26th International Conference on Computing
in High Energy & Nuclear Physics (CHEP 2023)

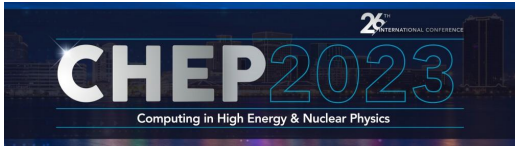
THE LHCB ULTRA-FAST SIMULATION OPTION, LAMARR

design and validation

08/05/2023

Matteo Barbetti

on behalf of the LHCb Simulation Project



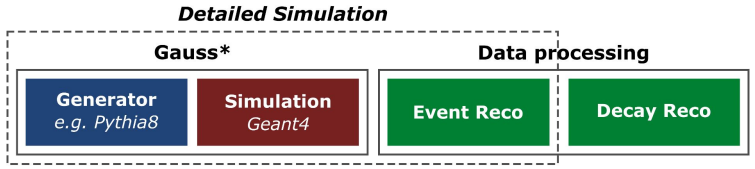
Simulating the LHCb experiment

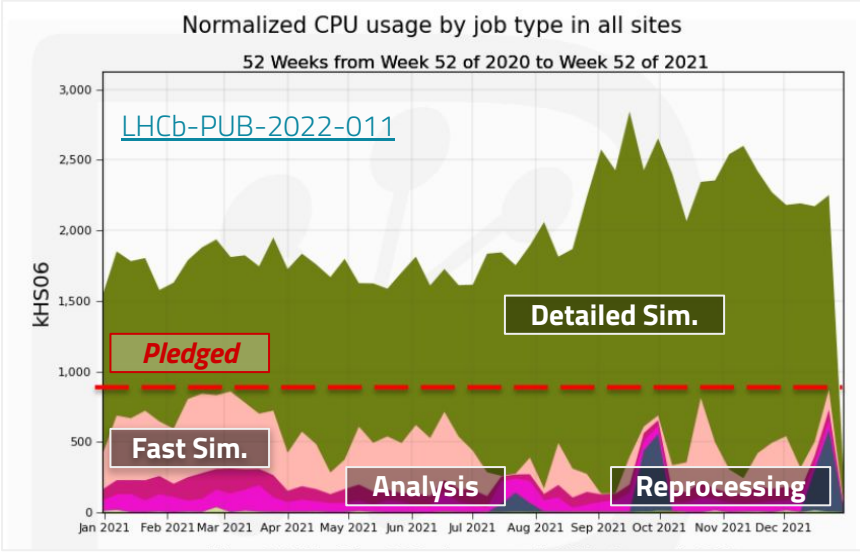
The standard for simulation at LHCb is ***Detailed Simulation***:

- simulation of all radiation-matter interactions
- simulated hits processed as real data
- extremely expensive in terms of CPU time (**more than 90%** used during LHC Run 2)
- unsustainable in the long term (*i.e.*, LHC Run 3 and those to come next)

Using Detailed Simulation only for LHC Run 3 needs will **far exceed the pledged resources** of LHCb.

Developing ***faster* simulation** strategies is mandatory to meet the upcoming and future requests for simulated data samples.





* **Gauss** is the LHCb simulation framework based on Gaudi [1]

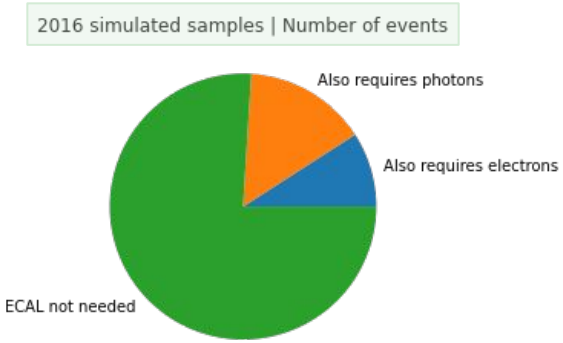
How does LHCb simulate events?

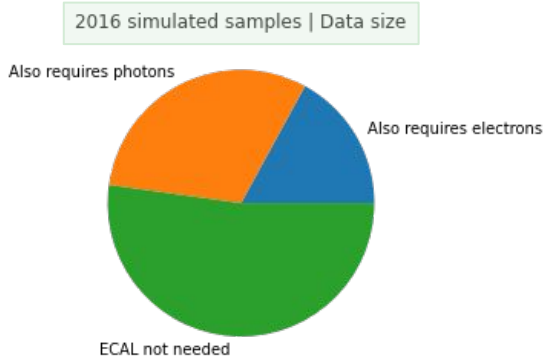
Simulations production driven by the LHCb physics program, *i.e.* most of the simulated decay modes are **heavy hadron decays**.

The detector will provide very "similar response" to, *e.g.*, a kaon from any source as long as with the same kinematics and detector conditions.

We could **save a lot of computing resources** by parameterizing the detector (low-level) response to that kaon and applying it to whatever decay model.

