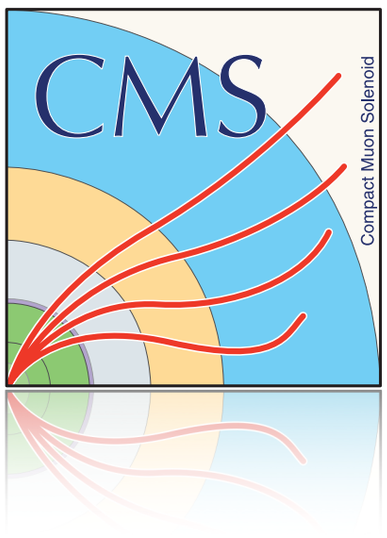
DeepAK8: Multi-class Boosted Jet Tagger

Owen Colegrove[1], Loukas Gouskos[1], Huang Huang[2], Joseph Incandela[1], Jan Kieseler[3], Qiang Li[2], Matthias Mozer[4], Huilin Qu[1], Paris Sphicas[3,5,6], Markus Stoye[3,5], Mauro Verzetti[3]

1) UCSB, 2) Peking University, 3) CERN, 4) KIT, 5) aMVA4NewPhysics, 6) Athens



CMS Heavy Flavour Tagging Workshop
April 13, 2018

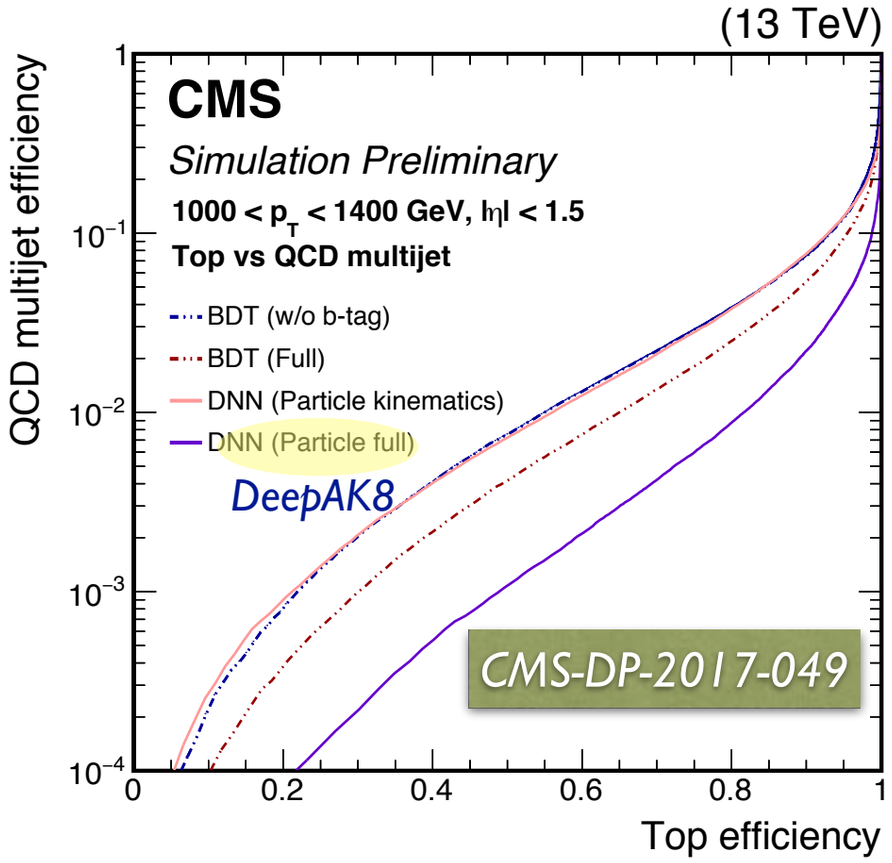
UCSB

Introduction

- Reconstruction and identification of boosted heavy particles (top/W/Z/Higgs) can provide powerful handles for both new physics searches and SM measurements at the LHC
- Very challenging task
  - large background from QCD jets, difficult to distinguish
- One of the hottest topics at the LHC, with rapid advancements
  - developments in jet substructure technique
    - exploiting differences in the energy/angular correlation of particles inside the jet
    - powerful tools for separating heavy particle decays from QCD jets
  - dedicated flavour tagging tools for boosted jets
    - exploiting unique flavour signatures of heavy particle decays (e.g., top->bW, H->bb, etc.)
    - complementary to substructure techniques and significantly improves the performance
  - growing interest in advanced machine learning techniques
    - aiming at further extending the frontier of performance

DeepAK8: Overview

- DeepAK8 tagger:
  - multi-class classifier for top, W, Z, Higgs and QCD jets, based on standard anti-k_T R=0.8 (AK8) jets
  - deep neural network (DNN) using low-level inputs (PF candidates, secondary vertices) directly
  - exploits substructure and flavour information simultaneously
  - improves the performance significantly



Inputs

Substructure

Particles

- Up to 100 PF candidates(*)
- Sorted in descending p_T order
- Uses basic kinematics, Puppi weights, etc.
  - and properties (quality, covariance, displacement, etc.) of the associated tracks for the charged particles

Flavour

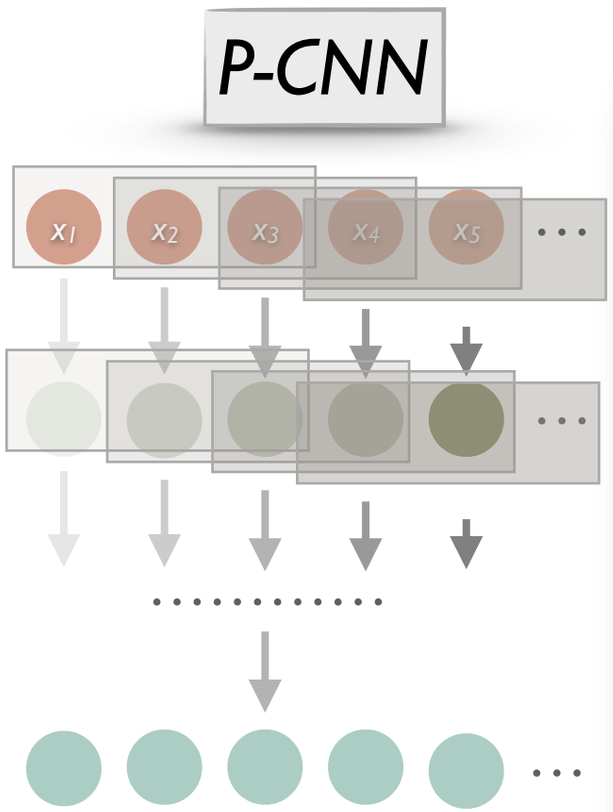
Secondary vertices

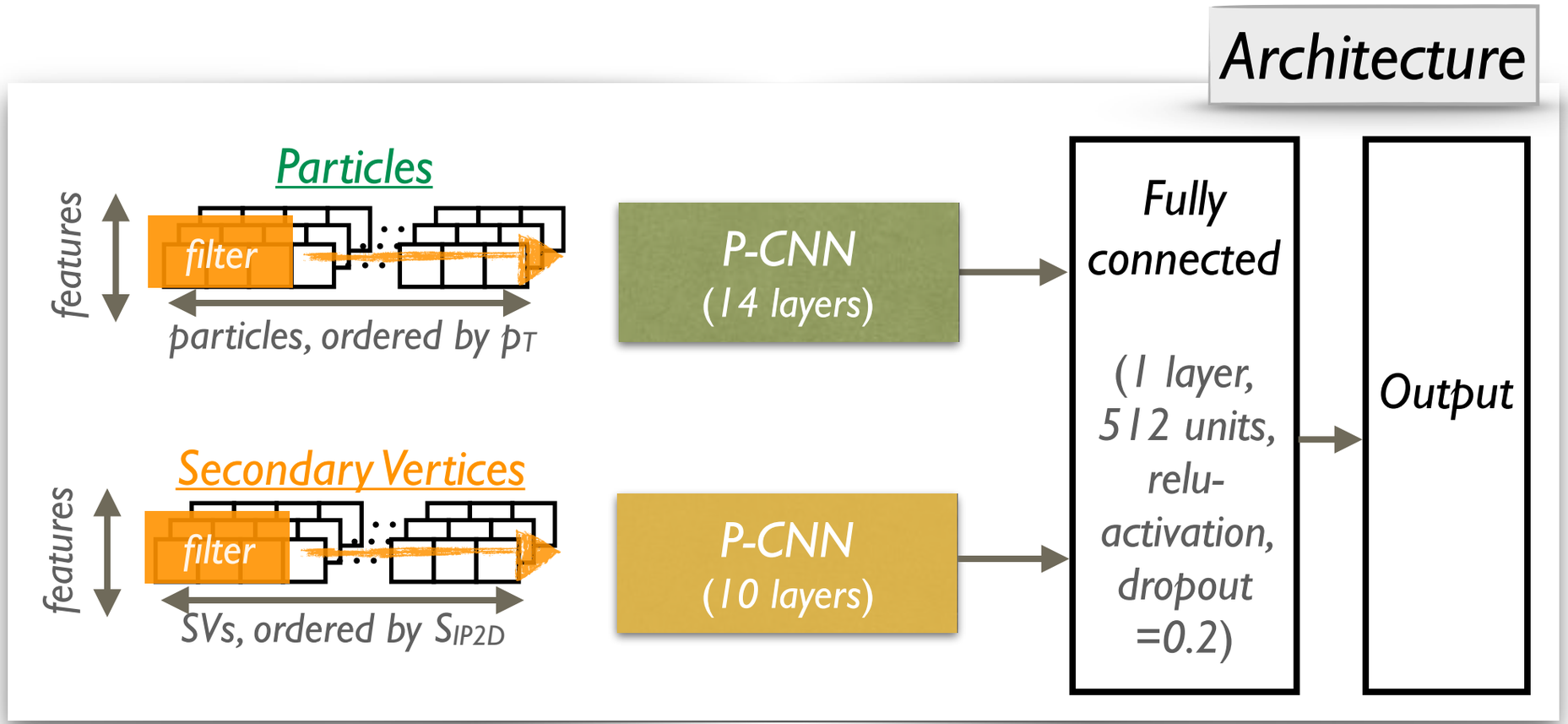
- Up to 5 SVs(*) (inside jet cone)
- Sorted in descending S_{IP2D} order
- Uses SV kinematics and properties (quality, displacement, etc.)

(*) *Number chosen to include all candidates for ≥ 90% of the events*

DEEPAK8: ARCHITECTURE

- Particles and SVs are first processed separately by two P-CNNs to extract useful features, and then combined with a fully connected network to produce the final prediction
  - P-CNN: particle-level convolutional neural network
    - essentially a one-dimensional CNN over a sequence of particles
    - progressively processing nearby particles to transform and aggregate information
    - stacking many layers to better get the correlation between particles





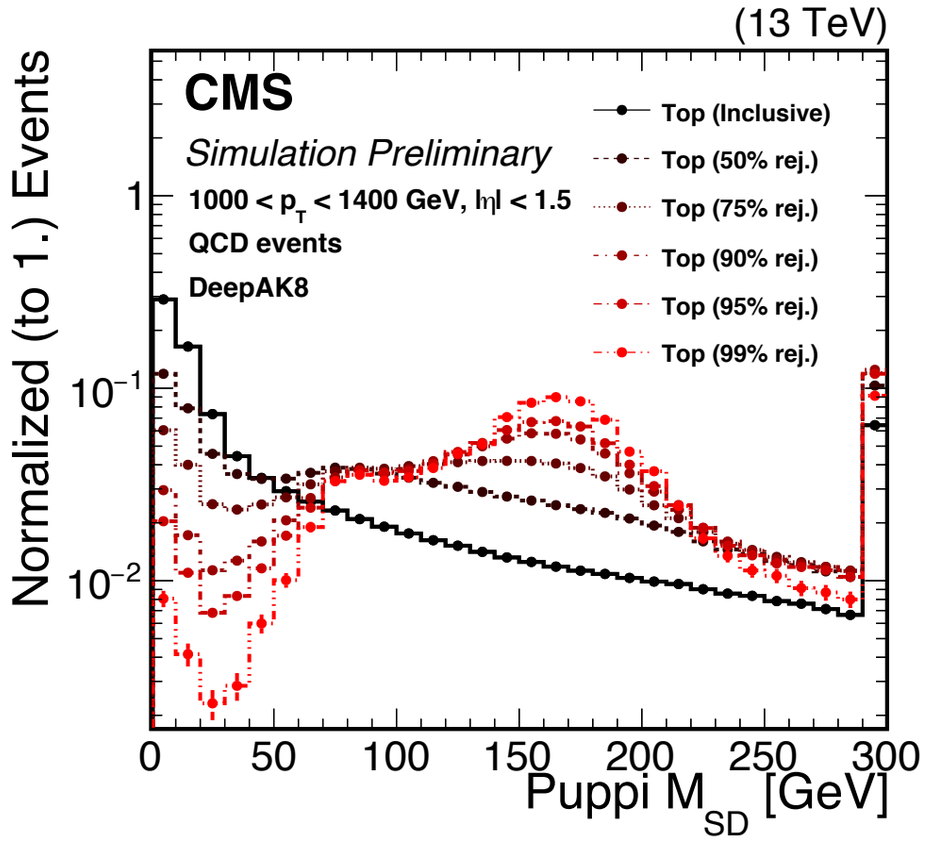
DEEPAK8: LABELS

- DeepAK8 features a fine-grained label definition
  - major categories defined for Higgs, top, W, Z by requiring the quarks from the heavy particle decay to be contained within the jet cone [i.e., $\Delta R(\text{jet}, q) < 0.8$]
    - minor categories are further defined based on the flavour content of the matched quarks
  - for QCD jets, the minor categories are defined based on the number of b- or c-hadrons inside the jet
  - full list shown in the table, with decreasing priority from top to bottom
- This makes DeepAK8 a versatile tagger
  - prediction scores are be thought as "probabilities" and can be easily transformed/aggregated for various needs, without the need of a dedicated re-training

Category	Label
Higgs	H (bb)
	H (cc)
	H (VV*→qqqq)
Top	top (bcq)
	top (bqq)
	top (bc)
	top (bq)
W	W (cq)
	W (qq)
Z	Z (bb)
	Z (cc)
	Z (qq)
QCD	QCD (bb)
	QCD (cc)
	QCD (b)
	QCD (c)
	QCD (others)

Correlation with the Jet Mass

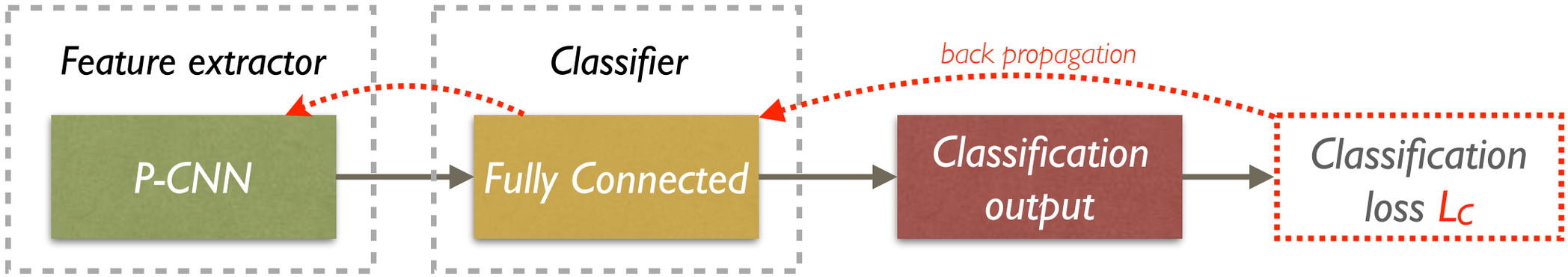
- The base version of DeepAK8 shows significantly improved performance
  - but also strong mass sculpting
- Mass sculpting itself is not necessarily a problem
  - dependent on the analysis strategy
  - DeepAK8 being explored in a few SUSY/ Exotic searches and looks very promising



- Nevertheless, a mass-independent tagger is more desirable in some other cases, e.g.,
  - preserves more events in the side band for background estimation
  - allows for signal extraction by fitting on the mass spectrum
- Is it possible to "decorrelate" DeepAK8 with the jet mass?

