Autoencoders for real-time event selection at the LHCb experiment

XVII CPAN days
COMCHA session

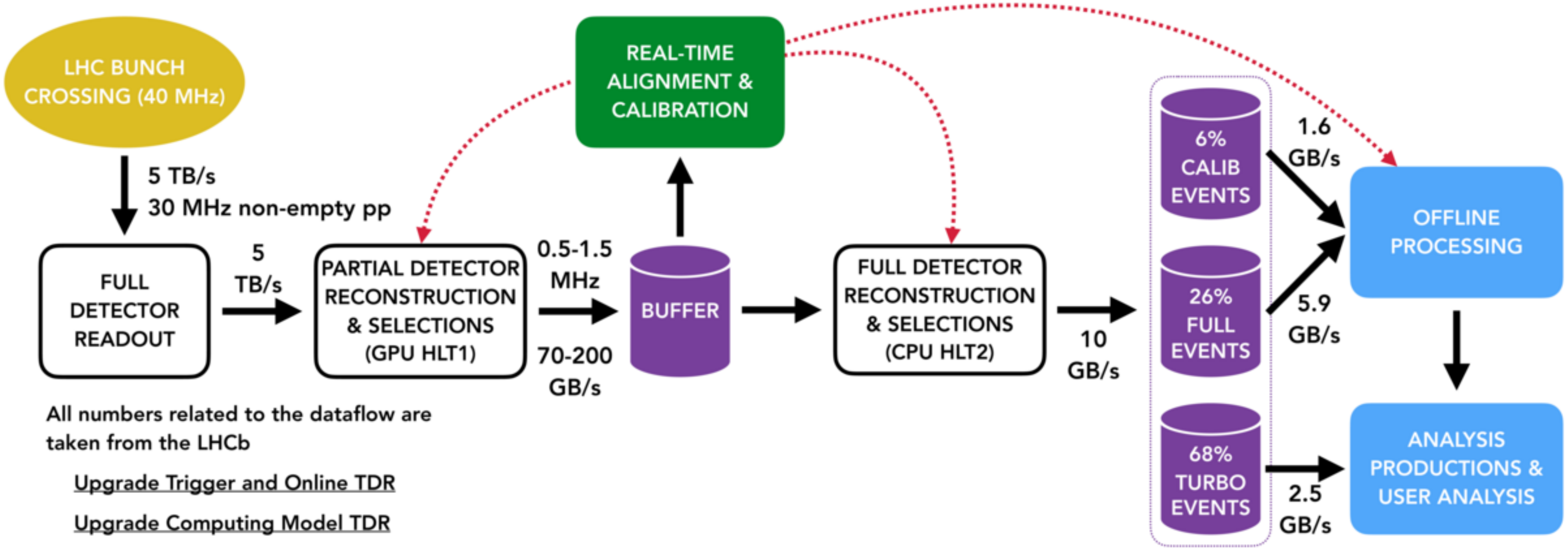
Carla Marin, Lukas Calefice,
Paloma Laguarta

19/11/2025

The LHCb trigger system

$BW = rate \times event\ size$

- Full detector readout at 40MHz without hardware trigger
- Fully **software-based trigger** in two stages:
  - **High Lever Trigger 1** (**HLT1**): partial reconstruction on GPUs at 30MHz, mostly inclusive selections
  - **High Level Trigger 2** (**HLT2**): full offline-quality reconstruction in CPUs, mostly exclusive selections
    - selective persistency: for each selection the part of the event to save offline is chosen



All numbers related to the dataflow are taken from the LHCb

Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

- Offline quality level thanks to real-time alignment and calibration
- For HLT2: limited output bandwidth of **10GB/s**
- Key computing challenge: efficiently select signal while keeping rate low