Analyses involving $h^{\pm}$ and $\mu^{\pm}$ only, often **drop** simulated raw detector information immediately → parameterizing directly the **high-level response** of the detector allows to save even more computing resources.





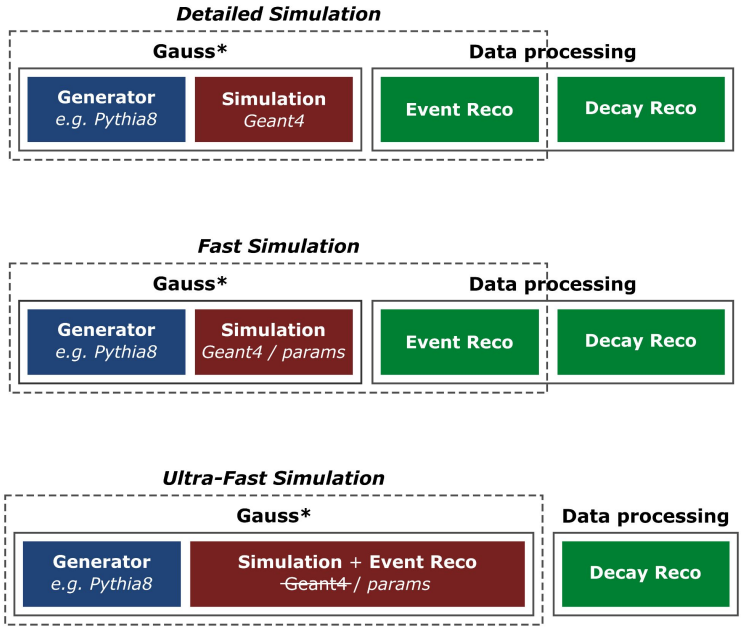
Fast simulation VS. ultra-fast simulation

Fast Simulation techniques aim to speed up the Geant4-based simulation production:

- Simulation framework upgrade (see first Michal's talk)
- Reducing the detector geometry (*e.g.*, track-only sim)
- Reuse of the underlying events, **ReDecay** [2]
- Parameterizing **energy deposits** instead of relying on Geant4 (*e.g.*, shower libraries [3] or CaloGAN [4])

Ultra-Fast Simulation strategies replace Geant4 with parameterizations able to transform generator-level particles into analysis-level reconstructed objects [5].

* **Gauss** is the LHCb simulation framework based on Gaudi [1]

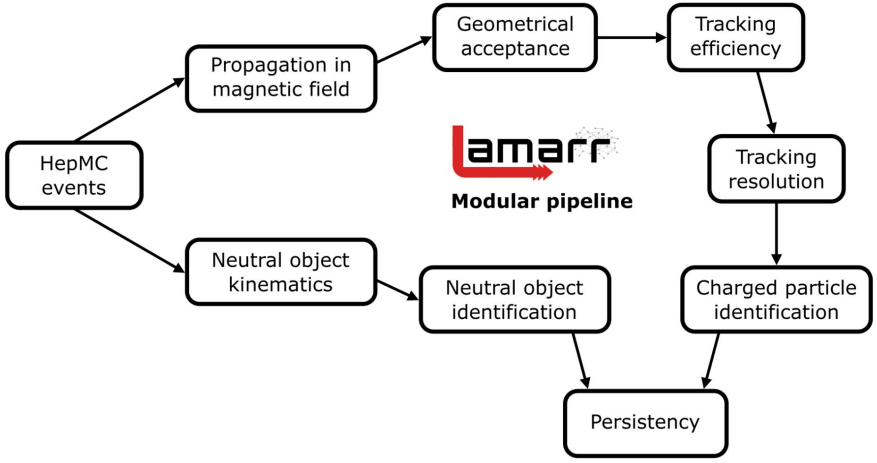


Lamarr: the LHCb ultra-fast simulation option

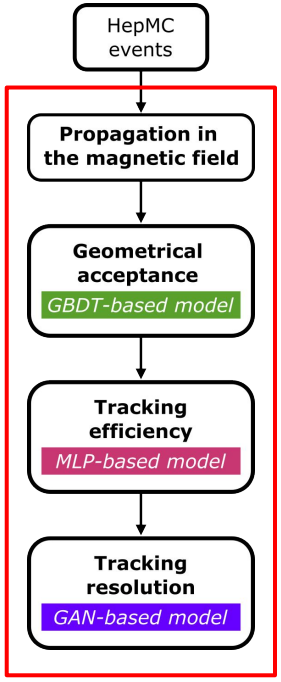
Lamarr is the novel ultra-fast simulation framework of LHCb, able to offer the fastest options for simulation. Lamarr consists of a **pipeline of** (ML-based) **modular parameterizations** designed to replace both the simulation and reconstruction steps [6, 7].

