This Report is part of a project that has received funding from the **European Union's Horizon 2020 research and innovation programme under grant agreement N°675440**

Deep learning in jet reconstruction at CMS

Markus Stoye[1,2] for the CMS collaboration

CERN[1], ITN aMVA4newphysics[2]

ACAT, 22nd August 2017

Content

DeepCSV:

- Heavy-flavor jet-tagger with human engineered variables and track pre-selections for heavy flavor tagging

DeepJet:

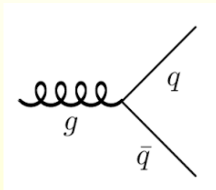
- Jet-tagger (heavy flavor, quark/gluon) using more raw information from jet constituent

Data/MC

- Some new strategy proposals (not CMS)

DeepCSV

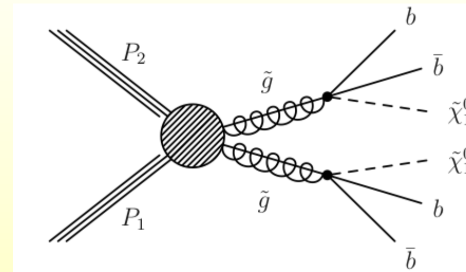
Jet flavour tagging

QCD: 

q = c or b

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

SUSY:
4 bs in separate jets



Inspired by this we defined 4 ***exclusive*** categories:

- Exactly **one b hadron** in the jet
- Exactly **one c hadron**, with no b-hadron in the jet
- **Two** or more **b hadrons** in jet
- Light quark/gluon jets (udsg)

→ Jet flavour tagging is ***intrinsically*** a multi-class classification problem

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(4) tracks (impact parameter sorted, pre-selected):

['trackJetDistVal','trackPtRel','trackDeltaR','trackPtRatio','trackSip3dSig','trackSip2dSig',' trackDecayLenVal','TagVarCSV_trackEtaRel']

From 1st secondary vertex:

['vertexMass','vertexNTracks','vertexEnergyRatio','vertexJetDeltaR','flightDistance2dVal','flightDistanc e2dSig','flightDistance3dVal','flightDistance3dSig'],

- Same variables used for the former standard CMS tagger "CSVv2"
- Red are changes with respect to CSVv2, i.e. DeepCSV uses slightly (factor 2) more

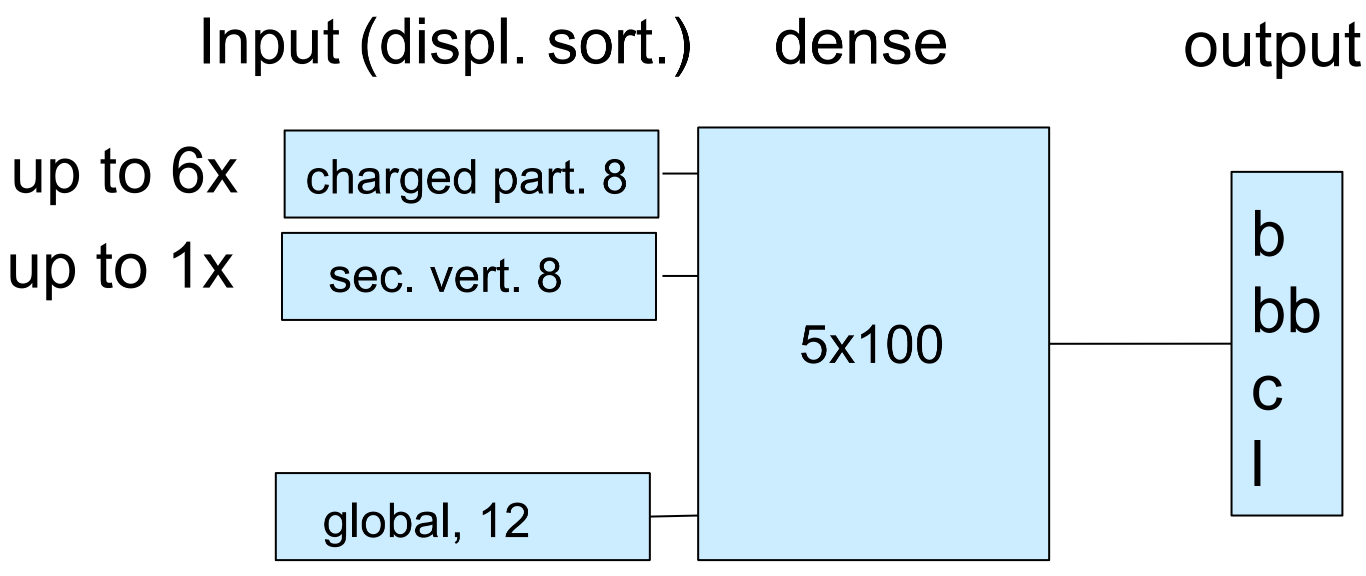
Training Stratgy

Three 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
- Optimal training: Large jets sample to avoid over-fitting