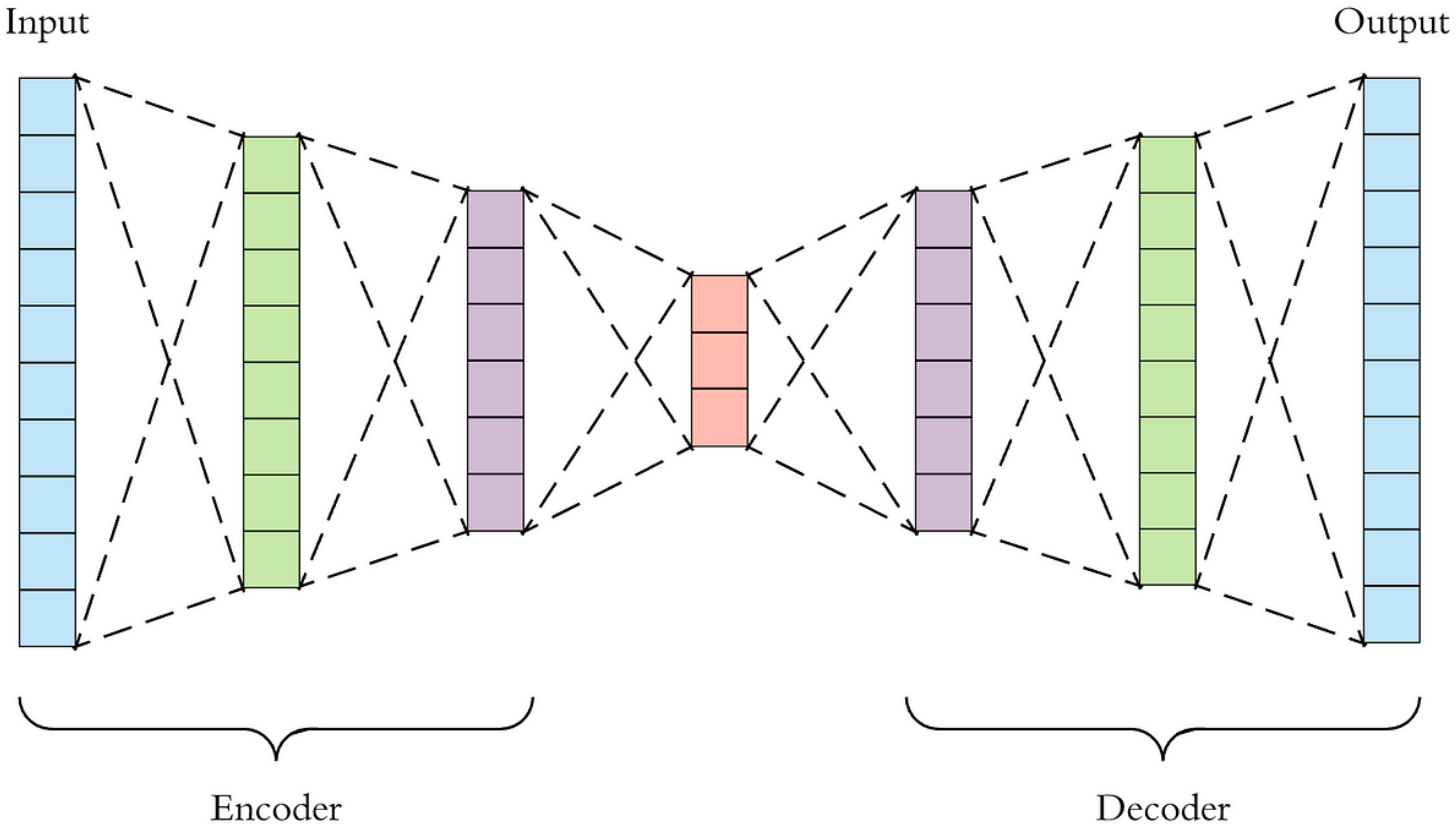
From cut-based to unsupervised ML selection strategies

- Traditional approach: **cut-based** selections to keep signal and suppress backgrounds
- Improved performance with supervised Machine Learning (ML) algorithms such as BDTs and MLPs
- Our work: alternative **unsupervised method** based on an **autoencoder** (AE) trained only on signal for model-independent background rejection