DECORRELATION WITH THE JET MASS

- The DeepAK8 architecture can be viewed as a two-step network

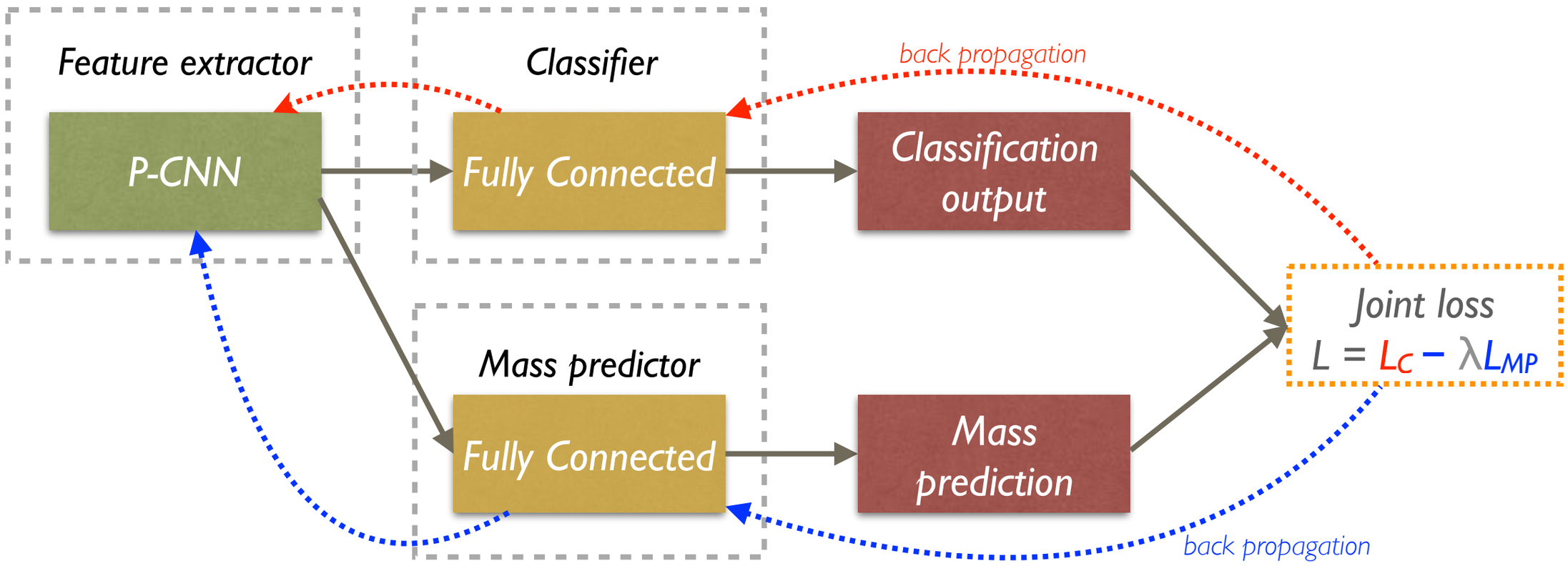


  - the P-CNNs first extract useful "features" from the particle and SV inputs
  - the FC layers then classify the jet based on the extracted features
    - the full network is trained as a whole by minimizing the classification loss L_C
- Without any constraints, the P-CNNs can learn all possible features contributing to the jet classification, including features that are highly correlated with the jet mass
  - this correlation is then reflected in the classifier outputs
- On the other hand, if we could regulate the P-CNNs so that the extracted features are not correlated with the jet mass, then the subsequent classifier will naturally inherit the mass-independence

ADVERSARIAL TRAINING

- We use adversarial training to regulate the behavior of the P-CNNs
  - a mass prediction network is introduced with the goal of predicting the jet mass from the features extracted by the P-CNNs
  - its loss, L_{MP}, is an indicator for mass correlation
    - smaller L_{MP} $\Longleftrightarrow$ more accurate mass prediction

      $\Longleftrightarrow$ the inputs (i.e., features extracted by the P-CNNs) has a higher correlation with the jet mass
  - a joint loss is introduced as $L = L_C - \lambda L_{MP}$, with the 2nd term as a penalty on mass correlation
    - minimizing L $\Longleftrightarrow$ simultaneously improving classification and reducing mass correlation
    - λ: a hyperparameter balancing between performance and mass independence

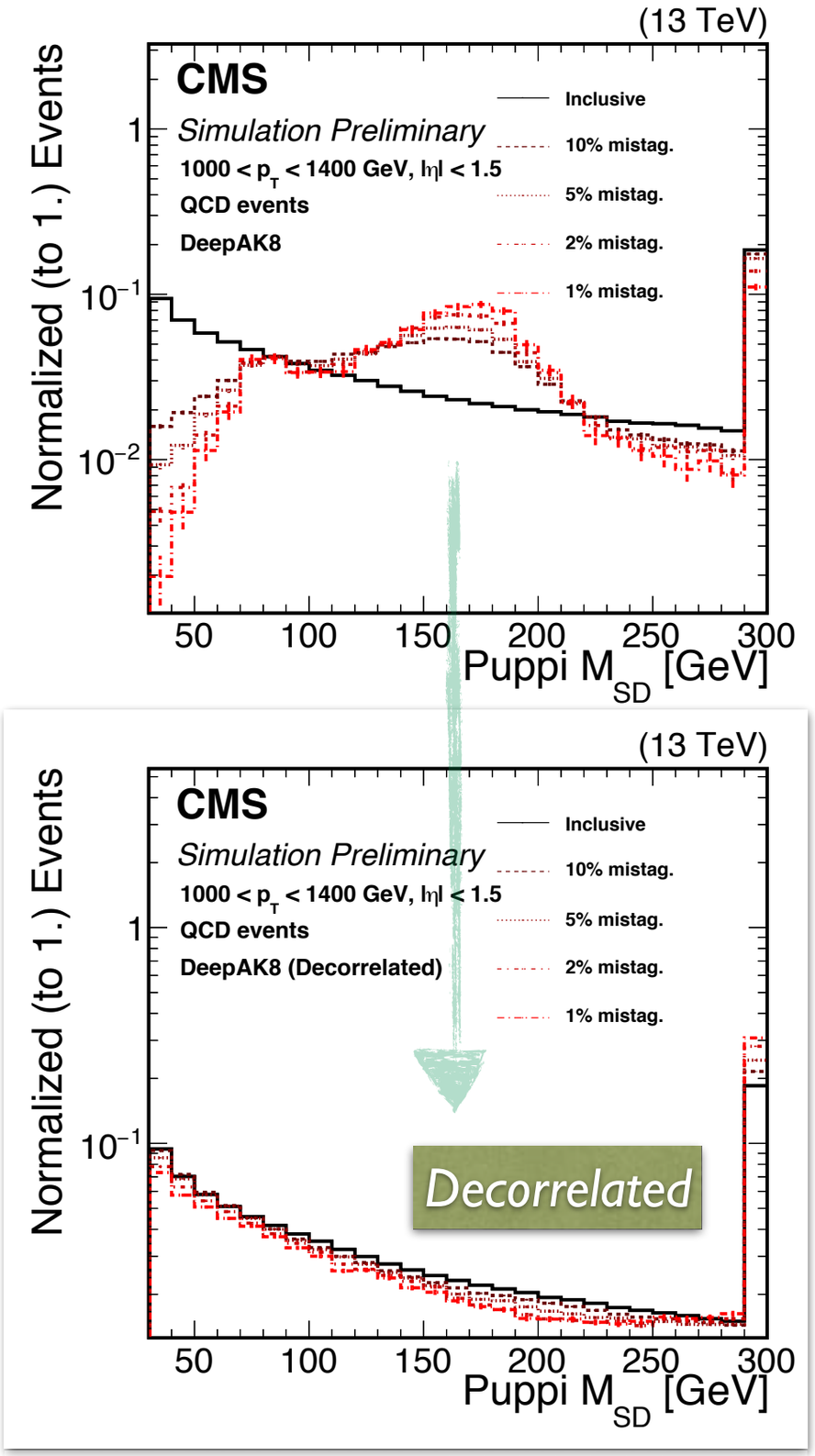


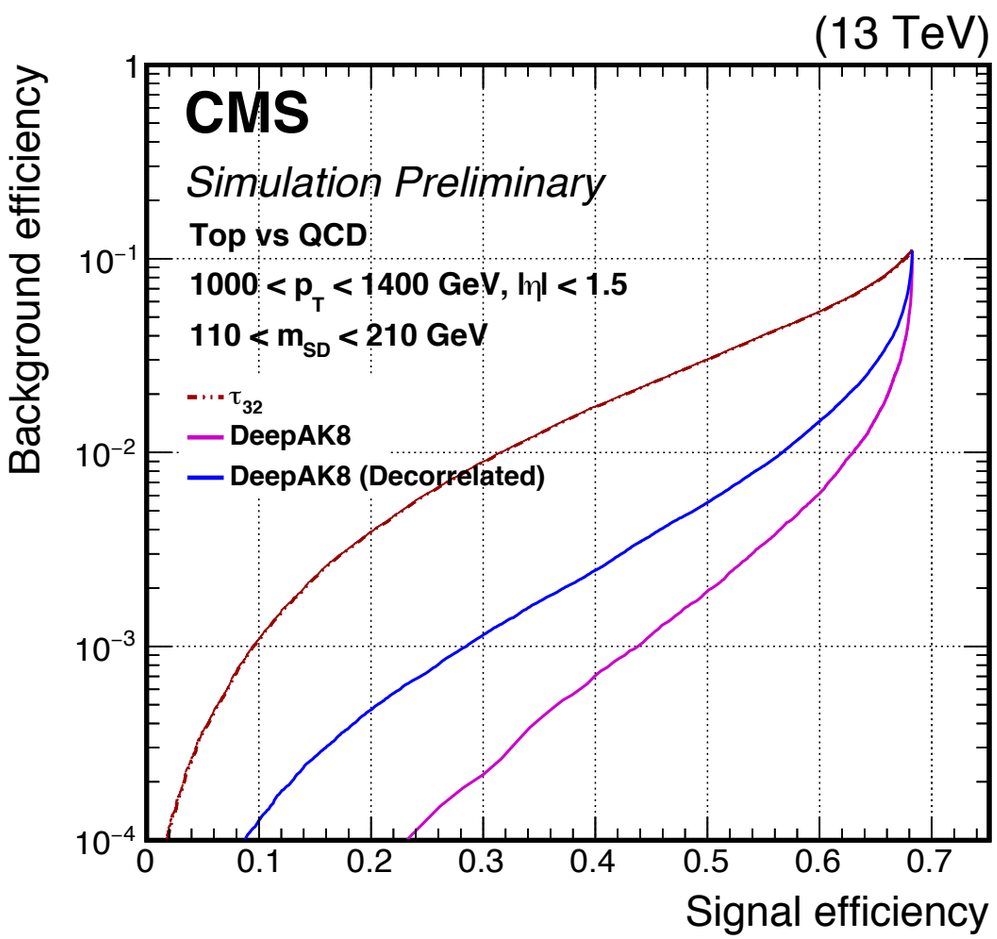
ADVERSARIAL TRAINING (II)

- Continued:
  - the mass predictor itself needs to be learned from the training data too
    - for the adversarial training procedure to work, the mass predictor needs to do a good job all the time, otherwise its loss will not reflect the level of mass correlation correctly
    - this is realized by iterating between the training of the classification network and the training of the mass prediction network
      - first train the mass predictor with its own loss using a mini-batch of data (e.g., O(100) ~ O(1000) events)
        - the P-CNN + classifier network is fixed at this stage
      - then train the full network (i.e., the P-CNN + classifier) with the joint loss using a mini-batch of data
        - the mass predictor network is fixed at this stage
      - iterating between them until the training converges
- more about adversarial training:
  - arXiv: 1611.01046, arXiv: 1703.03507
  - and many more ML literature on “domain adaption” and “learning fair representations”
    - e.g., 1705.11122, 1606.01614, 1409.7495, etc.

PERFORMANCE: TOP TAGGING

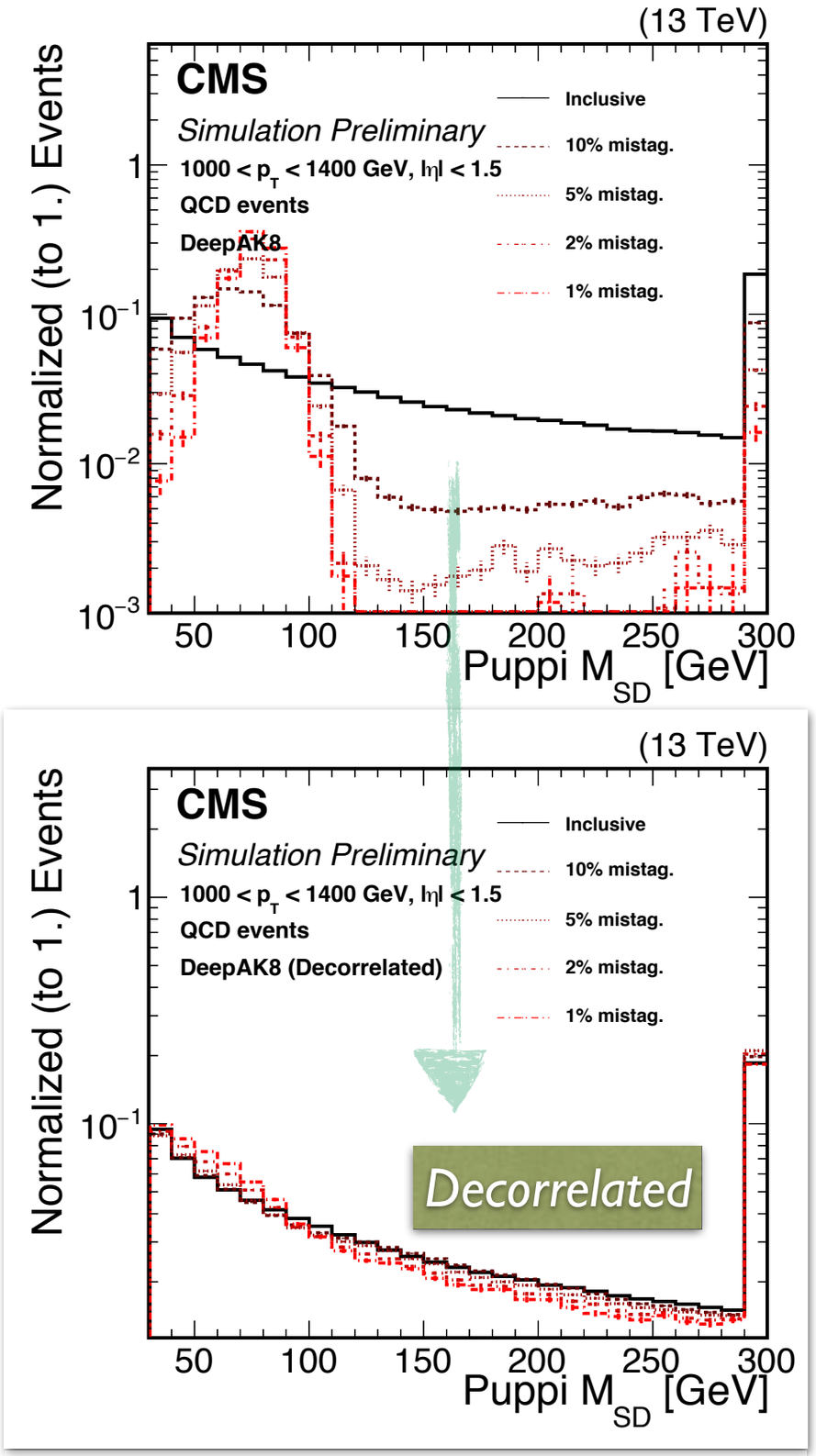
- Performance for top tagging
  - significantly reduced mass sculpting
  - at the price of some performance loss
    - trade-off between performance and mass-independence
  - still much more powerful than traditional approach (e.g., τ_{32})

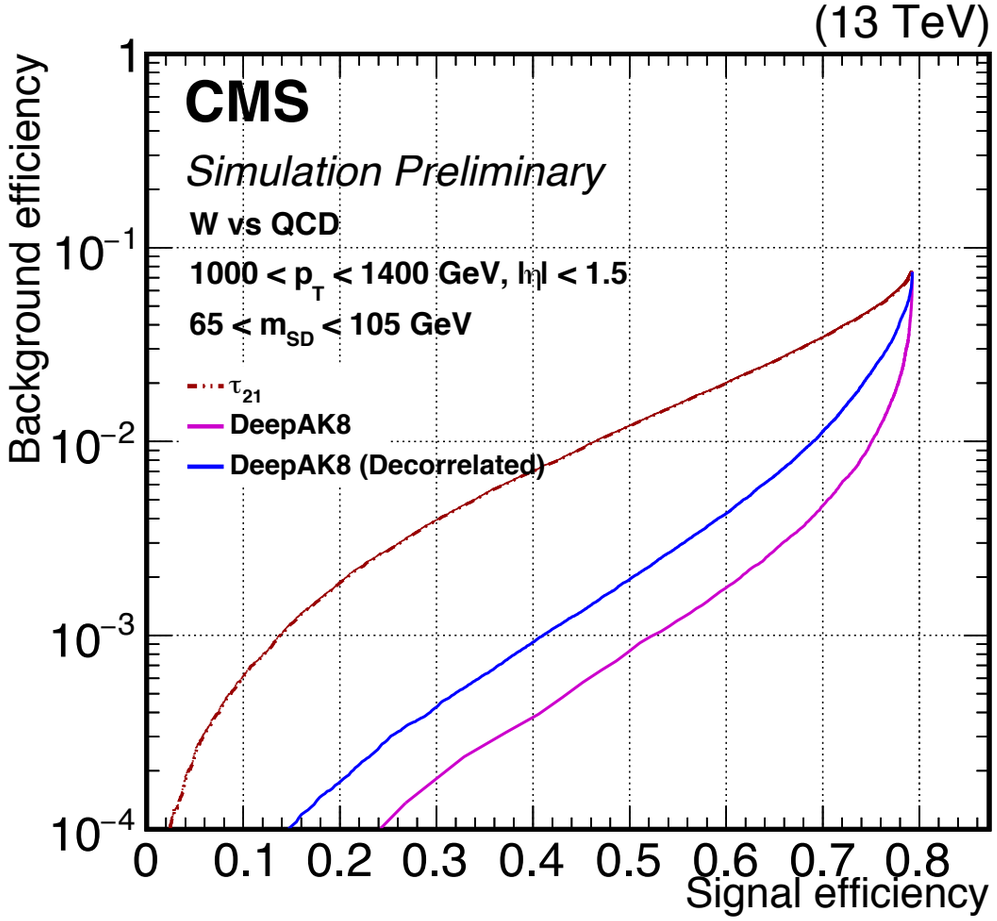




PERFORMANCE: W TAGGING

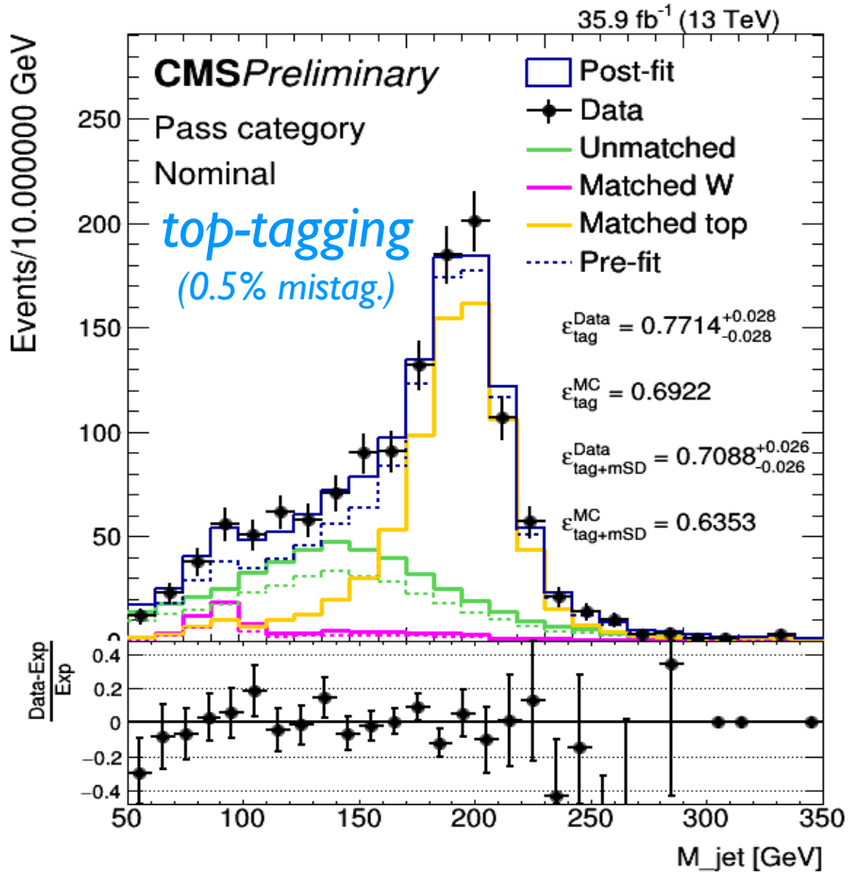
- Performance for W tagging
  - significantly reduced mass sculpting
  - at the price of some performance loss
    - trade-off between performance and mass-independence
  - still much more powerful than traditional approach (e.g., τ_{21})