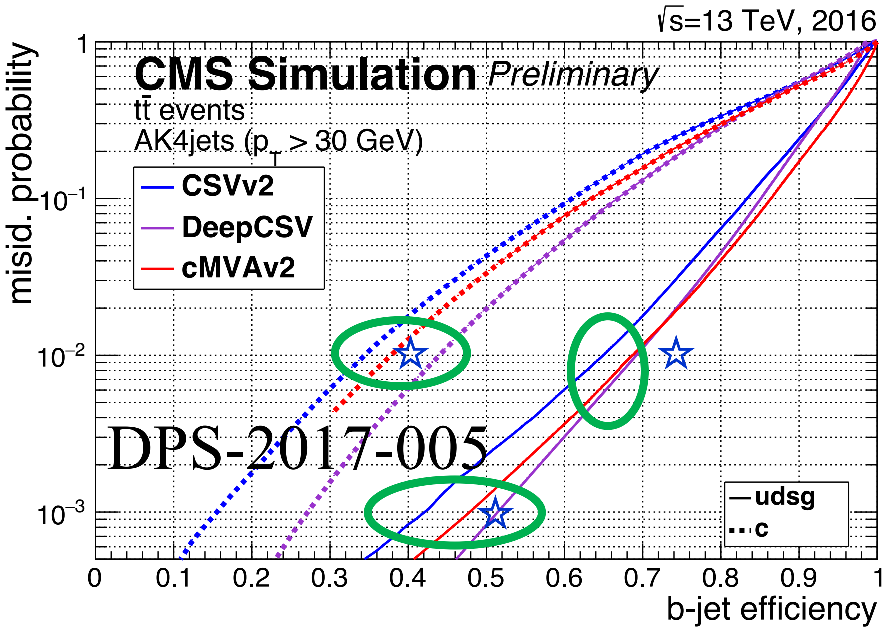
- QCD and tt for training
- 50M jets!

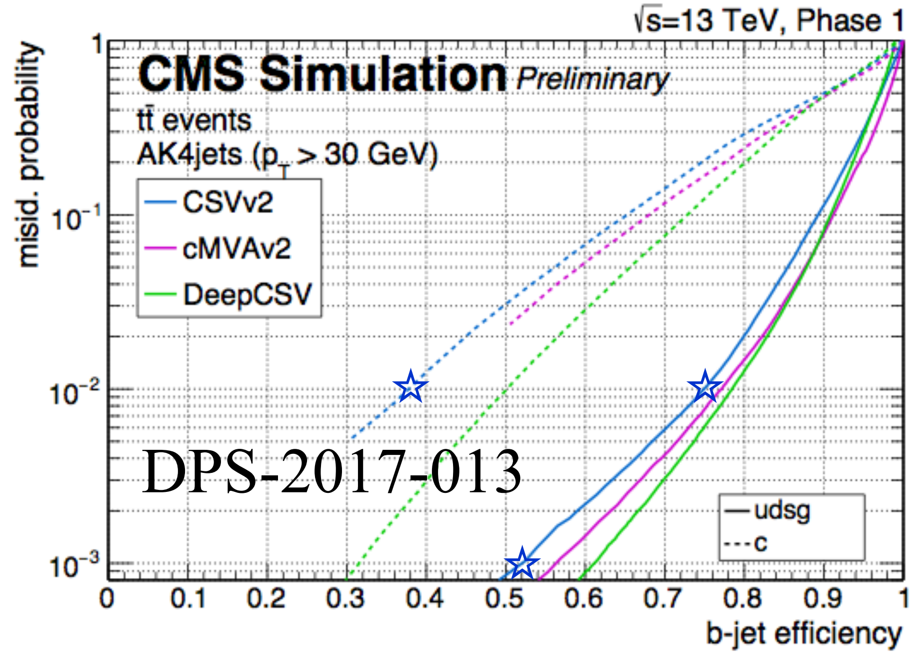
DeepCSV: DNN details



Good results achieved with relative dense DNN structure

Performance

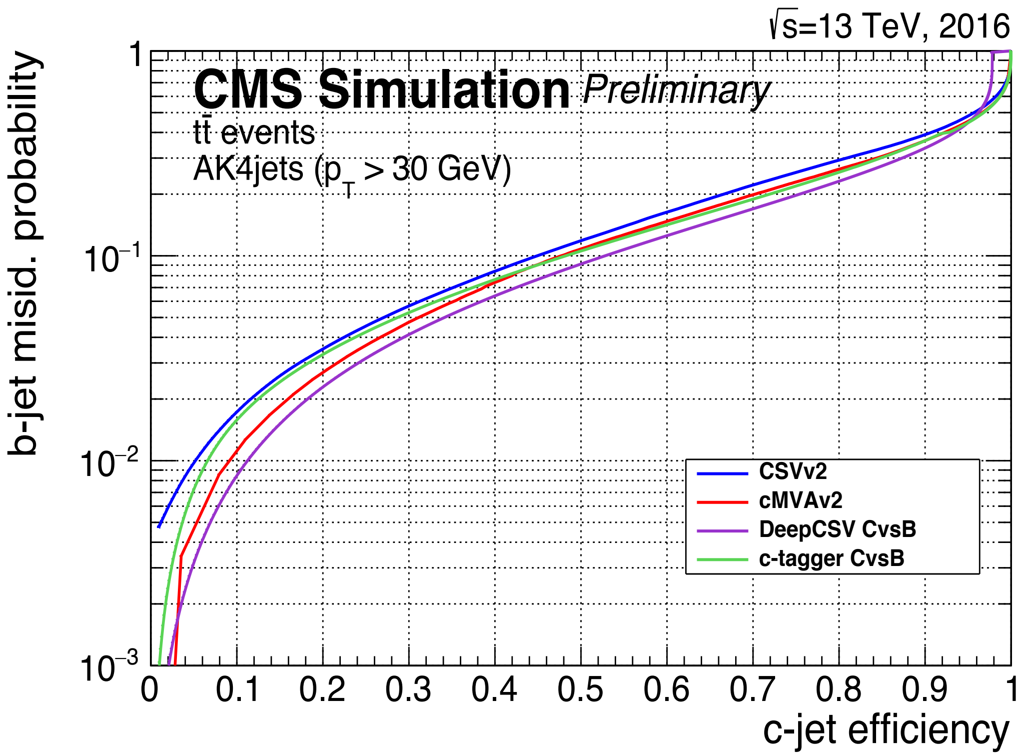




- DeepCSV does not use muons to allow for more validation option. With muons (as in cMVA) performance improves further
- At **0.1%** light fake rate, **20%** (40%→50%) more efficient for b-jets
- At **1%** CSVv2 b-jet efficiency **40% less fake** rate
- For **0.1%** fake rate: new pixel and new software lead to ***same gain***, for **1%** fake rate, software **gain 40%** of new hardware and more than hardware for b vs. c-jets

