DeepFlavour in CMS

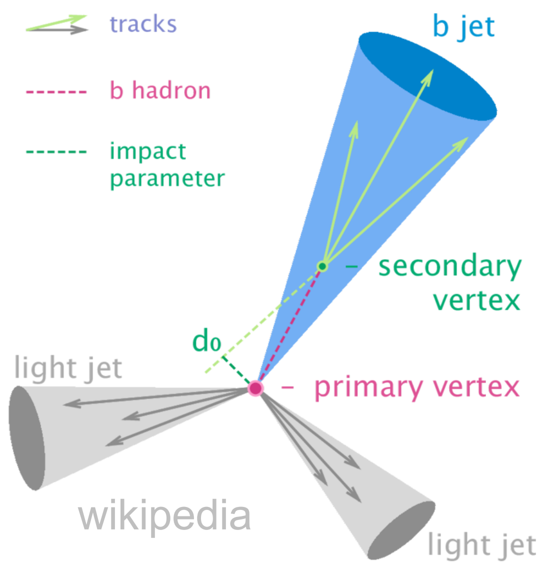
Markus Stoye[1,2] for the CMS collaboration

CERN[1], ITN aMVA4newphysics[2]

Heavy Flavour Tagging Features

Key features:

- Displaced tracks from longer lifetimes of heavy flavour jets
- Secondary verticies
- Eventually leptons in jets from W* in b → W*c or c → W*s
- Slightly wider jets
- …



Several complementary taggers in CMS using the above features

Jet-Flavour Taggers at CMS

Jet probability (btag):
- Likelihoods of *tracks* to be from PV

Soft lepton tagger (electron&muon) (btag):
- Muon and electron information
- NN

CSVv2 (btag):
- Combines information from secondary vertex and track information
- Combination of higher level features like masses of vertices and relatively raw information like significance of impact parameter per track.
- Shallow NNs + “likelihood method”

c-tagger:
- Uses CSV like variables and lepton information
- BDT

cMVAv2:
- BDT combines above b-taggers

PAS BTV-15-001 and BTV-15-002

New taggers “DeepFlavour”

DeepCSV:

- Multiclassification
- Include all CSVv2 features
- Additionally to CVSv2 few more “relatively raw” information, e.g. not only 2D impact parameter significance, but also it’s value, …
- More tracks than in CVSv2 used (up to 6).
- Deep Neural Network
- Lepton ID information not used to allow using them for validation in real data (thus Deep**CSV**)

DeepcMVA:

- I.e. soft lepton and JP taggers added to DeepCSV input
- Trained, but *not yet* validated in data

New CMS DP-2017/005

DeepCSV input features

(for detailed list of acronyms: BTV 15-001)

Per jet (sample):

```
['jet_pt', 'jet_eta','jetNSecondaryVertices', 'trackSumJetEtRatio',
'trackSumJetDeltaR','vertexCategory',
'trackSip2dValAboveCharm','trackSip2dSigAboveCharm',
'trackSip3dValAboveCharm', 'trackSip3dSigAboveCharm',
'jetNSelectedTracks','jetNTracksEtaRel']
```

Per 1st 6 tracks (impact parameter sorted, pre-selected):

```
['trackJetDistVal','trackPtRel','trackDeltaR','trackPtRatio','track
Sip3dSig','trackSip2dSig','trackDecayLenVal','TagVarCSV_trackEtaRel
']
```

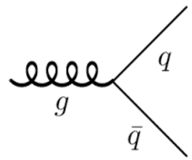
From 1st secondary vertex:

```
['vertexMass','vertexNTracks','vertexEnergyRatio','vertexJetDeltaR','fligh
tDistance2dVal','flightDistance2dSig','flightDistance3dVal','flightDistanc
e3dSig'],
```

- Red are on top of CSVv2
- All variables were set of b-tag commission ***before*** DeepCSV, i.e. tested/established features

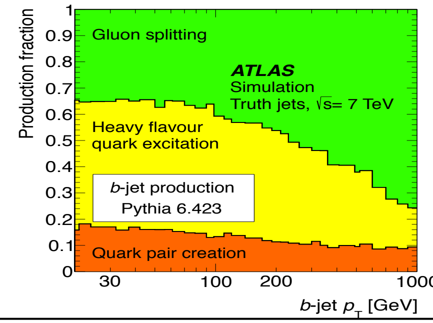
Multiclassification

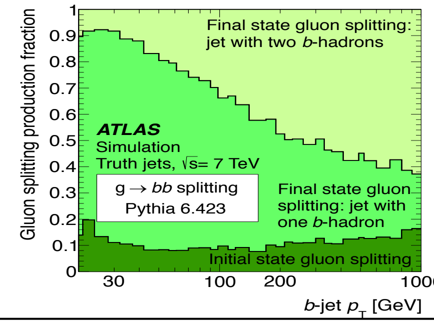
QCD:



