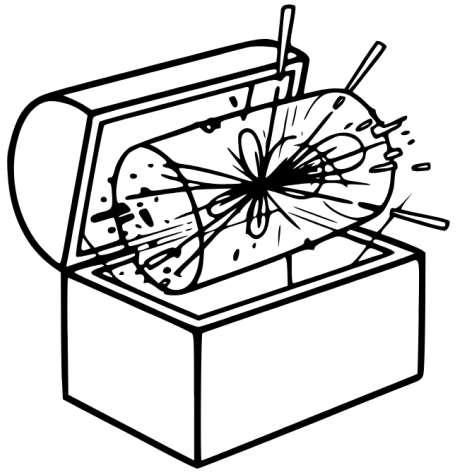

Brookhaven
National Laboratory
U.S. DEPARTMENT
of ENERGY
Treasure and Friends
Viviana Cavaliere (BNL)
@BrookhavenLab

What is Treasure?

- *Tokenized Representations for Energy-frontier AI Searches via Understanding and REasoning*
  - DOE HEP American Science Cloud (AmSC) Intelligent Data Activities Pilot



Contacts:

Viviana Cavaliere, Haider Abidi, Gabriele D'Amen (BNL)

Paolo Calafiura (LBNL)

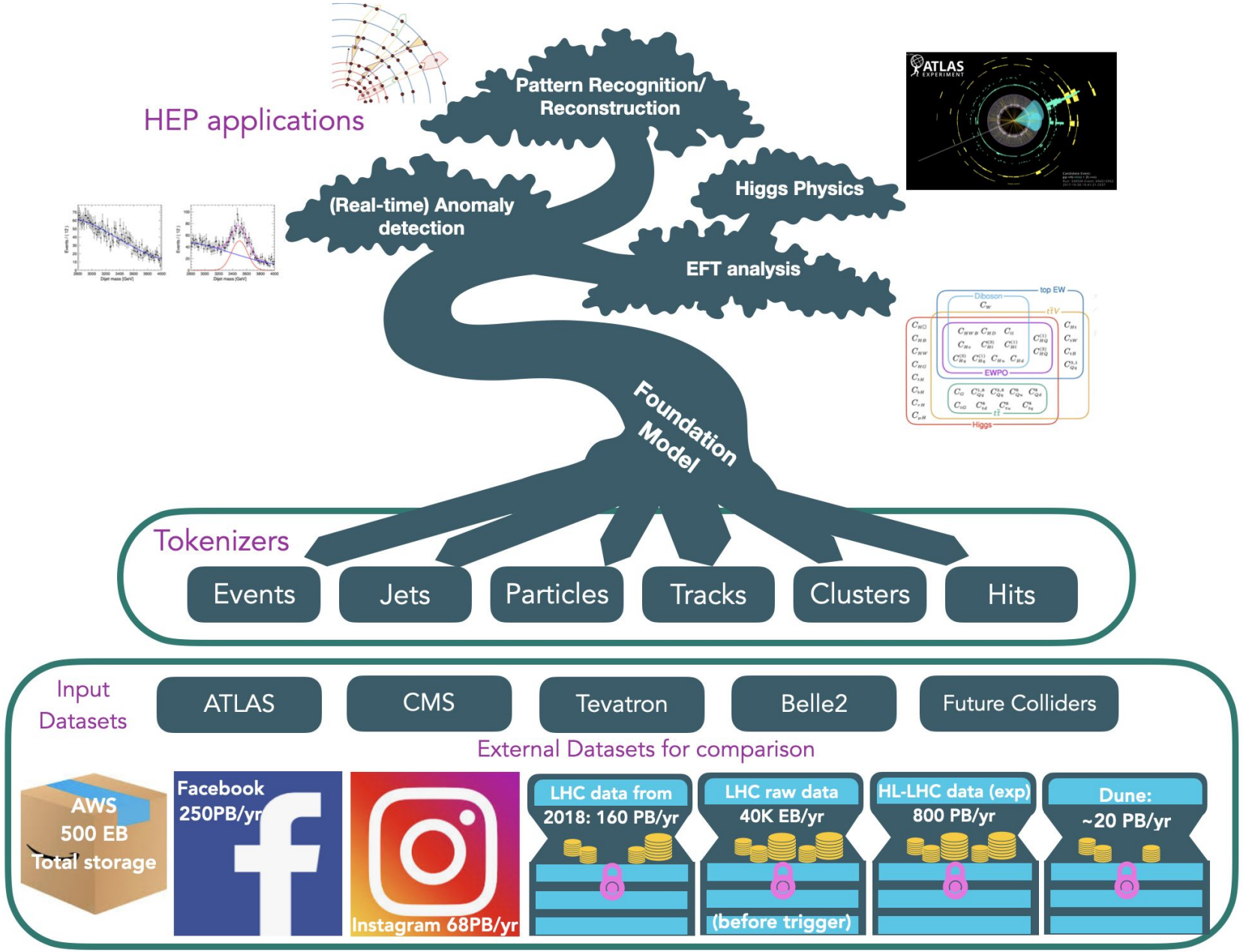
Walter Hopkins (ANL)

Michael Kagan (SLAC)

Kevin Pedro (FNAL)

Brookhaven National Laboratory · Fermilab · Berkeley Lab · SLAC National Accelerator Laboratory · Argonne National Laboratory

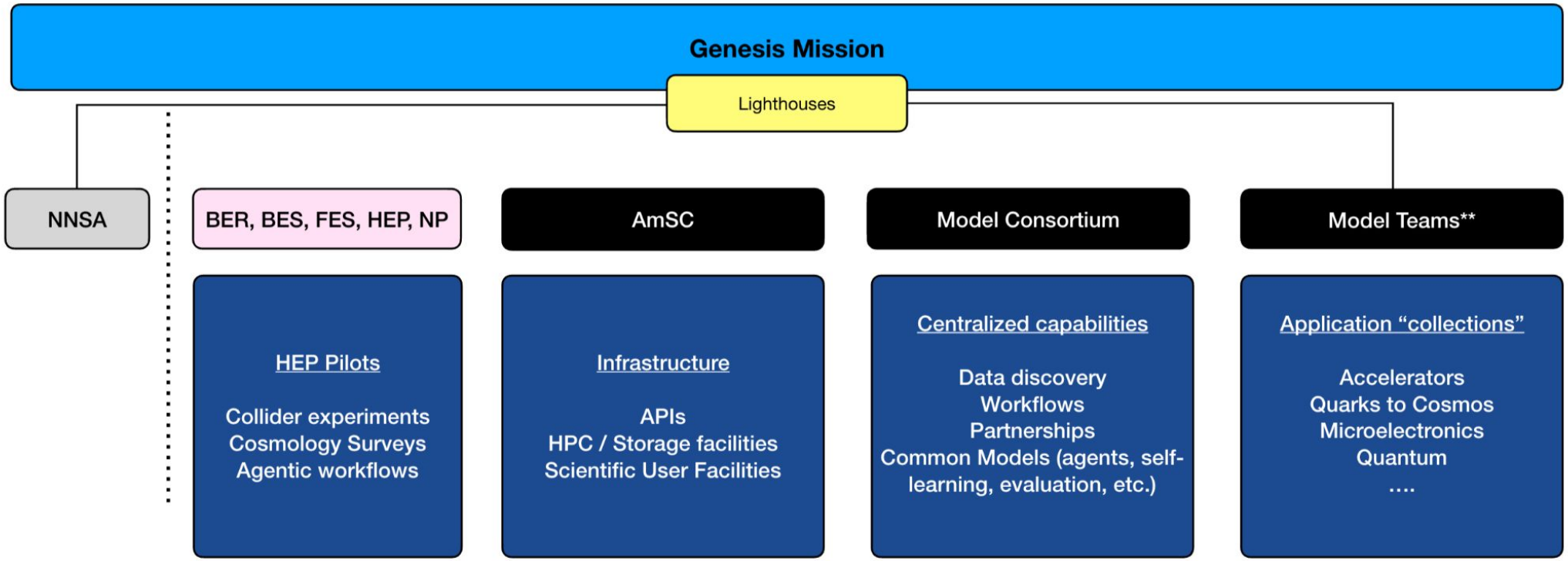
What is Treasure?



- *Treasure prepares **collider data** so it can be effectively used by modern AI methods*
- It transforms heterogeneous, experiment-specific data into standardized, **tokenized**, AI-ready representations.
- **Builds Foundation Models** to learn across experiments, detectors, and eras of data.

Goal: unlock new discovery potential from both current and legacy high-energy physics datasets.

Where do we fit in the Genesis Ecosystem?



from Nhan Tran

Why This Matters

Collider experiments generate enormous data volumes in incompatible formats.

This fragmentation limits cross-experiment analysis and reuse of legacy data.

Multi-Experiment Training

Learn detector-independent physics features.

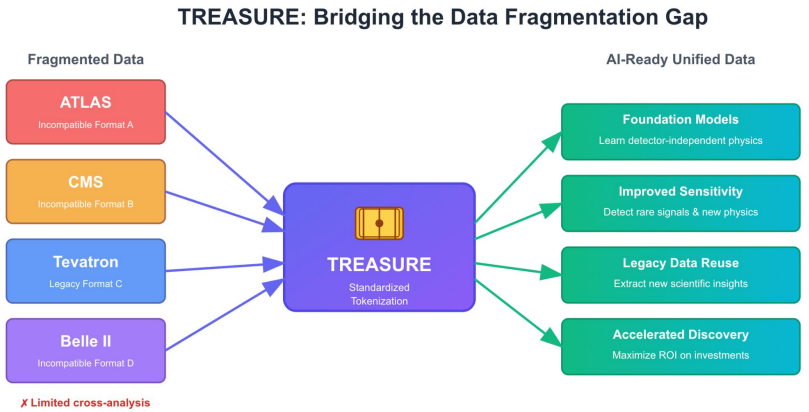
- Improved sensitivity to rare signals.
- More data!