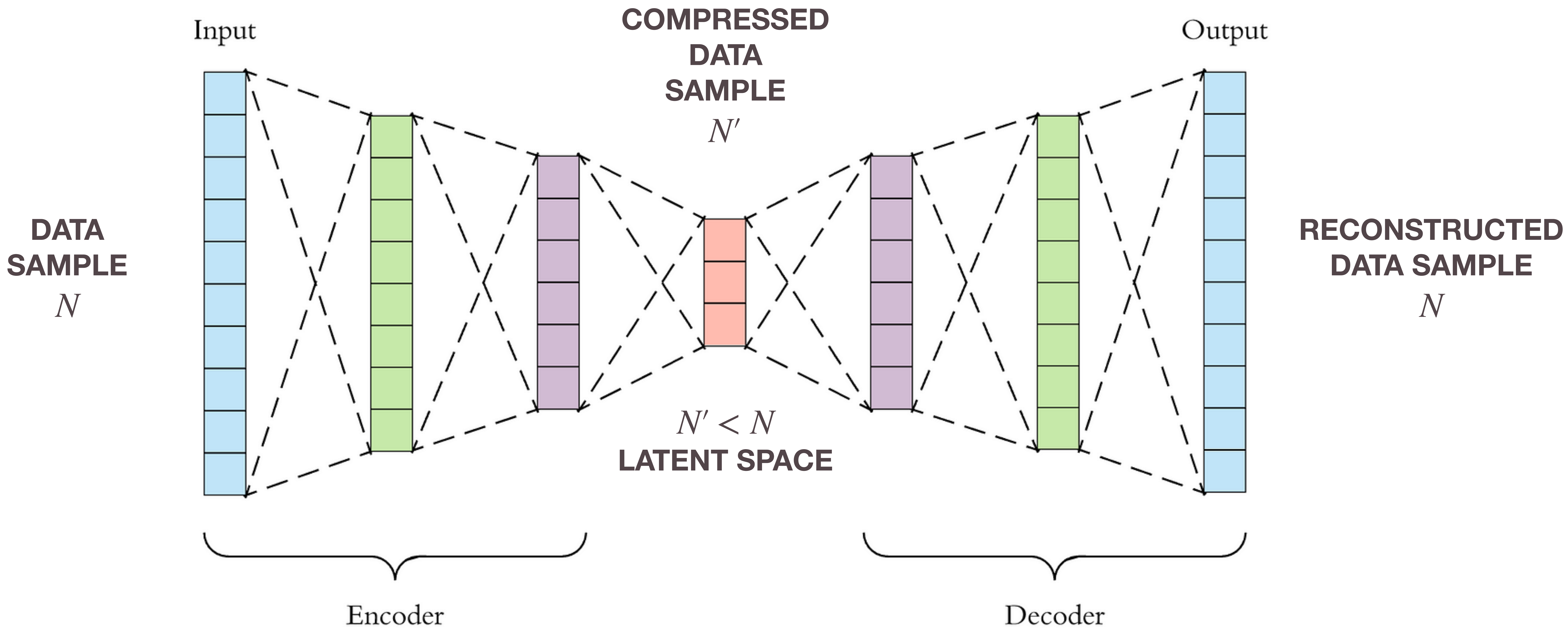
Cut-based	Supervised ML	Autoencoder
+ Simple, interpretable − Hard to optimize globally − Limited discrimination power in high-dimensional spaces − Manual tuning	+ Can capture non-linear correlations between variables − Requires explicit background samples − Limited robustness for unseen backgrounds	+ Can learn the essential patterns of data + Only trained on signal + Does not require explicit background samples − Requires larger training samples

What is an autoencoder (AE)?

- A type of Neural Network (NN) trained to **output the same that was inputted**
- Fancy way of learning the identity function? No!
- Dimensionality is reduced before getting reconstructed using a **bottleneck architecture**
- NN learns a compact representation with a higher degree of abstraction and a deeper understanding of data
- Traditionally used for anomaly detection: trained on majority class, can trigger on anomalous events (LLPs)
- Our approach: **train on signal** (minority class), use to discard background

