The RDF $\boldsymbol{t\bar{t}}$-analysis implementation

Analysis Grand Challenge

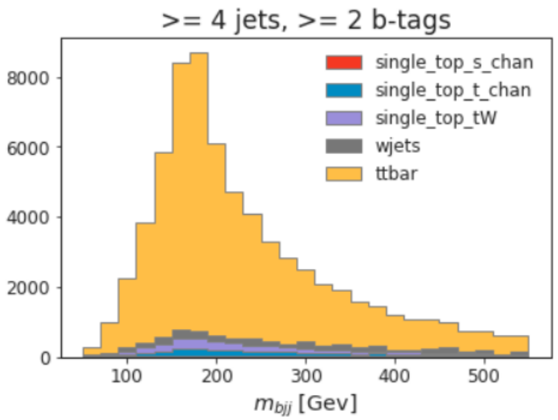
IRIS-HEP Fellow: *Andrii Falko*,

Taras Shevchenko National University of Kyiv

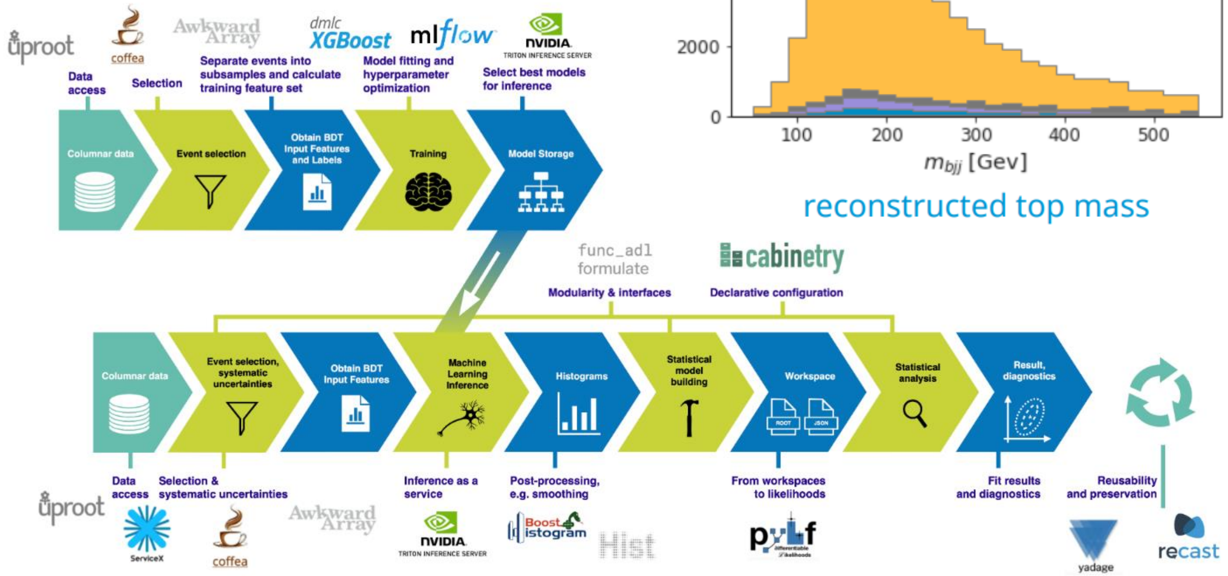
Mentors: *Enrico Guiraud*, *Alexander Held*

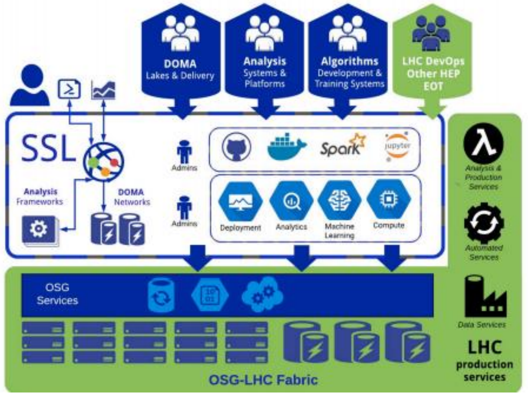
Introduction

- Analysis Grand Challenge
  - Developing and testing workflows envisioned for the HL-LHC
  - Showing the performance and scalability
- Specification of a physics analysis
  - $t\bar{t}$ cross-section measurement
  - Top-quark mass reconstruction
    - via simplified conventional approaches
    - via machine learning
  - 2015 CMS Open Data
  - Handling systematic variations
- IRIS-HEP's reference implementation
  - iris-hep/analysis-grand-challenge
- RDataFrame & AGC project
  - root-project/analysis-grand-challenge



