending on use case

- For NNs, the **most attractive option is** `ONNX`, which supports `TensorFlow` and `PyTorch`

⚠ **Normalizing Flows model structures not currently supported by** `ONNX`**...**

Examples of these architectures already appearing in CMS analysis

- Python's `pickle` ruled out due to version-dependency overhead
- `LWTNN`, which is used/developed in ATLAS, long-term support unclear (?)

§ Some different options for BDTs

- `petrify-bdt`: light-weight framework for converting BDTs into literal code, HEP focus
- `treelite`: could be viewed as the `ONNX` for BDTs, but more overhead/dependencies

⧖ **Some BDT formats supported by one and not the other**

4. Verifying ML Model Behavior

§ **Once an external user acquires a preserved model, they would like to know that it behaves as expected**

- For some particular cases of inputs, provide outputs to be checked against
- More generally, provide distribution of input and output variables for particular physics processes

⚠ **Complemented by detailed definition of inputs, how they are constructed, how they are ordered, etc.**

Ideal ML Preservation Recipe for CMS

I. Convert ML model to ONNX format for minimal reliance on dependencies

If trouble with model support, contact your local, friendly CMS ML group convener ☺

II. Setup area on Hugging Face to save ONNX file

✓ Obtain a DOI number

III. *Document* model on Hugging Face

- Description of inputs, how they are computed, how they are ordered

IV. Provide performance metrics

- For given set of inputs, what does output(s) look like ?
- Provide input distribution and output distributions

Concluding Remarks

There is a path forward for ML model preservation within CMS with some careful considerations

Like general analysis preservation, ML preservation still needs to build more momentum within CMS to reach "critical mass"