Legacy Data Revaluation

Legacy datasets underutilized due to technical obstacles.

⇒ *Extract new information and extend scientific impact.*

TREASURE bridges this gap, accelerating discovery and maximizing return on investments.



Input Datasets Include

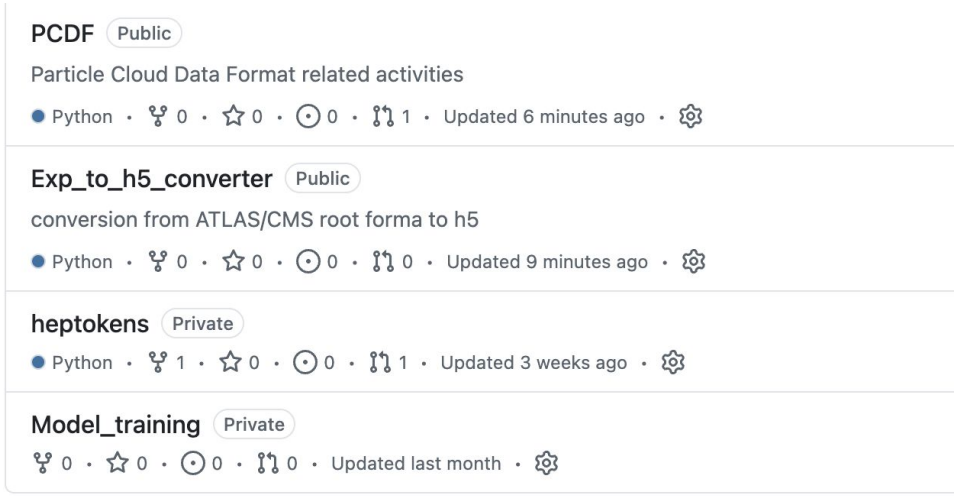
aspen
jetclass
jetset
atlas
cms
belle 2
...

Fermilab BERKELEY LAB SLAC NATIONAL ACCELERATOR LABORATORY Argonne NATIONAL LABORATORY

The TREASURE Pipeline

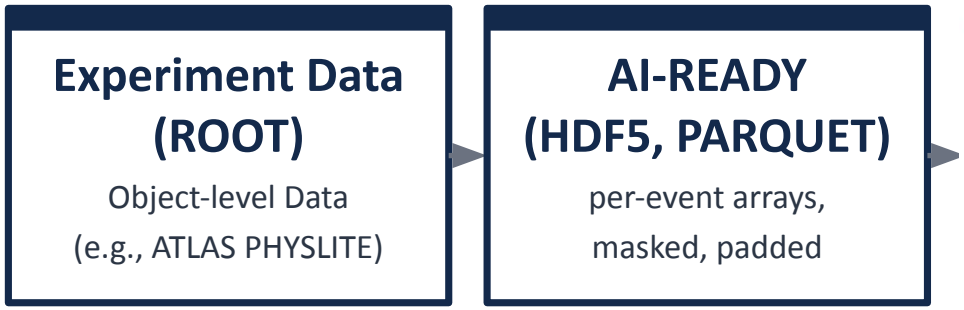
Each step is going to be made public

- **Decoupled by design.** Tokenization is separated from training: the same Parquet file can feed multiple trainings — **no re-tokenization**. Parquet is compatible with Spark, Arrow and DuckDB for distributed serving.
- However if re-tokenization is necessary for a different task that is also possible



Brookhaven National Laboratory

Conversion from Experiment Ntuples to hdf5

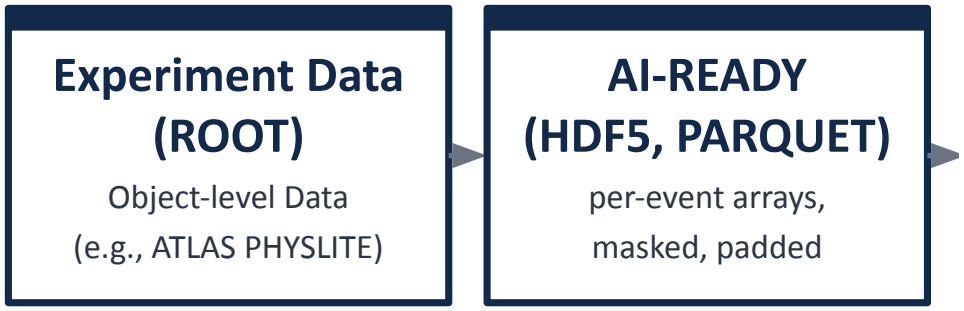


standalone converter that reads experiment-native files and writes flat HDF5 arrays;

https://github.com/Treasure-AmSC/Exp_to_h5_converter/blob/main/atlas_to_h5_converter.py

```
Converts ATLAS Run 2 Open Data (PHYSLITE format) into analysis-ready HDF5
files for the TREASURE foundation model pipeline.

HDF5 Layout
-----------
  /common/    — Cross-experiment variables (same keys for ATLAS and CMS).
                All energies/momenta in GeV, angles in radians.
  /atlas/     — ATLAS-specific variables (ID flags, b-tagging, etc.).
  /truth/     — MC truth particles (electrons, muons, photons, taus,
                bosons, jets, MET). Only present for simulation.
  /metadata   — HDF5 group attributes: experiment, DSID, process name,
                cross-section, selection cuts, provenance.

Schema
------
  common/electrons : pt, eta, phi, charge, trk_iso03, mask, n
  common/muons     : pt, eta, phi, charge, trk_iso03, mask, n
  common/taus      : pt, eta, phi, charge, mask, n
  common/photons   : pt, eta, phi, trk_iso03, mask, n
  common/jets      : pt, eta, phi, mass, n_trk, mask, n
  common/tracks    : pt, eta, phi, d0, z0, mask, n
  common/met       : pt, phi, sumet                    (scalar per event)
  common/event     : pvx, pvy, pvz, mu, experiment_id, is_simulation

  atlas/electrons  : LHMedium, LHTight, topoetcone20, ptvarcone30, mass
  atlas/muons      : topoetcone20, ptvarcone30
  atlas/taus       : NNDecayMode, RNNJetScore, RNNEleScore
  atlas/photons    : isTight, topoetcone20, topoetcone40, ptcone20
  atlas/jets       : DL1d_pb/pc/pu, GN2_pb/pc/pu, QG_nTracks/Width/C1
  atlas/tracks     : qOverP, chiSquared, nDoF
  atlas/event      : event_number, run_number, mcChannelNumber
```

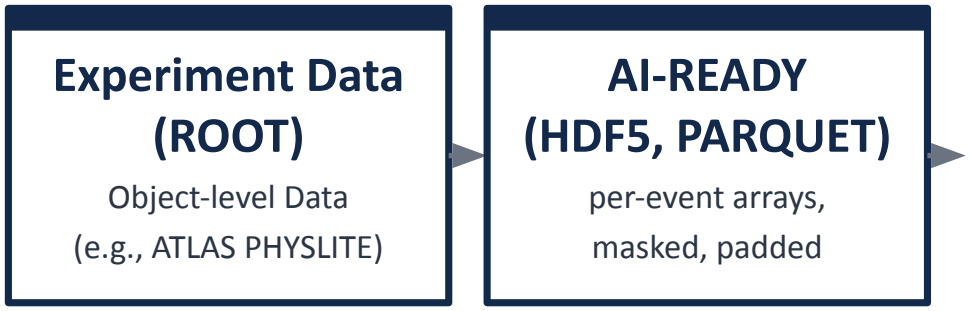
Conversion from Experiment Ntuples to hdf5



Some very basic selection applied to objects

```
Electrons:
  - pT > 5 GeV
  - LH Loose ID (DFCommonElectronsLHLoose)
  - Author == 1 (single-track) or 16 (forward)
  - Object Quality: (OQ & 1446) == 0
  - |n| < 2.47
  - |z0·sinθ| < 3 mm

Muons:
  - pT > 5 GeV  (calo-tagged: pT > 15 GeV)
  - DFCommonMuonPassPreselection (standard ATLAS muon preselection)
  - |n| < 2.7
  - Combined + CaloTagged: |d0| < 3 mm, |z0·sinθ| < 3 mm
  - StandAlone: no impact parameter cut

Photons:
  - pT > 15 GeV
  - |n| < 1.37 or 1.52 < |n| < 2.37 (crack veto)
  - DFCommonPhotonsIsEMLoose

Taus:
  - pT > 20 GeV
  - |n| < 2.5, excluding 1.37 < |n| < 1.52 (crack veto)
  - nTracks == 1 or 3
  - |charge| == 1
  - RNNJetScore > 0.15 (~Loose WP)

Jets:
  - pT > 25 GeV
```