reconstructed top mass



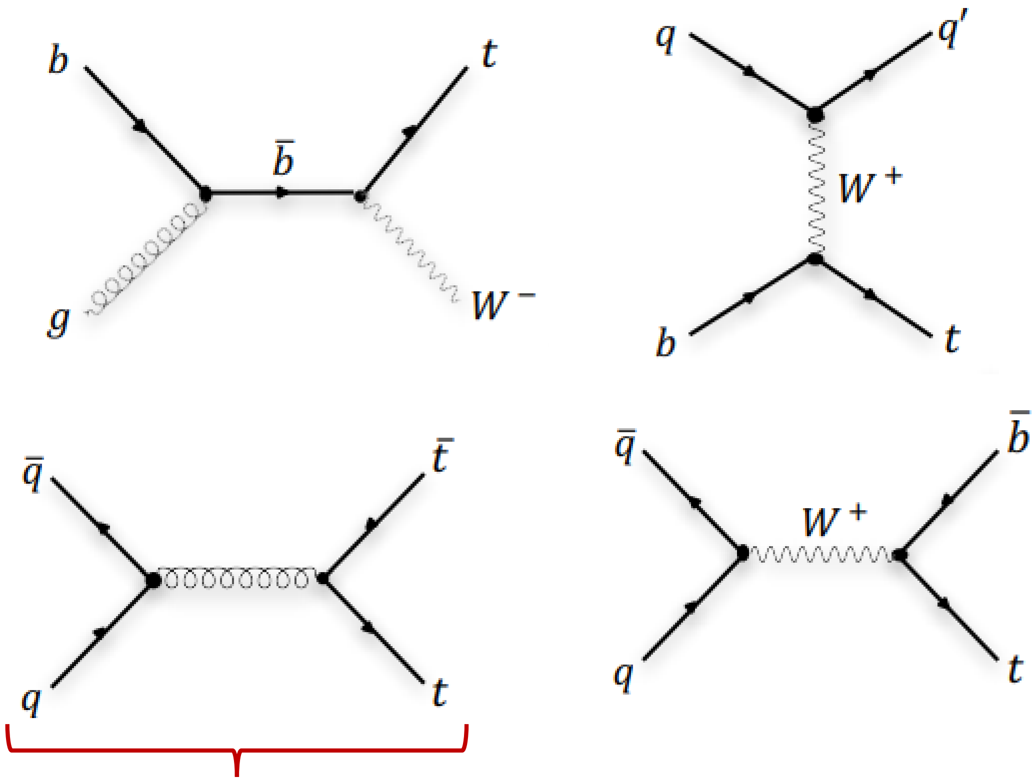


RDF & AGC project

- Implementation of Analysis Grand Challenge with ROOT's modern analysis interface
- RDataFrame
  - flexibility to express virtually any HEP analysis
  - allows execution of any C++ code
  - seamless scaling out to computing clusters
- Project's goals for summer
  - Update RDataFrame implementation to AGC v1
  - Update RDataFrame implementation to AGC v2

$t\bar{t}$-analysis specification

1. Analysis task: $\boldsymbol{t\bar{t}}$ cross-section measurement

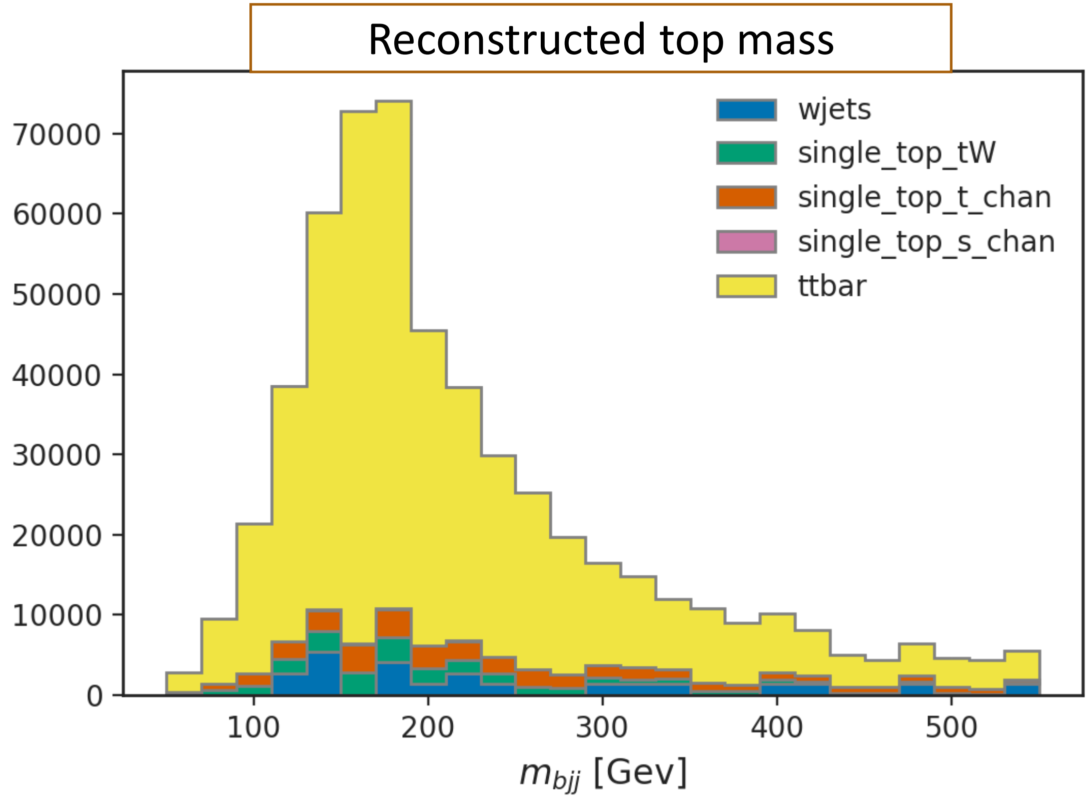


2. Input dataset: [2015 CMS Open Data]

3. Event selection
   - Semi-leptonic decay channel
   - 2 jets 2 b-jets 1 lepton

4. Account for weights

5. Calculation of observables



6. Systematic uncertainties
   - Jet kinematics variations
   - Event-weights variations

7. Machine Learning Component
   - Jet-parton assignment
   - BDT

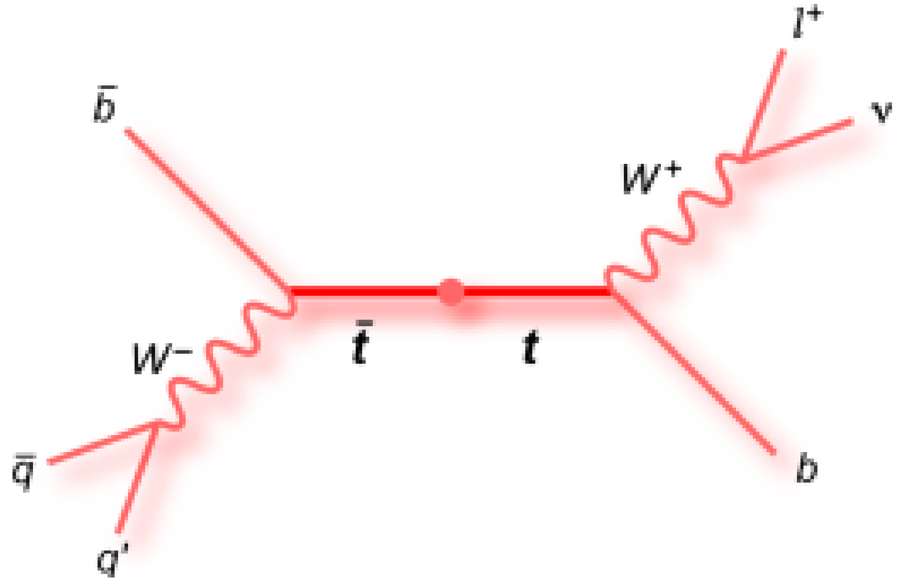
Moving to the latest AGC version

V	Data schema	Selection cuts			Machine learning
		jets	$\text{jets}_{\text{b-tagged}}$	leptons	
0	POET	• P_T > 25 GeV	• P_T > 25 GeV • b-tag > 0,5	• P_T > 25 GeV	None
1	NanoAOD				
2	NanoAOD	• P_T > 30 GeV • $\|\eta\|$ < 2,4 • jetId == 6	• P_T > 30 GeV • $\|\eta\|$ < 2,4 • b-tag > 0,5 • jetId == 6	• P_T > 30 GeV • $\|\eta\|$ < 2,1 • sip3d < 4 • cutBased_e==4 • tightId_μ • $\text{pfRelIso04\_all}_\mu$ < 0.15	BDT to predict jet-parton assignment in $t\bar{t}$ events