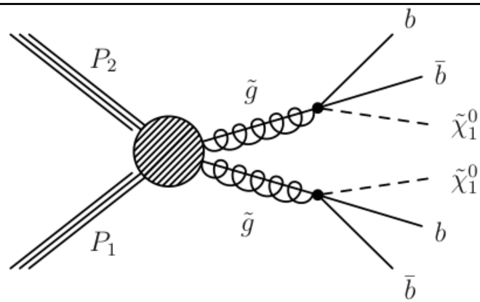
q = c or b

Often two b-hadron in a single AK4 jet for gluon splitting

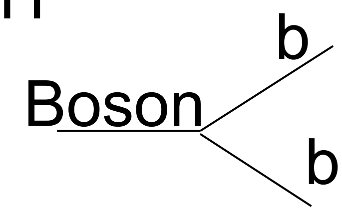




SUSY:
4 rather isotropic single bs



Very boosted H or Z:



Inspired by this we defined 5 ***exclusive*** categories:

- Exactly one b hadron in jet
- Exactly one c hadron, but no b-hadron in jet
- Two or more b hadrons in jet
- Two or more c hadrons, but no b-hadron in jet
- Light jets (udsg)

Training physics-process selection

Two aims:

- A generic tagger, use admixture of different processes that produce heavy flavour
- Robust tagger: train including realistic special cases, e.g. we do keep jets with accidental lepton overlap or alike

QCD:

- very clean, e.g. no accidental overlap of lepton from and jet
- Good source of gluon splitting sample, flavour excitation, flavour creation

ttbar:

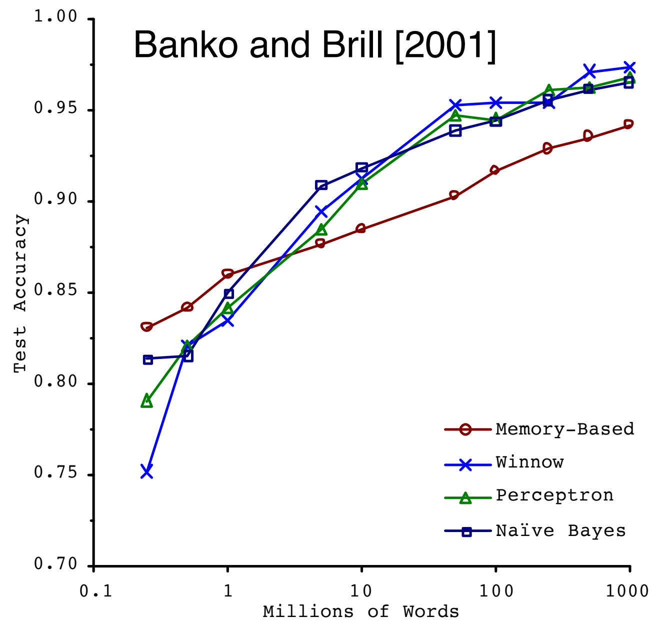
- Less clean, i.e. includes accidental overlap of leptons
- bs from top decay and cs from W.

Use QCD and ttbar for training

Training sample size

Of course "more data" not always helps, but sometimes it can.

- CMS has >10 billion jets simulated for 2016 conditions
- This is a ***massive*** number, i.e. in HEP we stand out by having relatively "*cheap"* data.
- In DeepCSV we use about **50M** jets, which is 25 x more sample/feature than e.g. arxiv:1607.08633
- Producing e.g. well 250M well labeled jets (e.g. 50M ttbar events) not a big deal!
- Used 0.5:1:2 ratio for c:b:udsg to have good statistics in each class
- Flattened PT/eta shape up to GeV and than used PT/eta shape of bs



➢ We are generally able to use huge sample training datasets
➢ For DeepCSV 50M were used

DeepCVS Deep Neural Network

Arguments for DNN:

- Good classification performance
- Application speed (will be applied billions of times)
- Scalability for future studies

DNN details:

- 66 input features
- 4 hidden layers with 100 nodes each
- Relu activation
- Softmax activation for last layer
- Loss: x-entropy
- Learning rate 0.0003
- Adam optimizer
- 500 epochs
- dropout