Conversion from Experiment Ntuples to hdf5



Some basic metadata information for now

```
Metadata (/metadata attrs)
--------------------------
  converter_version, converter_changelog, experiment, experiment_id,
  input_file, n_events, DSID, process name, generator, cross-section (pb),
  filter efficiency, k-factor, √s, data-taking year, MC campaign,
  license, DOI, citation.  All selection cuts recorded for reproducibility.
  Uses atlasopenmagic package if installed, else hardcoded lookup table.
```

PCDF: The Particle Cloud Data Format

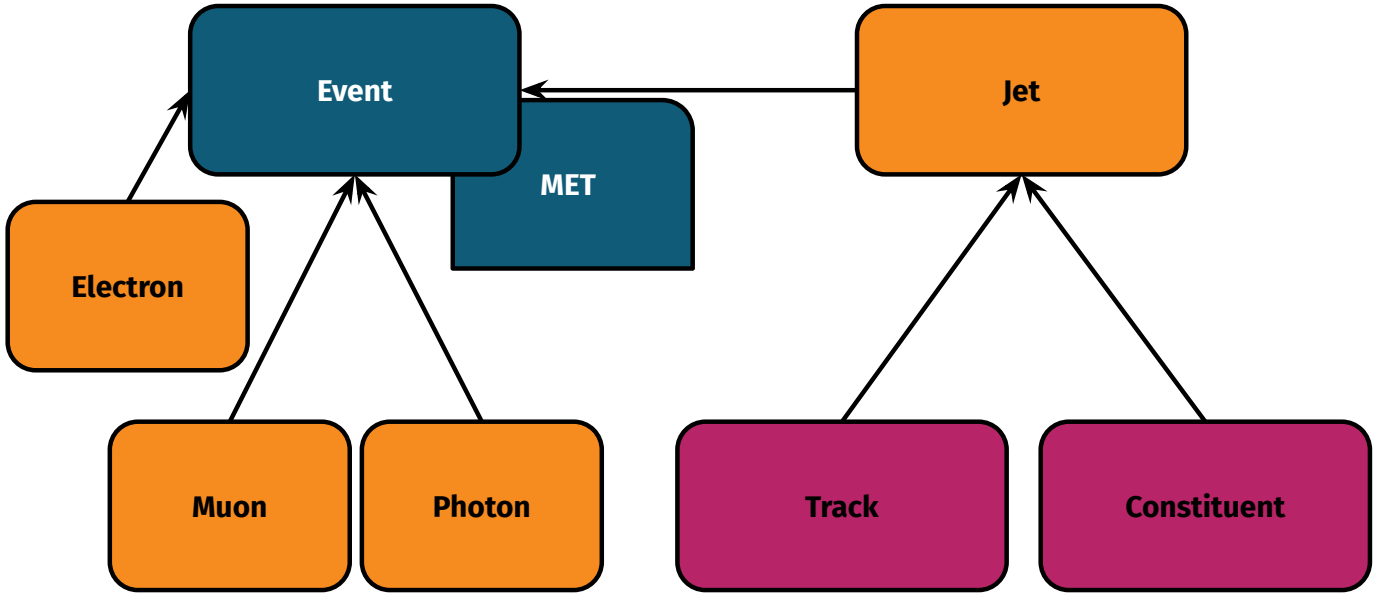
Beojan Stanislaus

Flat Table Architecture

Use of indices as cross-references allows tables to be **flat**.

Enables storage as **Parquet** files with rich metadata (column names, statistics).

Modern software (e.g., **Polars**) can group by indices to generate n-dimensional **HDF5** tables efficiently.



Next step: develop **web API** to automate pre-determined **transformations**.

Event-level Tokenization step

TOKENIZER

VQ-VAE / FSQ / PQ / RVQ per type

Each object type has its **own separate VQ-VAE** (or FSQ/PQ/RVQ) with its own codebook, because each object type has completely different features:

- Electron: pT, η, φ,
- Jet: pT, η, φ, mass, btagging etc
- Track: qOverP, η, φ, d0, z0, pT, χ^2, nDoF

Encoder > **Codebook** > **Decoder**

Compresses 8–16 physics features into a compact latent vector z

Dictionary of N templates; snaps z to the nearest entry → integer k

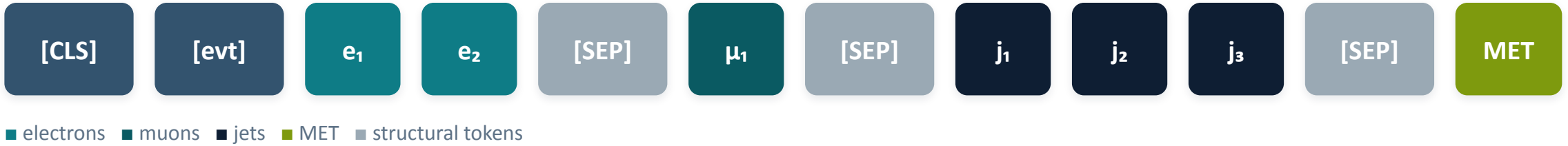
Reconstructs the features from the codebook entry (training only)

Inspired by https://arxiv.org/abs/2401.13537: **Masked Particle Modeling on Sets: Towards Self-Supervised High Energy Physics Foundation Models**
and OmniJet-α (arXiv:2412.10504)

Brookhaven National Laboratory

From objects to one event sequence

Objects are tokenized independently — the electron VQ-VAE never sees the jets. We tell the transformer what each token is with a type embedding, then let self-attention discover the cross-object physics.



How each token is built

```
token_repr  =  token_embedding(id)
            +  type_embedding(type)
            +  position_embedding(pos)
```

Type IDs (electron=4, muon=5, jet=6, …) tell the model what each token is — no one-hot feature needed.

Self-attention recovers the relationships

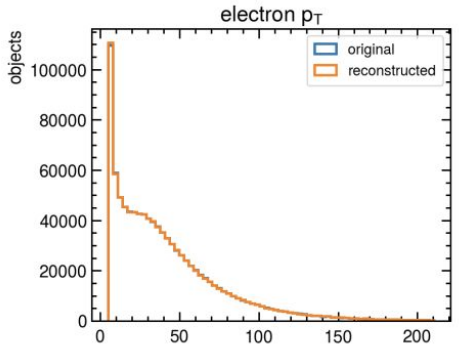
At tokenization time we never compute ΔR, Δφ or invariant masses between objects.

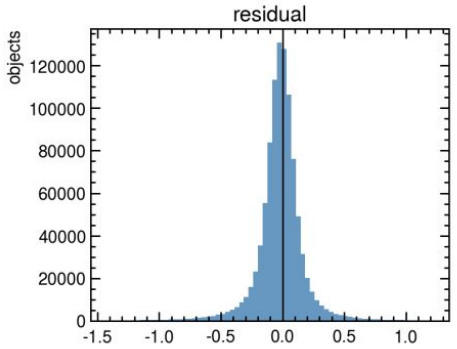
But every token attends to every other token. The attention layer sees both objects' φ values and can learn "e_1 and j_1 are close" — recovering the cross-object structure downstream.

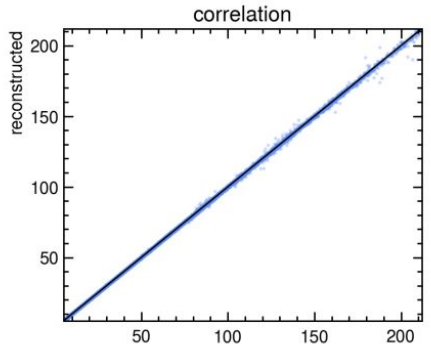
How to estimate tokenization loss

Nazlim Agaras, Nicholas Luongo

We look at how each variable is reconstructed after tokenization.





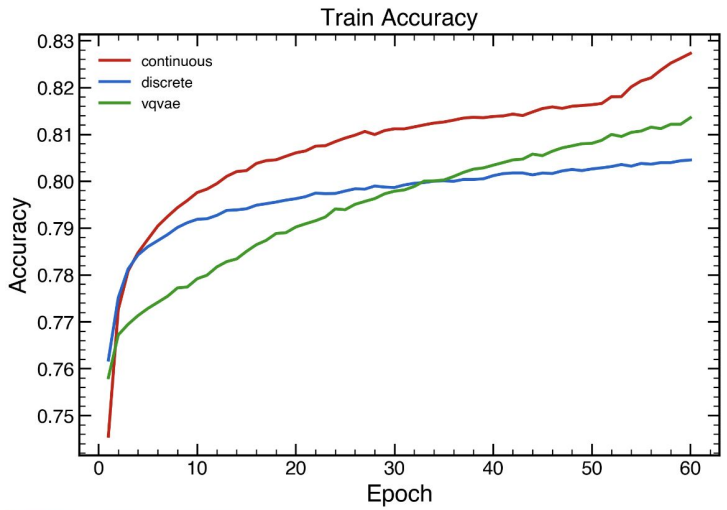


We use a downstream easy task to quantify the loss

Task 1

$H \rightarrow ZZ \rightarrow 4\ell$

Train classifier to separate Higgs signal from background. Direct comparison: standard classifier vs foundation model.