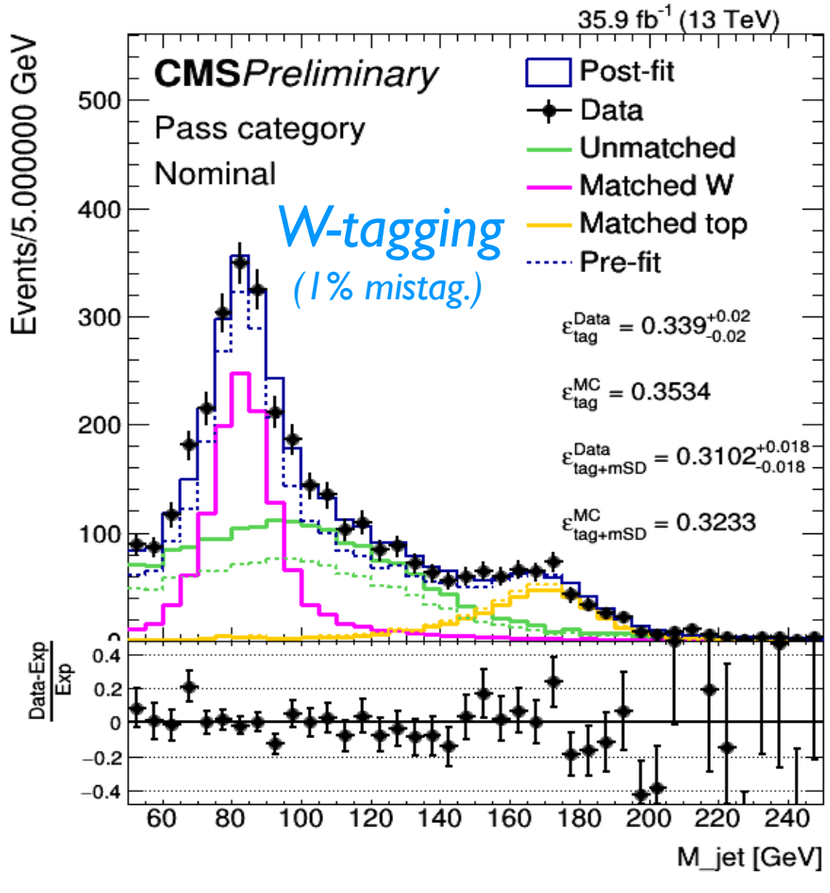


Validation in Data

- Performance for top/W tagging tested in data with a ttbar dominated sample
  - 1 tight muon (p_T>45 GeV, |η|<2.1), MET>50 GeV, N_j(ak4)≥2, N_b(tight)≥1
  - select highest p_T AK8 jet opposite to the muon as the candidate
  - define three mass templates: top-matched, W-matched and unmatched
  - simultaneously fitting the "pass" and "fail" categories to extract SFs for the tagging efficiency
- Performance in data and MC agrees well



Data/MC = 1.12 ± 0.04 *

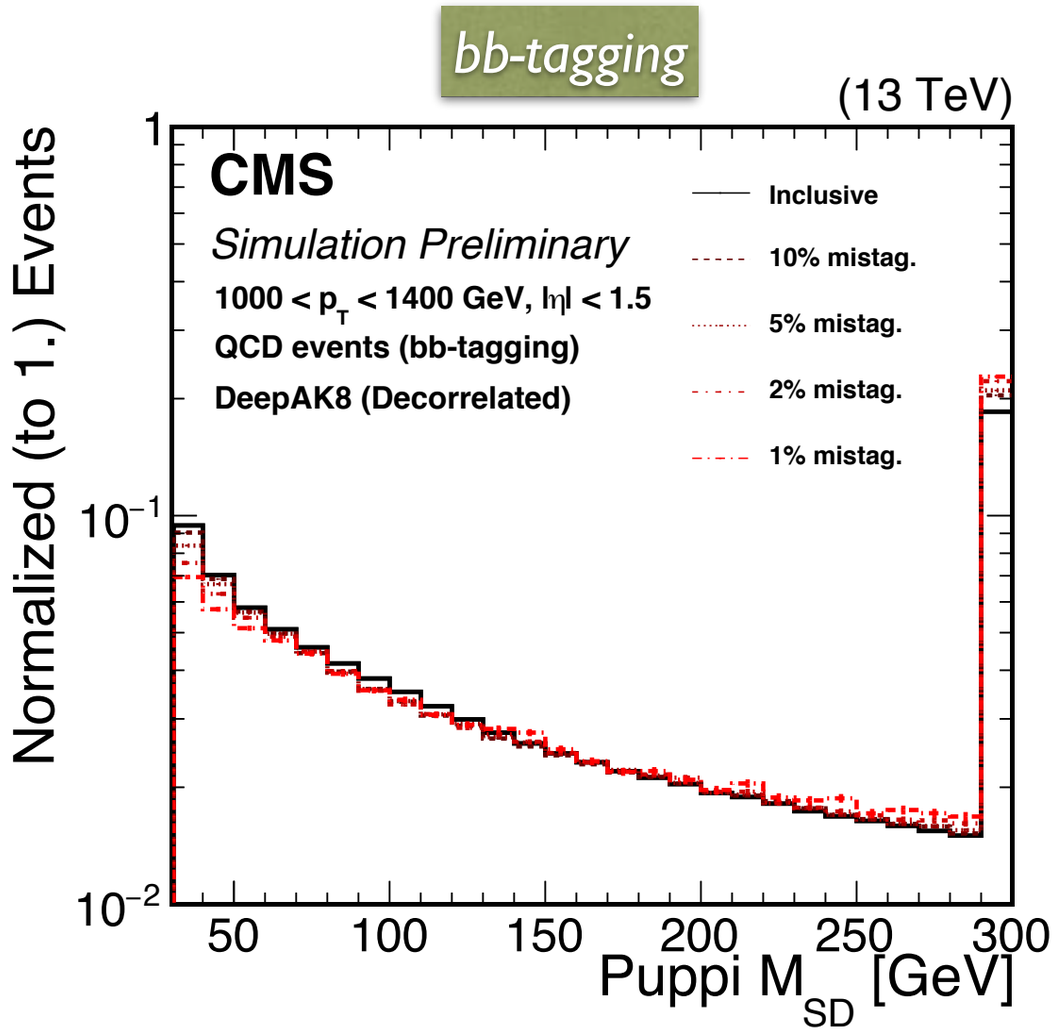


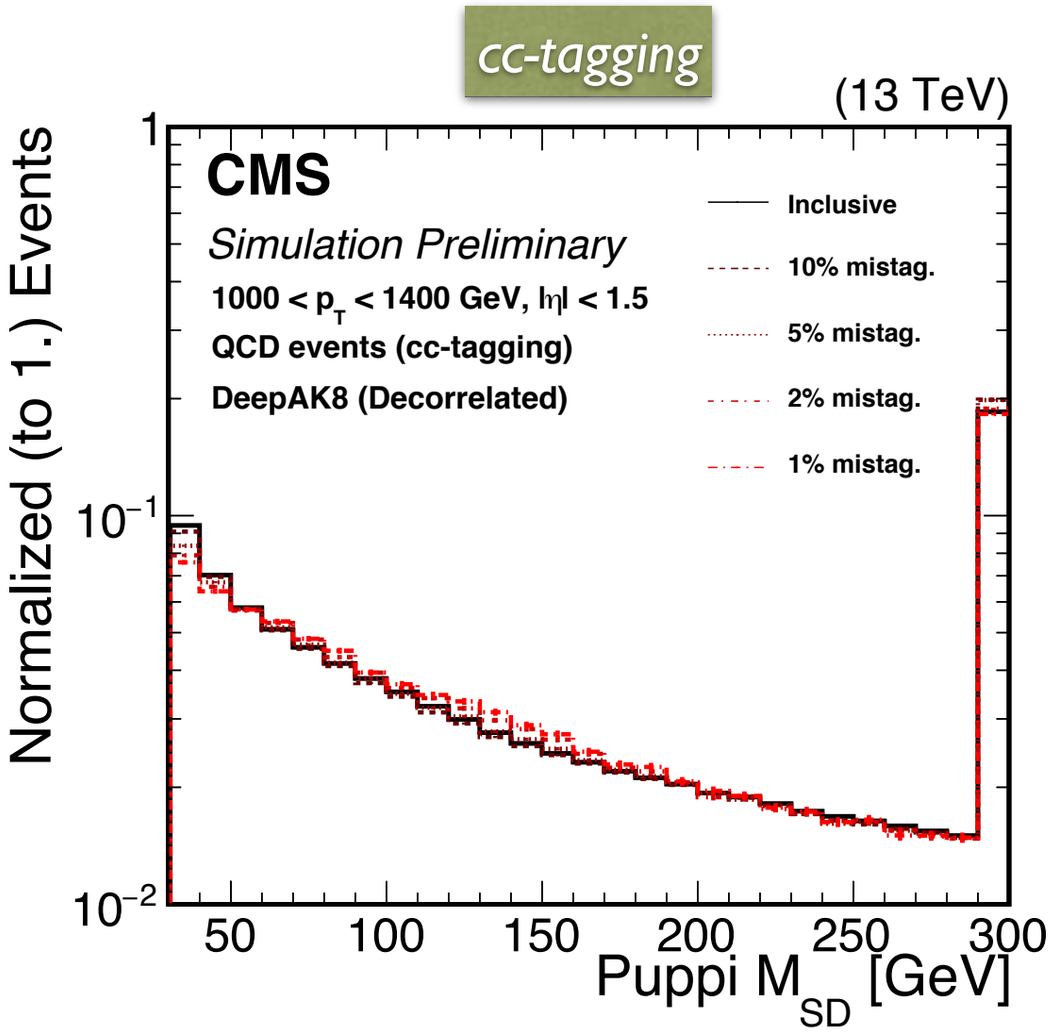
Data/MC = 0.96 +/- 0.06 *

(*) *systematic uncertainties not included yet*

PERFORMANCE: FLAVOUR TAGGING

- The decorrelated DeepAK8 can also be used as a boosted flavour tagger (bb vs. cc vs. light)
  - score (bb) := (H(bb) + Z(bb) + QCD(bb)) / (H(bb) + Z(bb) + H(cc) + Z(cc) + QCD (all))
  - score (cc) := (H(cc) + Z(cc) + QCD(cc)) / (H(bb) + Z(bb) + H(cc) + Z(cc) + QCD (all))
  - where QCD (all) = QCD(bb) + QCD(cc) + QCD(b) + QCD(c) + QCD(others)
- Mass sculpting well under control

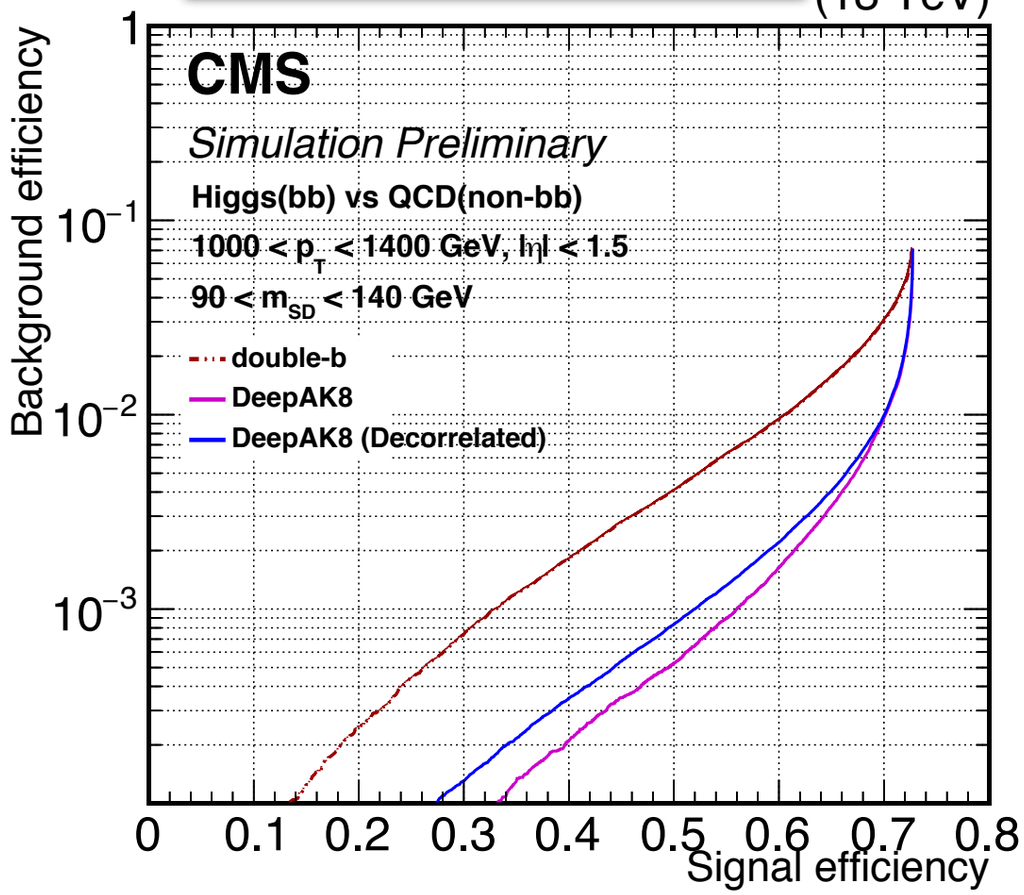




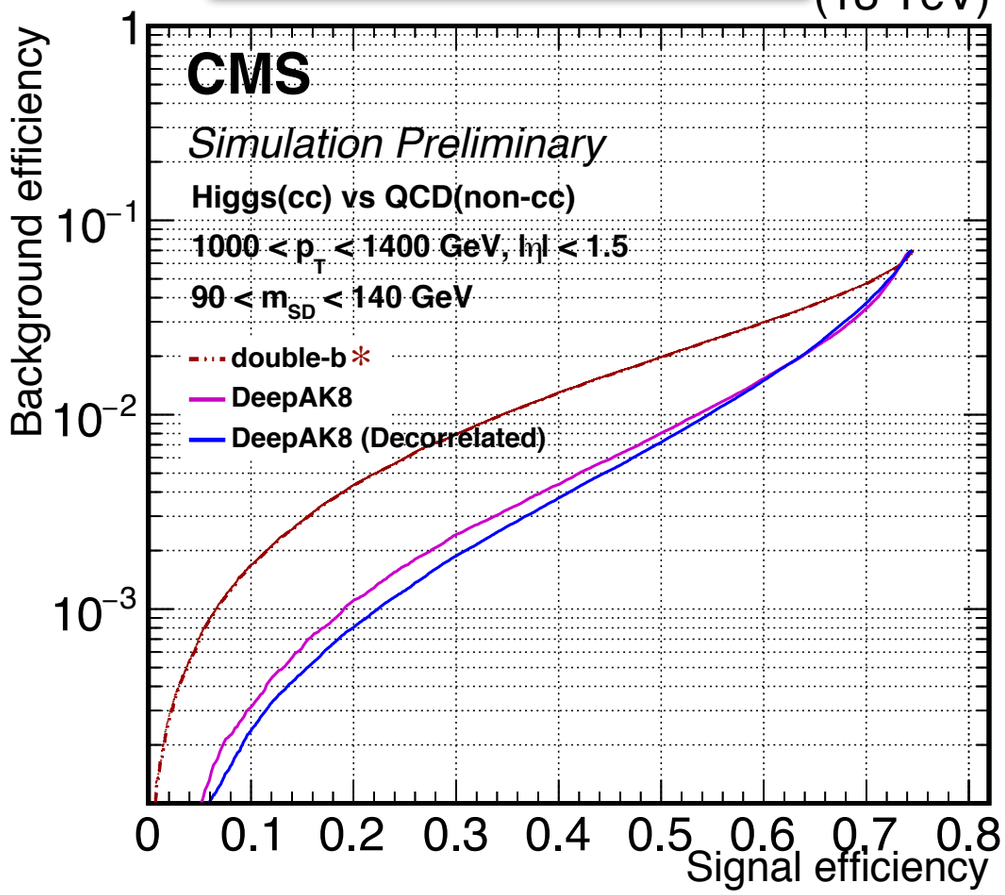
PERFORMANCE: FLAVOUR TAGGING (II)

- Performance is compared with existing methods and the base version of DeepAK8
  - signal: BulkGraviton->HH, with H->bb (or H->cc)
  - background: QCD sample with bb (or cc) class excluded
- The performance is still strong

Higgs(bb) vs QCD(non-bb)



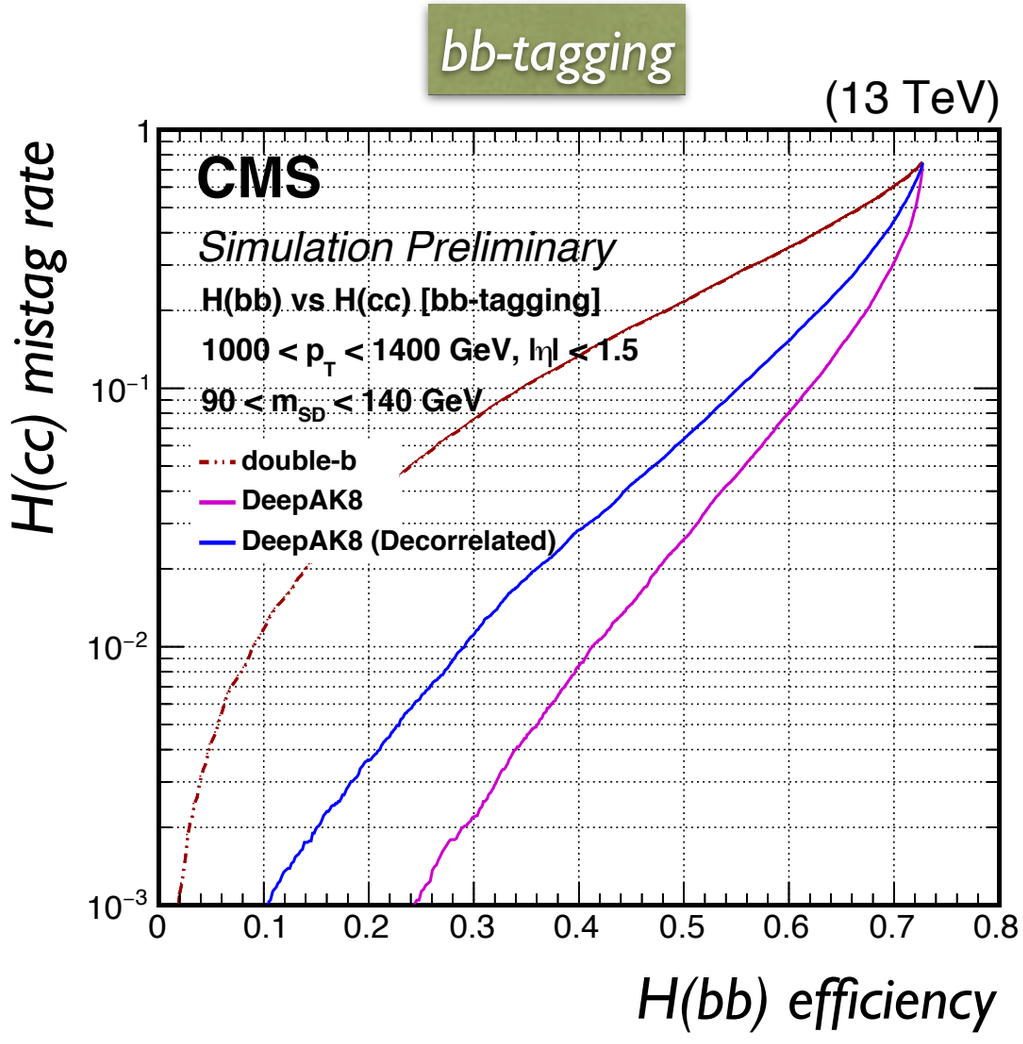
Higgs(cc) vs QCD(non-cc)