- Switched POET new input data schema (NanoAOD)
  - tag v1
- Added new selection cuts of version 2
  - root-project/analysis-grand-challenge
- Added boosted decision tree inference for the jet-parton assignment
  - andriiknu/agc-root/tree/add_inference

Top mass reconstruction

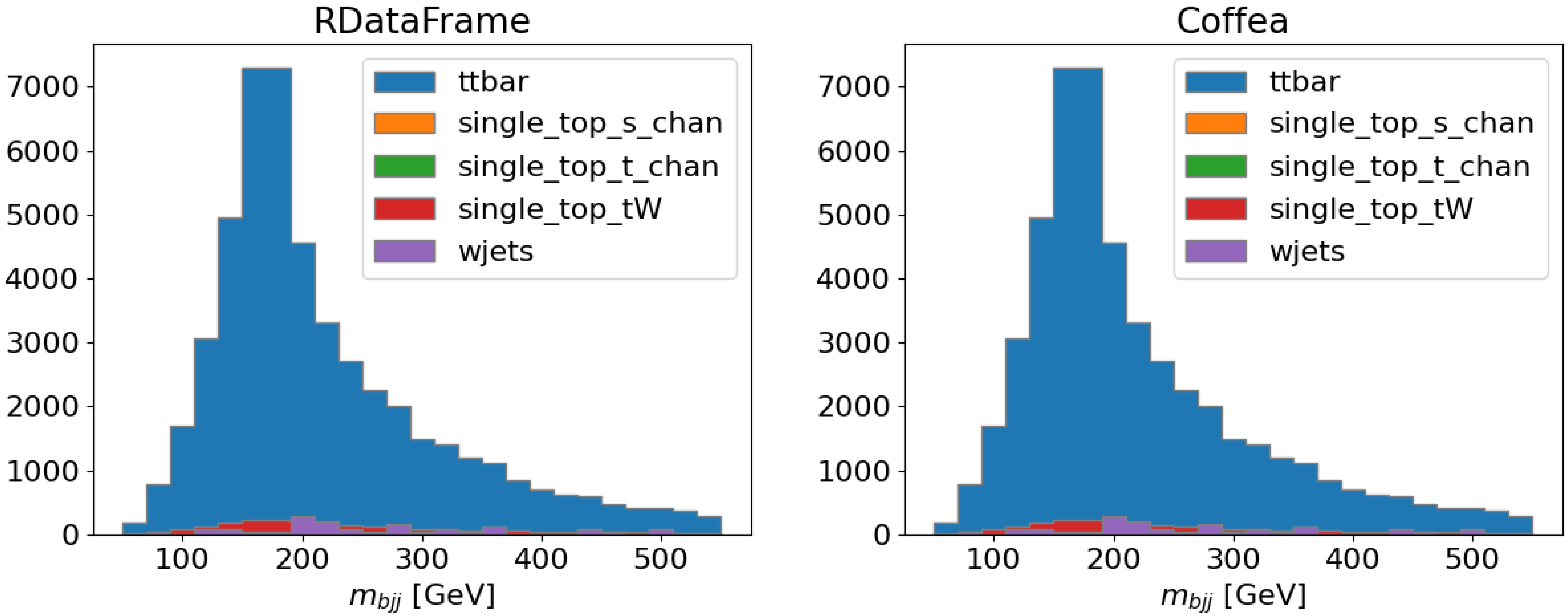
- Semi-leptonic decay channel schema
- The top mass is a mass of three jets $(\overline{b}, \overline{q}, q)$
- At least four jets in every event
- Choose three jet combination which is the product of top decay



Trijet combination

- Consider all trijet combinations
- Assign them some properties (*features*)
  - Is there at least 1 b-tagged jet? (*bool*)
  - Total transverse momentum
- Set some criteria how to conclude from these properties (features) which trijet is the best candidate for being top decay product
  - Choose the combination with the largest combined transverse momentum

Histograms validation against reference



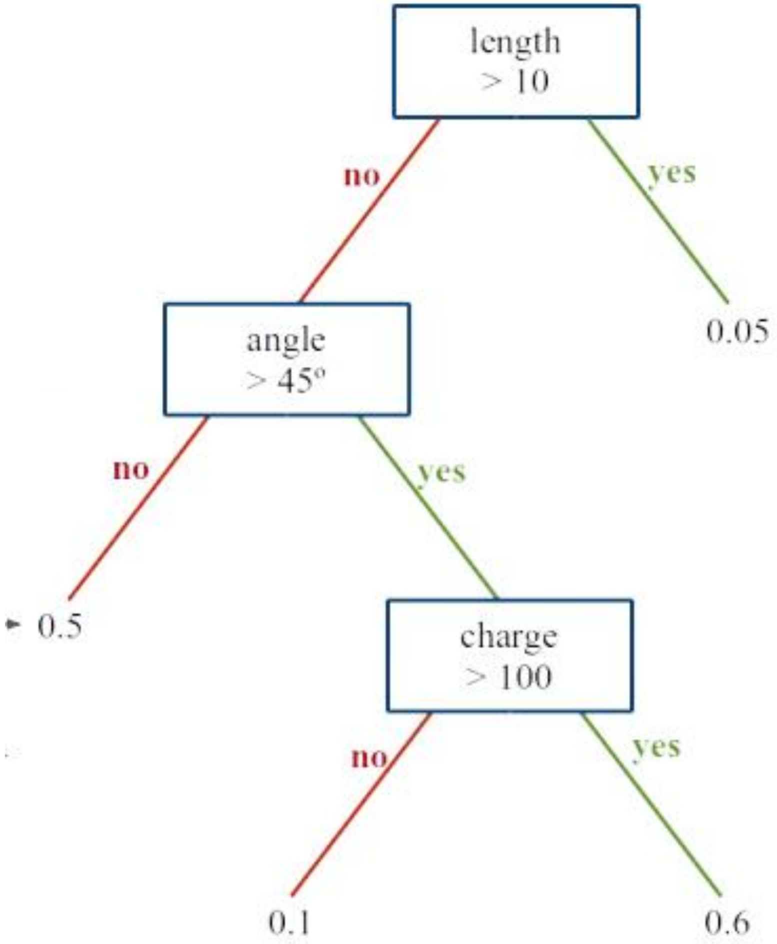
All of 122 histograms are in perfect matching bin-by-bin alignment

Machine learning component: why is needed?