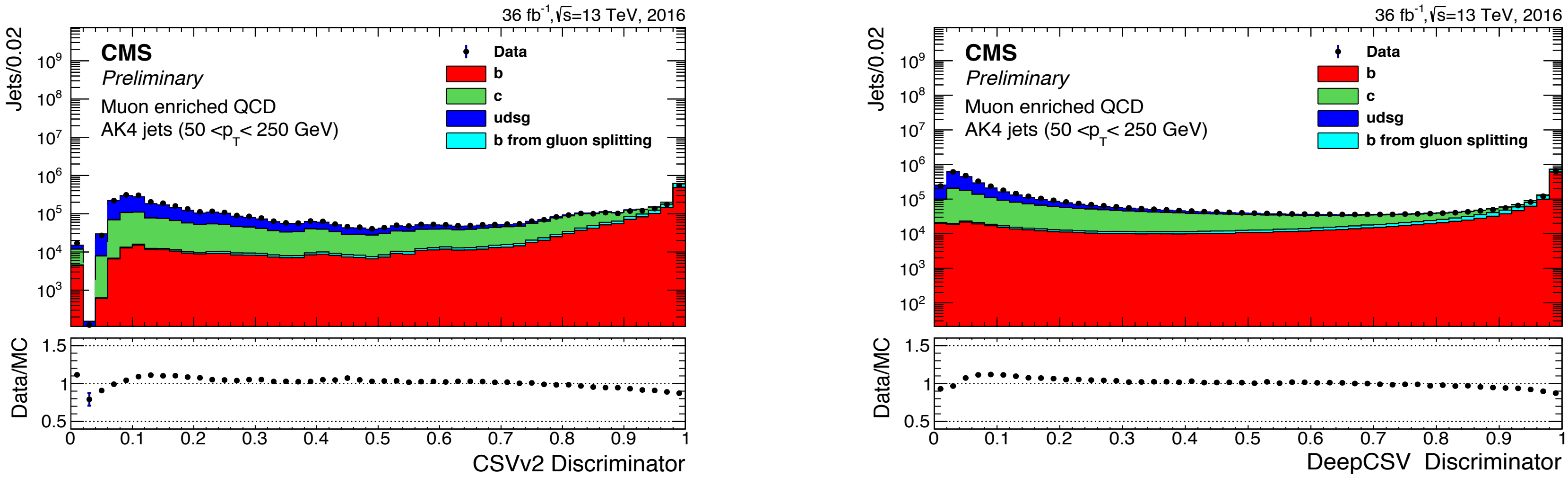
Relatively simple DNN structure lead to good results

ML tools used

- Compressed data-format “miniAOD” of CMS used
- Bare root-tupels from CMSSW converted to python (root_numpy)
- Preprocessing (zero-padding, mean subtraction, PT/eta flattening of classes, …) mostly python
- Pure Tensorflow and Keras with Tensorflow as backend was used for training studies
- LWTNN (pure C++) used to implement DNN in CMSSW

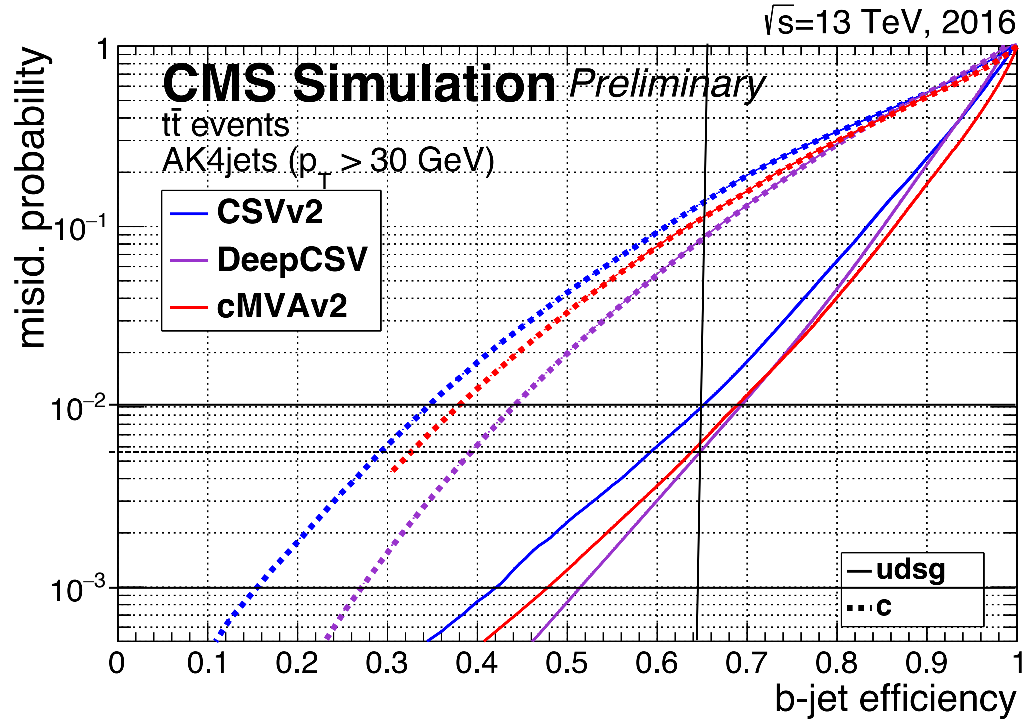
➢ Separated training an application tools
➢ LWTNN presented at IML (from UCI).
➢ Used tools widely spread outside HEP

Discriminators p(b)+p(bb)