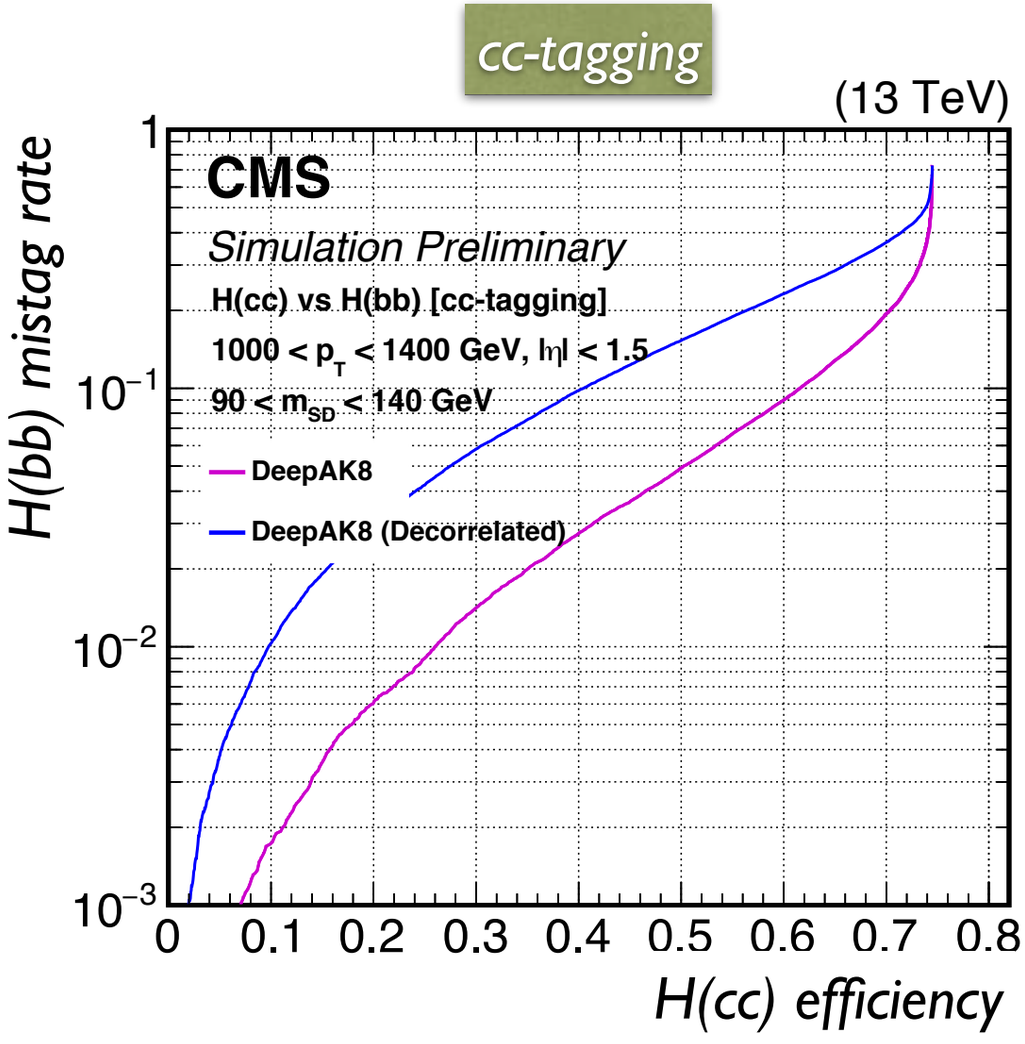


** double-b tagger not designed for cc-tagging*

PERFORMANCE: FLAVOUR TAGGING (III)

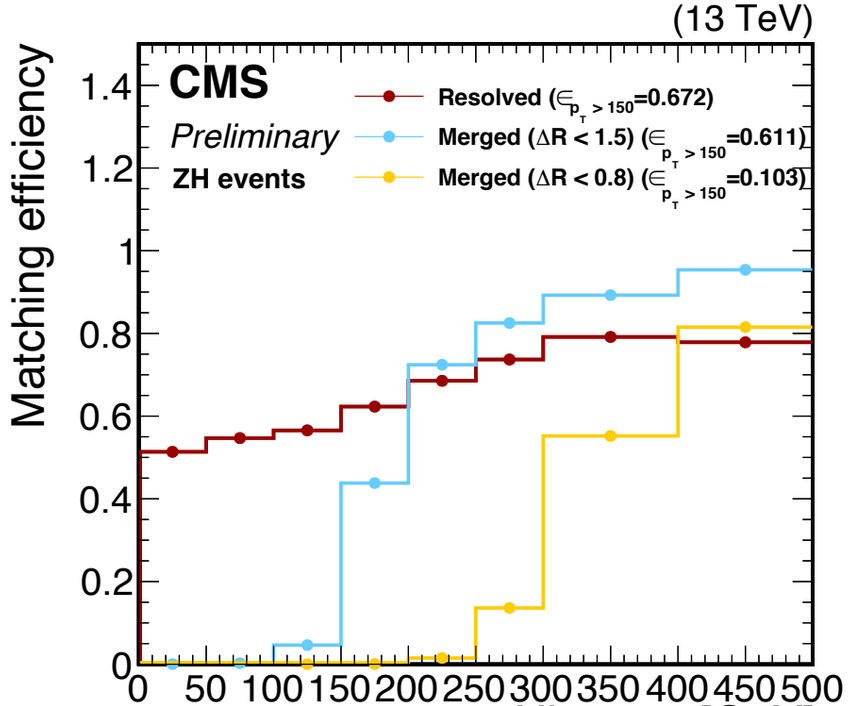
- Also powerful bb vs cc separation
  - demonstrated with BulkGrav->HH sample, with H->bb vs H->cc
- bb vs cc separation comes out-of-the-box, without dedicated re-training
  - advantage of the multi-class approach

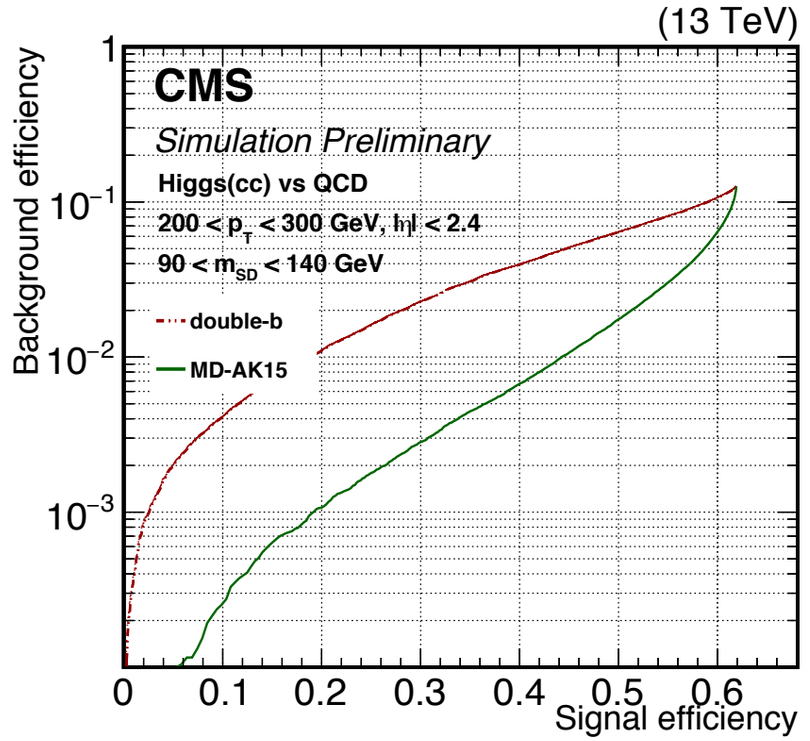




H→cc Measurement with DeepAK8

- DeepAK8 shows powerful cc-tagging capability as well as cc vs bb separation
  - this makes it a promising tool for the SM H->cc measurement
- Started looking into VH(cc) with DeepAK8
  - AK8 jets not very efficient for VH as the Higgs boson is typically not so boosted
  - we extend DeepAK8 to R=1.5 jets to better cover Higgs candidates with lower p_T
    - threshold for merging lowered from ~300 GeV (R=0.8) to ~150 GeV (R=1.5)
    - tagger retained with the same inputs and architecture as DeepAK8
- preliminary studies in VH(cc) look very encouraging
  - more details in the presentation at the Hbb meeting





SUMMARY

- DeepAK8: a versatile multi-class boosted jet tagger
  - fine-grained label definition allows for flexible usage by simply aggregating/ transforming the prediction scores
    - as boosted resonance tagger (top/W/Z/H)
    - as boosted flavour tagger (e.g., bb vs. cc vs. light)
    - etc.
  - significant improvement in performance compared to existing approaches
- Decorrelated DeepAK8:
  - adopts adversarial training technique to decorrelate DeepAK8 with the jet mass
  - shows significantly reduced mass sculpting but still strong performance
    - always a trade-off between discrimination power and mass-independence
- Work in progress to integrate DeepAK8 into CMSSW
  - preliminary recipe in place for interested groups
- New training with 2017 samples planned to fully exploit the new pixel detector

Backups

P-CNN

PARTICLE-LEVEL CNN

- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles

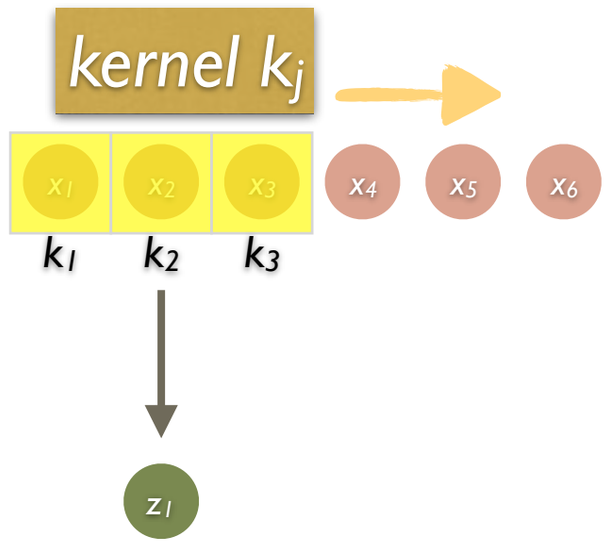
particle sequence

only one feature: x_i
(e.g., p_T of the i^{th} particle)

PARTICLE-LEVEL CNN

- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles

particle sequence



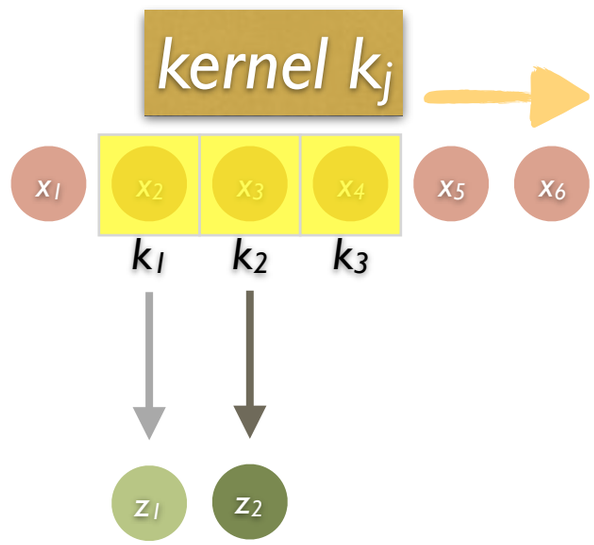
only one feature: x_i
(e.g., p_T of the i^{th} particle)

$z_1 = k_1 * x_1 + k_2 * x_2 + k_3 * x_3$

PARTICLE-LEVEL CNN

- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles

particle sequence



only one feature: x_i
(e.g., p_T of the i^{th} particle)

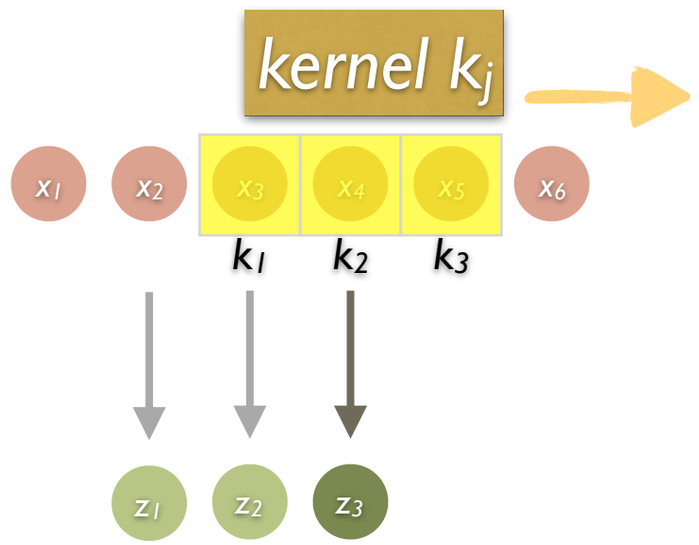
$z_1 = k_1 * x_1 + k_2 * x_2 + k_3 * x_3$

$z_2 = k_1 * x_2 + k_2 * x_3 + k_3 * x_4$

PARTICLE-LEVEL CNN

- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles

particle sequence



only one feature: x_i
(e.g., p_T of the i^{th} particle)

$z_1 = k_1 * x_1 + k_2 * x_2 + k_3 * x_3$

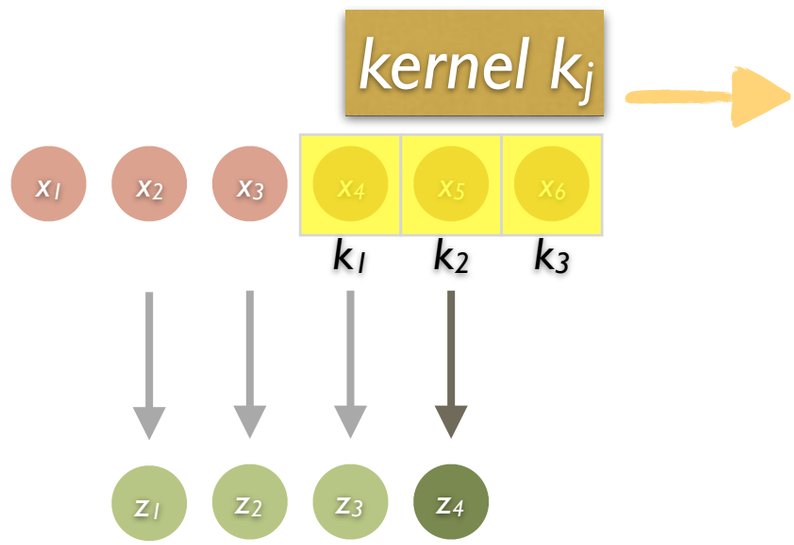
$z_2 = k_1 * x_2 + k_2 * x_3 + k_3 * x_4$

$z_3 = k_1 * x_3 + k_2 * x_4 + k_3 * x_5$

PARTICLE-LEVEL CNN

- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles

particle sequence



only one feature: x_i
(e.g., p_T of the i^{th} particle)

$z_1 = k_1 * x_1 + k_2 * x_2 + k_3 * x_3$

$z_2 = k_1 * x_2 + k_2 * x_3 + k_3 * x_4$

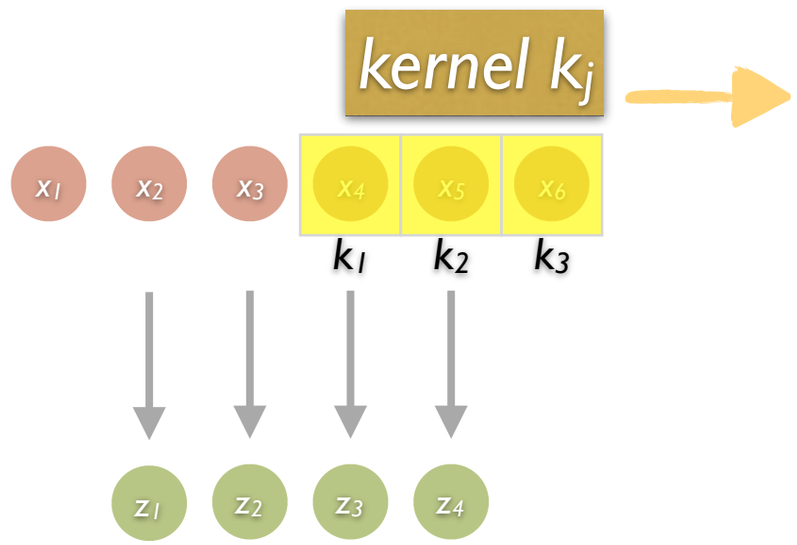
$z_3 = k_1 * x_3 + k_2 * x_4 + k_3 * x_5$

$z_4 = k_1 * x_4 + k_2 * x_5 + k_3 * x_6$

PARTICLE-LEVEL CNN

- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles

particle sequence



only one feature: x_i
(e.g., p_T of the i^{th} particle)