- The way explained previously to get a trijet combination is **too rough**
  - Why only two properties (features)? (max P_T and *b*-tag)
  - The criteria were not very reliable
- Machine learning allows us to create more complicated schemas to choose the best option
- We can construct many more properties for jet combinations to express more physics
- By using ML we delegate the work of making decisions to the computer

Boosted decision trees

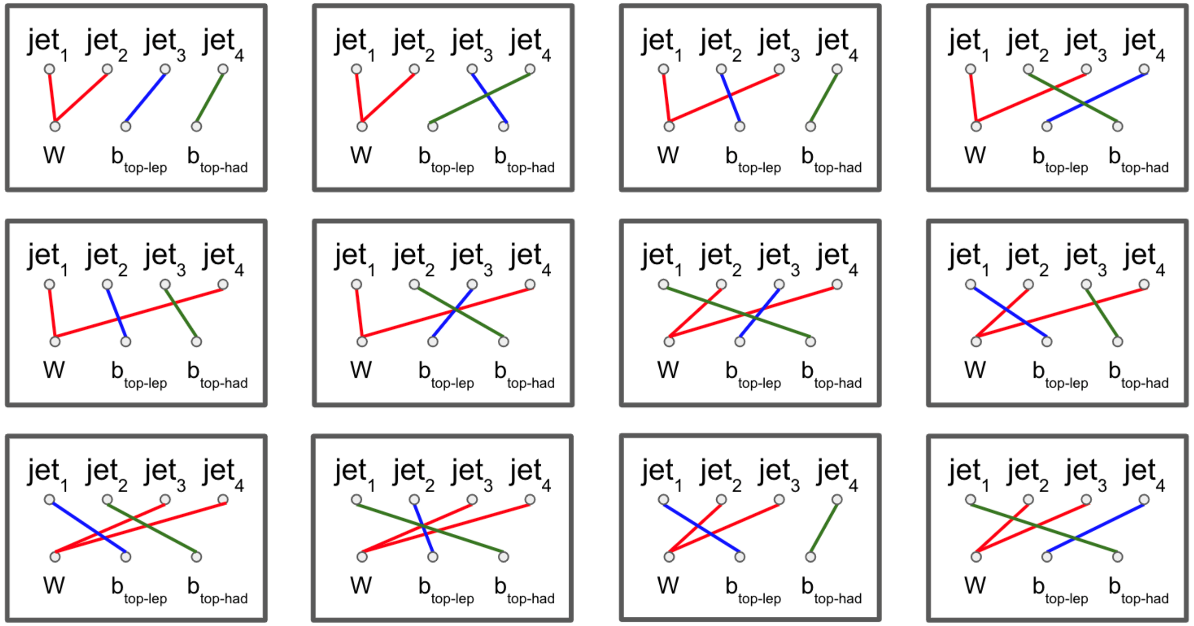
- Decisions are made via decision trees
- A decision tree takes a set of input features and splits input data recursively based on those features
- Allows estimate probability to be a candidate of top decay by much more complicated logic: chain of decision
- Increasing the depth of the tree allows for expressing the logic of arbitrary complexity
- The magic occurs because we can:
    - create a forest of decision trees
    - delegate to the computer to draw the content of each tree by fitting the model with data (training)



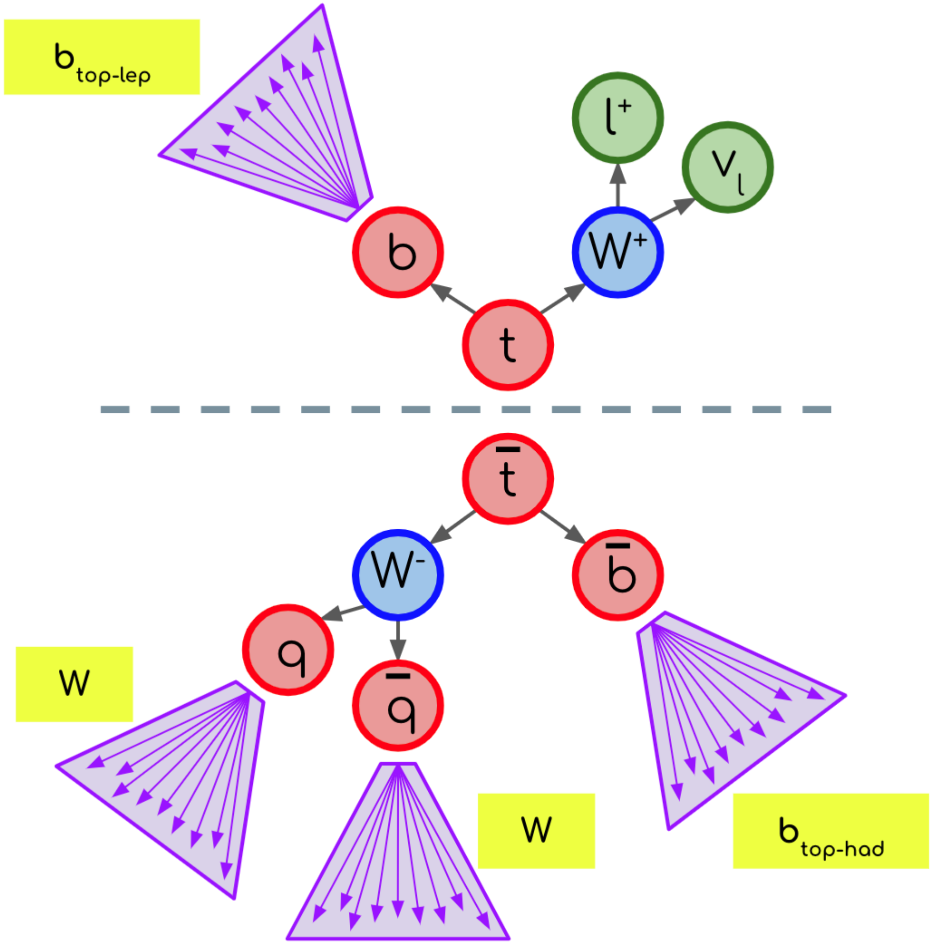
The image taken [there]