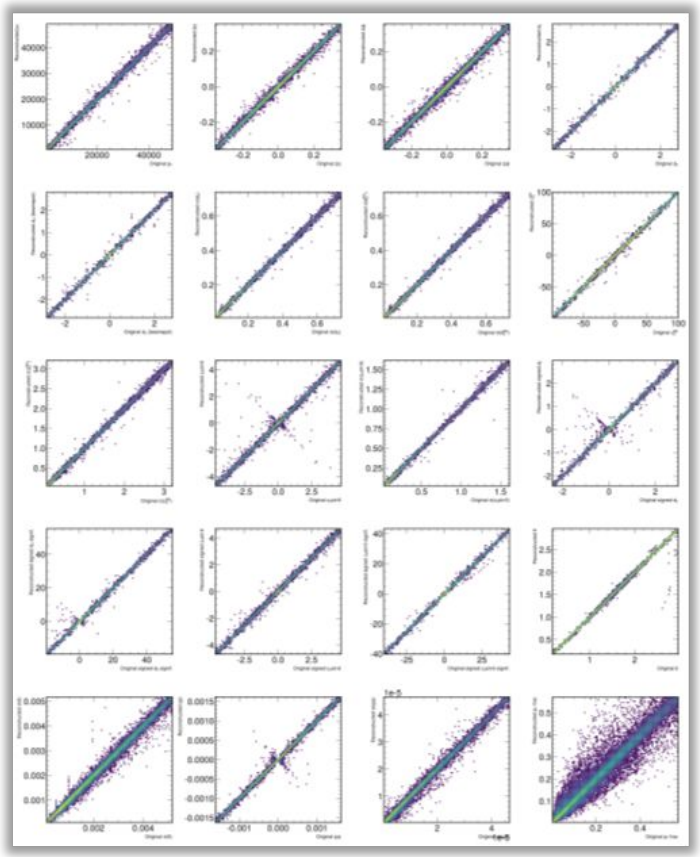
Brookhaven National Laboratory

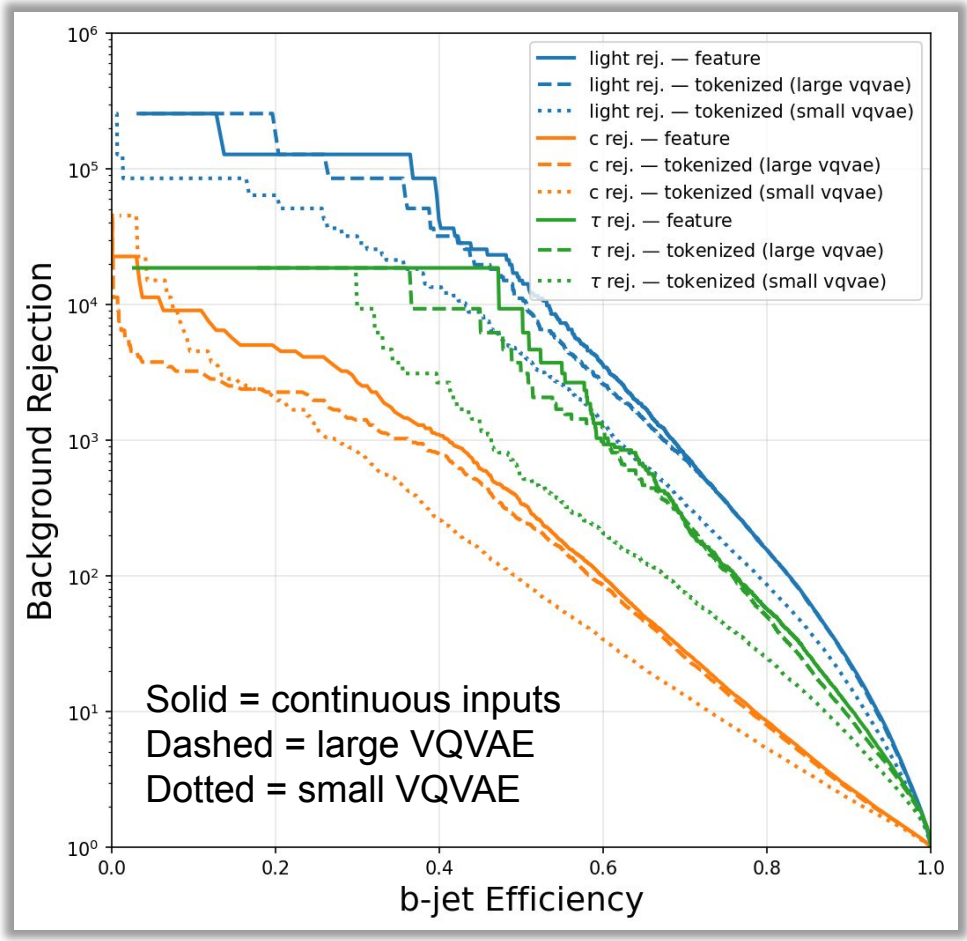
heptokens: A library for learning jet representations

Quick start

```
from heptokens.data import
  SingleFileMapModule

from
heptokens.models.vq_vae
import
  LitVqVae

dm = SingleFileMapModule(
  data_path="mc-ttbar.h5",
  num_csts=40)

model = LitVqVae(
  codebook_size=512,
  num_quantizers=3)

trainer.fit(model, dm)
```

Jeff Krupa, Michael Kagan



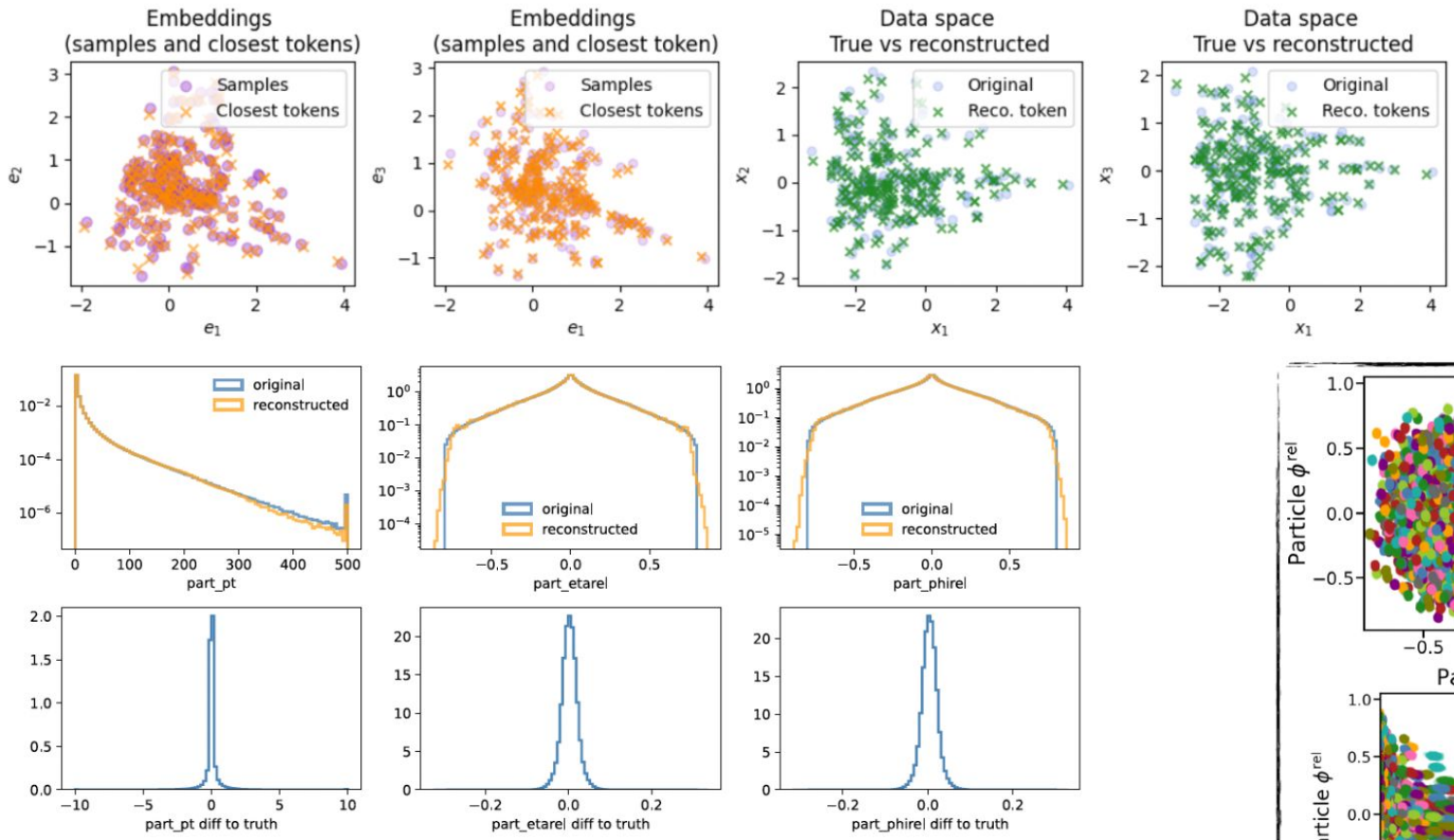


Tokenization of CMS jet open datasets

Oz Amram

Aspen open jet dataset tokenization

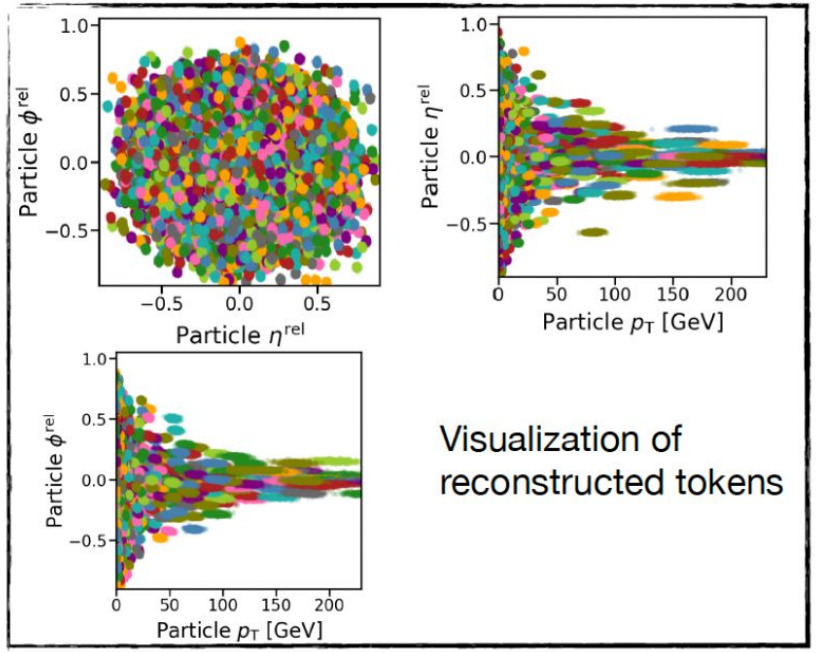
(https://www.fdr.uni-hamburg.de/record/16505)