Big performance improvements

ROC for c vs b

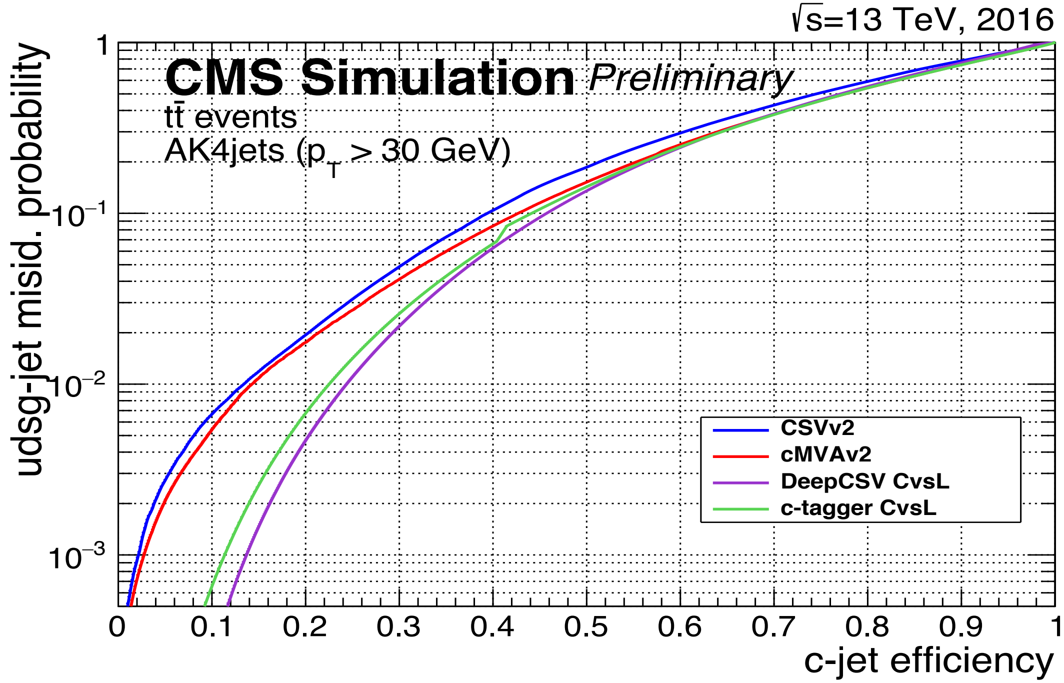


- Default c-tagger is ***binary***, i.e. trained on b vs c only.
- DeepCSV can be made binary (p'(b)+p'(c)=1) after the training by:

$$p'(c) = \frac{p(c)}{p(c)+p(b)}$$

- DeepCSV best c-tag performance
- Note, the c-tagger uses some lepton information

ROC c vs. light



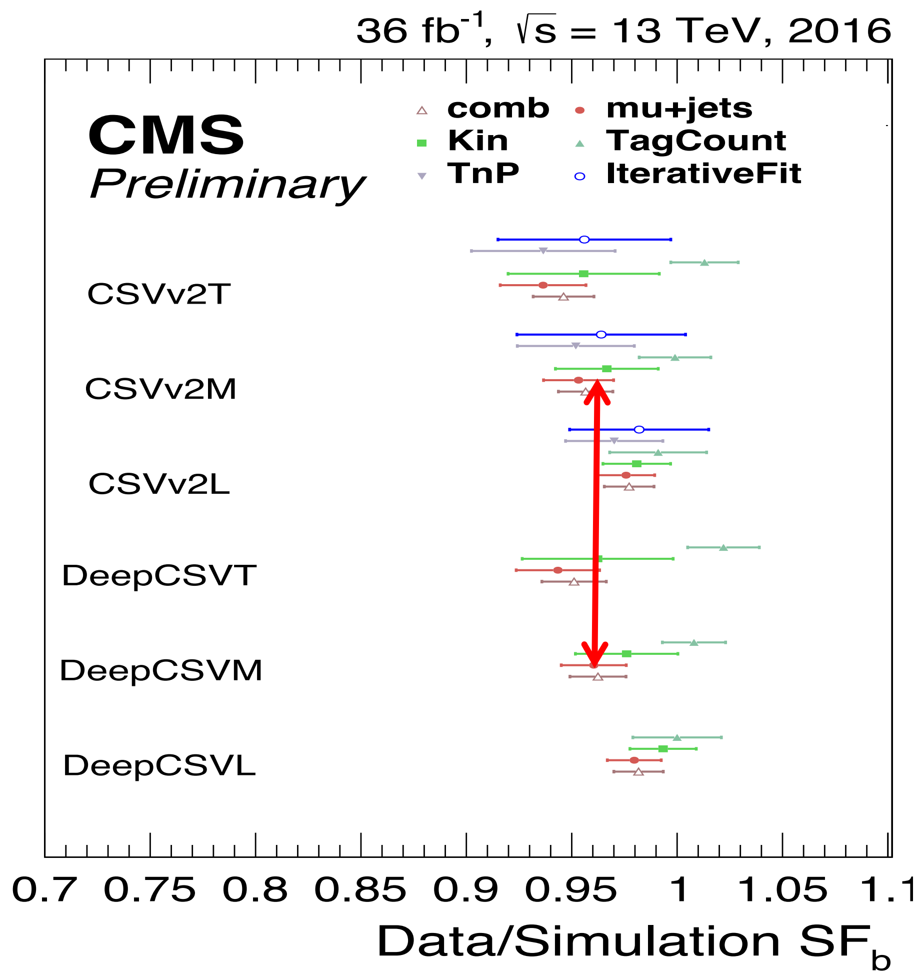
$$p'(c) = \frac{p(c)}{p(c)+p(udgs)}$$

Three CMS taggers become one:

- **CSVv2** (b vs. light&c with given c/light ratio), **c-tag** (c vs light) and **c-tag** (c vs b) → **DeepCSV** multi-class tagger

Tagging simplified by single tagger

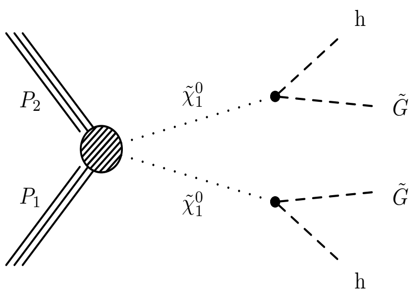
Performance in real data



Save data/simulation agreement

Application in physics analysis

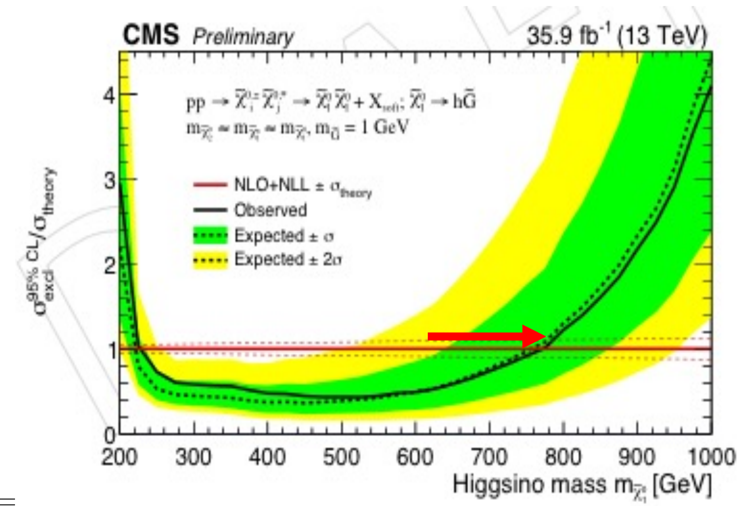
SUS-16-044:
Search for events with two h->bb and MET



2b ≡ $N_{b,T} = 2$, $N_{b,M} = 2$

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

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



CSVv2 $\mathcal{L} = 35.9$ 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 $\mathcal{L} = 35.9$ 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	3.3	5.7
Baseline, $p_T^{miss} > 300$, 4b	4.6±0.4	2.5	4.0

Significant Improvement: e.g. up to ~50% more signal for 15% more bkg
→ Significantly improved lower mass limit (150 GeV in Higgsino mass)

DeepFlavour

DeepFlavour & DeepJet

Include more raw data (DeepFlavour):

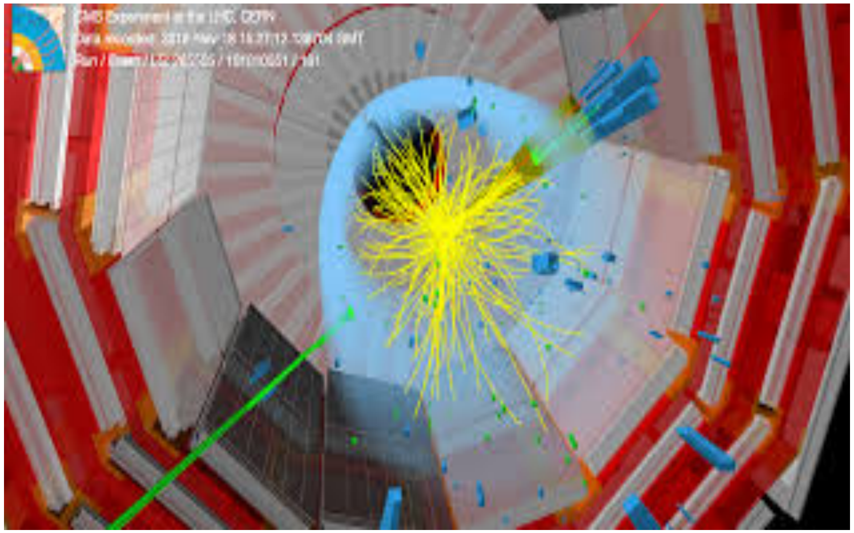
- Collect relevant information of jet constituents, both designed features for tagging and more raw features
- Design a custom DNN structure that is able to deal with the large input by using domain knowledge

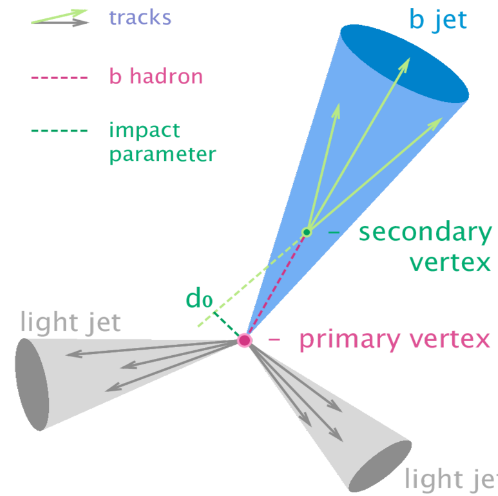
Include more classes, regression (DeepJet):

- quark flavours, quark/gluon, jet p_T, …
  - + Correlations between tagging of different IDs and even p_T are taken correctly into account
  - + Multi-class more flexible and takes correlations of inputs and outputs into account.