ML implementation

- Imagine event with four jets
- Assign the label to each jet:
  - $b_{top-lep}$, W, or $b_{top-had}$
- Build all unique permutations of 4-jets

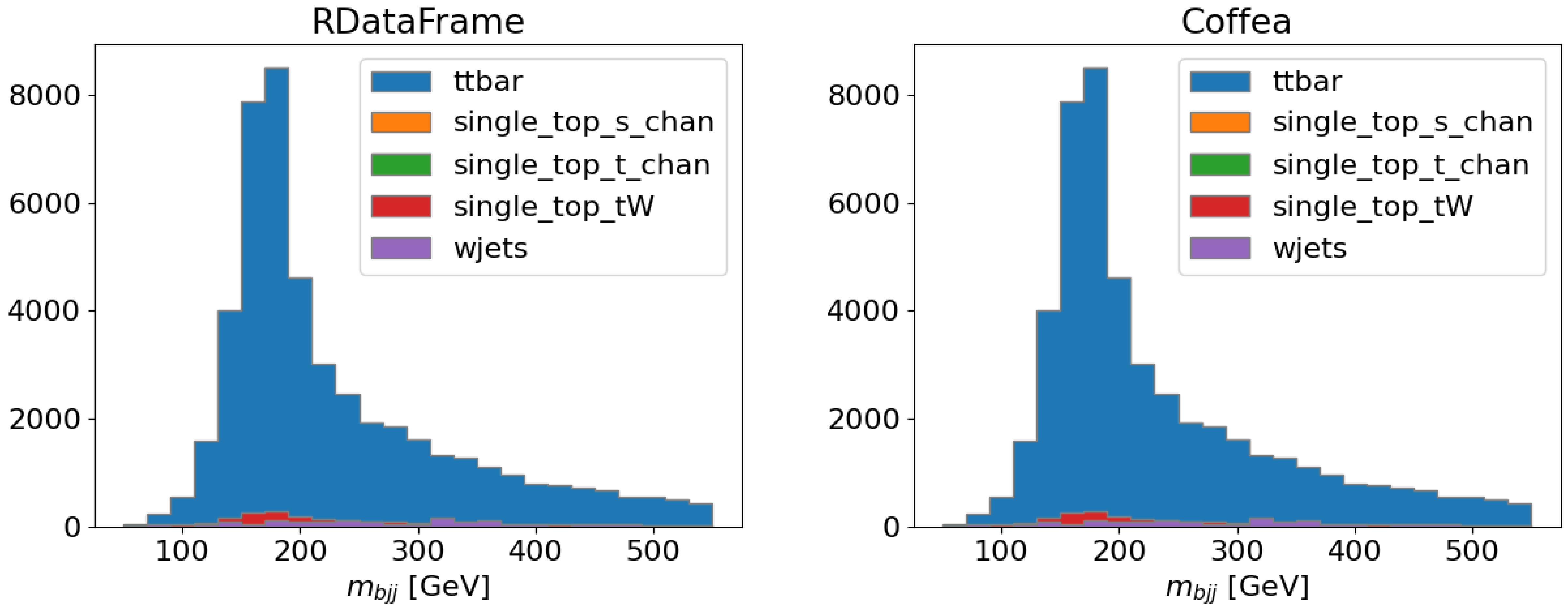


- Calculate 20 features for every permutation (not only 2)
- BDT inference gives the best candidate



see full details at official AGC documentation

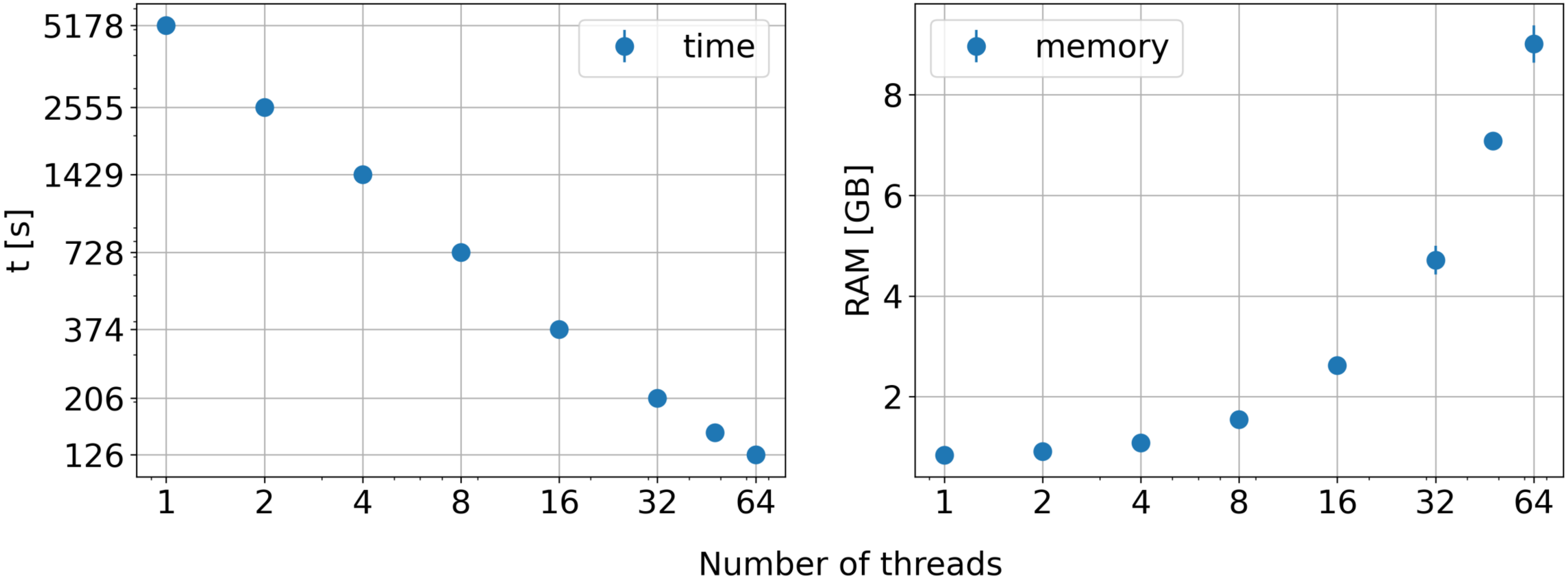
Validation of ML output histograms



- 1211 out of a total of 1220 histograms are perfectly matched bin-by-bin**!!**
- 9 of them are within tolerance of 1%**!**