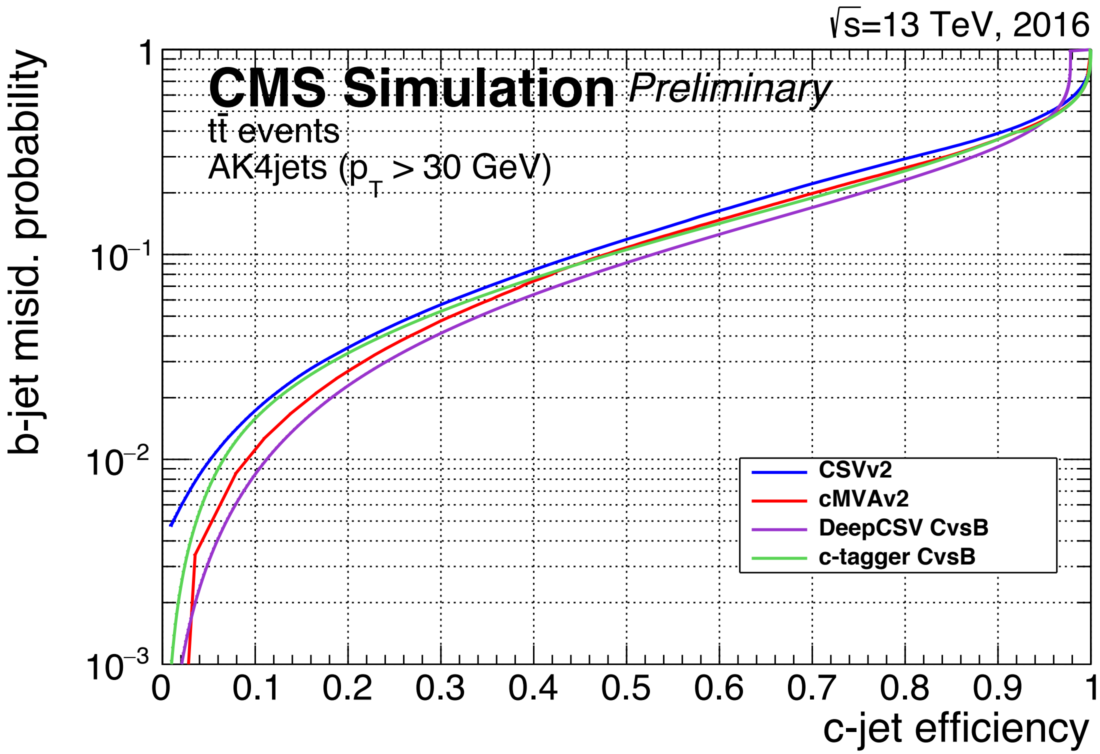
- We use the probability to have at least one b-hadron in the jet as discriminator for default b-tagging, i.e. p(b)+p(bb).
- Events without any *pre-selected* track are put first bin (underflow) for DeepCSV
- DeepCSV has a very smooth distribution
- CVSv2 and DeepCSV similar trends

ROC b-jet vs. light and c-jet



- DeepCSV 40% smaller fake (0.6%) rate at same b efficiency as medium WP CSVv2
- **20%** relative (10% absolute) better efficiency for 0.1% misid. probability.

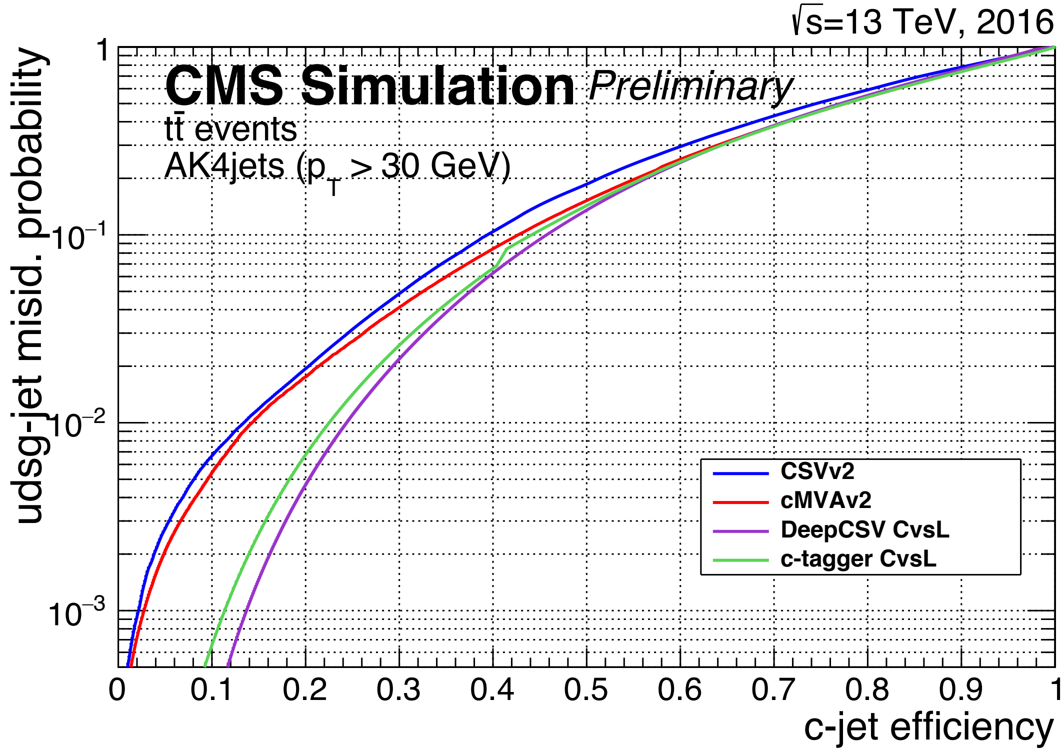
ROC for c vs b



$$\text{Discr.} = \frac{p(c)+p(cc)}{1-p(udsg)}$$

- Better performance than c-tagger
- Note, the c-tagger uses some lepton information
- DeepCSV more stringent in not accepting jets, thus less close to 1 (no track events).