- NN can do multi-class classification easily, simpler than training many binary classifiers and combining these

Inputs from jets



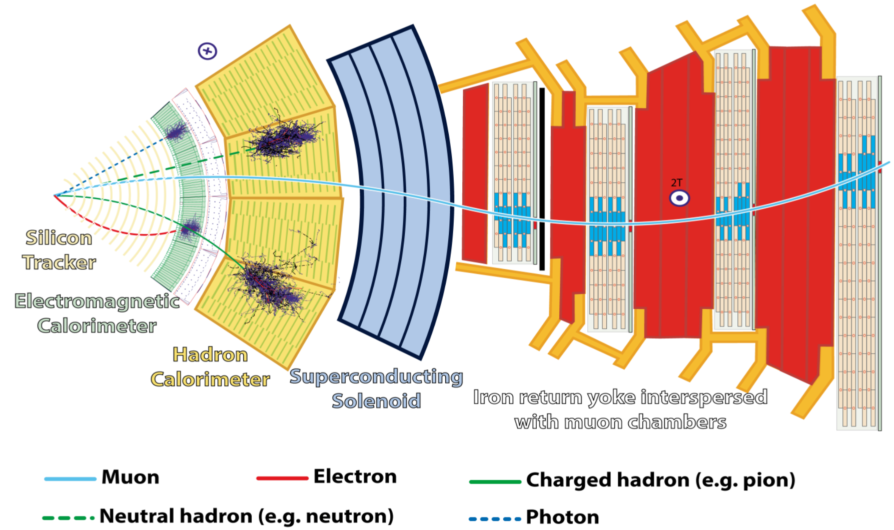


- All Particles candidates (separate for charged and neutral) of a jet
- Secondary vertices' of inclusive vertex finder in jet cone
- We sort these the above by displacement or (if not displaced) by P_T

Extract features from particles/vertices

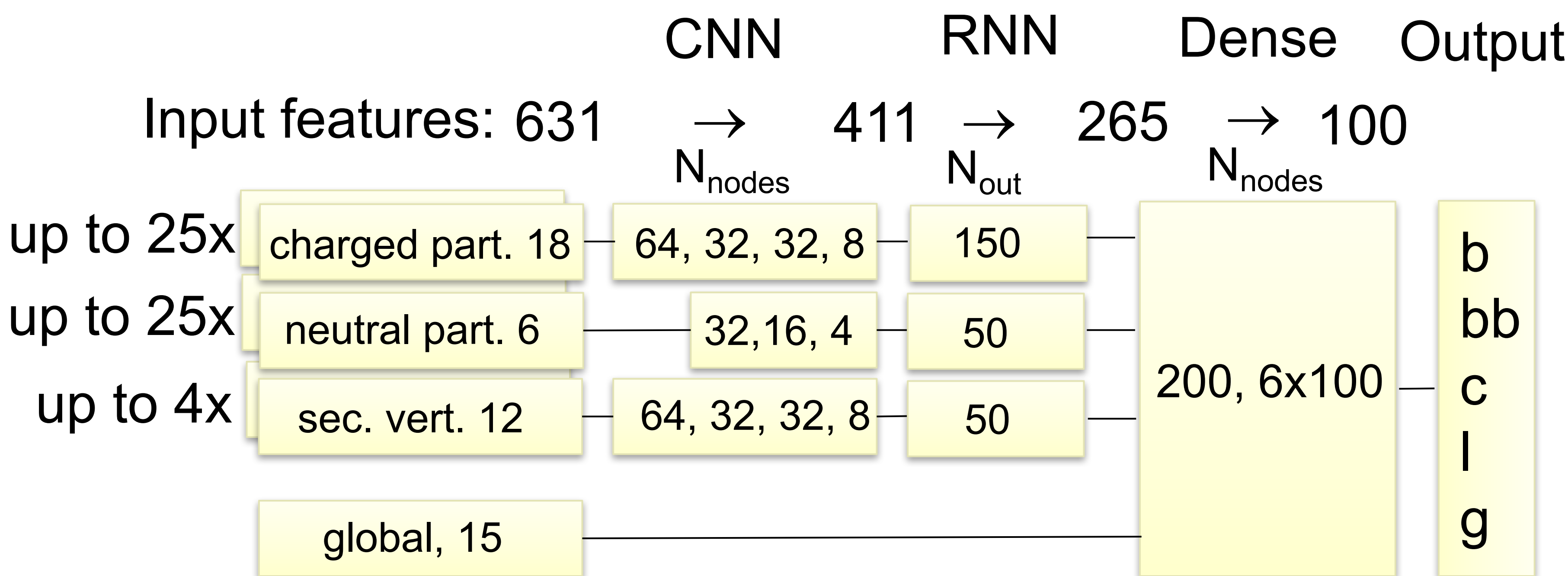
Input from each particle:

- High level features:
  - CSV variables
- Lower level features:
  - like χ^2



- Secondary vertices and particles are feed through multiple convolutional layers
- These layers transform the large input of “likely useful” features of a particle to a smaller set of features optimal to minimize the loss
- Scales well with adding more features per particles, because it hardly increase complexity of model