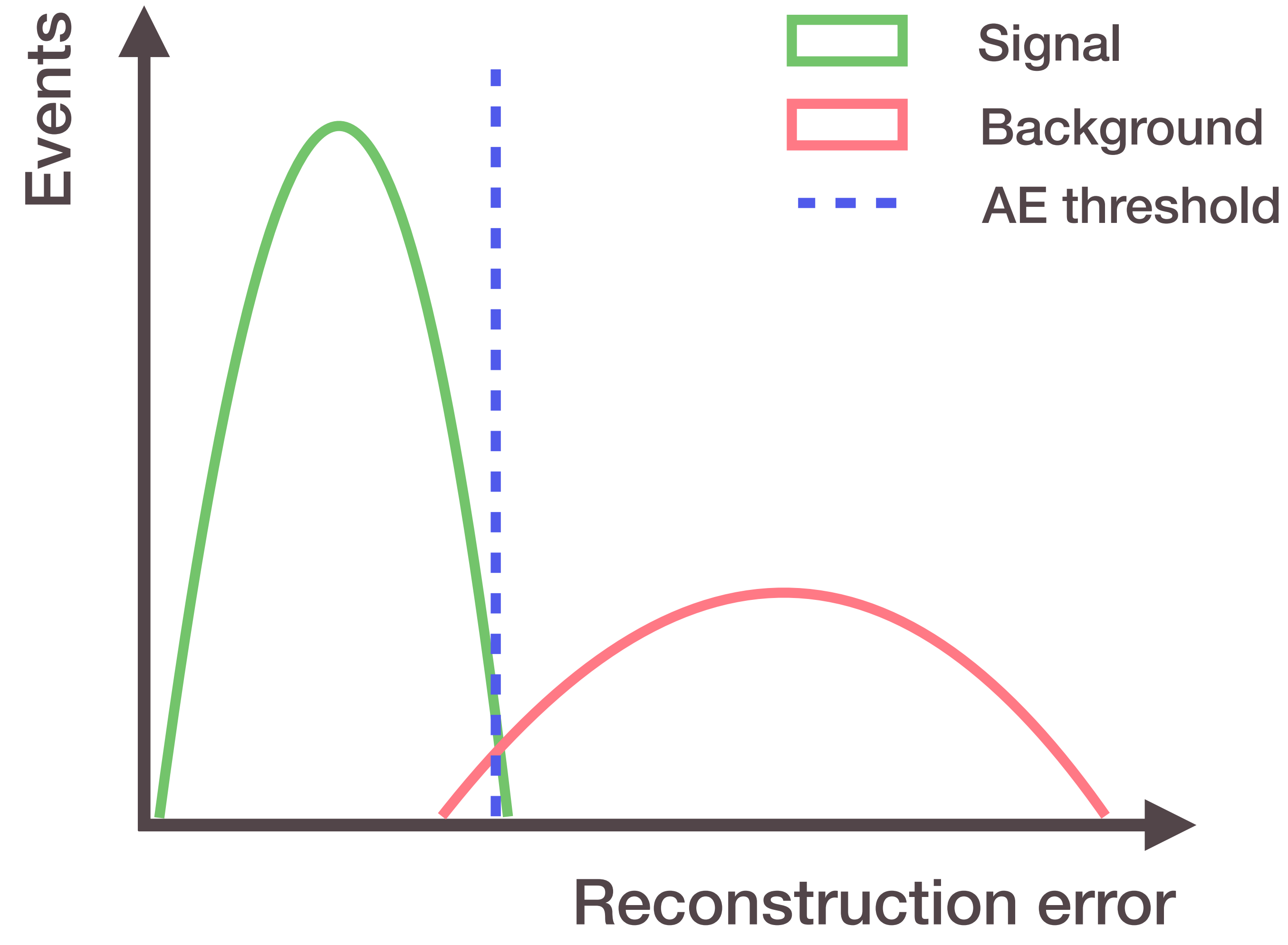


How does an AE work?