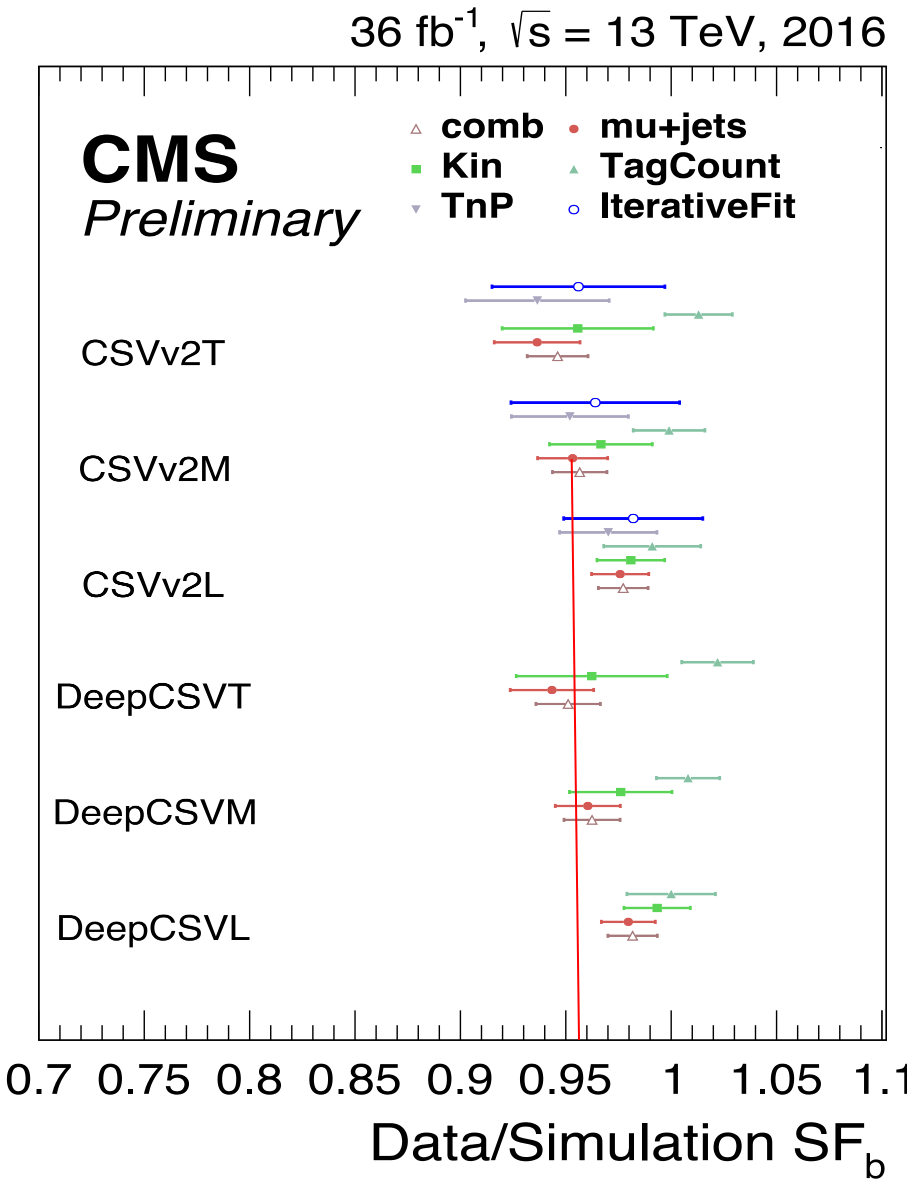
ROC c vs. light



$$\text{Discr.} = \frac{p(c)+p(cc)}{1-p(b)-p(bb)}$$

Slight improvement w.r.t. c-tagger

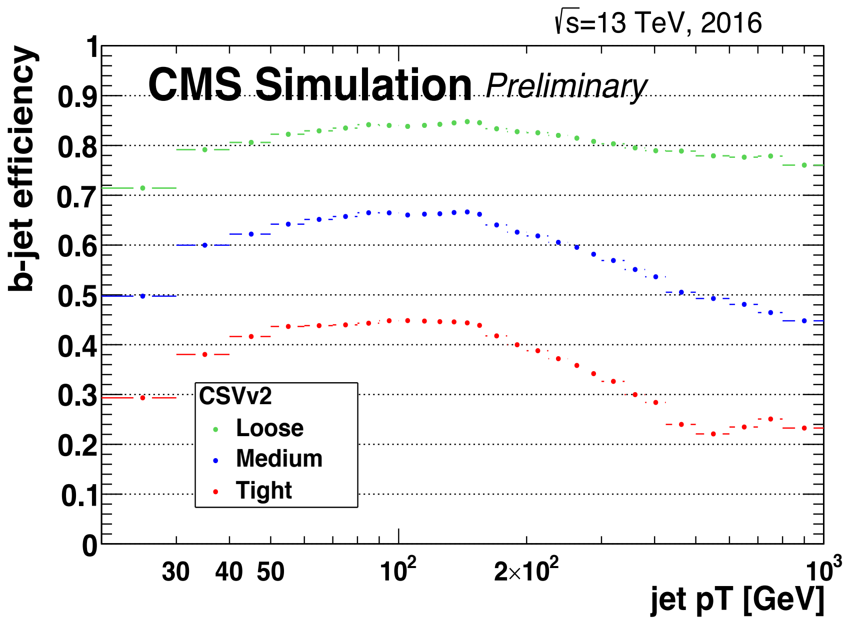
Performance in real data

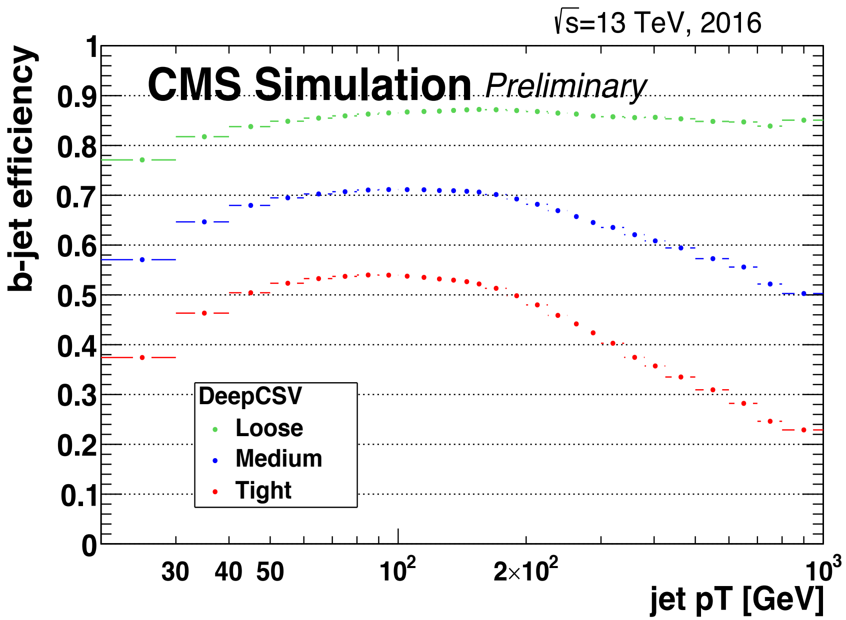


- Data/simulation agreement same within uncertainties
- Central values slightly better data/MC agreement for DeepCSV