Comparison of original and reconstructed particle pt, η^{rel}, ϕ^{rel} inside jets

$\eta^{rel} = \eta^{particle} - \eta^{jet}$, $\phi^{rel} = \phi^{particle} - \phi^{jet}$

Codebook size 8192



Visualization of reconstructed tokens

First Foundation Model development

Pre-training

Masked-Event-Modelling

Data	ALL events (real + MC), no labels
Method	mask 15% of object tokens at random
Task	predict the masked tokens from context
Learns	"what does a typical collision look like?"

Output → a pre-trained transformer backbone

Fine-tuning

task-specific, one head at a time

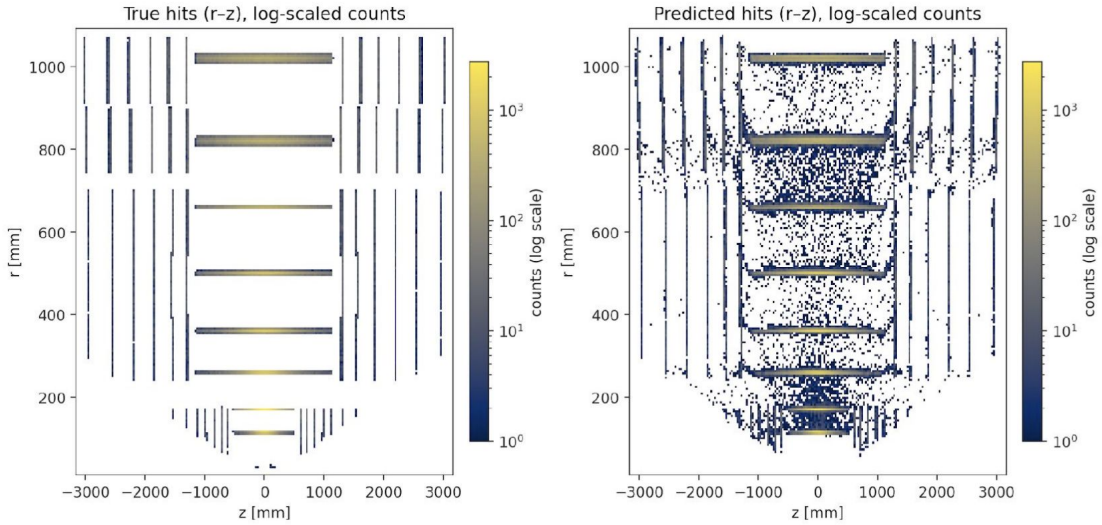
Freeze (or lightly tune) the backbone; add a small task head on the [CLS] token.

- **Task 1 for example — H→ZZ→4ℓ classification**
- **Task 2 — H→γγ classification**
- **Task 3 — >?**

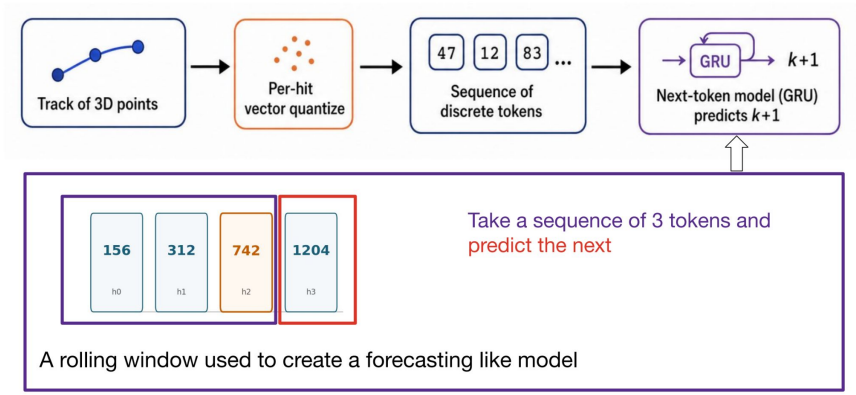
Same backbone, different heads. *Pre-train once (expensive); fine-tune cheaply per task.*

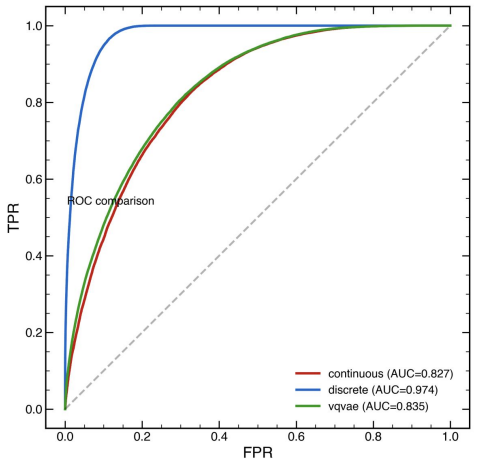
Looking forward: Low-Level & Other Experiments

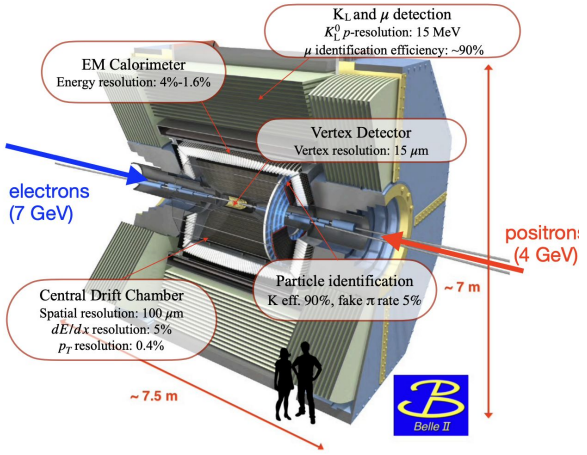
First tries at track reconstruction and token non-LHC data have started