Benchmarks

- Dataset of total size 1.78 TB / 940160174 events
- The time scale is logarithmic



- Speedup up to 41x
- Total throughput for 64 cores 7.4M events/s

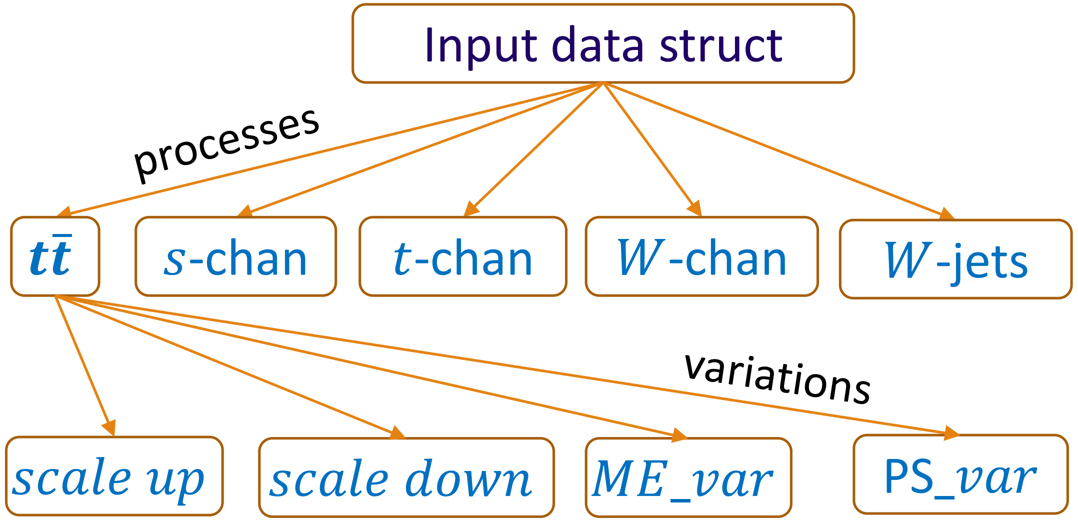
The main achievements

- Switched ROOT's RDataFrame implementation of AGC to version 1.
    - Used new input data schema (NanoAOD)
- Switched ROOT's RDataFrame implementation of AGC to version 2.
    - Defined new selection cuts (slide 5)
    - Added calculation variables which are input features for machine learning inference
    - Added implementation of ML inference
    - ROOT's RDataFrame integrated with the C++ FastForest library
- Validation
    - We are in good agreement with IRIS-HEP reference implementation
    - The performance benchmarks show good scalability

Thanks a lot for your attention!
Questions?

Backup

Input data

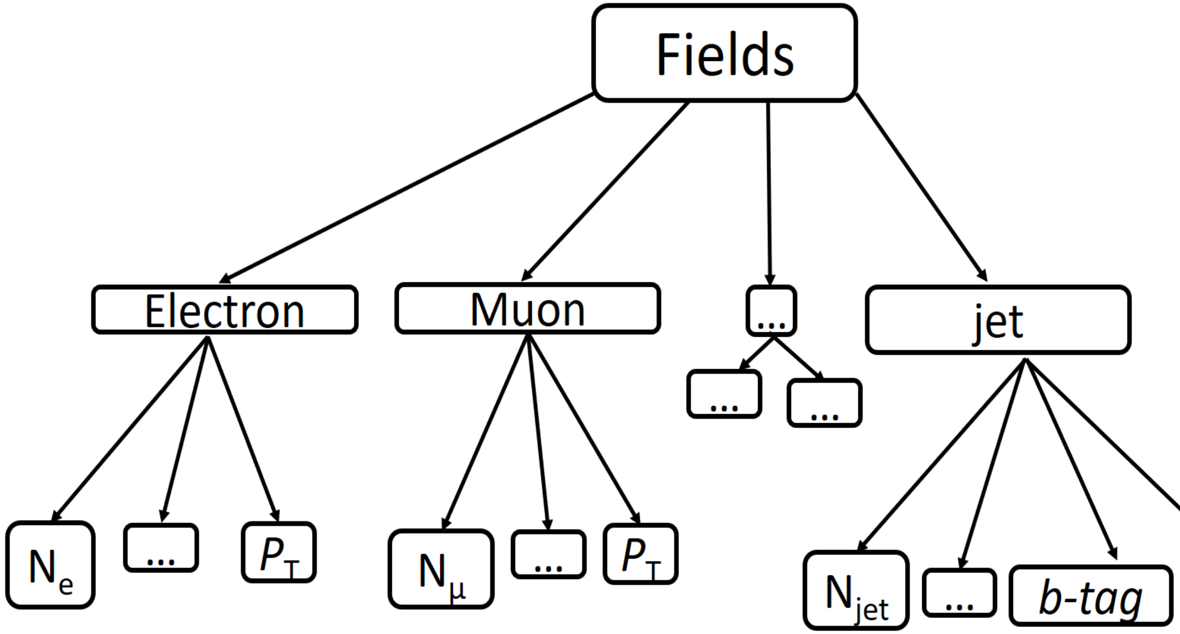


Datasamples

- 2015 CMS Open Data
- 9 subsets of root files produced in MC simulation
- 5 interaction channels → 5 processes involved
- 4 kinds of variations → given as 4 additional sets