Improvement by revisited ML strategy confirmed in real data

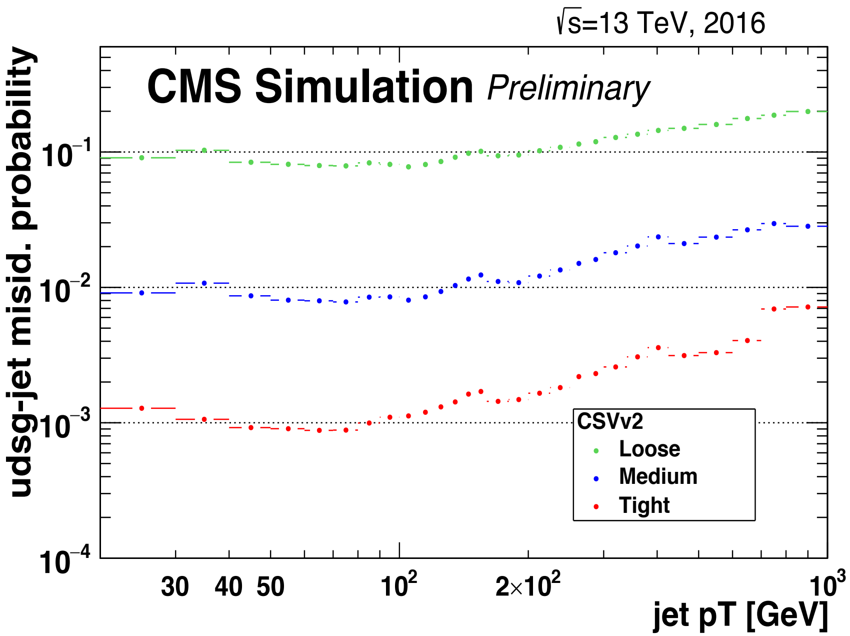
Efficiency as function of P_T

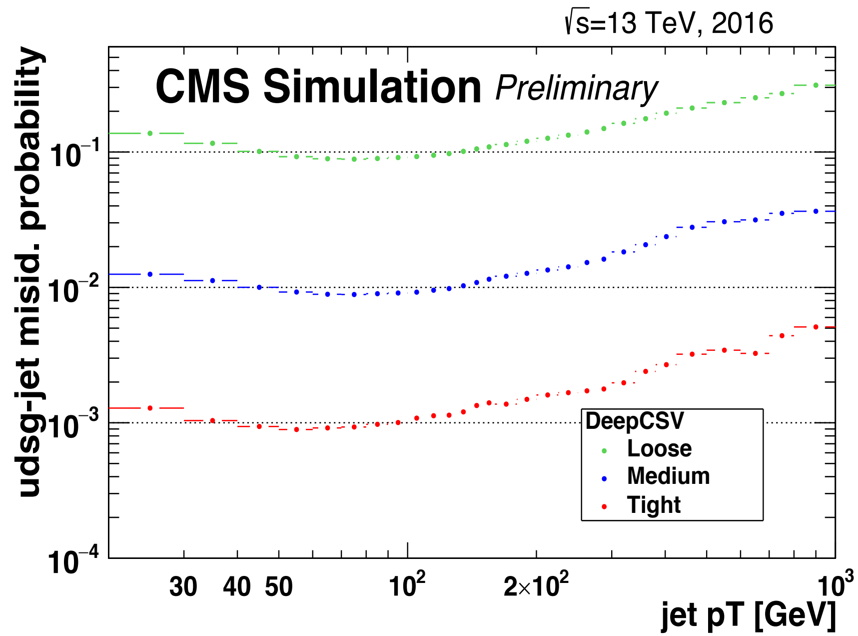




- ttbar sample used for evaluation
- DeepCSV same trends as CSVv2
- Easiest region for tagging between 50-200 GeV