$z_1 = k_1 * x_1 + k_2 * x_2 + k_3 * x_3$

$z_2 = k_1 * x_2 + k_2 * x_3 + k_3 * x_4$

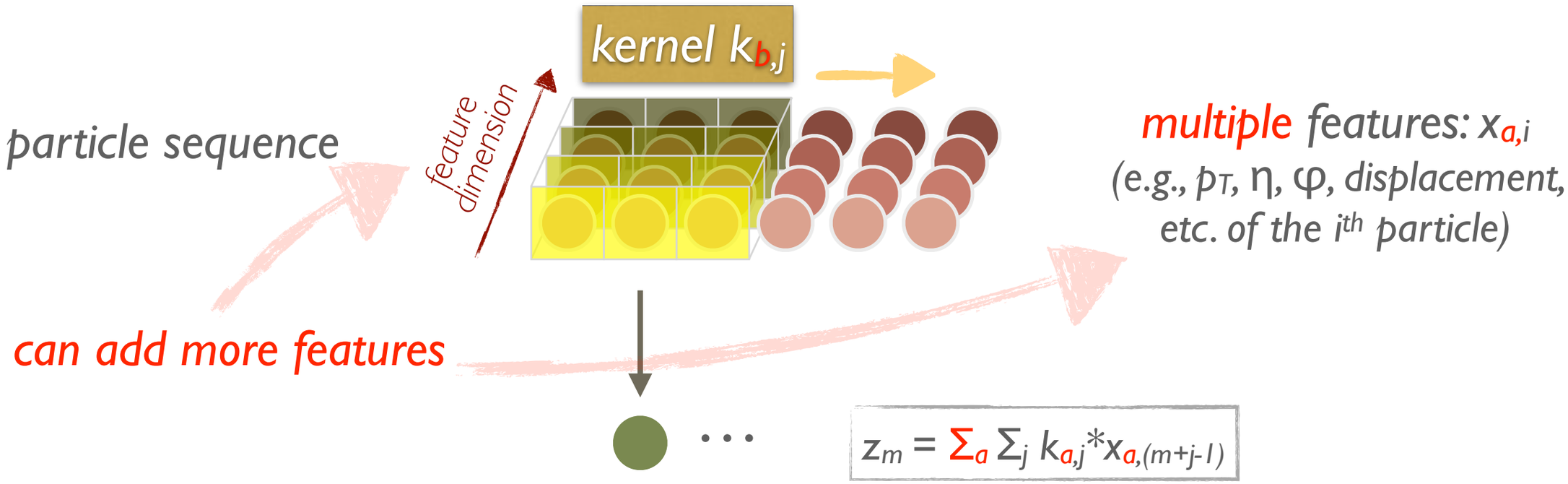
$z_3 = k_1 * x_3 + k_2 * x_4 + k_3 * x_5$

$z_4 = k_1 * x_4 + k_2 * x_5 + k_3 * x_6$

$z_m = \sum_j k_j * x_{(m+j-1)}$

PARTICLE-LEVEL CNN (II)

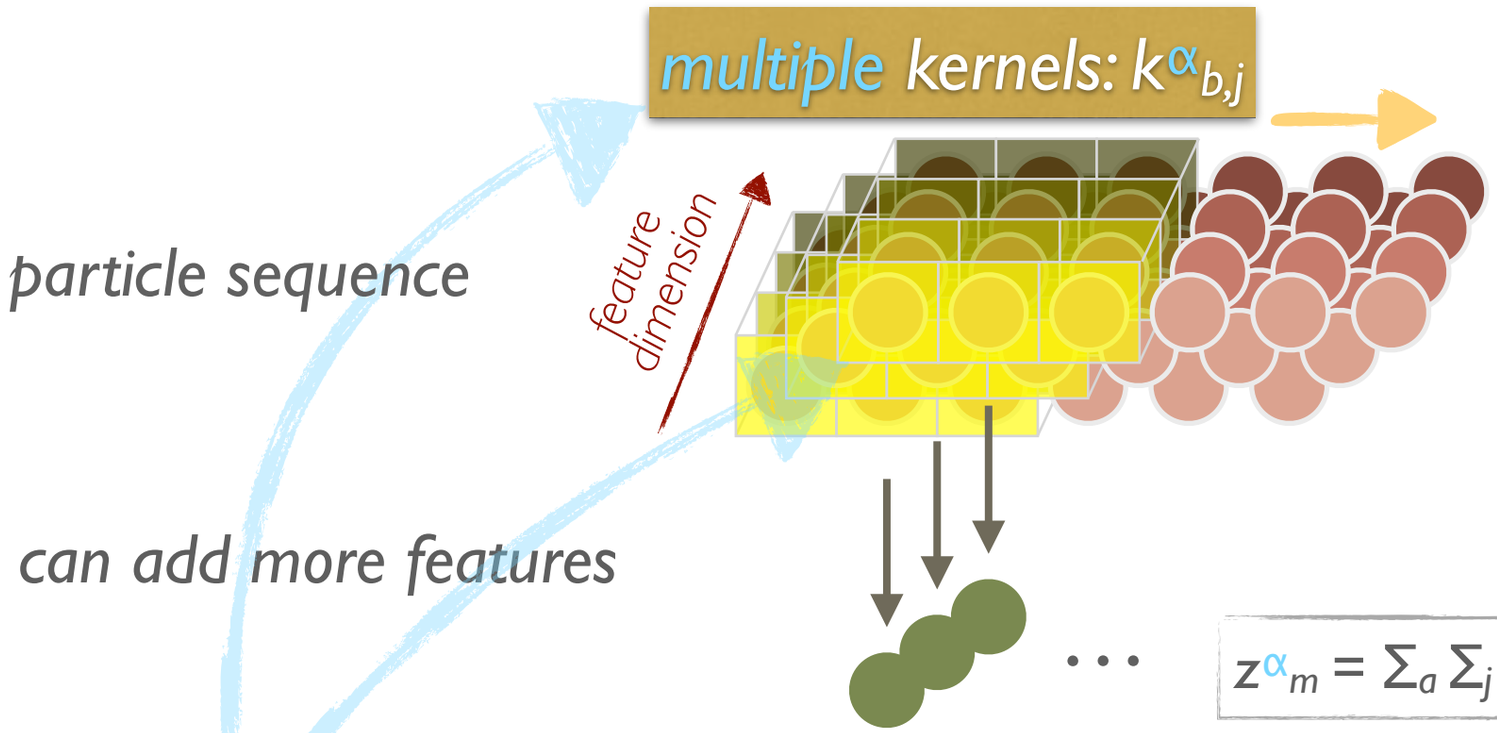
- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles



  - *a transformation that combines all features of nearby particles*

PARTICLE-LEVEL CNN (III)

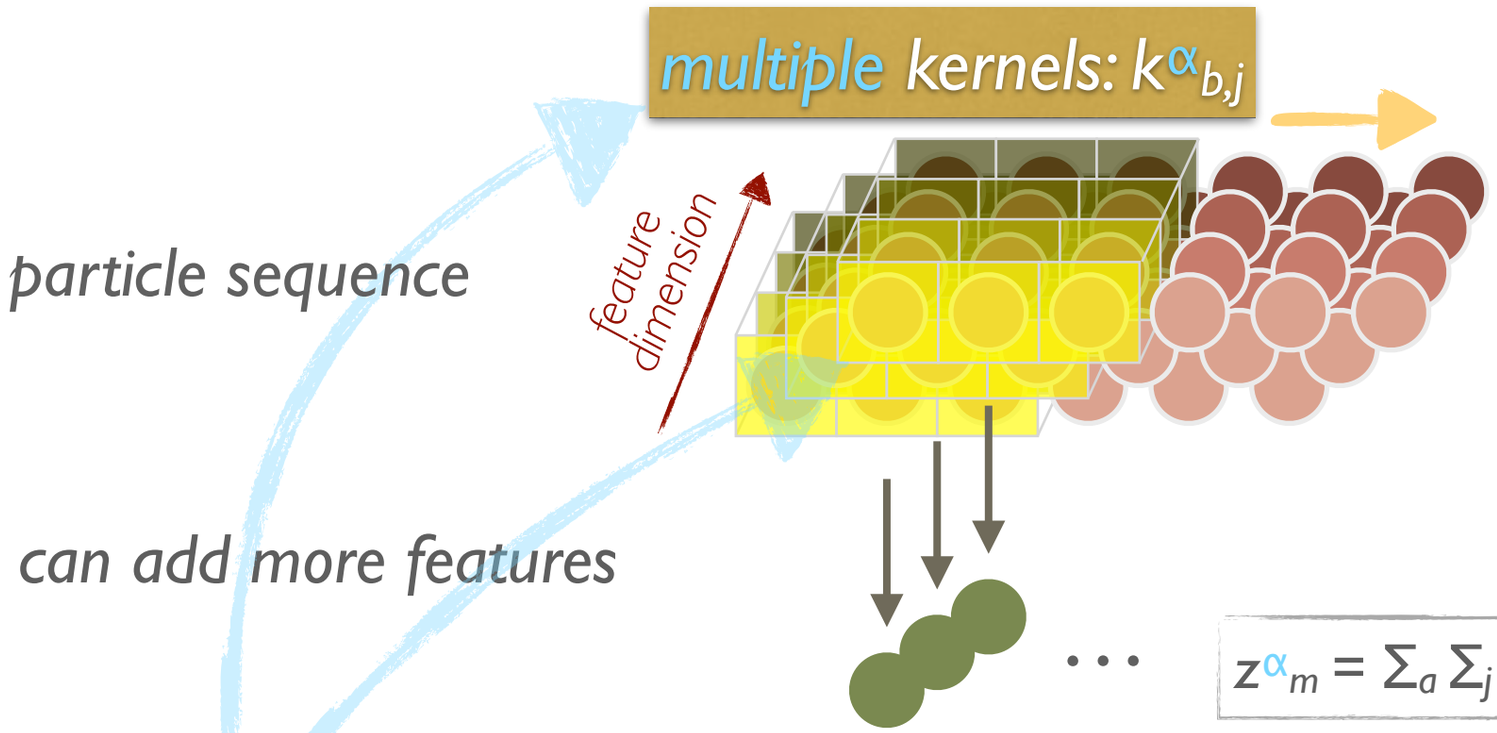
- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles



- each kernel gives a different transformation

PARTICLE-LEVEL CNN (III)

- Particle-level CNN (P-CNN)
  - one dimensional CNN over a sequence of particles



- each kernel gives a different transformation
- these transformations are "learned" from the training sample to best extract information that are useful for boosted jet tagging

SAMPLES

SAMPLES

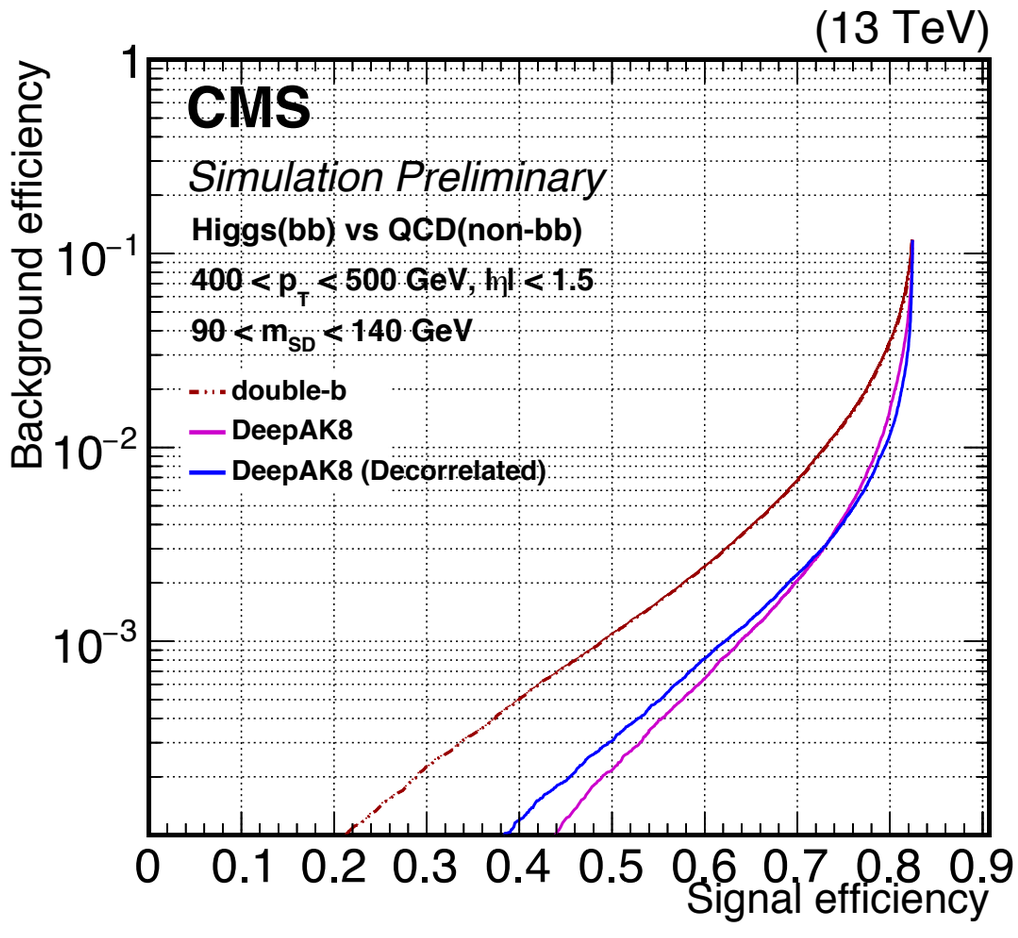
- High p_T:
  - $1000 < p_T < 1400$ GeV, $|\eta| < 1.5$
  - mass cut: [90, 140] GeV; apply the Puppi soft drop correction [Moriond17corrections]
  - signal:
    - H->bb: BulkGravTohhTohbbhbb_narrow_M-3000_13TeV-madgraph
    - H->cc: GluGluToBulkGravitonToHHTo4C_M-3000_narrow_13TeV-madgraph-pythia8
    - truth matching: deltaR(jet, q)<0.8
  - background:
    - QCD_Pt_1000to1400_TuneCUETP8M1_13TeV_pythia8
- Low p_T:
  - $400 < p_T < 500$ GeV, $|\eta| < 1.5$
  - signal:
    - H->bb: BulkGravTohhTohbbhbb_narrow_M-1000_13TeV-madgraph
    - H->cc: GluGluToBulkGravitonToHHTo4C_M-1000_narrow_13TeV-madgraph-pythia8
  - background:
    - QCD_Pt_470to600_TuneCUETP8M1_13TeV_pythia8

Extra Performance Plots

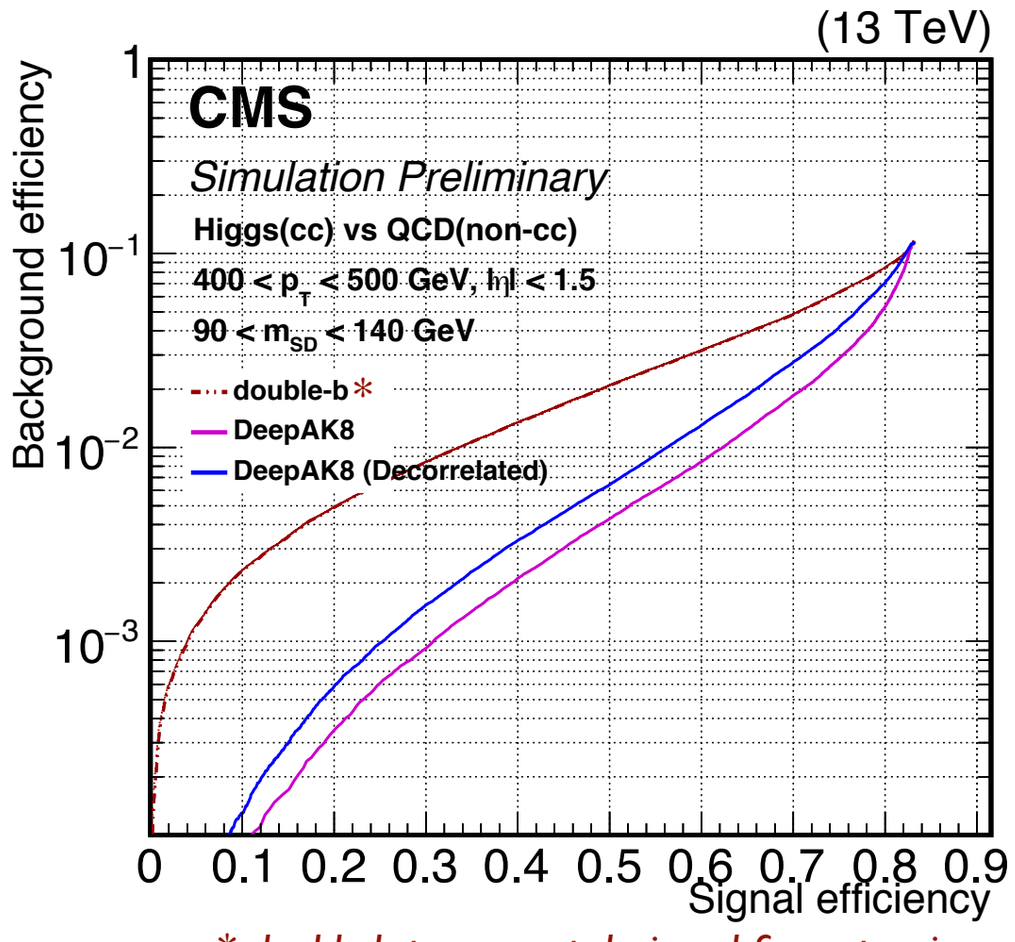
Low Pt

- Background:
  - QCD (excludig flavours including bb and cc)

Higgs(bb) vs QCD(non-bb)