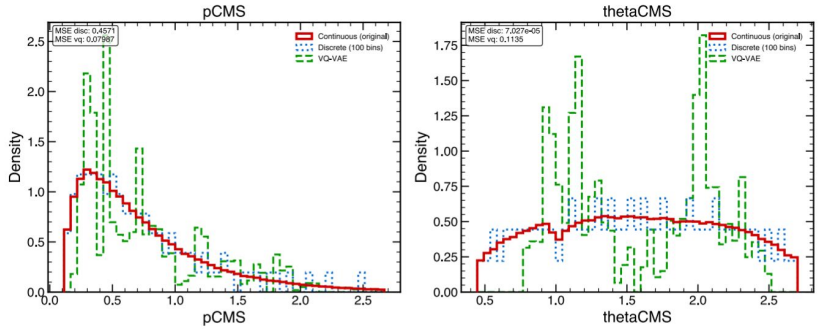
Punit Sharma







Riccardo Manfredi

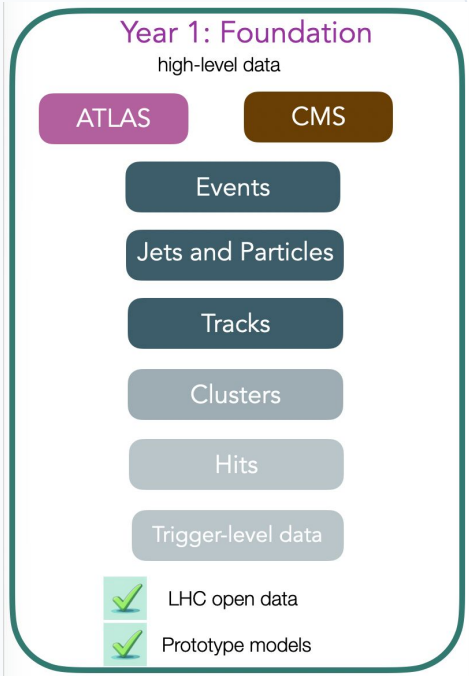


Project Timeline

Year 1

Foundations

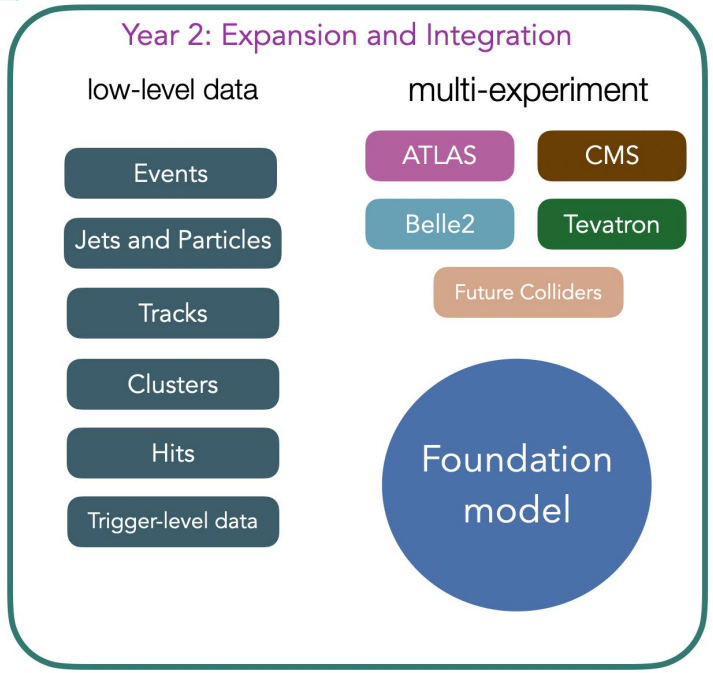
- AI-ready datasets from LHC Open Data
- Common data models, metadata, and tokenization schemes
- Assess AI-readiness and quantify tokenization impact
- **Demonstrate cross-experiment learning with prototype AI models**



Year 2

Expansion and Integration

- **Detector data:** hits, clusters, streaming compression
- **Beyond the LHC:** Tevatron (ppbar), Belle II (e+e-), future experiments
- **Scale up cross-experiment foundation model training**



Brookhaven National Laboratory · Fermilab · Berkeley Lab · SLAC National Accelerator Laboratory · Argonne National Laboratory

What's next & How to get involved?

- Next steps:
  - First paper out on event-level tokenization
  - Establish Treasure workflow on AmSC - use the opportunity to setup all the technical pieces
  - Provide CERN open data and tokenized through ModCon
  - Create a context-aware event-level tokenization with HepTokens
  - Focus on low-level information tokenization - work with experiments to publish low-level information
- **Happy to collaborate!** - reach out !
  - 2nd workshop is planned for later year
  - Depending on how many genesis proposals/interested people are there, setup a common meeting for collaboration
- **Aim to bring together Phase 1 + collaborators for a combined submission for Phase 2 in december**

backup

Hierarchical Feature-Group Tokenization

Multiple sub-tokens per object, aggregated by inner transformer

Flat (current)

Each object ⟶ 1 position in the sequence.
All features are projected through a single linear layer.
The transformer must figure out which features are kinematics, which are b-tagging, etc.

Jet ⟶ Linear([pT, η, φ, mass, ..., GN2_pu]) ⟶ hidden dim

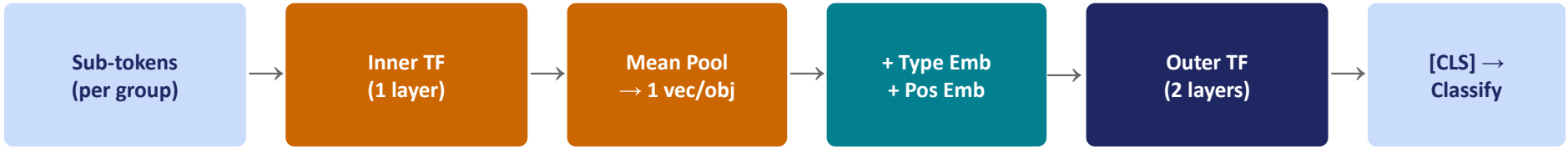
Hierarchical (feature groups)

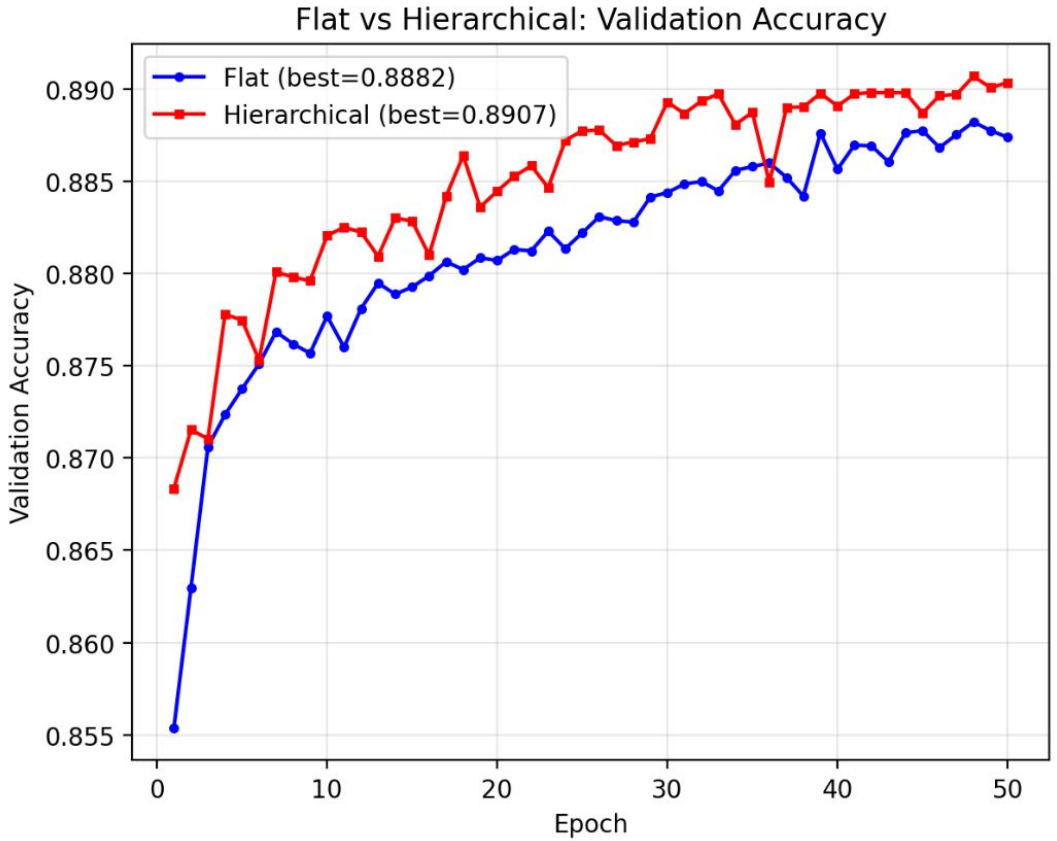
Each object ⟶ 2-3 sub-tokens by semantic group.
Inner transformer aggregates sub-tokens ⟶ 1 vector.
Outer event transformer reads aggregated objects.

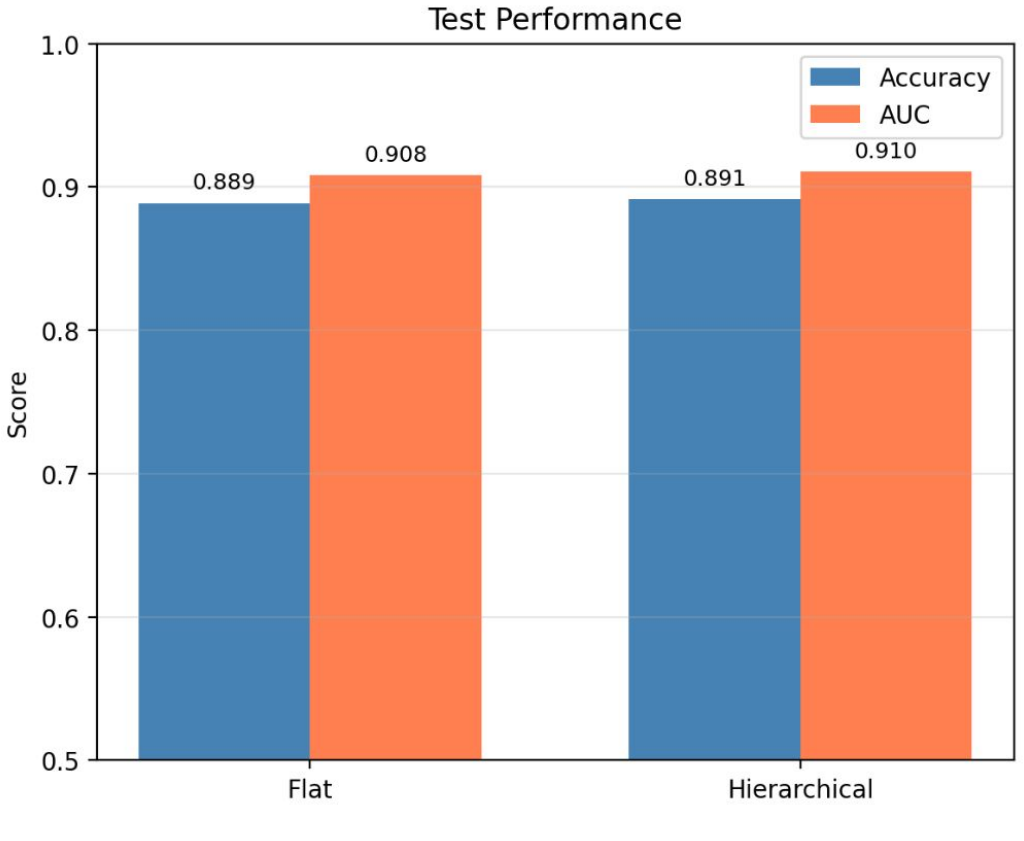
Jet ⟶ [kinematics: pT,η,φ,m] + [substructure: QG]
+ [b-tagging: DL1d, GN2]
⟶ inner attention ⟶ 1 jet vector ⟶ outer transformer