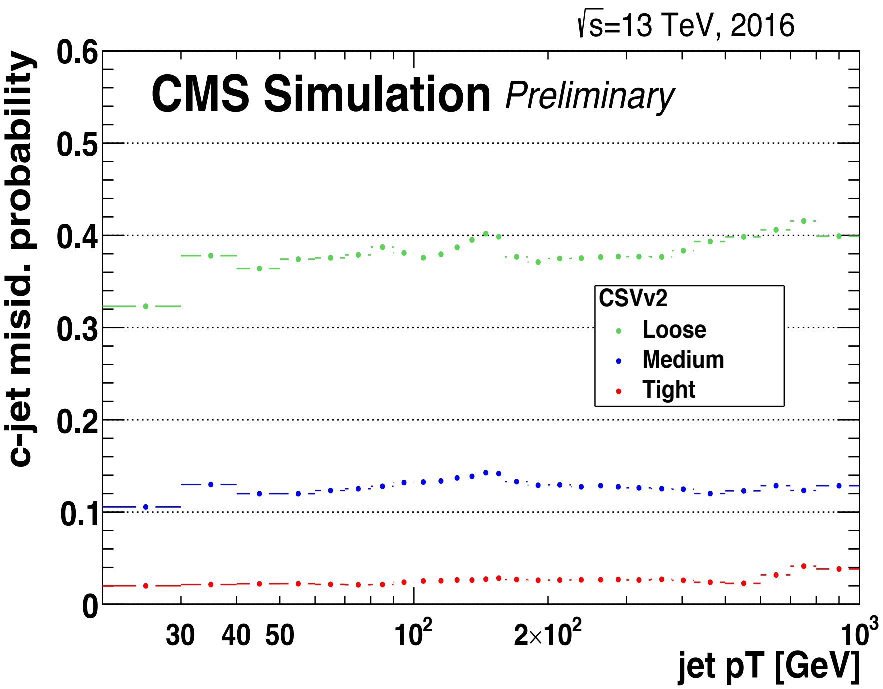
Light-jet misid prob. as a function of p_T

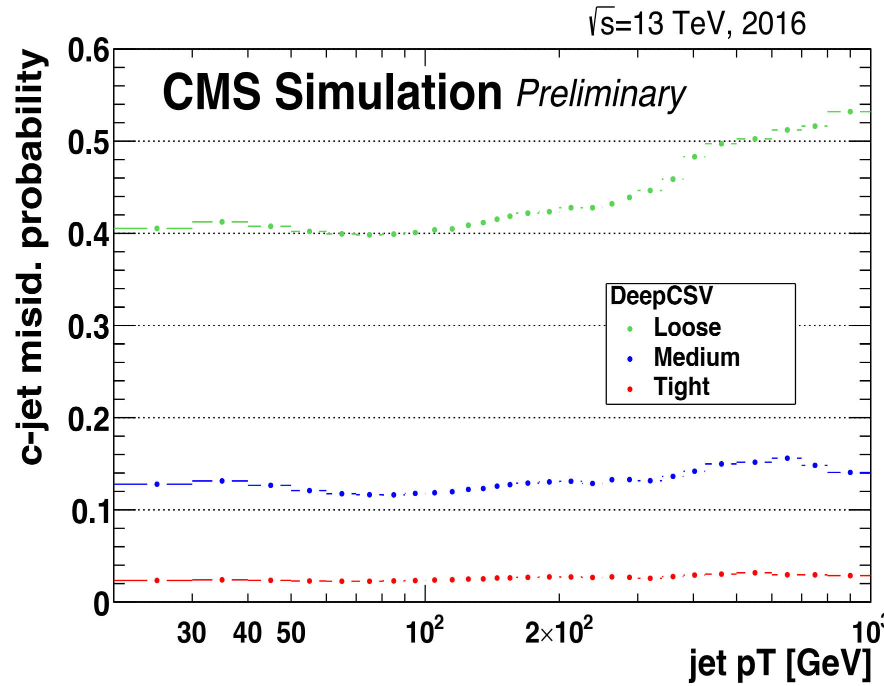




- Note, Working points defined (e.g. 1% mistag rate) in QCD sample with P 80-120, and not *ttbar* as shown
- DeepCSV same trends as CSVv2
- Increasing mistag rate at high P_T