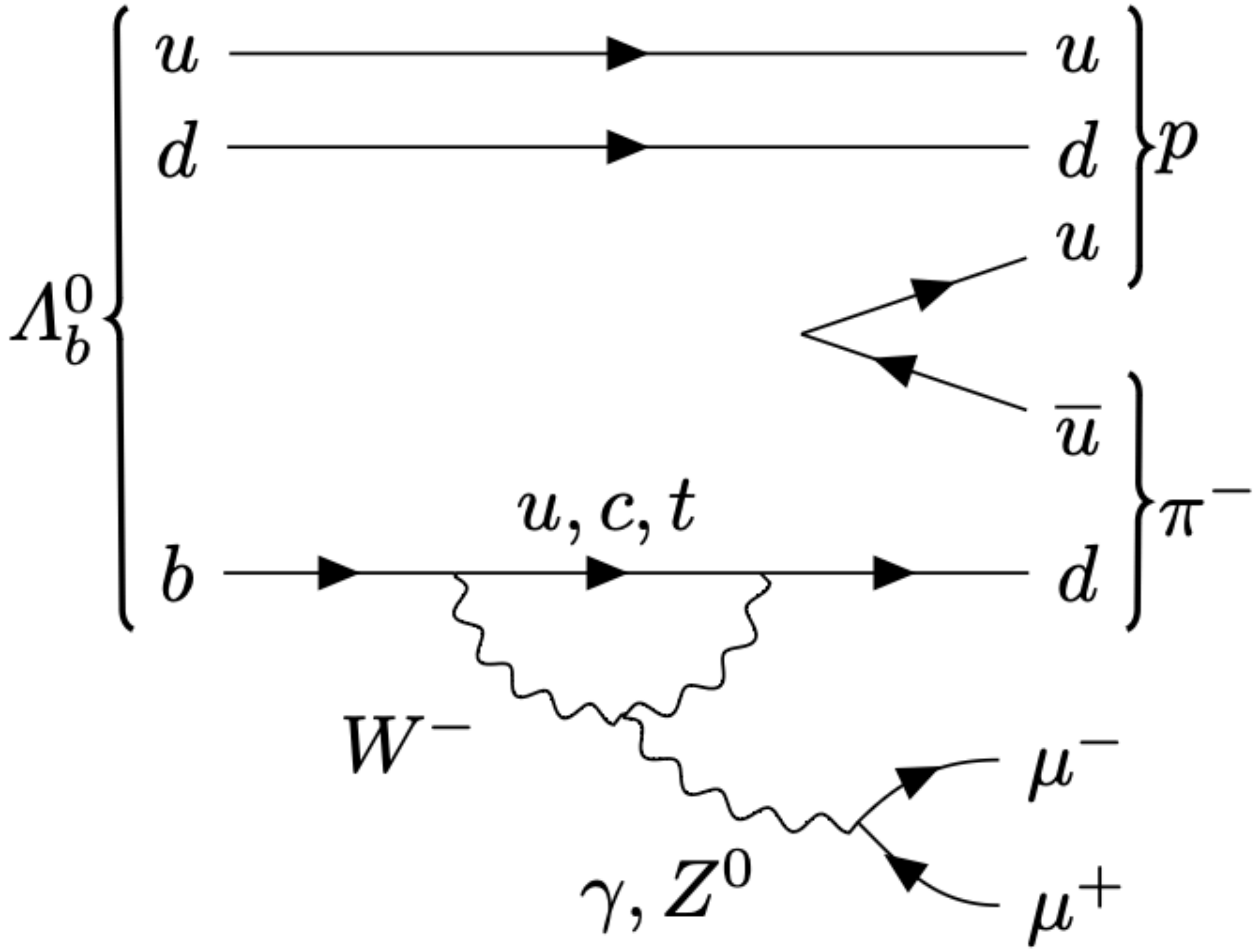
Why is an AE useful for our problem?

- The AE is trained on a <u>specific signal sample</u>
- <u>Learning process</u> → minimization of $\mathscr{L}_{loss} = F\,(\,output - input\,)$, **reconstruction error**
- The <u>weights of the network converge</u> into a **meaningful representation of the data**
- By setting a **threshold** in the reconstruction error, we can <u>filter signal and background</u> without needing a background-like training sample (i.e. model independent background rejection)