Lamarr is integrated with the LHCb simulation framework:

- compatibility with all the **LHCb-tuned generators**
- compatible with the **distributed computing** middleware (LHCbDirac) and production environment
- able to provide datasets in the same format used for analysis



Tracking system models

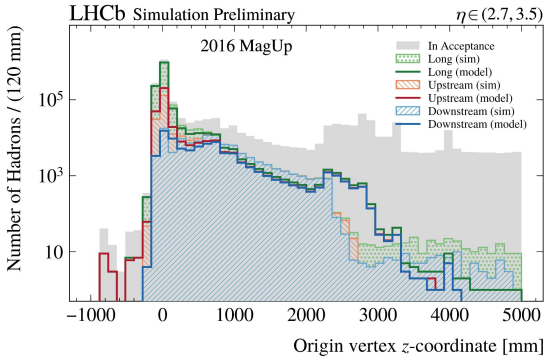


Lamarr Tracking pipeline

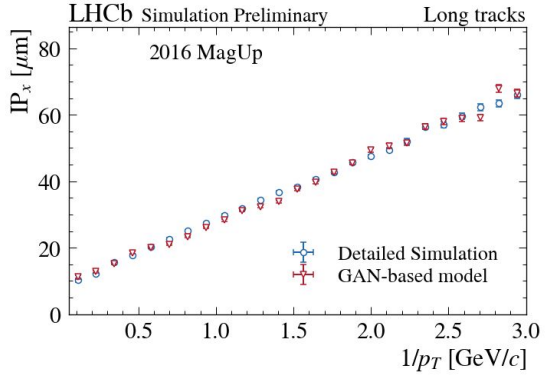
Lamarr parameterizes the **LHCb Tracking system** mostly relying on a set of (ML-based) modules:

- **acceptance** → predict which of the generated tracks fall in the geometrical acceptance of the experiment
- **efficiency** → predict which of the generated tracks in acceptance are properly reconstructed by the detector
- **resolution** → convert the generator-level parameters of the reconstructed tracks into analysis-level ones, including track-quality features

A major effort is ongoing to model correctly the Tracking system in function of the **type of tracks**.



LHCb-FIGURE-2023-008



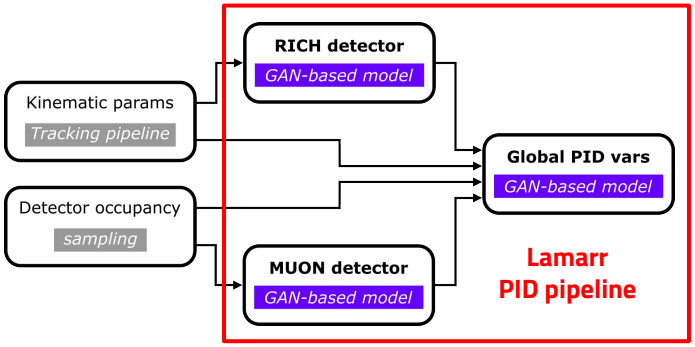
LHCb-FIGURE-2022-004

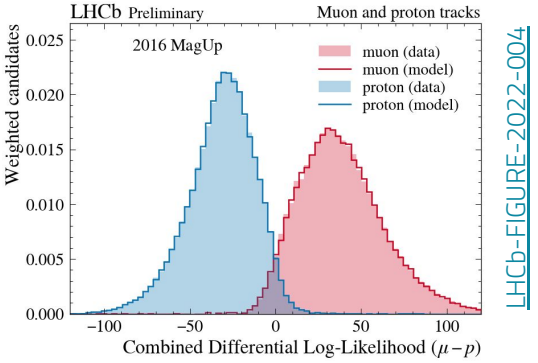
Particle identification system models

The high-level response of the LHCb PID system relies on ***Generative Adversarial Networks*** (GAN) [8, 9] trained on either detailed simulated samples or real data.

Lamarr provides RICH and MUON models for for **muon**, **pion**, **kaon** and **proton** tracks based on the kinematics of the reconstructed tracks and a description of the detector occupancy.

This information alone aren't enough to parameterize the **Global PID variables**, that also need the response of the RICH and MUON systems. Hence, Lamarr provides the higher-level response of the PID system relying on a **stack of GAN-based models**.





LHCb-FIGURE-2022-004

Electromagnetic calorimeter model

Parameterization of the **LHCb calorimeter** available in Lamarr designed for detector studies → not suitable for simulation production

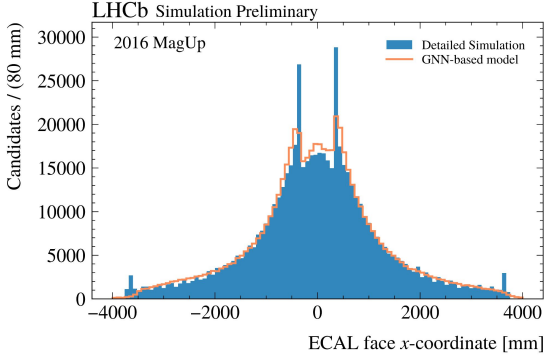
Improving the ECAL models is a necessary step if we want that Lamarr provides reliable parameterizations also for **photons** and **electrons**.