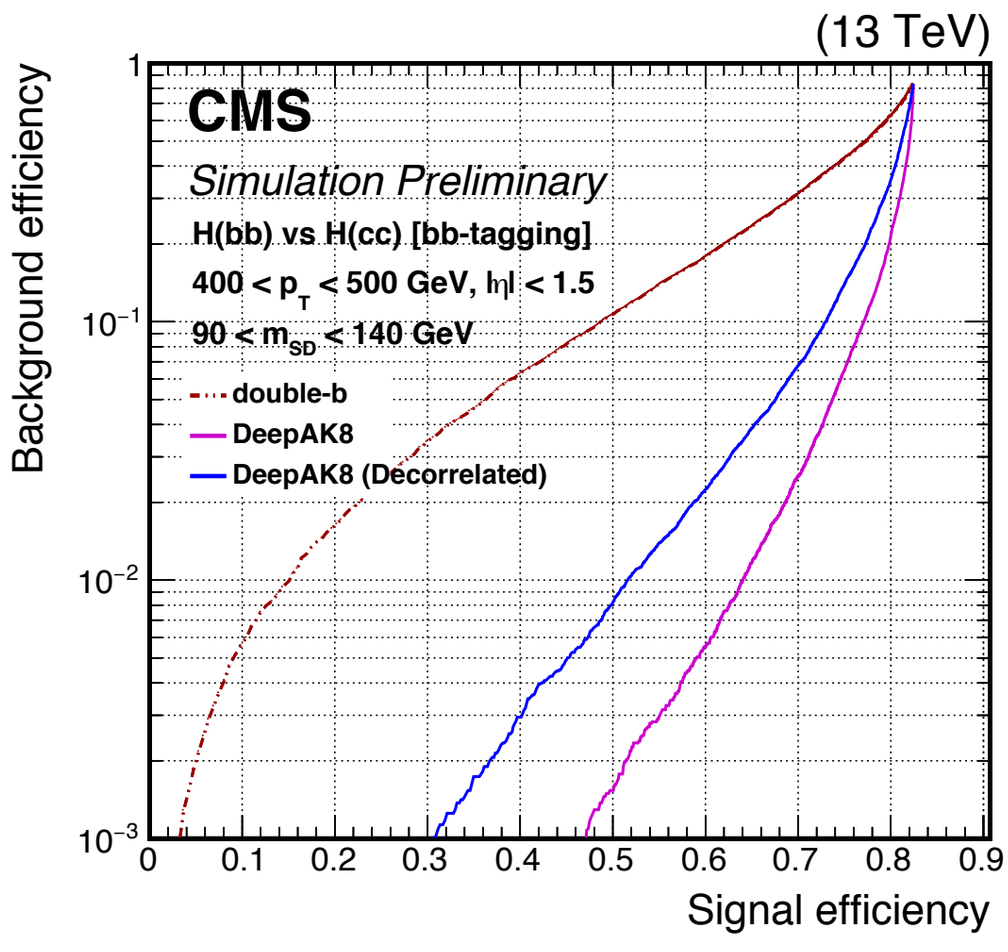
Higgs(cc) vs QCD(non-cc)



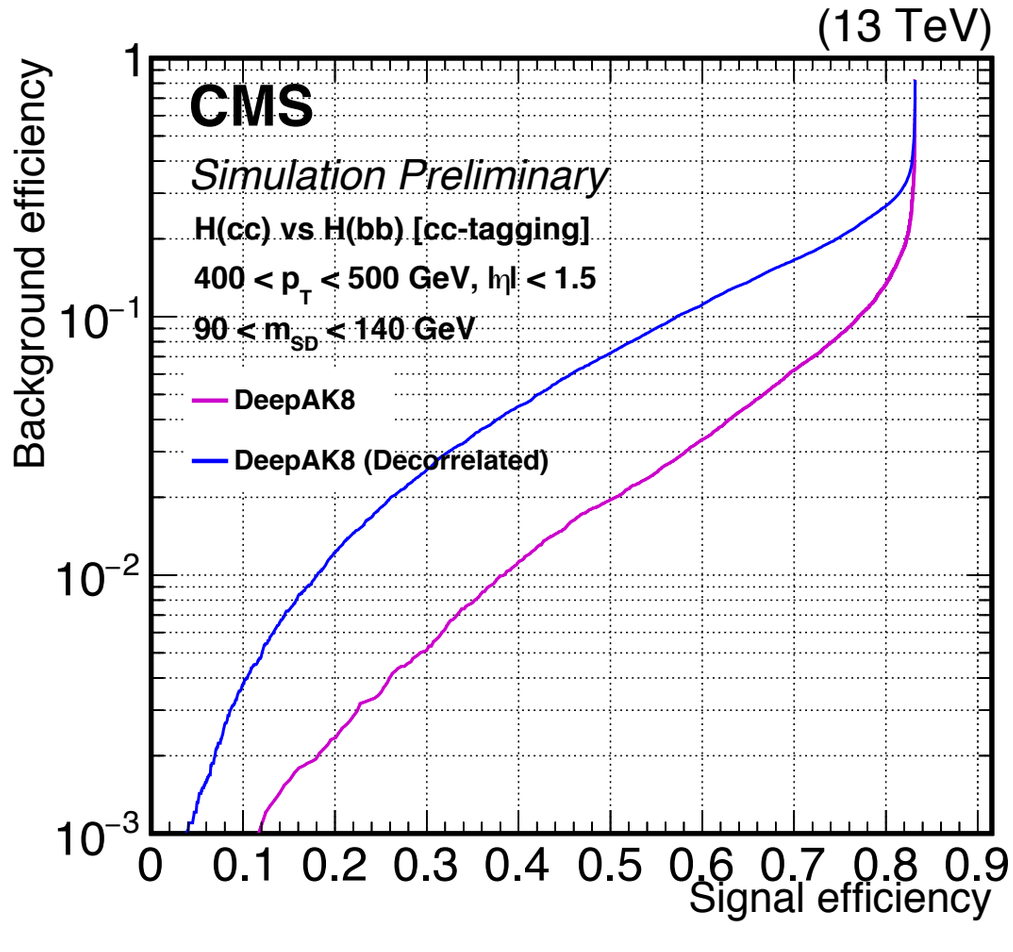
** double-b tagger not designed for cc-tagging*

Low Pt (bb vs cc)

bb-tagging



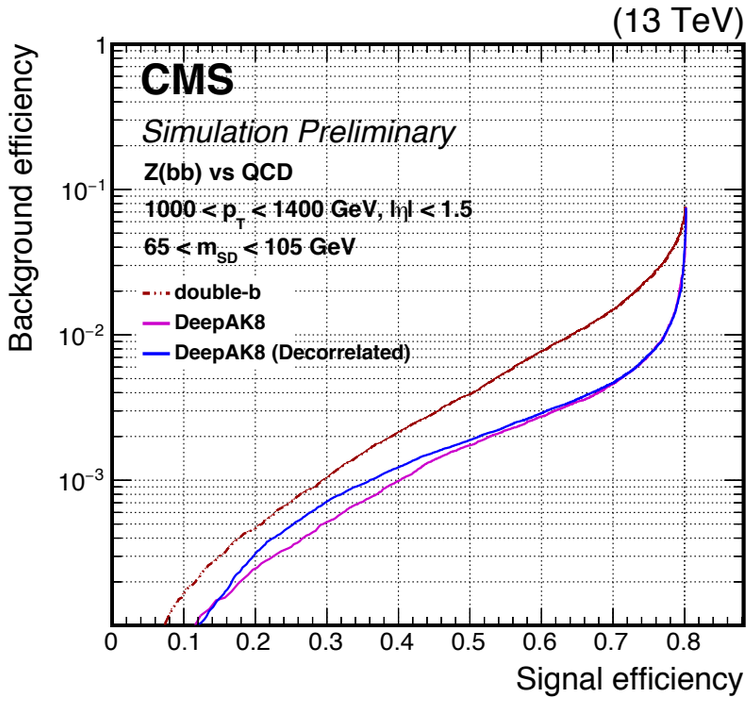
cc-tagging

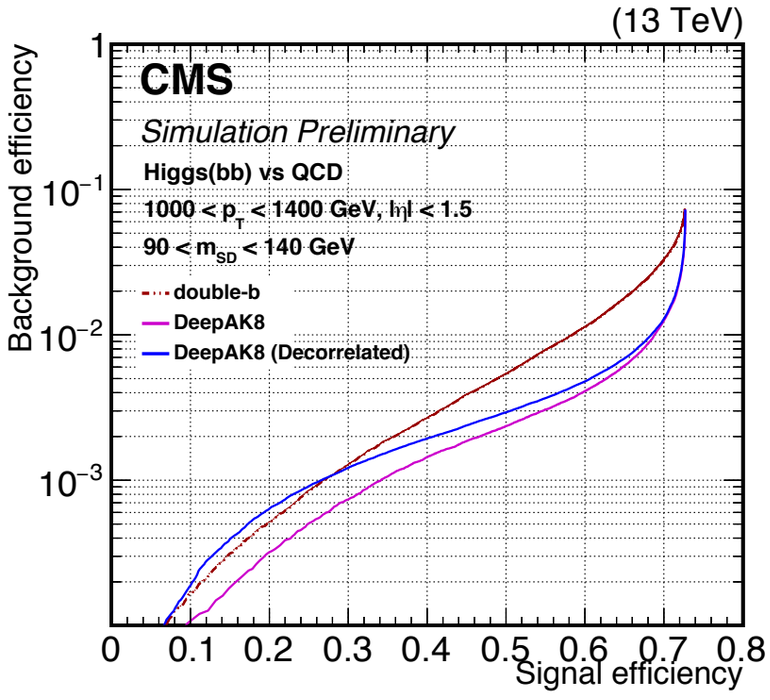


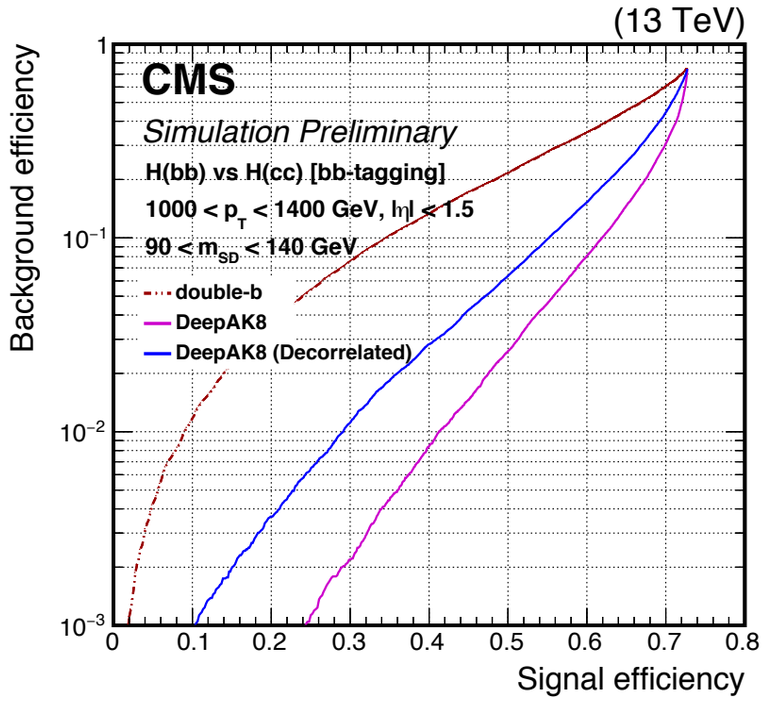
Z+H+QCD Scores

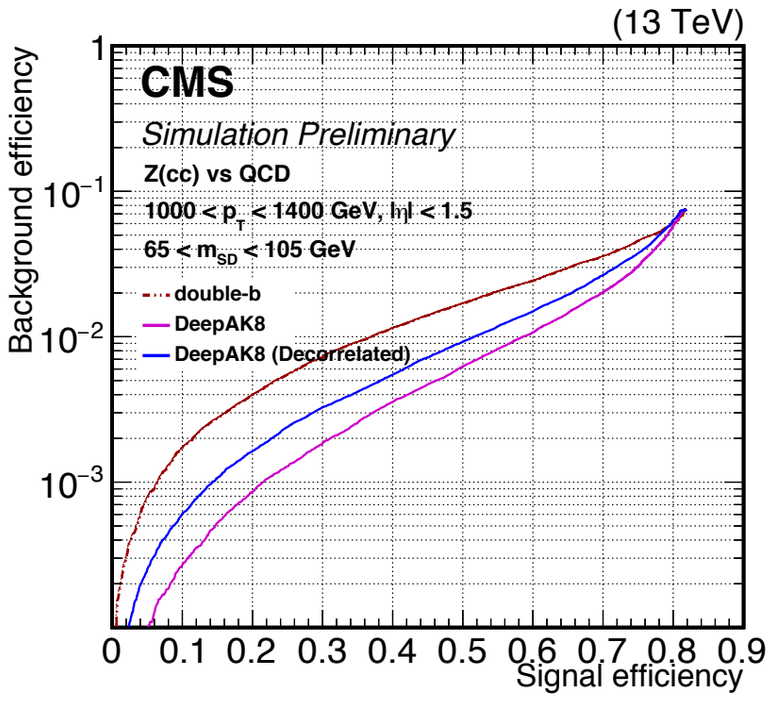
- Score definition:
  - score (bb) := (H(bb) + Z(bb) + QCD(bb)) / (H(bb) + Z(bb) + H(cc) + Z(cc) + QCD (all))
  - score (cc) := (H(cc) + Z(cc) + QCD(cc)) / (H(bb) + Z(bb) + H(cc) + Z(cc) + QCD (all))
  - where QCD (all) = QCD(bb) + QCD(cc) + QCD(b) + QCD(c) + QCD(others)
- Background:
  - QCD (all flavours including bb and cc)

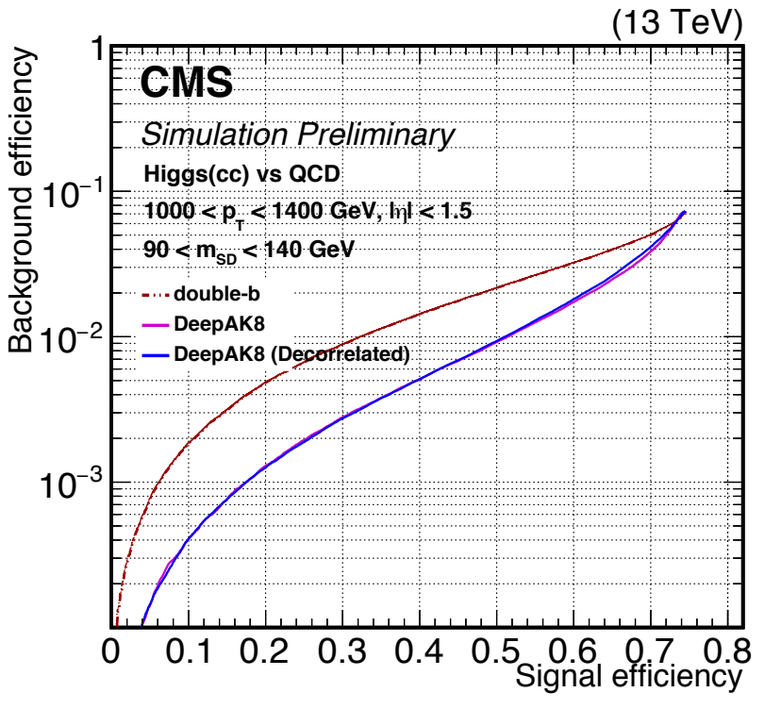
Z+H+QCD score (High pT)