Generic custom DNN for jets



~250.000 model parameters

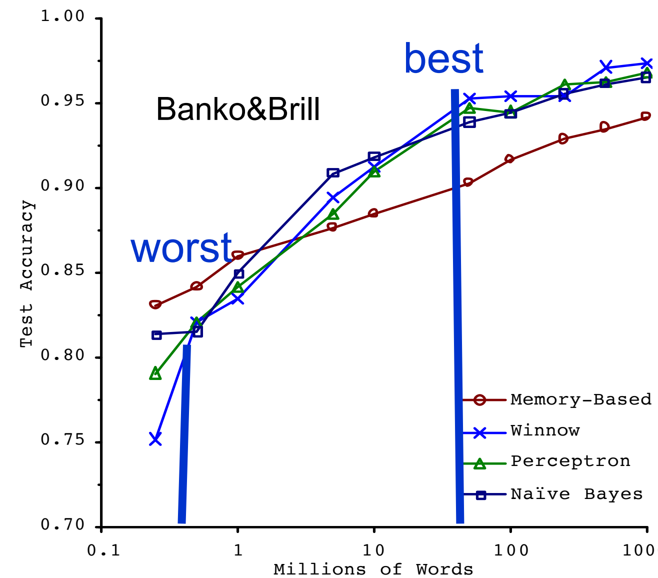
- The RNN (LSTM) layer improved the performance ~ 1%
- Key ***new*** element are the multilayer DNN (CNN) on particle level

Training sample

80M tt and QCD jets

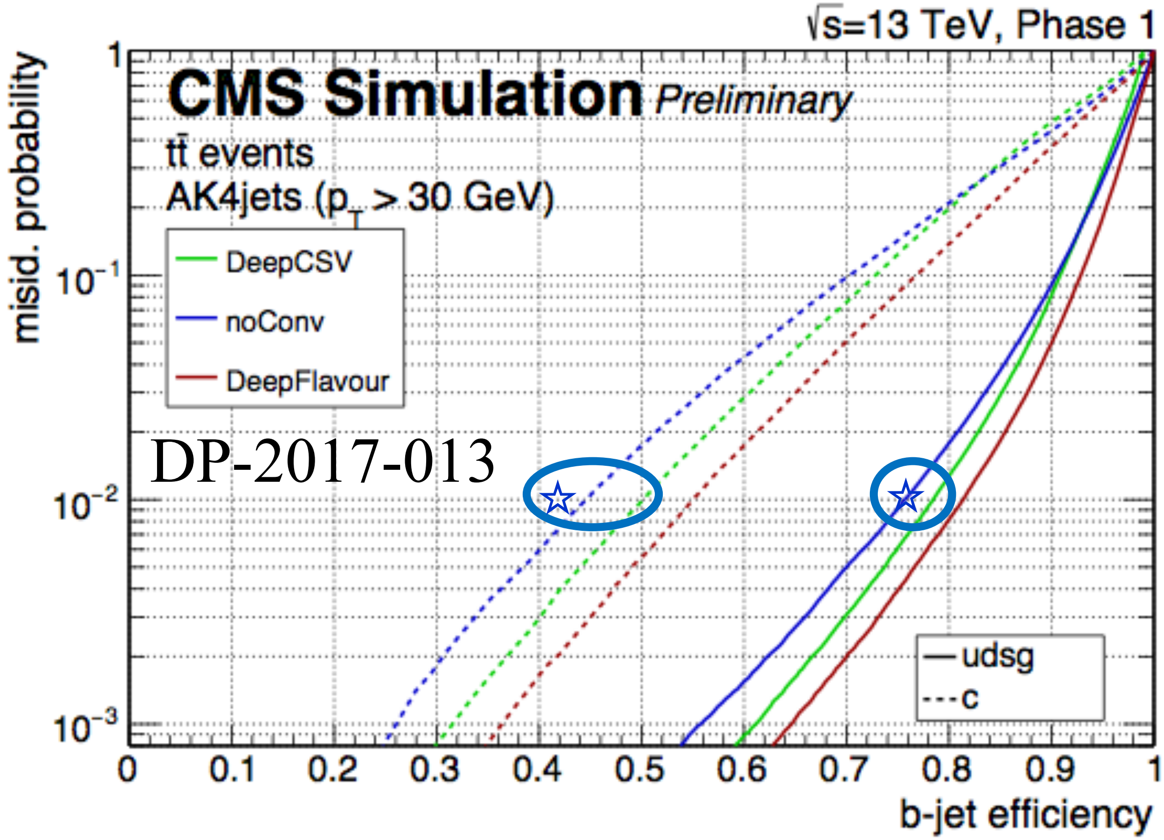
→ Largely over-constrained (>> more sample than model parameters), hardly regularization needed

→ Operate in the "data plateau", 20M < 40M ≲ 80M



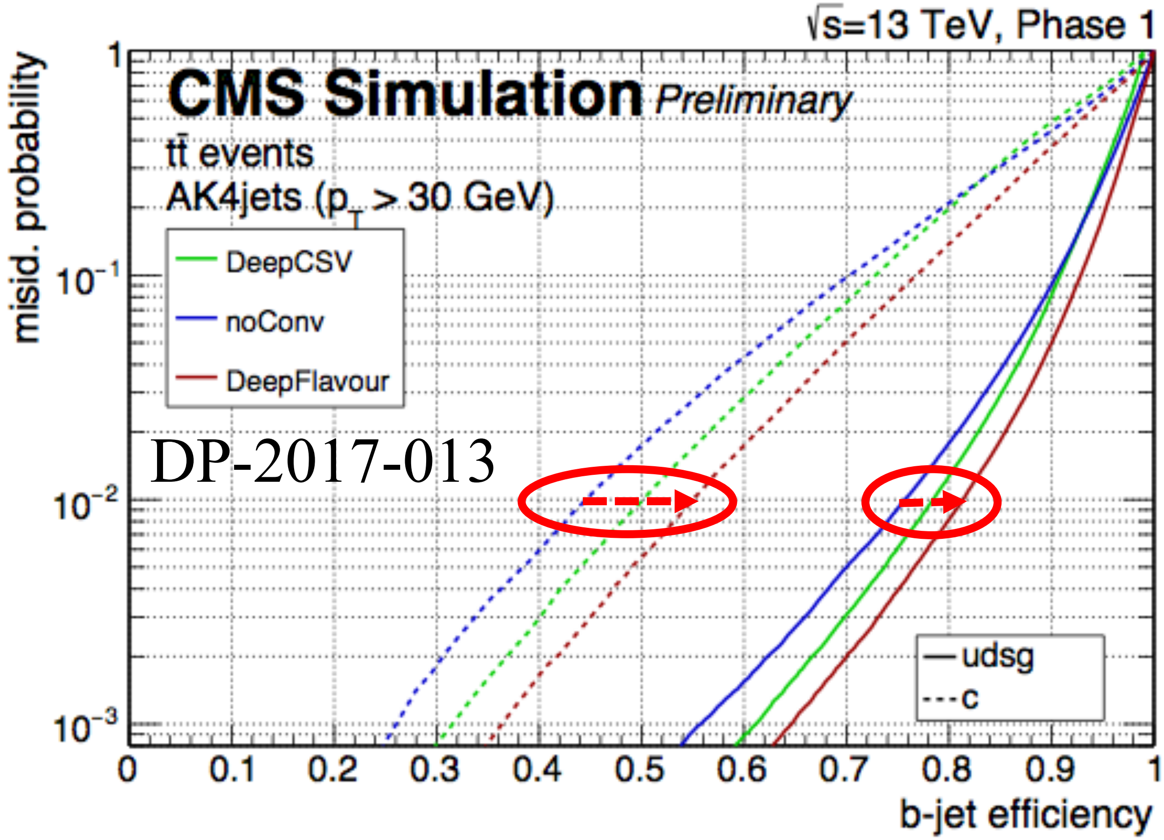
→ Optimal DNN strategy depends on availability data, jet reconstruction we have O(100) M