Feature groups per object:

Electron: [kinematics: pT,η,φ] + [ID: LHLoose/Med/Tight] Muon: [kinematics: pT,η,φ] + [ID: quality]
Tau: [kinematics: pT,η,φ] + [ID: nTracks, RNN scores] Photon: [kinematics: pT,η,φ] + [ID: isLoose/Med/Tight]
Jet: [kinematics: pT,η,φ,m] + [substructure: QG vars] + [b-tagging: DL1d+GN2] Track: [kin] + [impact: d0,z0] + [quality: χ^2,nDoF]



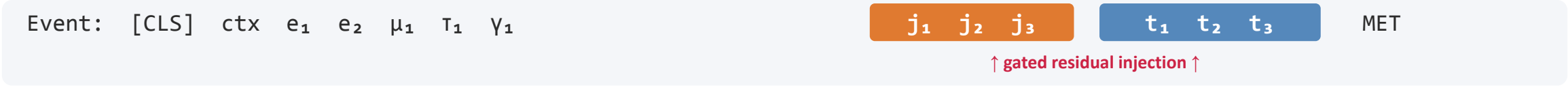




Brookhaven National Laboratory

Constituent + Hit Hierarchical Architecture

Sub-object info enriches jets (constituents) and tracks (hits) via gated residual



Jet Constituents (tracks/clusters inside jet)

Per jet: up to 50 constituent tracks/clusters
Each: pT, η, φ, charge, d0, z0

1. Project features → hidden dim
2. Constituent Transformer (1 layer)
3. Attention pooling → 1 vector per jet
4. Gated residual:
   jet_enriched = jet + σ(W·[jet // const]) × const

Gate learns how much constituent info to add.
When gate≈0: pure object-level. When gate≈1: constituent-driven.

Track Hits (detector measurements along track)

Per track: up to 30 detector hits
Each: η, φ, energy, layer, time

1. Project features → hidden dim
2. Hit Transformer (1 layer)
3. Attention pooling → 1 vector per track
4. Gated residual:
   track_enriched = track + σ(W·[track // hit]) × hit

Same architecture as jet constituents.
Hit patterns reveal particle type (e/μ/π),
track quality, and pile-up rejection.

Auto-detects constituent/hit data in H5. Falls back to object-only if absent. Works on current PHYSLITE data today.

Brookhaven National Laboratory

heptokens

A library for learning jet representations (Jeff Krupa, Michael Kagan (SLAC)

Tokenization model variants

MLP (no-context) · Transformer (in-context) — easily extendable to other models using LightningModule interface

Production ML stack

PyTorch Lightning · Hydra configs · Scalability with Snakemake · WandB logging · Pixi environments

Support for multi-stage pipelines

Ready for deployment into larger systems · Supports two-stage pipeline (decoupled tokenization+classification)

Tokenization performance

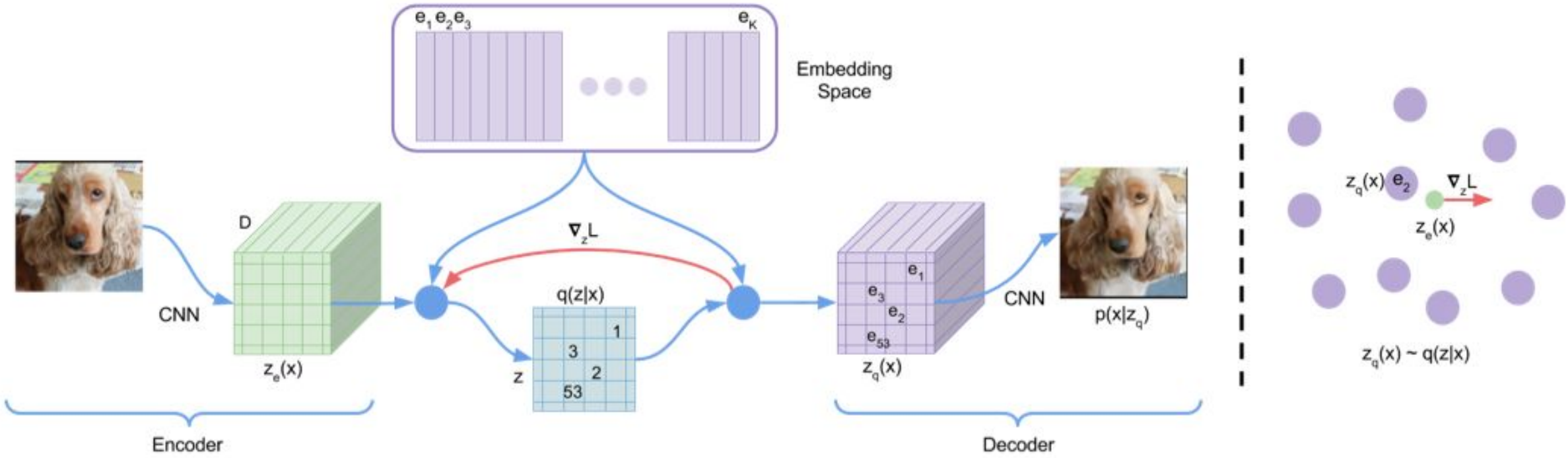
High-fidelity tokenization (<1 % jet resolution), high-performance classification on tokens (competitive with continuous features)

Quick start

```
from heptokens.data import
  SingleFileMapModule

from
heptokens.models.vq_vae
import
  LitVqVae

dm = SingleFileMapModule(
  data_path="mc-ttbar.h5",
  num_csts=40)

model = LitVqVae(
  codebook_size=512,
  num_quantizers=3)

trainer.fit(model, dm)
```

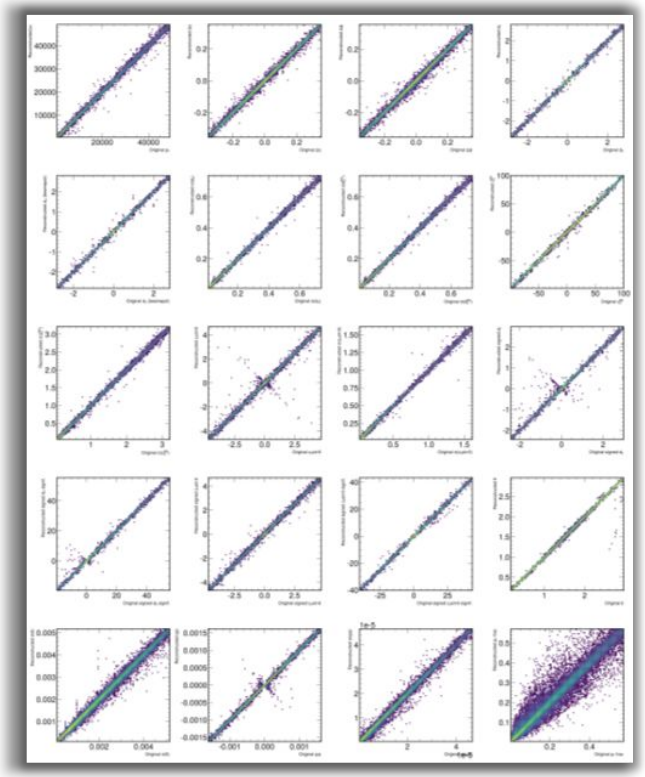
VQ-VAE

- Vector quantized variational autoencoder (VQ-VAE) compresses high-dimensional inputs into a learned codebook
  - Produces structured, compact tokens for use in downstream tasks

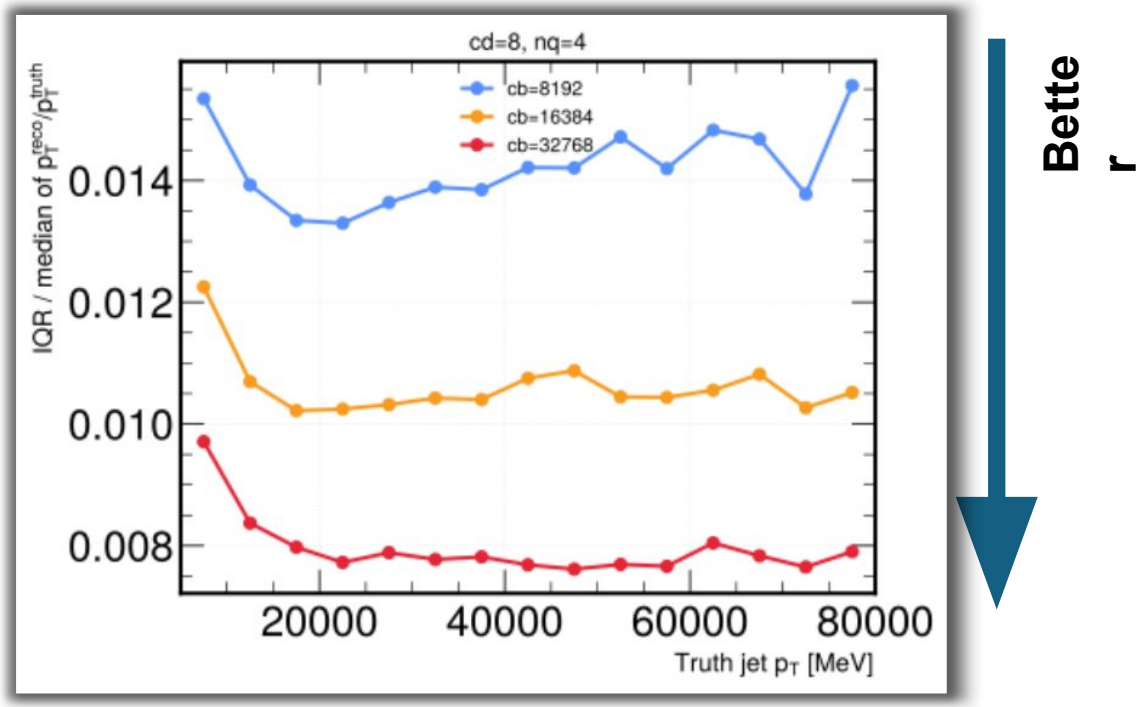


Tokenization

- JetSet = open ATLAS *tt* simulation with 200M jets for flavor tagging
  - 20 track features, event-level features
- Tokenizing tracks with variants of VQ-VAEs