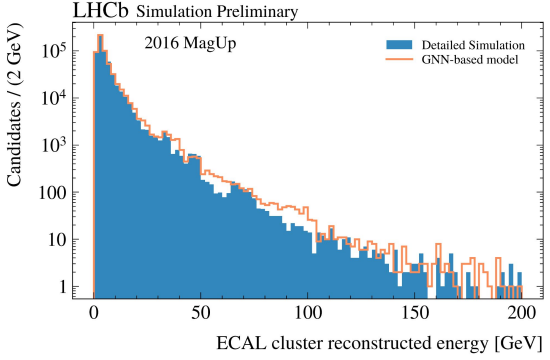
Simulating ECAL with an ultra-fast approach requires to face the **particle-to-particle correlation problem**:

- sequence of *n* generated photons → sequence of *m* reconstructed clusters (in general, with $n \neq m$)
- approached as a **translation problem**

Two strategies are currently under investigation:

- ***Graph Neural Networks*** (GNN) [10, 11]
- ***Transformer*** [12, 13]



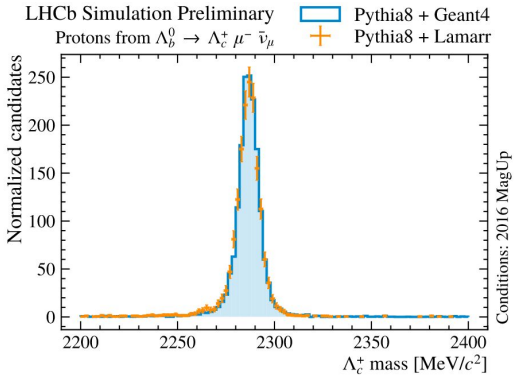


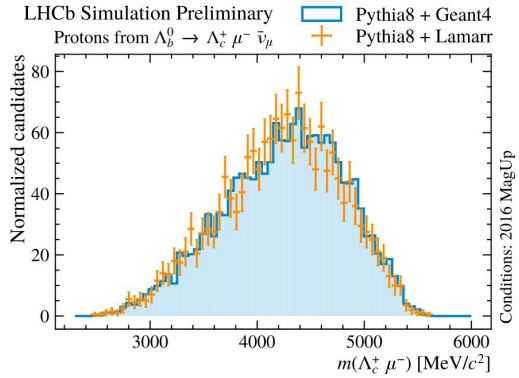
LHCb-FIGURE-2023-008

Lamarr validation campaign

Lamarr is currently under validation, comparing the distributions of the **analysis-level reconstructed quantities** parameterized with what obtained from Detailed Simulation.

- semileptonic decay mode: $\Lambda_b^0 \to \Lambda_c^+ \mu^- X$ with $\Lambda_c^+ \to p K^- \pi^+$
  - crucial the interface with LHCb-tuned generators
- **muons**, **pions**, **kaons** and **protons** in a single decay
  - all particle species for which Lamarr provides parameterizations
- Lamarr-based samples, detailed simulated samples and plots obtained from the **LHCb analysis software**
  - testing the integration with the current version of Gauss





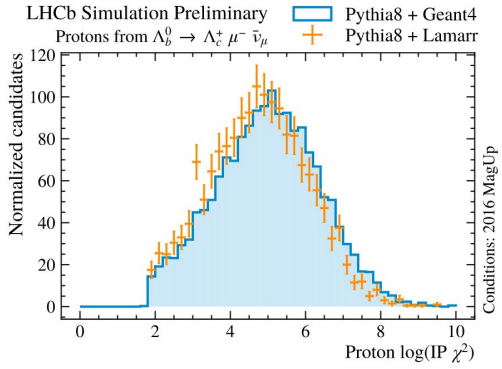
LHCb-FIGURE-2022-014

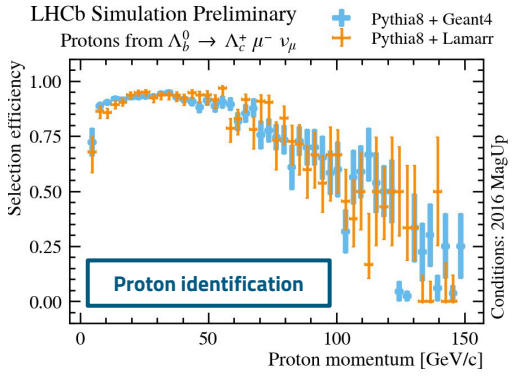
Some validation results

Smeared kinematic parameters of the generated particles are used to compute the **reconstructed masses**, the **impact parameters** and the **PID variables**.

Lamarr also provides information on *uncertainties* associated to the track reconstruction. For example, the **impact parameter χ^2** is a measure of inconsistency of a track trajectory with the PV.

Lamarr simulates the distribution of the detector response. But it's also crucial assessing that the **selection efficiencies** in function of the kinematic parameters and detector conditions for the parameterized quantities are well reproduced.