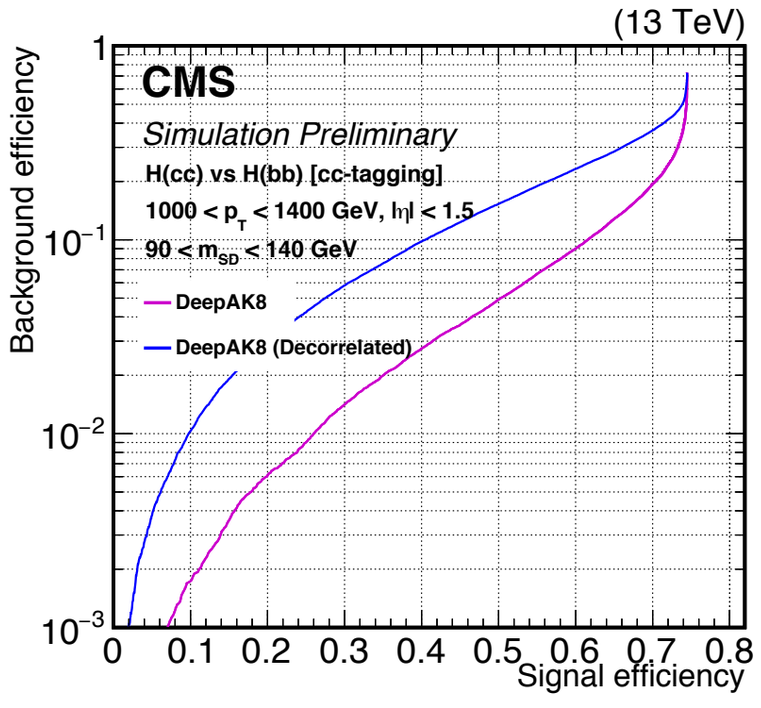




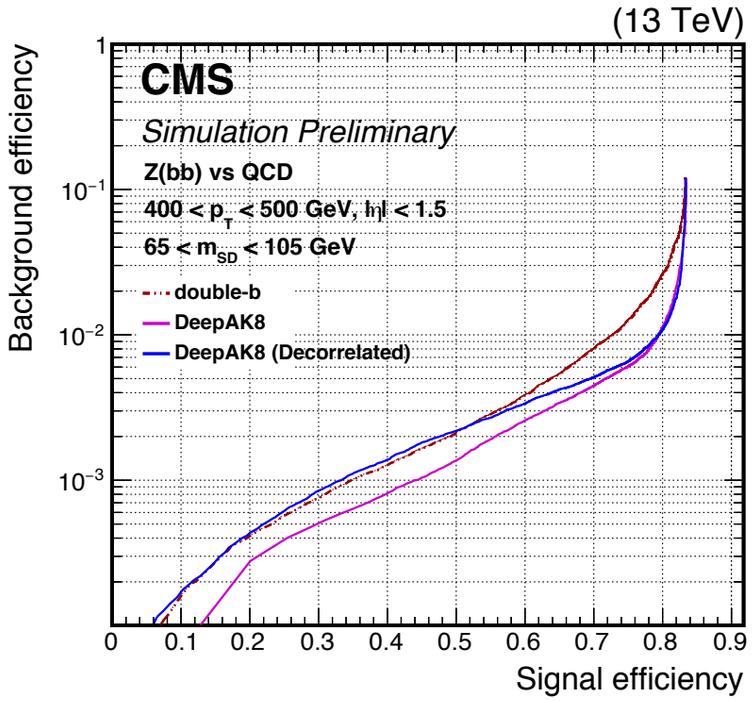


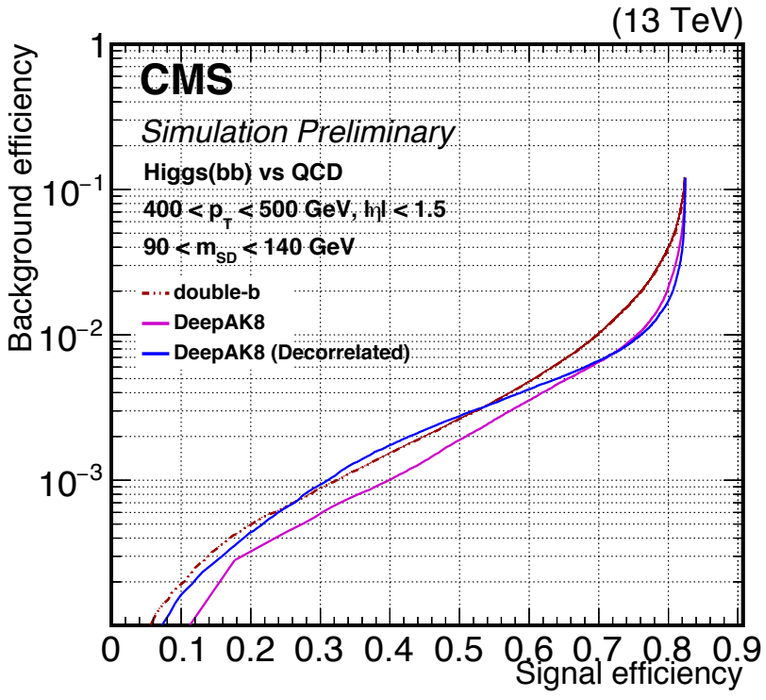


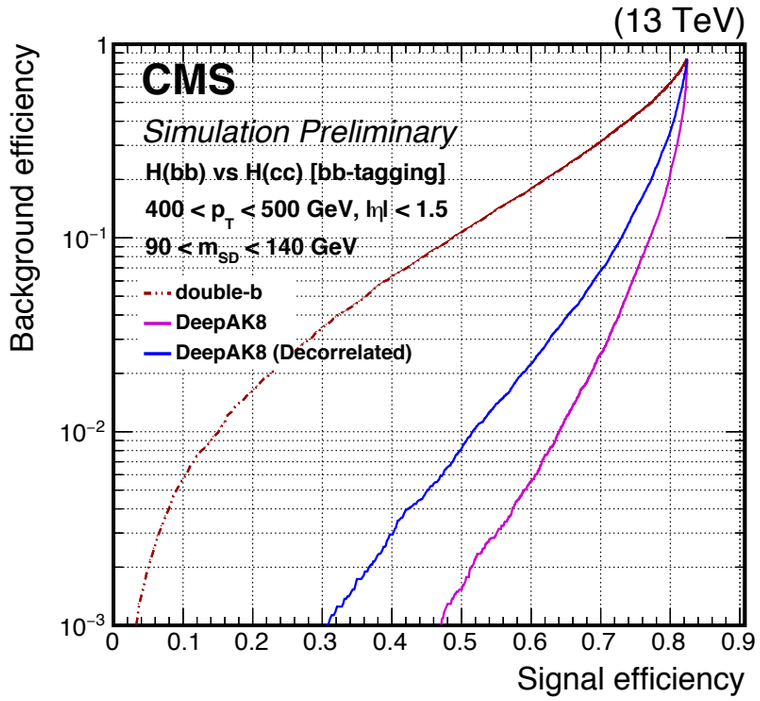


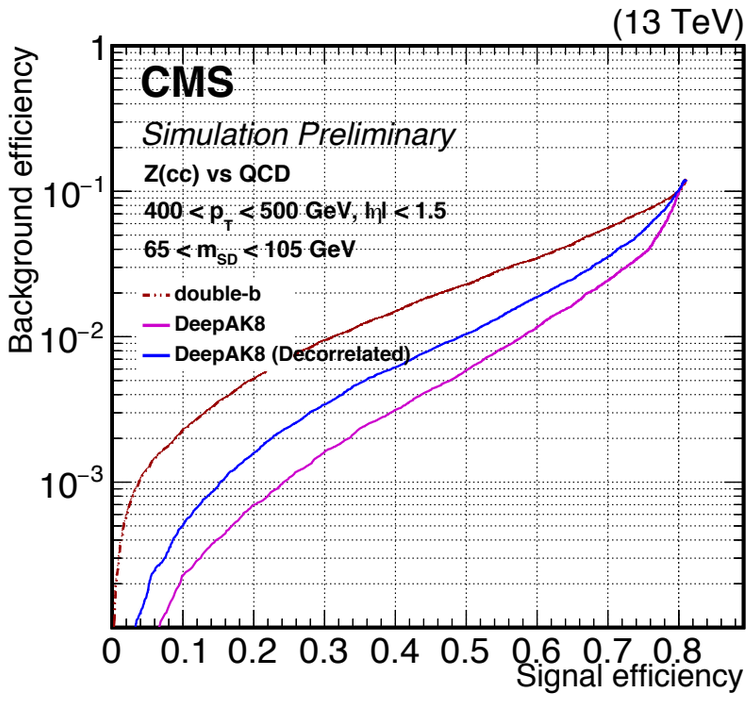


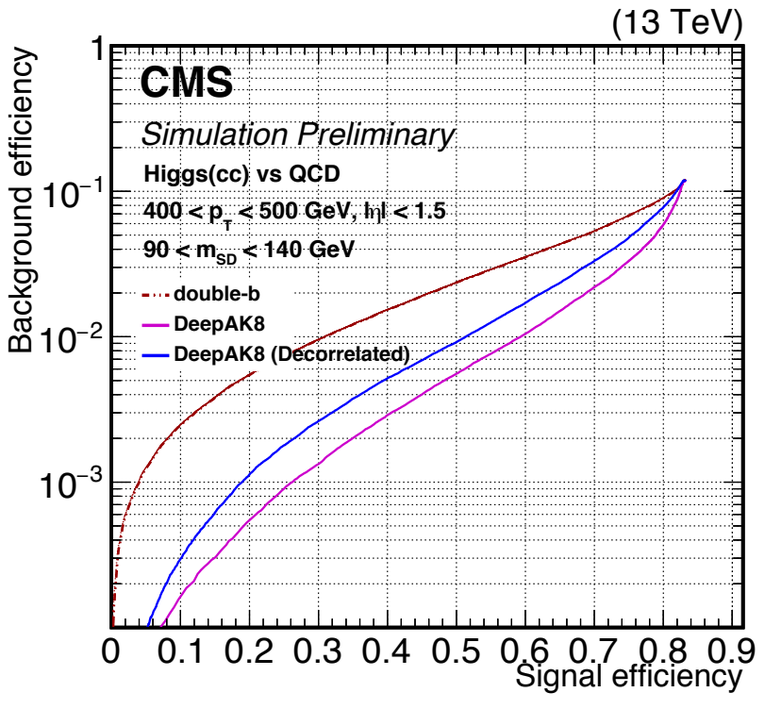
Z+H+QCD score (Low Pt)

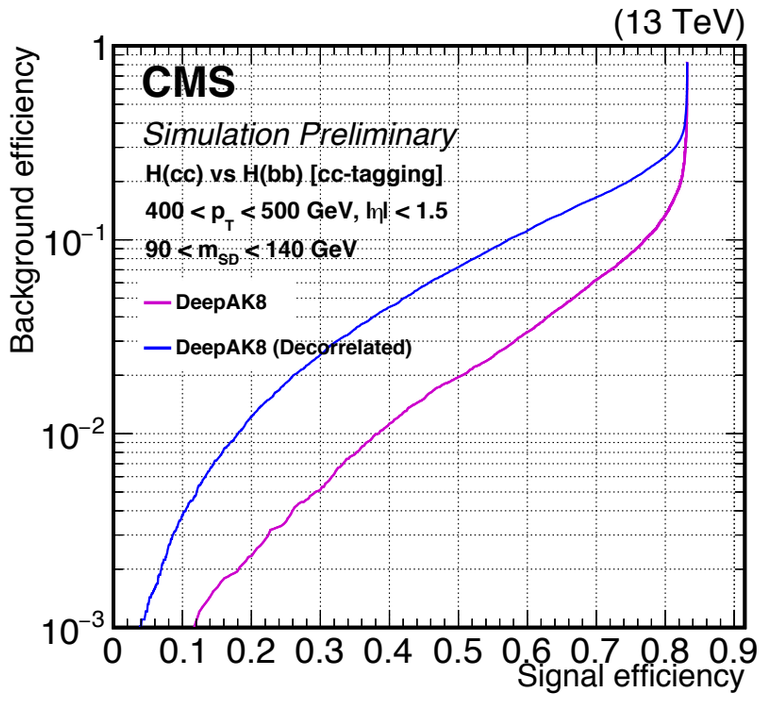




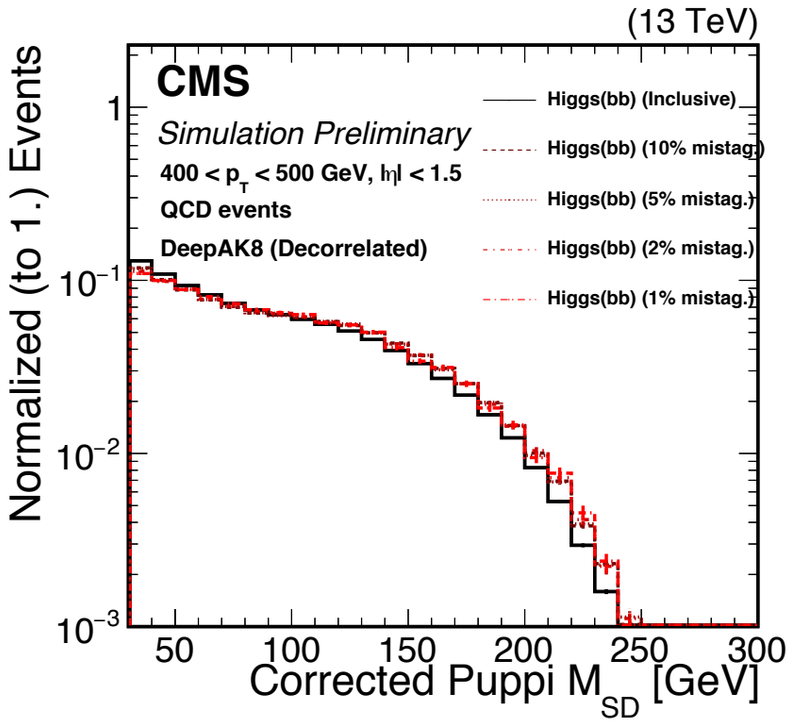


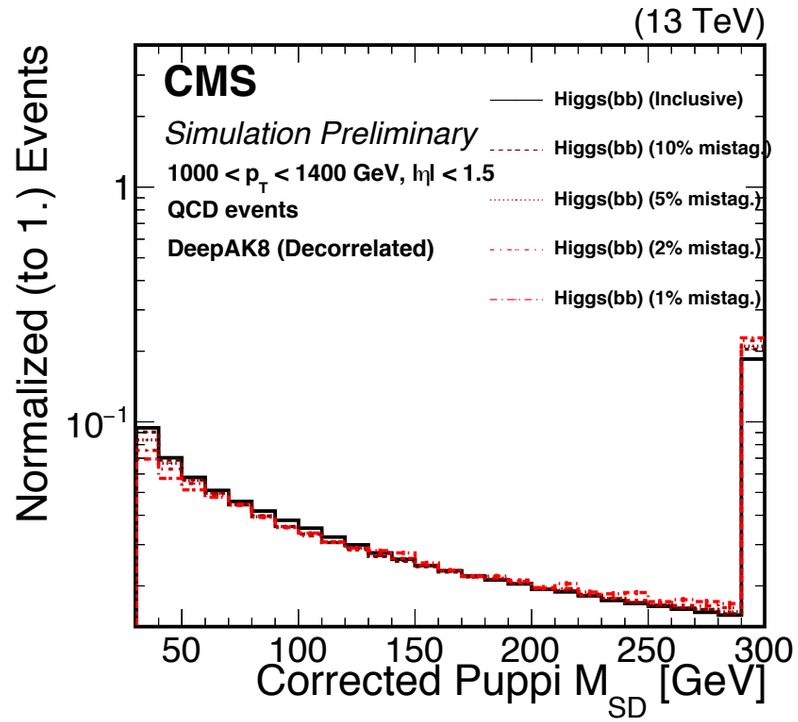


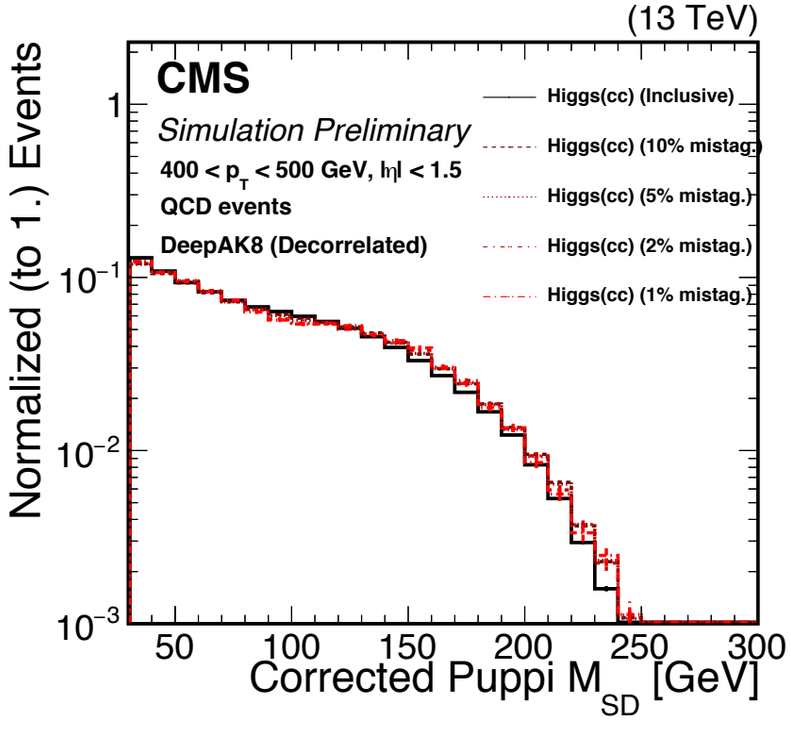


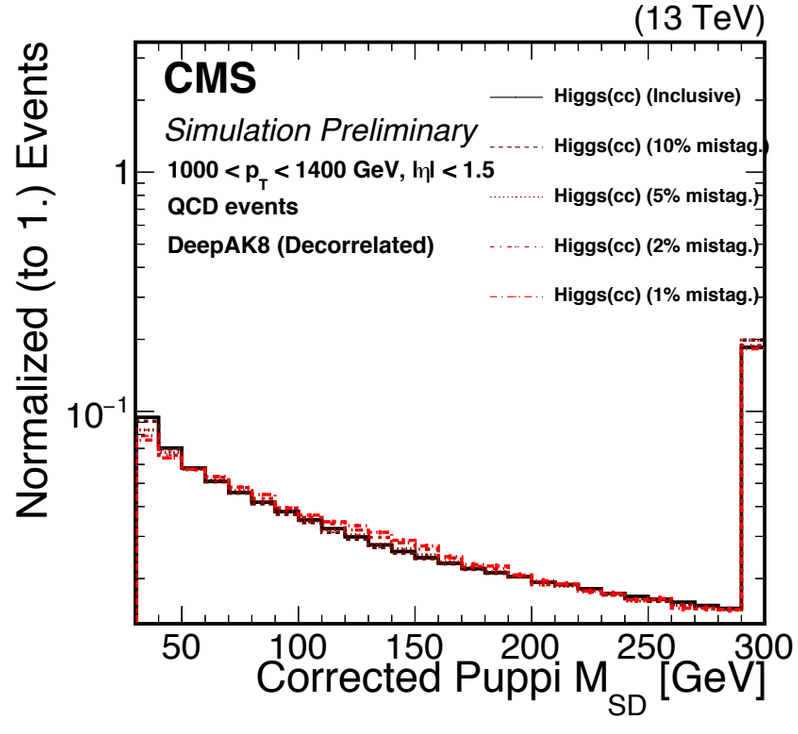


Z+H+QCD score (Mass)