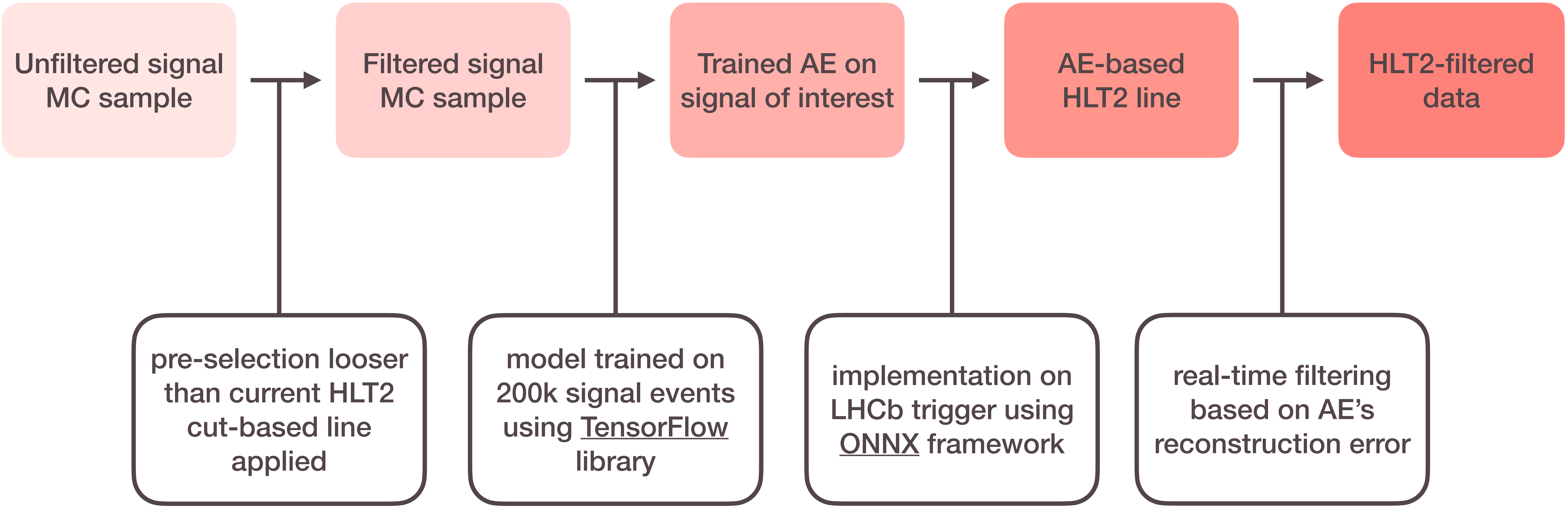


Signal of interest: $\Lambda_b^0 \to p\,\pi^-\mu^+\mu^-$

- Flavour changing neutral current only occurring via loop diagrams
- Suppressed by CKM and GIM mechanisms: very rare decay in the SM with $\mathscr{B} \sim 10^{-8}$ sensitive to NP
- Run 3 analysis in preparation: first step → improve trigger efficiency
- Cut-based selection already in HLT2, cutting on:
  - Vertex and track quality variables
  - Kinematic (p and p_T) variables
  - Particle identification variables