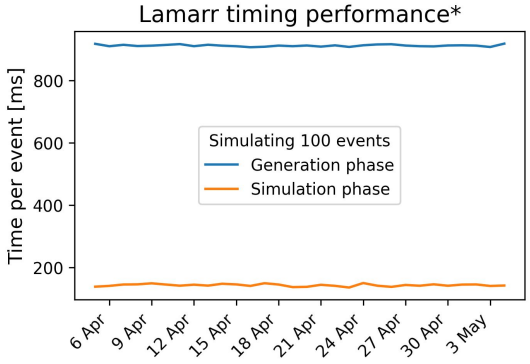
LHCb-FIGURE-2022-014

Preliminary timing studies

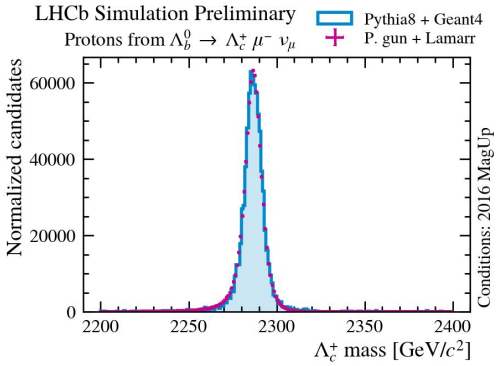
Comparing the normalized CPU spent for Geant4-based and Lamarr simulations of $\Lambda_b^0 \to \Lambda_c^+ \mu^- X$ decays we estimate a CPU reduction of **two-order-of-magnitude** for the Simulation phase.

Since the generation of *b*-baryons is exceptionally expensive, Pythia8 becomes the **major consumer of CPU** for simulation in the ultra-fast paradigm.

A **further speed-up** can be reached reducing the cost for generation, for example simulating only the signal particles (*i.e.* with ***Particle Guns***) and avoiding at all the simulation of the *pp* collisions, not needed since Lamarr models the detector occupancy.



* data obtained from the **LHCbPR** portal [14]



LHCb-FIGURE-2022-014

Implementing Lamarr with the future Gauss

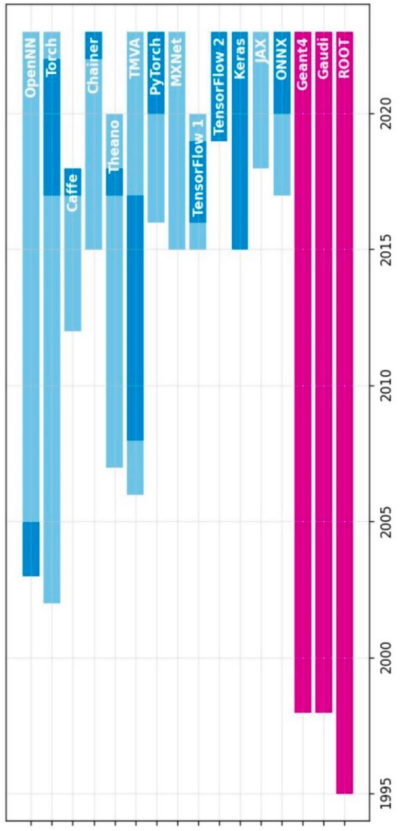
Developing an integration pipeline for ML-based models we should take account of:

- enable developments on **short time scales**
- put into production takes time (production quality studies)
- want to use in production for **many years**