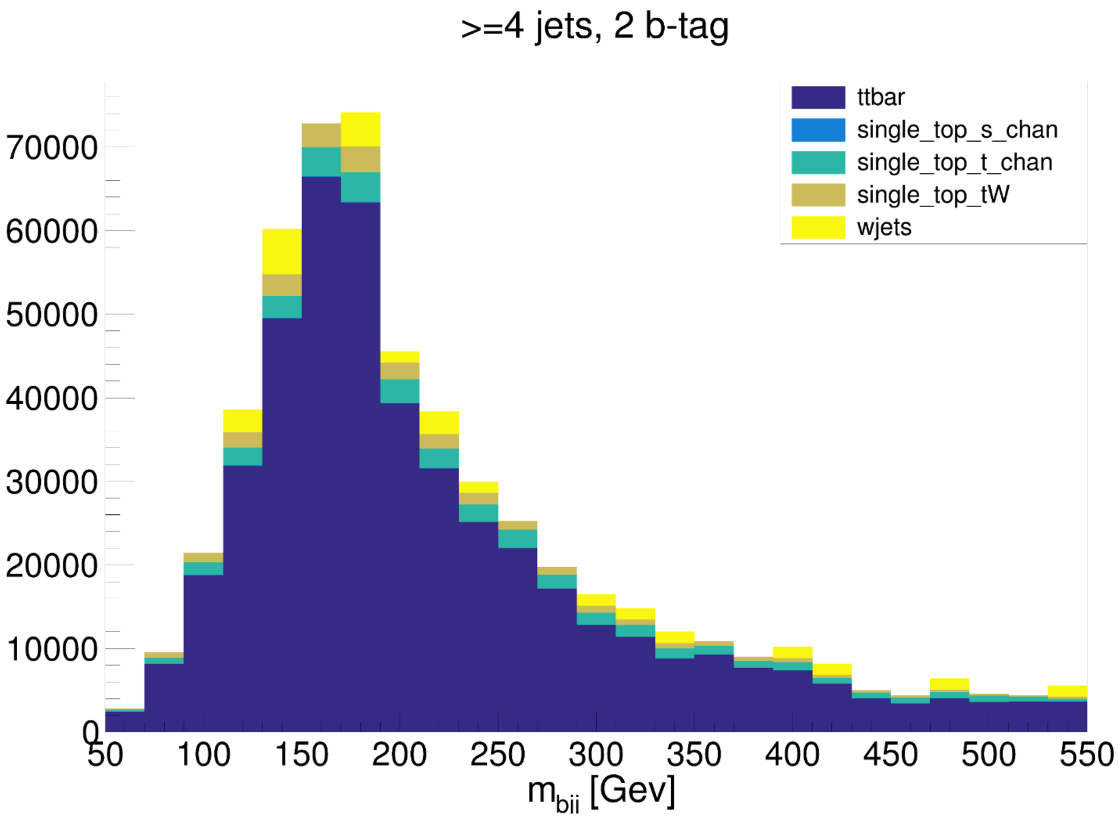
Data schemas

- Physics Objects Extractor Tool (POET)
- NanoAOD



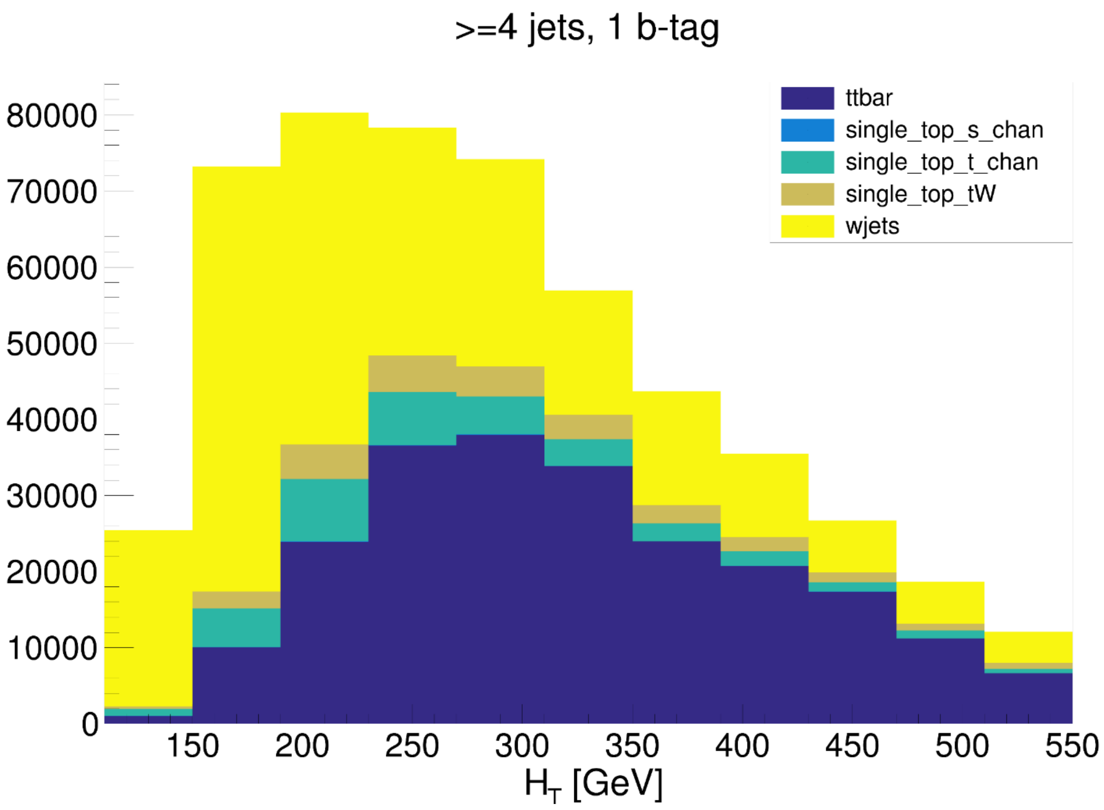
Observables

Signal region (*reconstructed **top mass***):