Light-jet misid prob. as a function of p_T



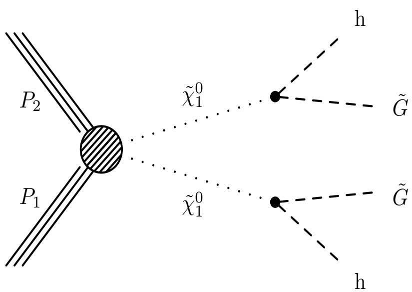


- Slightly stronger trend of c-jet rejection degrading with P_T
- For medium WP good c-jet rejection for DeepCSV

Application in physics analysis

SUS-16-044:

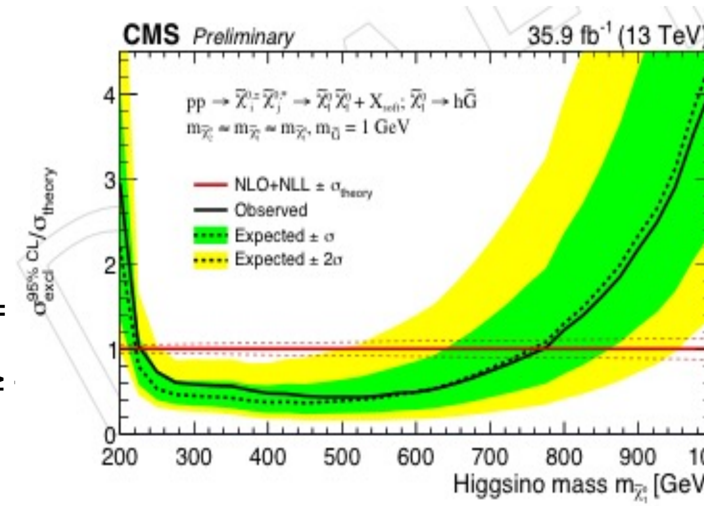
Search for events with two h->bb and MET



$2b \equiv N_{b,T} = 2, N_{b,M} = 2$

$3b \equiv N_{b,T} \geq 2, N_{b,M} = 3, N_{b,L} =$

$4b \equiv N_{b,T} \geq 2, N_{b,M} \geq 3, N_{b,L} \geq$



CSVv2 $\mathcal{L} = 35.9\ \text{fb}^{-1}$	All SM bkg.	TChiHH (225,1)	TChiHH (700,1)
$\geq$ 2b	–	3761.5	33.7
$\geq$ 3b	–	1999.1	19.0
4b	–	860.0	9.3
Baseline, $\geq$ 2b	2600.1±101.0	75.6	7.7
Baseline, $\geq$ 3b	276.9±5.5	49.6	5.4
Baseline, 4b	72.2±4.1	30.9	3.6
Baseline, $p_T^{miss} > 300$, $\geq$ 2b	104.2±2.4	2.8	6.0
Baseline, $p_T^{miss} > 300$, $\geq$ 3b	12.9±0.8	2.4	4.2
Baseline, $p_T^{miss} > 300$, 4b	4.0±0.4	1.7	2.8

DeepCSV $\mathcal{L} = 35.9\ \text{fb}^{-1}$	All SM bkg.	TChiHH (225,1)	TChiHH (700,1)
$\geq$ 2b	–	4625.6	39.7
$\geq$ 3b	–	2548.7	24.1
4b	–	1149.1	12.7
Baseline, $\geq$ 2b	3650.5±90.2	95.1	9.9
Baseline, $\geq$ 3b	385.2±9.0	68.6	7.4
Baseline, 4b	94.3±5.3	43.4	5.1
Baseline, $p_T^{miss} > 300$, $\geq$ 2b	144.8±2.8	4.0	7.7
Baseline, $p_T^{miss} > 300$, $\geq$ 3b	16.3±0.8	3.2	5.7
Baseline, $p_T^{miss} > 300$, 4b	4.6±0.4	2.5	4.0

- E.g. last row, 15% more background and up to ~ 50% more signal
- Significantly improved limit (150 GeV in Higgsino mass)

Conclusions

New tagger DeepCSV in CMS:

- More “relatively raw” input features used than before
- Adapted training strategy that includes large training dataset and two processes, ttbar and QCD
- Use Deep Neural Network for training.
- New tagger outperformed existing b and c-taggers
- Improvements confirmed in data
- First analysis used this tagger (more in the pipeline)
- Multiclassification (b,bb,c,cc,udsg) is lean to maintain and allows in future usage e.g. gluon->bb splitting tagging or similar applications
- Step towards exploring more deep-learning in CMS