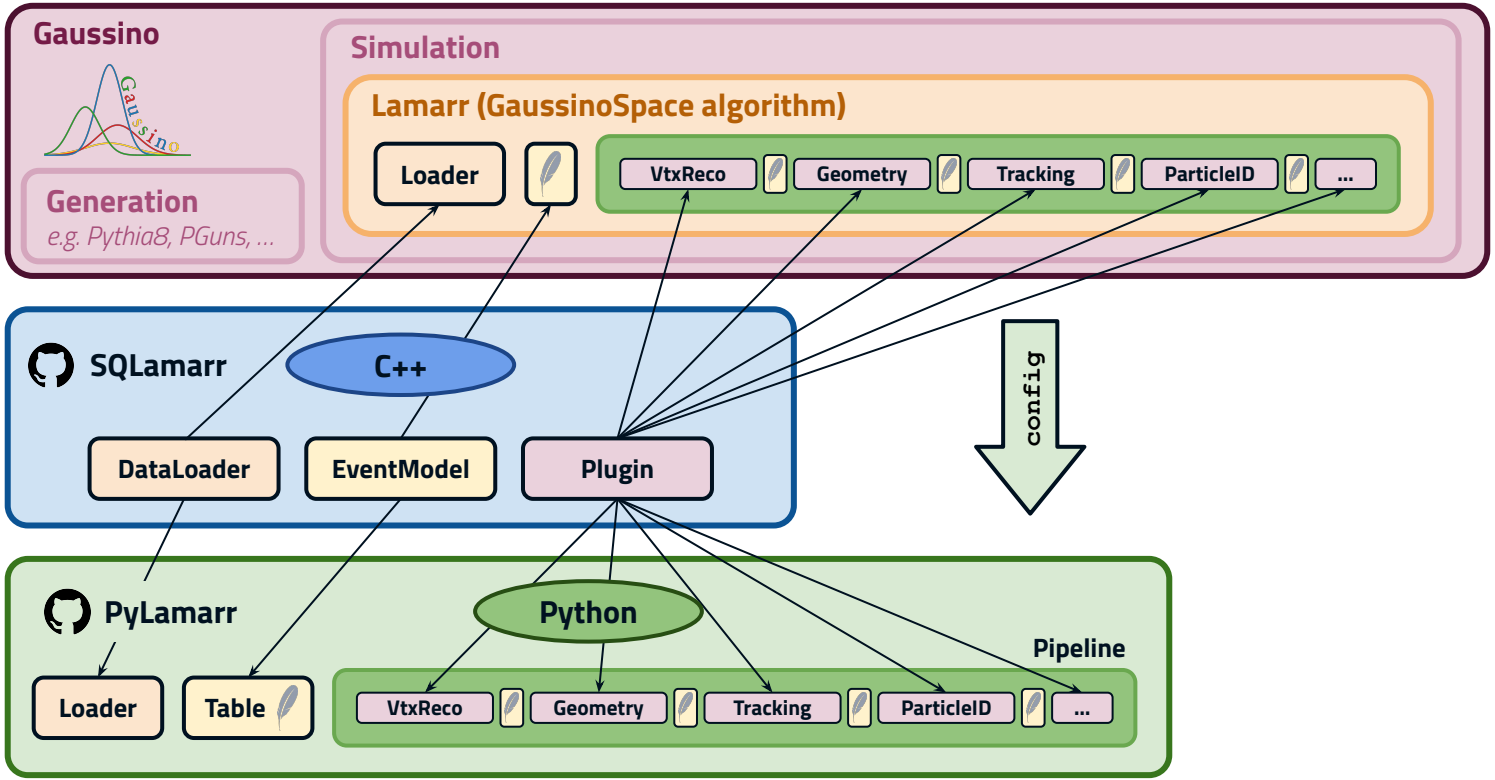
A solution is to deploy the trained models as C file (with a transpiling approach, *e.g.* scikinC [15]) to compile them to binaries, **dynamically linked** at runtime.

Integration of Lamarr with Gauss-on-Gaussino (see first Michal's talk) is currently under development:

- **SQLamarr** (repo, docs) is a C++ library based on SQLite3 that defines classes and interfaces for loading data and managing parameterizations
  - **hard dependency policy** to be compiled within Gaussino (see second Michal's talk)
  - stand-alone application provided
- **PyLamarr** (repo) is a pure-python project designed to configure pipelines
  - based on SQLamarr
  - pipelines can be executed in stand-alone mode



Using Lamarr within Gaussino



SUMMARY AND CONCLUSION

- The Lamarr framework offers to LHCb the **fastest option for simulation** needed to meet the upcoming and future requests for simulated samples
- Lamarr is integrated with the LHCb analysis framework and Lamarr-based simulation can be produced centrally using **LHC Grid resources**
- Great effort on improving the quality of the parameterizations (focus on 2016 data-taking) and developing a new parameterization for the calorimeter focused on **particle-to-particle problem**

Lamarr will never replace the Geant4-based simulation, but may provide soon a precious tool to reduce the pressure on CPU of Detailed Simulation. Lamarr is designed to meet most of the needs for simulation of physics groups, from **designing selection** strategies, **training multivariate classifiers**, to **studying systematics** or **correlation effects**.

THANKS!

Any questions or comments?

You can find me at: matteo.barbetti@fi.infn.it

ICSC
Centro Nazionale di Ricerca in HPC,
Big Data and Quantum Computing

This work is partially supported by ICSC – Centro Nazionale di Ricerca in High Performance Computing, Big Data and Quantum Computing, funded by European Union – NextGenerationEU

References

1. LHCb Collaboration, M. Clemencic *et al.*, J. Phys.: Conf. Ser. **331** (2011) 032023
2. D. Müller *et al.*, Eur. Phys. J. C **78** (2018) 1009
3. M. Rama and G. Vitali, EPJ Web Conf. **214** (2019) 02040
4. M. Paganini et al., arXiv:1712.10321
5. LHCb Collaboration, L. Anderlini, arXiv:2110.07925
6. L. Anderlini *et al.*, PoS **ICHEP2022** 233
7. M. Barbetti, arXiv:2303.11428
8. I. J. Goodfellow *et al.*, arXiv:1406.2661
9. D. Teljék, arXiv:1907.05681
10. F. Scarselli *et al.*, IEEE Trans Neural Netw **20** (2009) 61
11. P. Velickovic *et al.*, arXiv:1710.10903
12. A. Vaswani *et al.*, arXiv:1706.03762
13. A. Dosovitskiy *et al.*, arXiv:2010.11929
14. D. Popov, EPJ Web Conf. **214** (2019) 02043
15. L. Anderlini, M. Barbetti, PoS **CompTools2021** 034