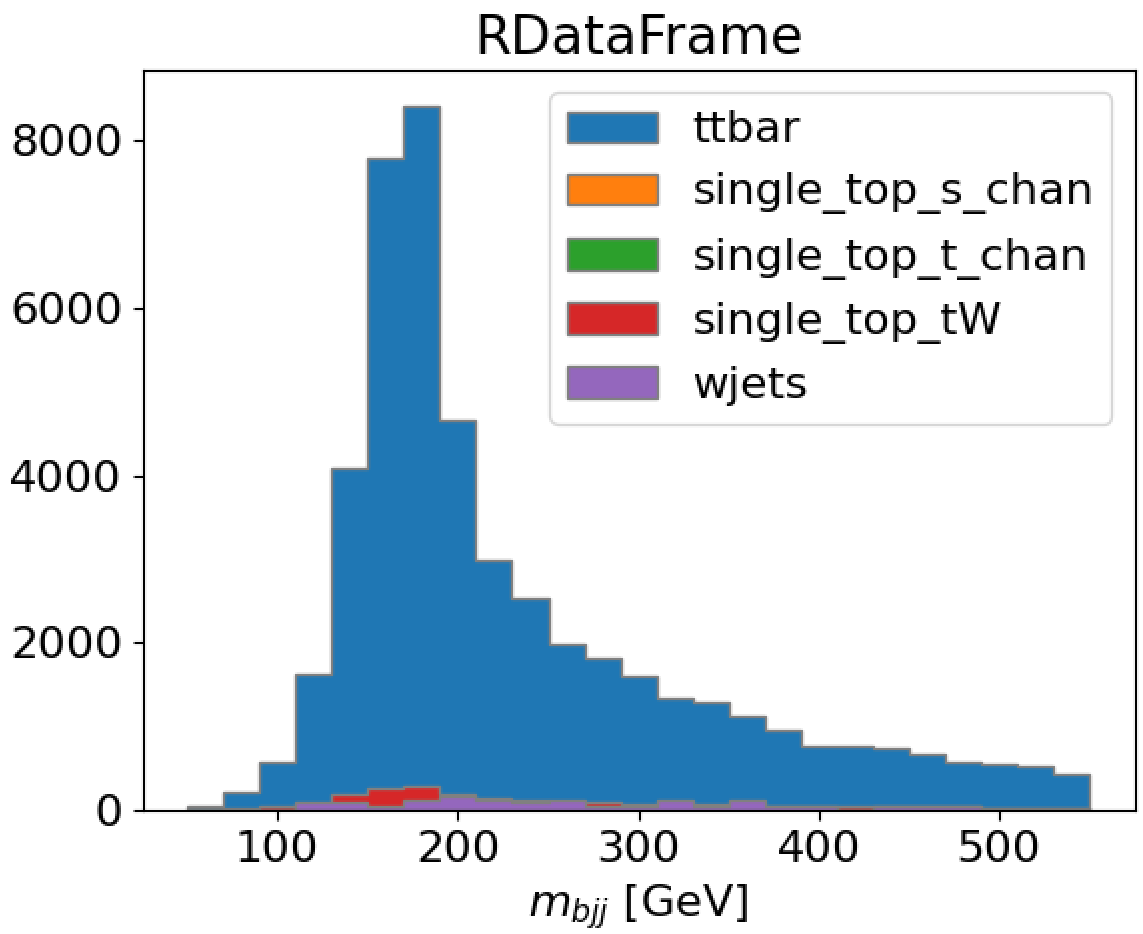
Top-quark mass peak plot

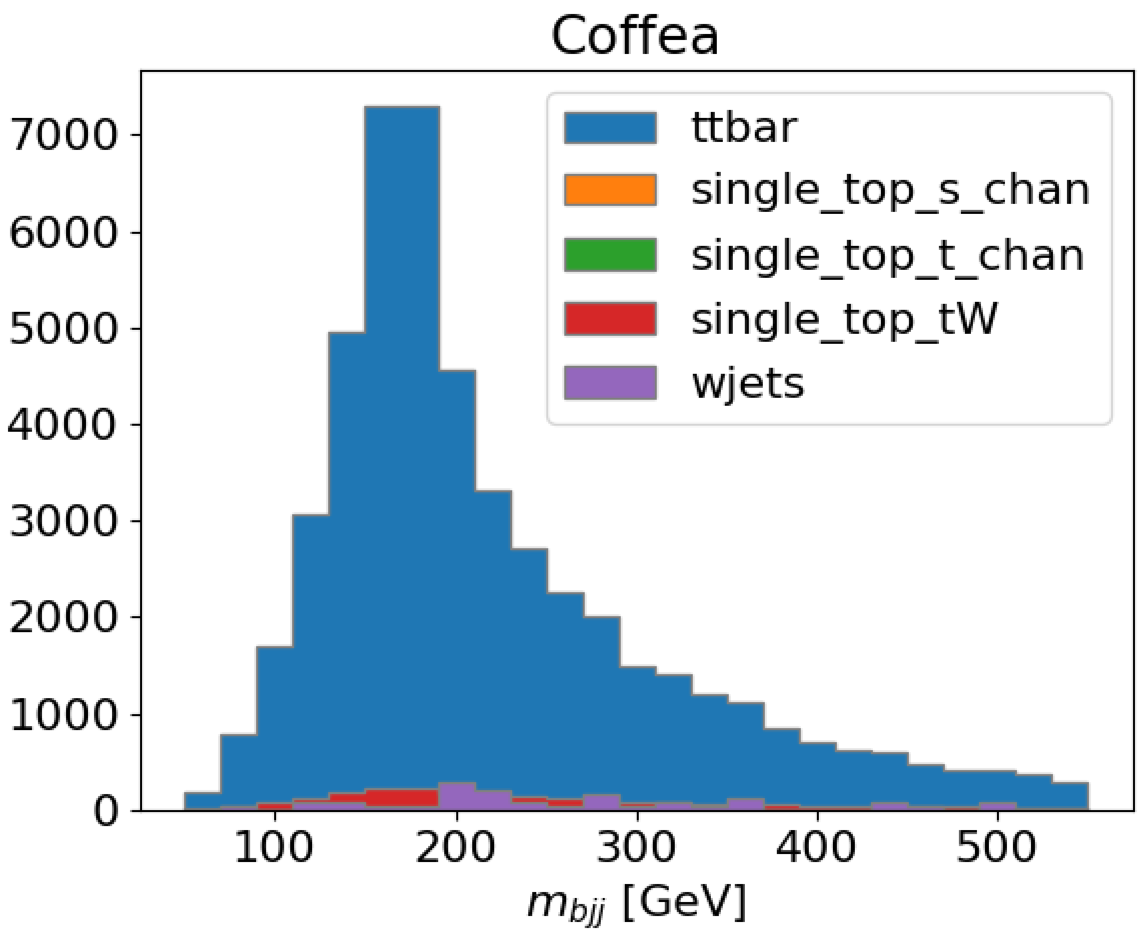
Control region (sum of the p_T of all jets in each event):



Scalar sum of transverse momenta plot

Comparison of ML output histograms





- built-in another way to generate permutations