Implementation for $\Lambda_b^0 \to p\,\pi^-\mu^+\mu^-$



Some technical details

model uses kinematic, vertex and track quality variables
24 features in total

5 internal layers with a total of 100k tunable parameters

mean squared error loss used for training

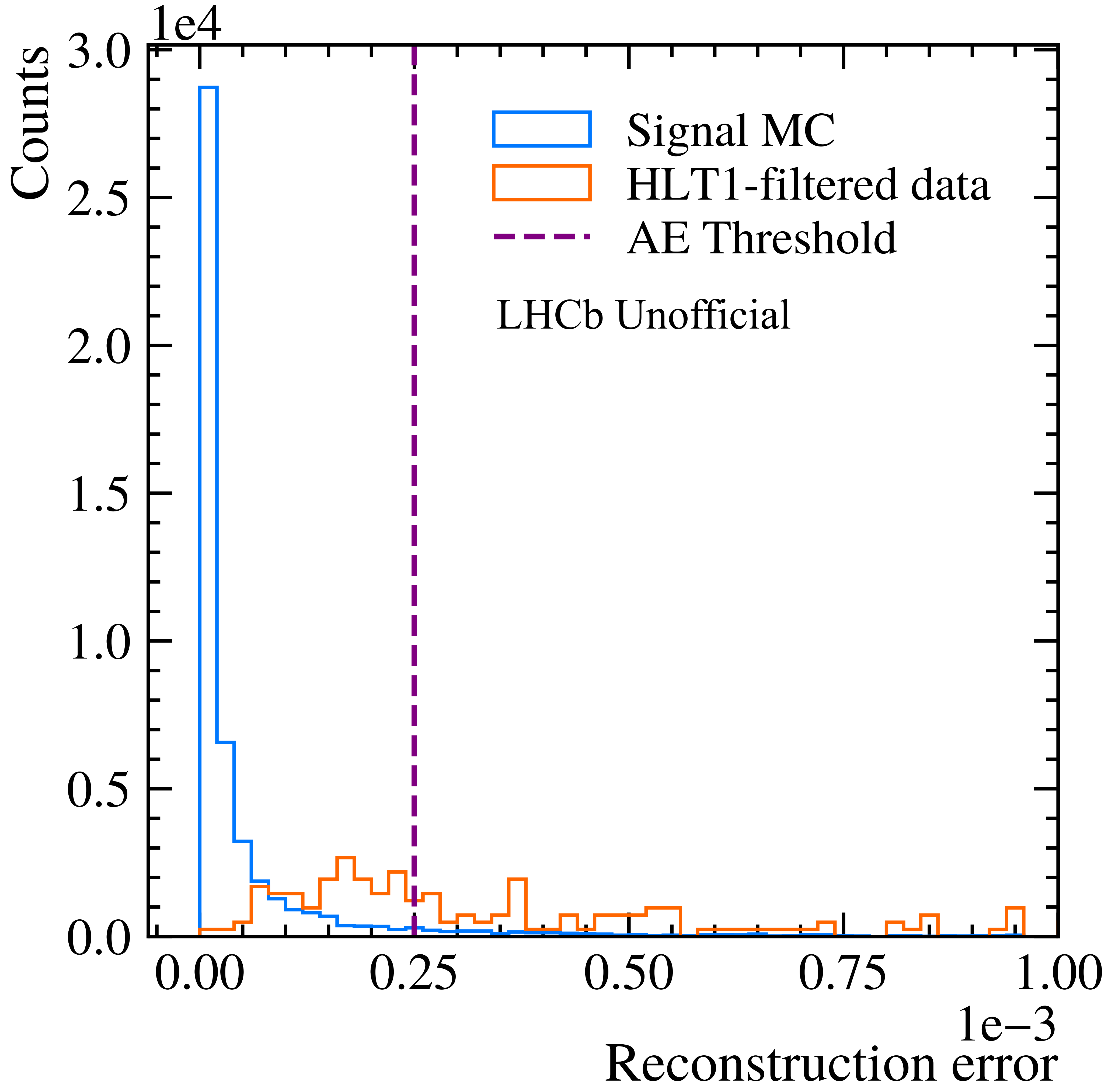
$$rec\ err = \frac{1}{N}\sum_{i=1}^{N}\frac{(x_i - y_i)^2}{x_i}$$