BACKUP

Long, upstream and downstream tracks

Upstream tracks

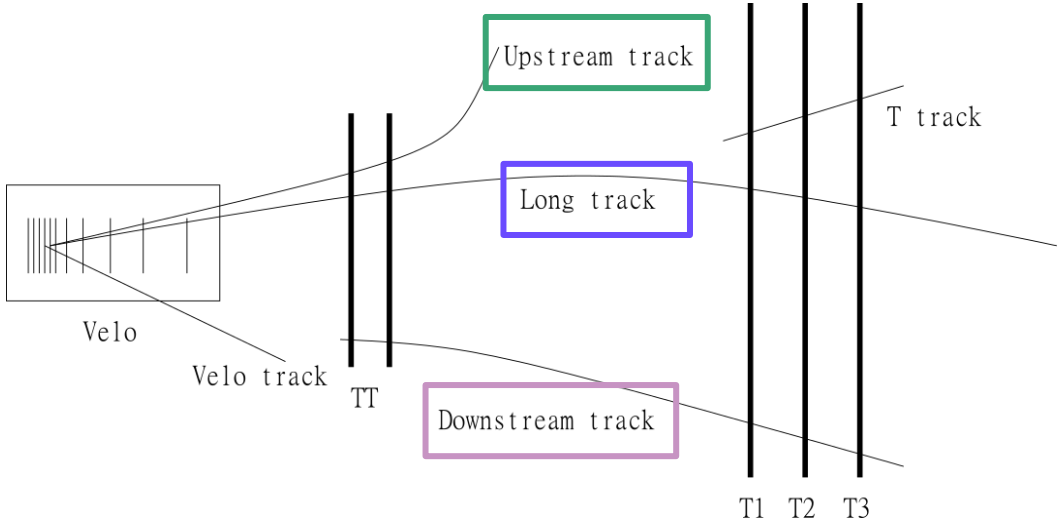
Tracks reconstructed by the tracking stations upstream the magnet.

Long tracks

Tracks reconstructed after having traversed the whole detector.

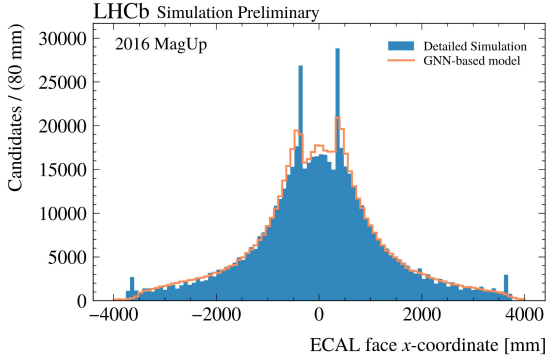
Downstream tracks

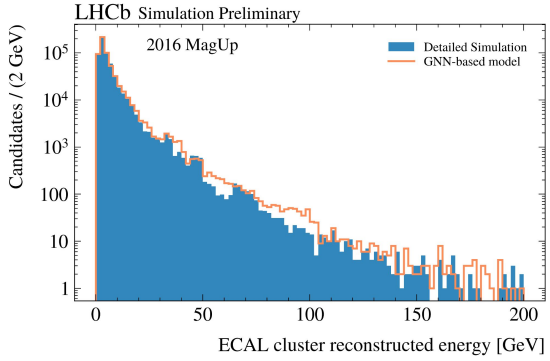
Tracks reconstructed by the tracking stations, but no hits found in the VELO.