Accurately reconstructs track features

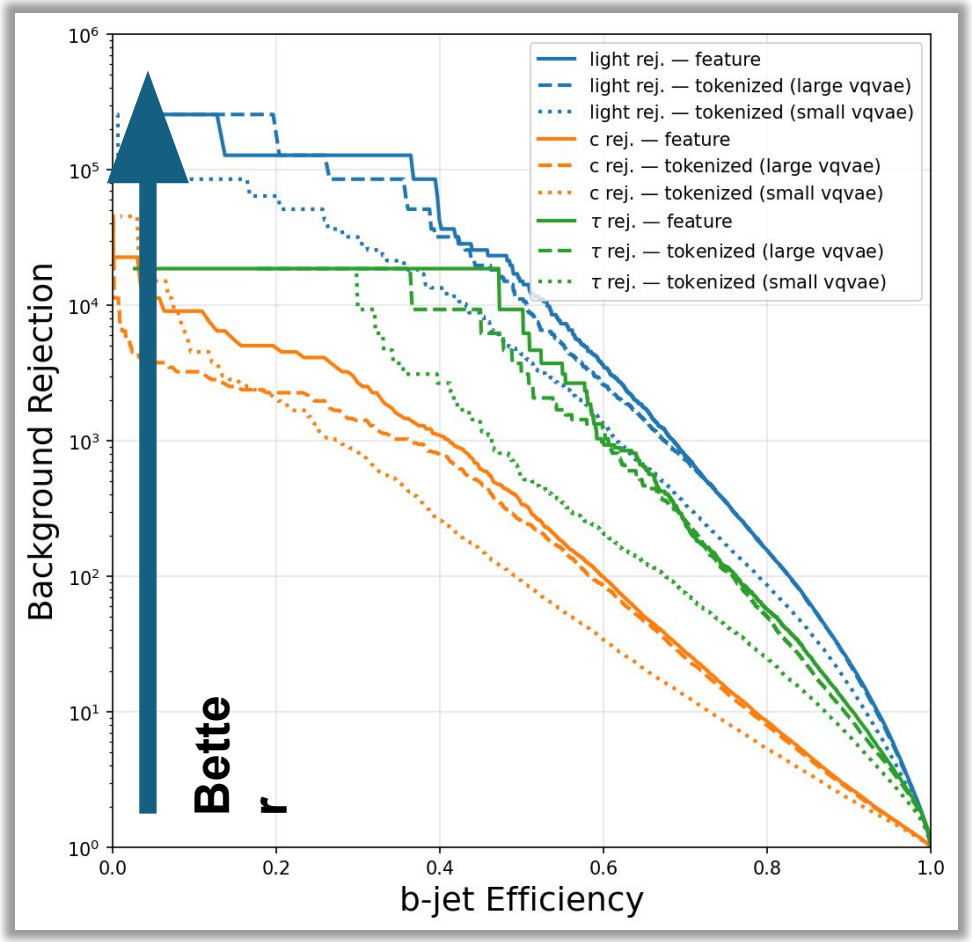


And achieves reconstructed jet resolution < 1% with large codebook size

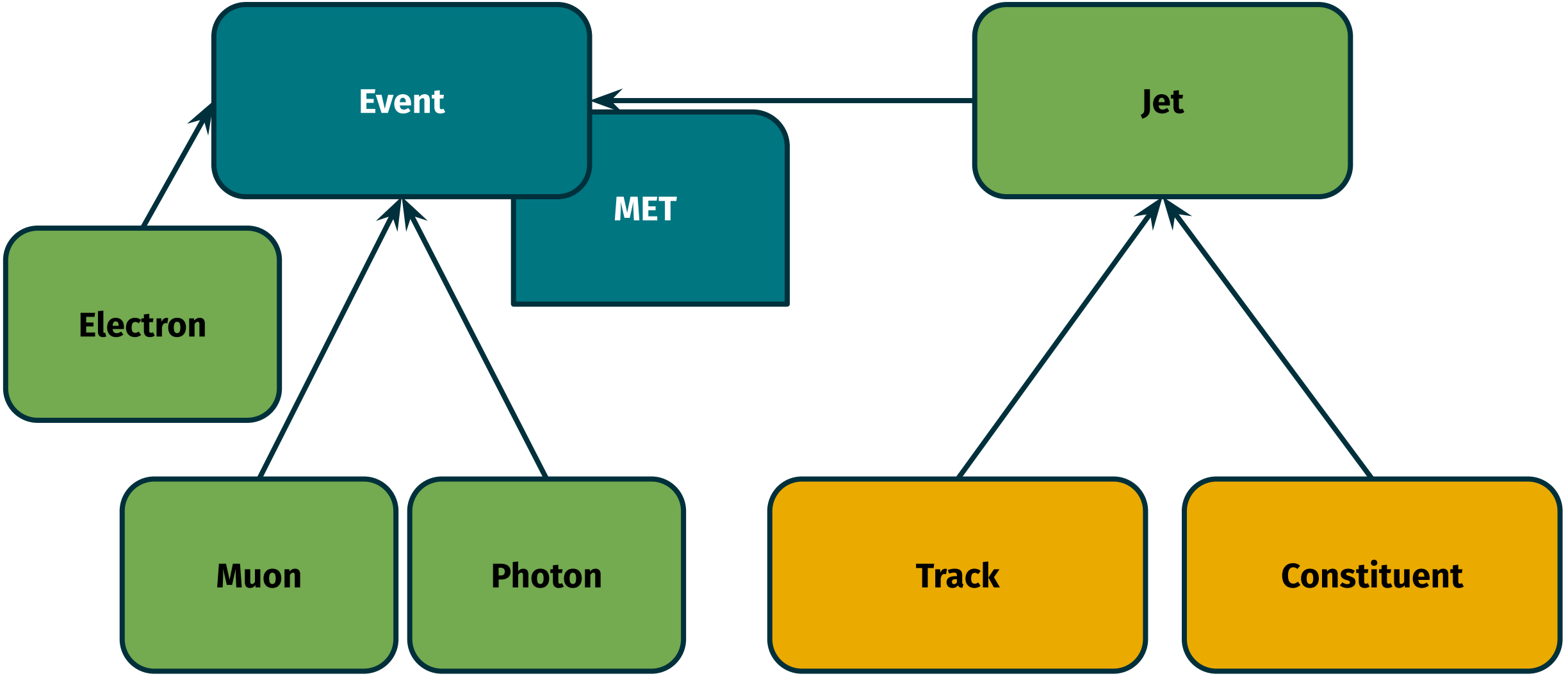
Flavor tagging (classification)

- Training flavor tagging algorithm on **large VQ-VAE** has close performance to **continuous features**

Solid = continuous inputs
Dashed = large VQVAE
Dotted = small VQVAE



Particle Cloud Data Format



Event

EventIndex **[Primary Key]**

Detector Index [Link to some other DB?]

Simulation tag

Run number

Event number

~~Primary Vertex x, y, z~~ Beam Position x, y, z

(Auxiliary Info ObjectID)

MET

MET_phi

Change to attach MET to event

Objects (Electron, Muon, Photon)

index	**[Primary Key]**
eventIndex	[Link to Event]
pt	
eta	
phi	
charge (electron and muon only)	
truth	

Jet (Large and Small *R*)

index	**[Primary Key]**
eventIndex	[Link to Event]
pt	
eta	
phi	
m	
truth / flavor (large-R / small-R)	

Constituents (Particle Flow)

index [Primary Key]

jetIndex

eventIndex

pt

eta

phi

m

- *For now, R = 0.4 only*
- *Read both charged and neutral constituents from PHYSLITE and follow jet → constituent element links*
- *NB: ElementLinks are not pleasant to use from Uproot*

Tracks

index **[Primary Key]**

eventIndex

jetIndex

q_p

theta

phi

d0

z0

Can later calculate a pt, eta, phi representation

File Format

- Use of indices as cross-references allows tables to be flat
- Enables storage as Parquet files with metadata such as column names and statistics
- Right software (e.g. Polars) can group by the indices to generate n-dimensional HDF5 tables