where $N = 24$ features

trained for 6k epochs, with batch size 32 using Adam optimizer

algorithm takes an average of $0.3\ \%$ of throughput in HLT2

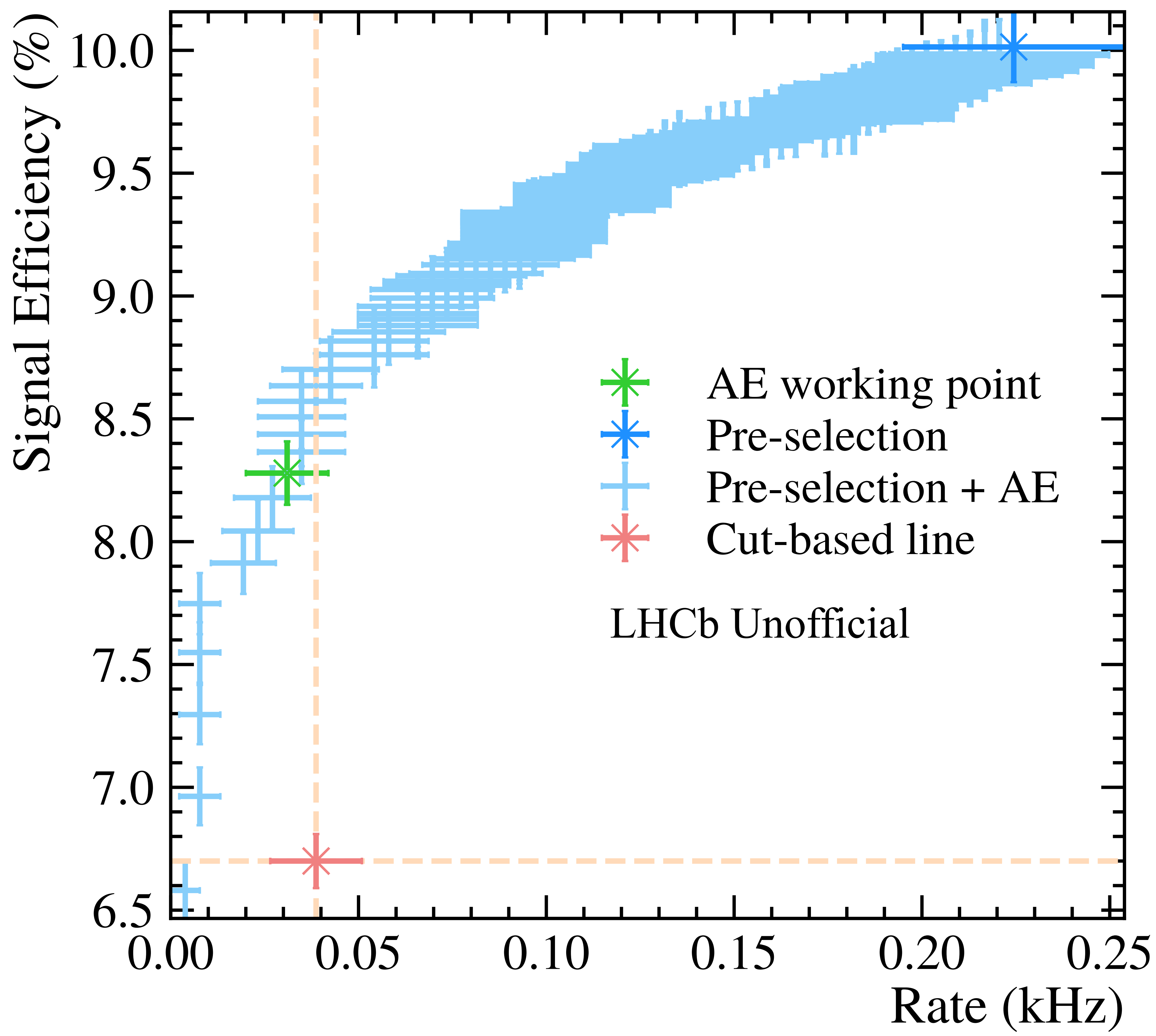
Results



- **MC simulation of 2024 data** used to evaluate model's performance on signal
- **HLT1-filtered data of 2024 run** used to evaluate model's performance on background
- Reconstruction error provides a **tunable trade-off** between signal efficiency and background rejection, which translates into rate

Results

- AE achieves **stronger separation** power than current cut-based line
- Same rate: **increase** in signal efficiency by <u>28 %</u> with AE
- Same signal efficiency: **decrease** in rate by <u>80 %</u> with AE



Conclusions and future work

- This proof-of-concept work has shown that an **autoencoder-based selection** at HLT2 level can achieve **improved performance** with respect to cut-based methods.
- This showcases the **potential of unsupervised ML algorithms** for real-time event selection.
- The enhanced signal efficiency could be crucial for analyses of **suppressed decays** studied at LHCb.
- AE has been **running** in the LHCb trigger since October!
- **Ongoing steps:** validate performance on data and evaluate impact on analyses.
- **Future** possible approaches with this method involve **inclusive selections** at both HLT1 and HLT2 level.
- Ideas are welcome!

Thanks for your attention!
Questions? :)

Results, towards LbToPpHmMuMu

- Similar procedure with LbToPpKmMuMu:
  - Get signal MC with looser cuts than line ‘Hlt2RD_LambdabToPKMuMu’
  - Apply same pre-cuts we had for LbToPpPimMuMu
  - Apply AE trained on LbToPpPimMuMu
  - Check signal efficiency dependency on AE output
- The idea is to do a similar study considering the rate, WIP