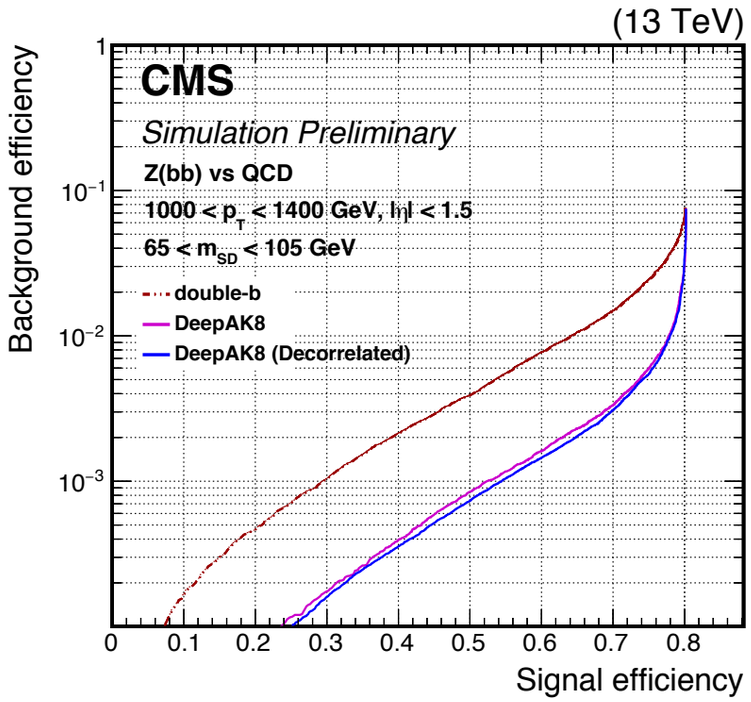


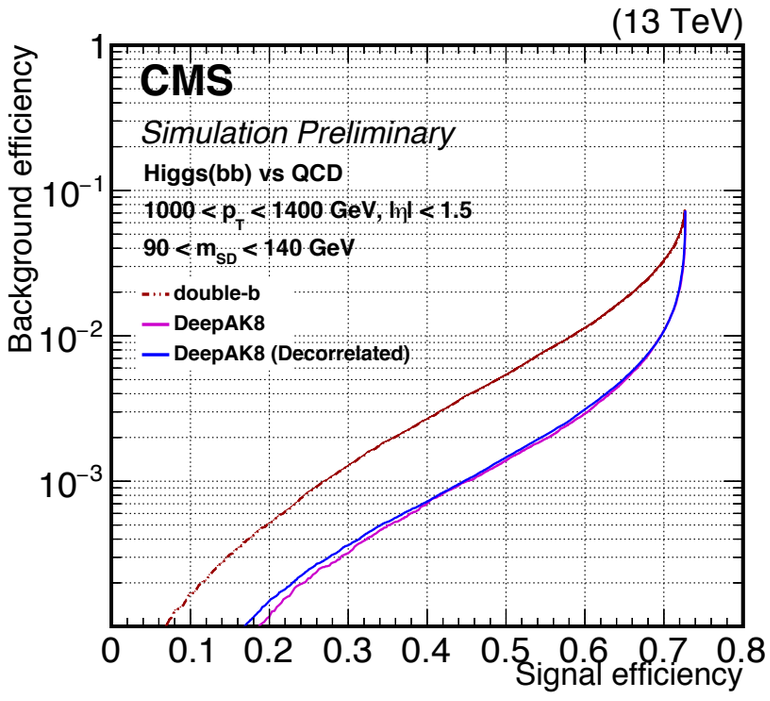


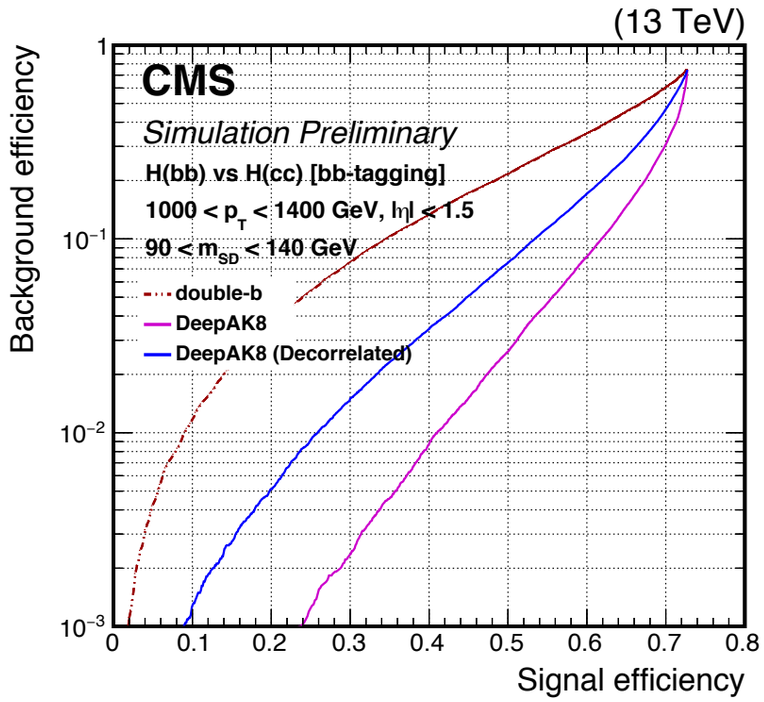
Z+H Scores

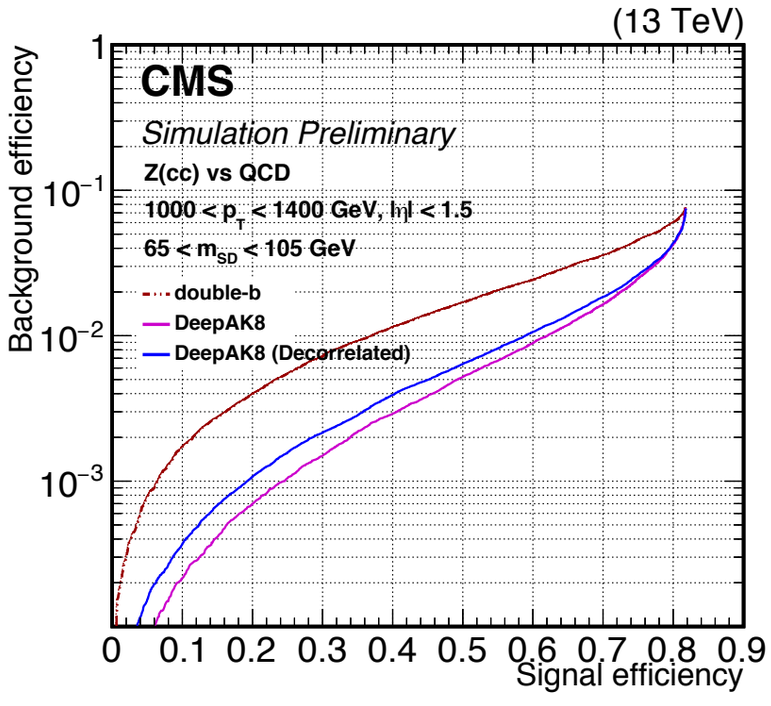
- Score definition:
  - score (bb) := (H(bb) + Z(bb)) / (H(bb) + Z(bb) + H(cc) + Z(cc) + QCD (all))
  - score (cc) := (H(cc) + Z(cc)) / (H(bb) + Z(bb) + H(cc) + Z(cc) + QCD (all))
  - where QCD (all) = QCD(bb) + QCD(cc) + QCD(b) + QCD(c) + QCD(others)
- Background:
  - QCD (all flavours including bb and cc)

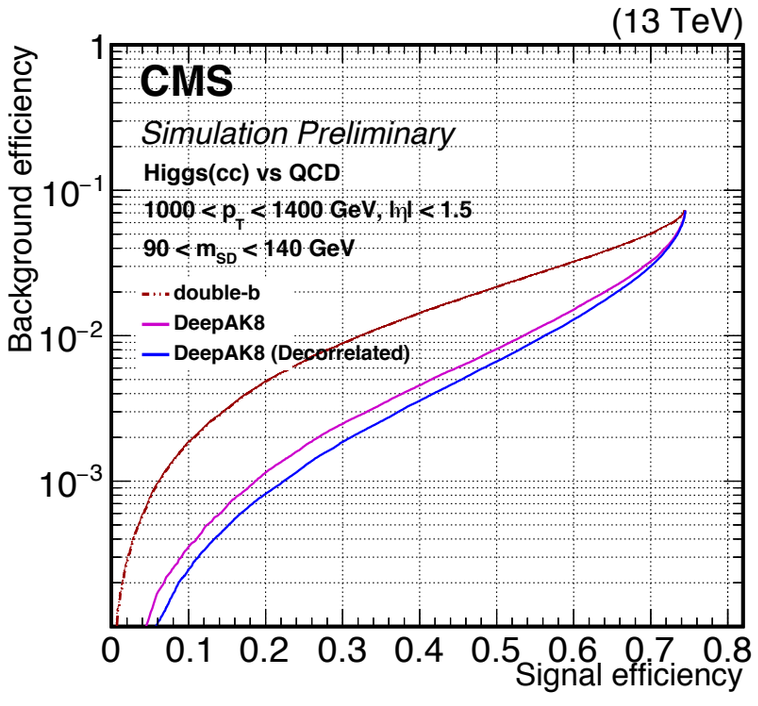
Z+H score (High Pt)

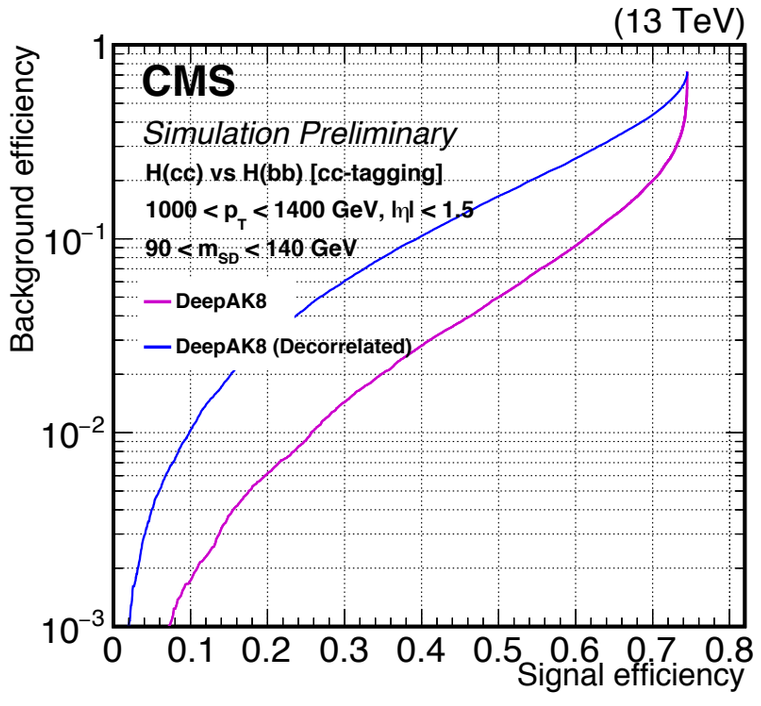




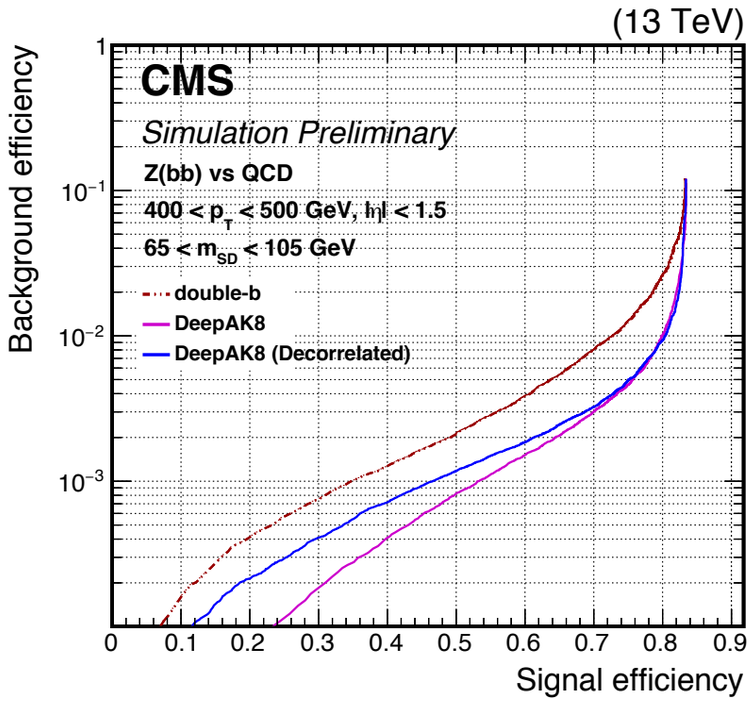


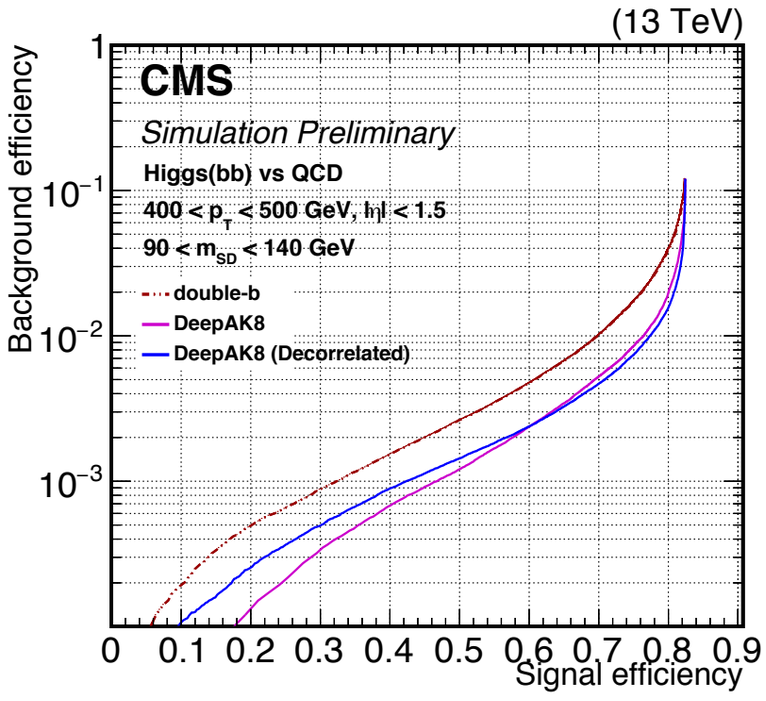


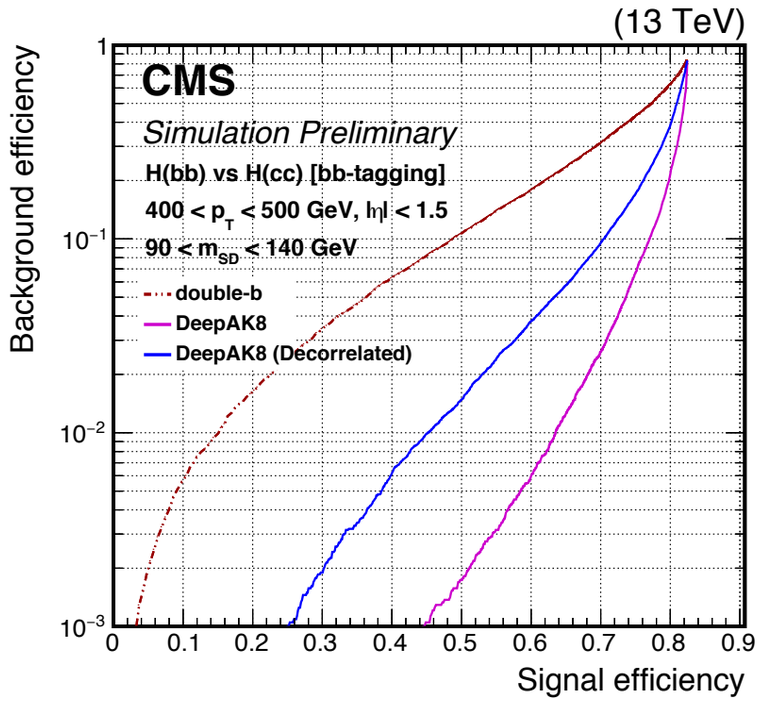


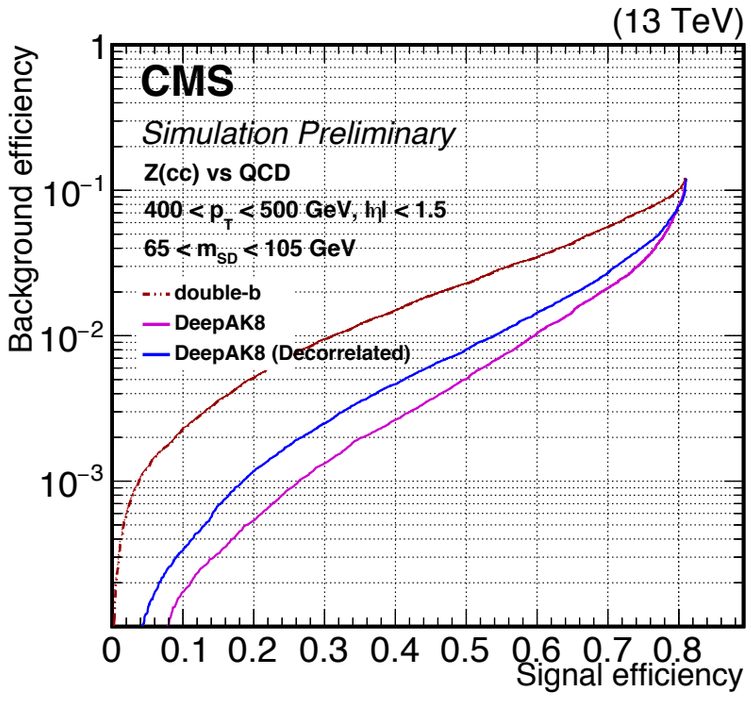


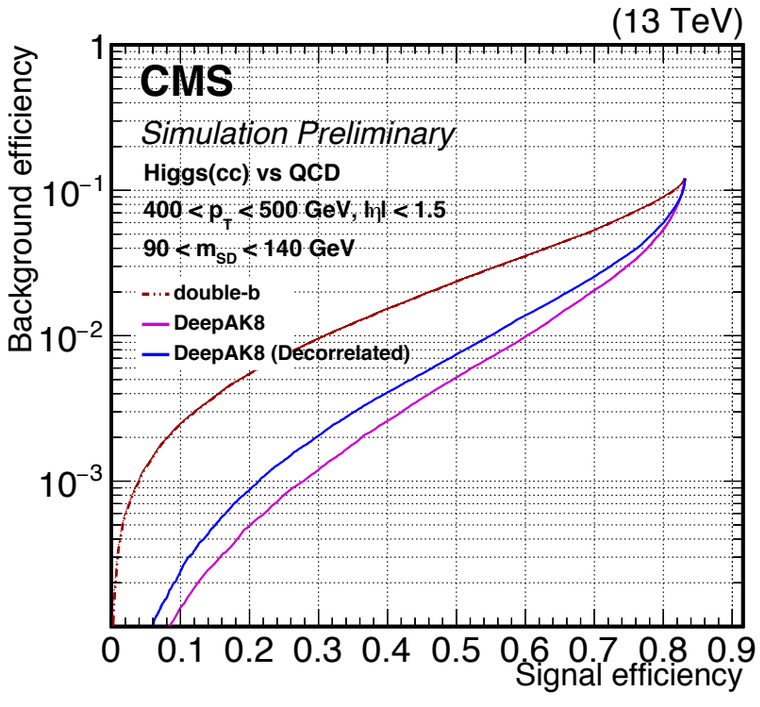
Z+H score (Low Pt)

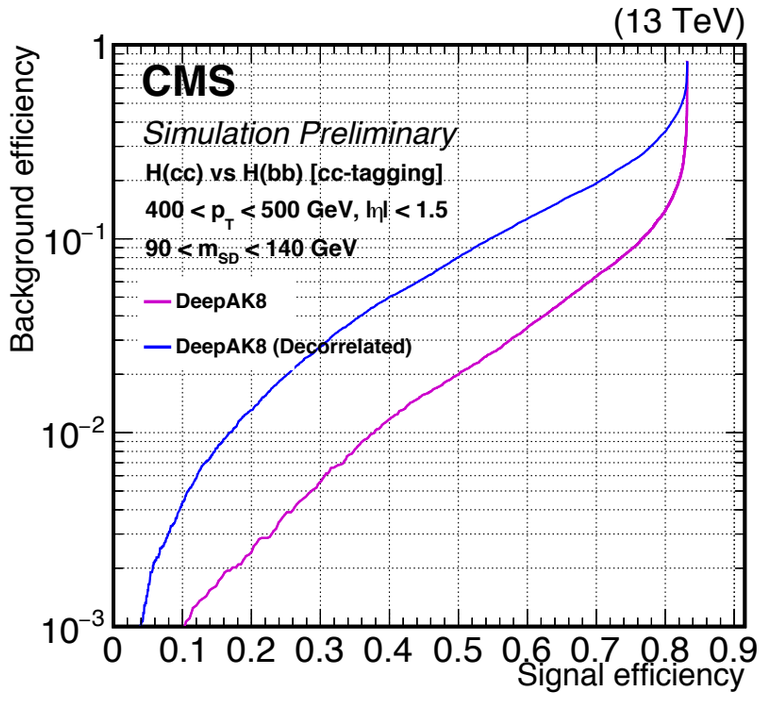












Z+H score (Mass)