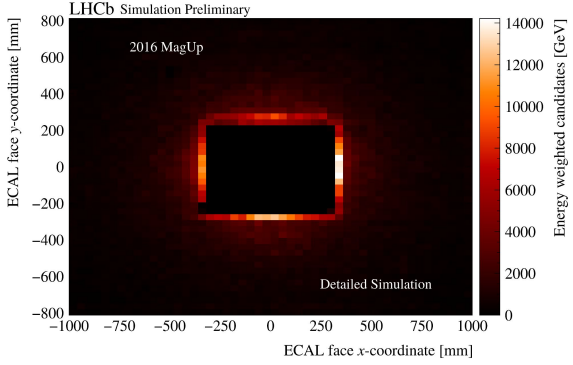


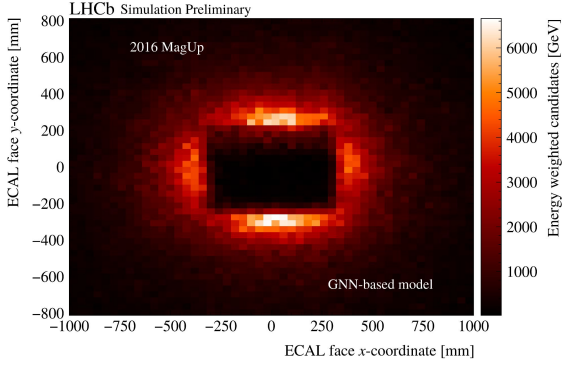
GNN results for calorimeter

LHCb-FIGURE-2023-008



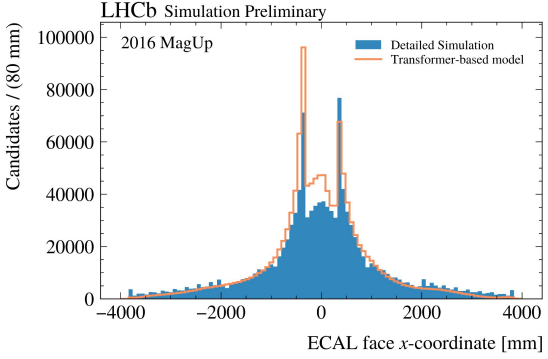


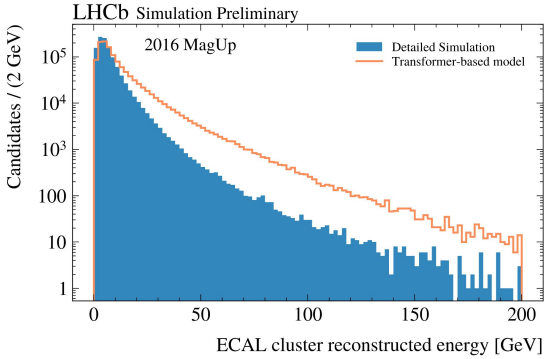


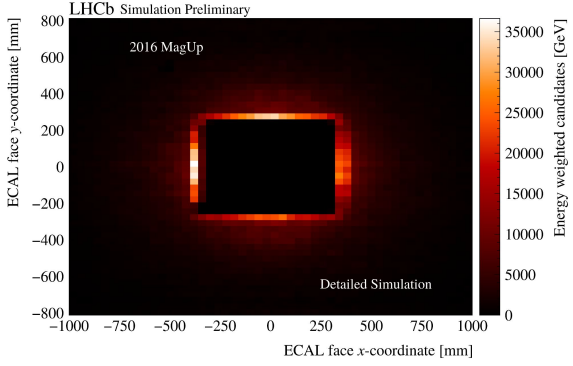


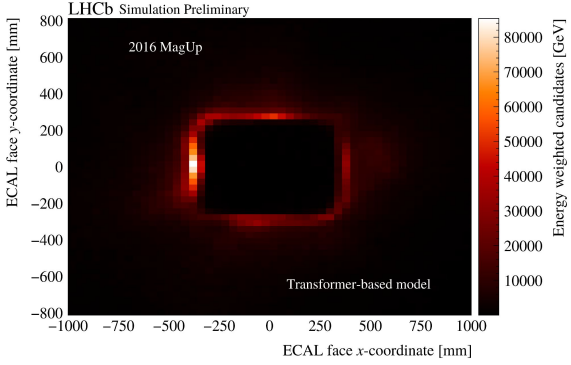
Transformer results for calorimeter

LHCb-FIGURE-2023-008





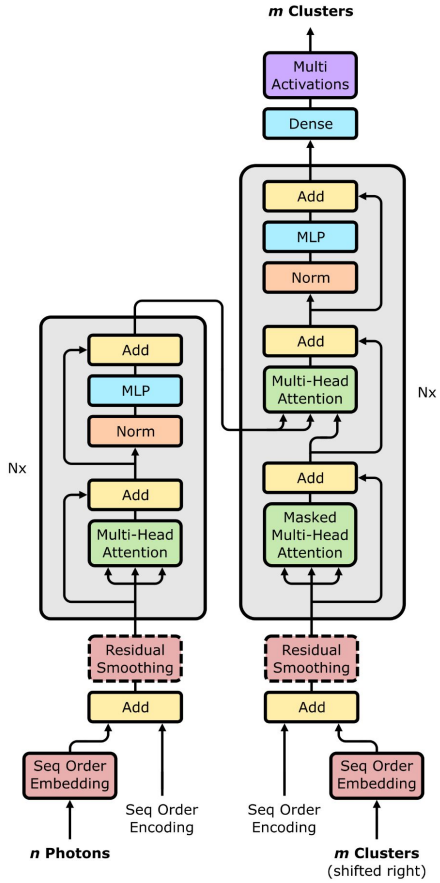




Transformer architecture

A Transformer is a encoder-decoder model powered by **self-attention layers**. The *encoder* processes the source sequence (photons) and allows to parameterize the photon-to-photon correlations, while the *decoder* processes the target sequence (clusters) and allows to parameterize both the cluster-to-cluster and photon-to-cluster correlations.

The attention layers see their input as a set of vectors with **no order**. Hence, the sequence order must be **embedded** within a set of features, otherwise the model will see the input sequence as a *bag of words* instance: `how are you`, `how you are`, `you how are`, and so on, are **indistinguishable**.



Models deployment

TensorFlow and **ONNX** use their own thread scheduler that can lead to huge overhead for HEP applications.

To make the most from the modular logic of Lamarr, we are interested in a deployment tool that allow to easily replace a specific parameterization, **without recompiling the whole pipeline**.

This is the idea behind the ***scikinC*** tool (landerlini/scikinC) where the ML-based models are defined to be dynamically linked to the main application. In this way, models can be **developed and released independently** (more details here).