Just more information



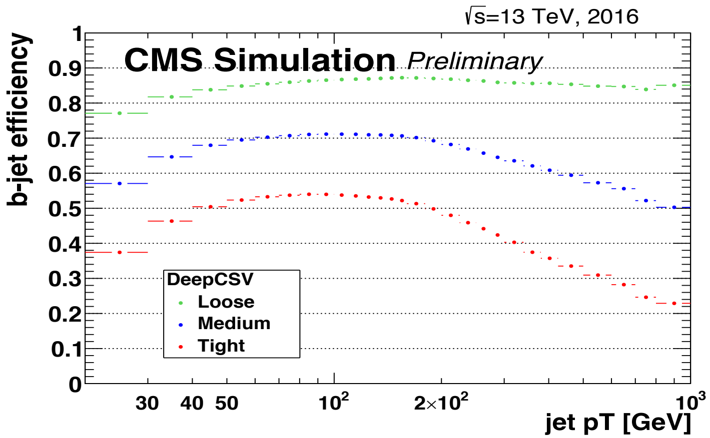
- Just adding more information, nodes and layers (+2) did degrade performance (DeepCSV→noConv)

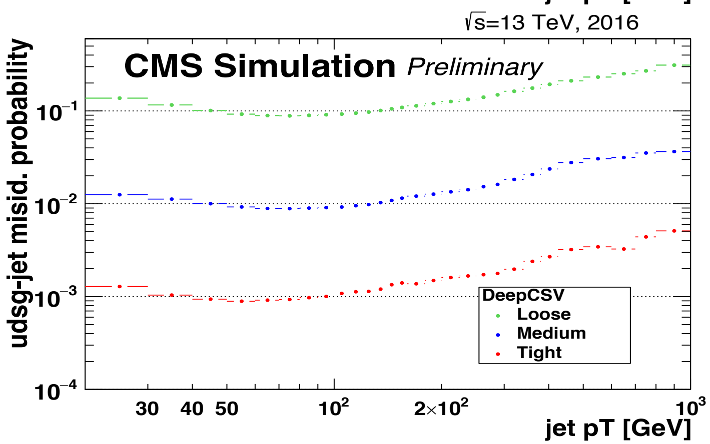
Adding CNNs

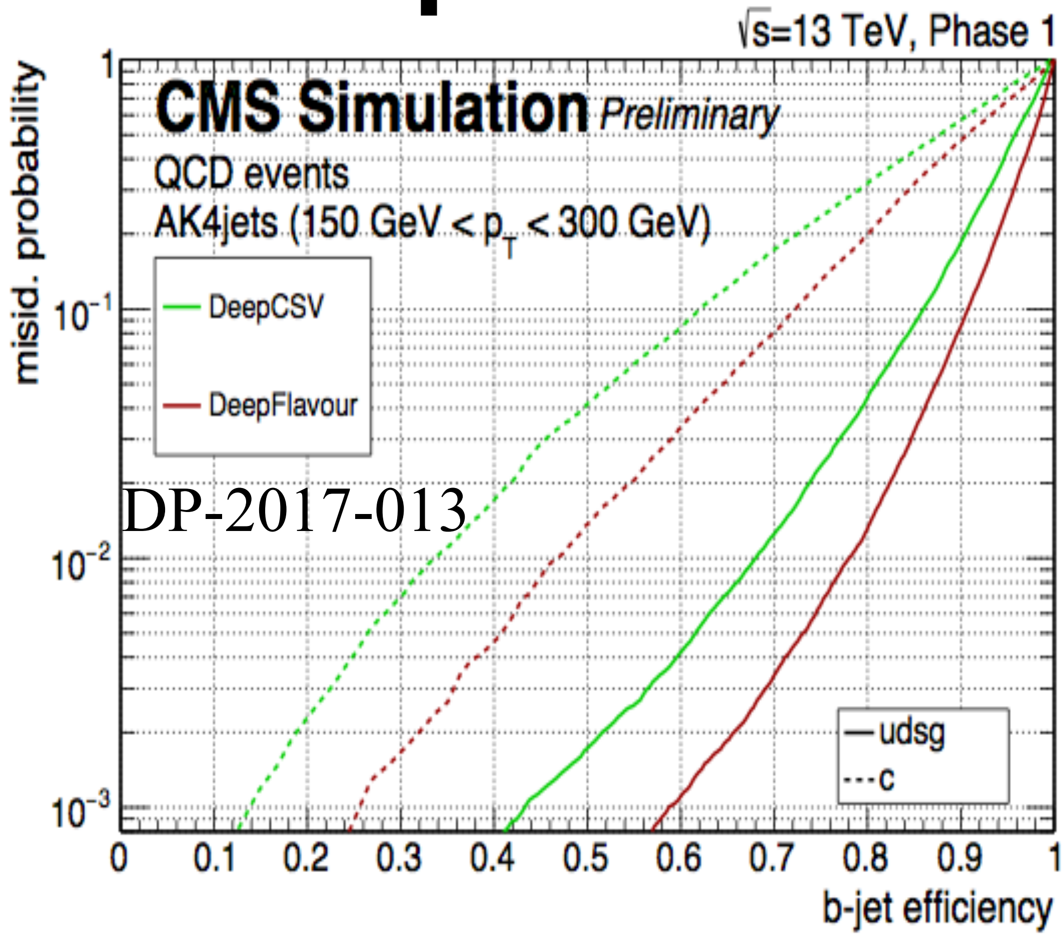


- Adding the convolution ***and*** more information improves performance (noConv→DeepFlavour)
- Adding RNN “only” helps ~ 1% (not shown)

DeepCSV vs. DeepFlavour

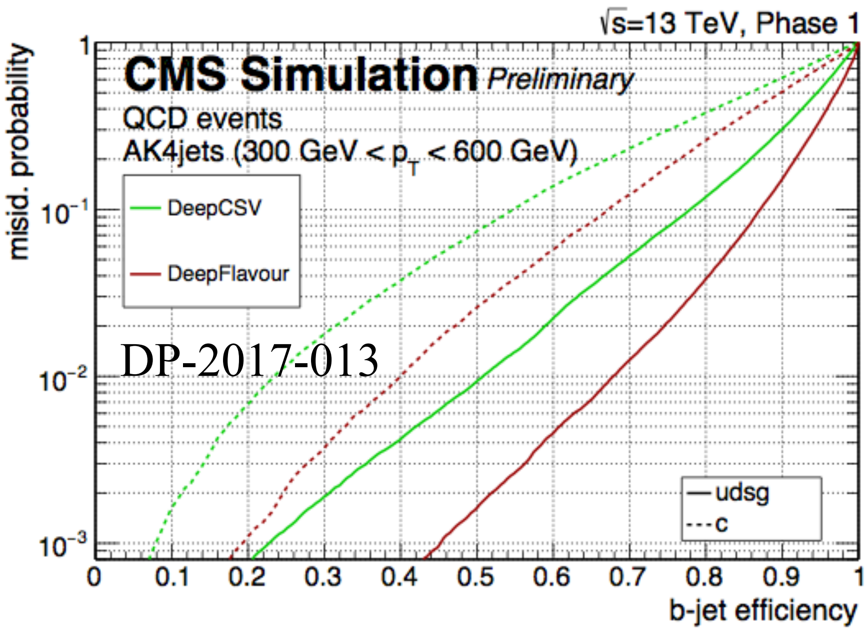


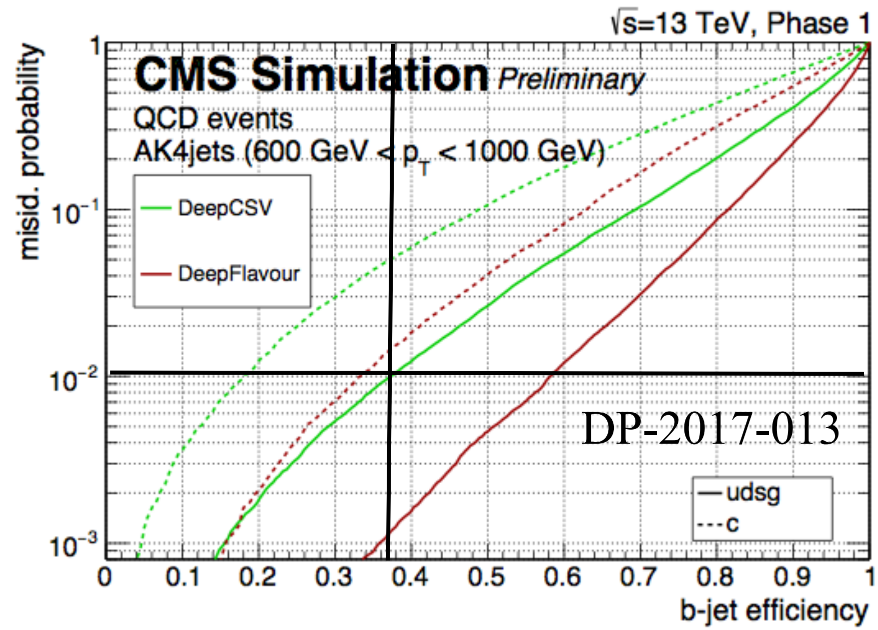




- A well-known problem: flavour tagging degrades as momentum increases
- The effect is largely reduced by DeepFlavour

DeepCSV vs. DeepFlavour





- At high momentum: at a DeepCSV fake rate of 1%, we obtain the same b tag efficiency for 10 times smaller bkg
- Alternatively: 50% more efficient at same 1% fake rate
- DeepFlavour recovers much of the degradation at high momentum

DeepJet

DeepFlavour + q/g → DeepJet

Investigate a few custom DNN q/g tagging:

Recurrent for q/g:

Input features, p_T descending: | RNN(LSTM) | Dense

p_W as in 1407.6013

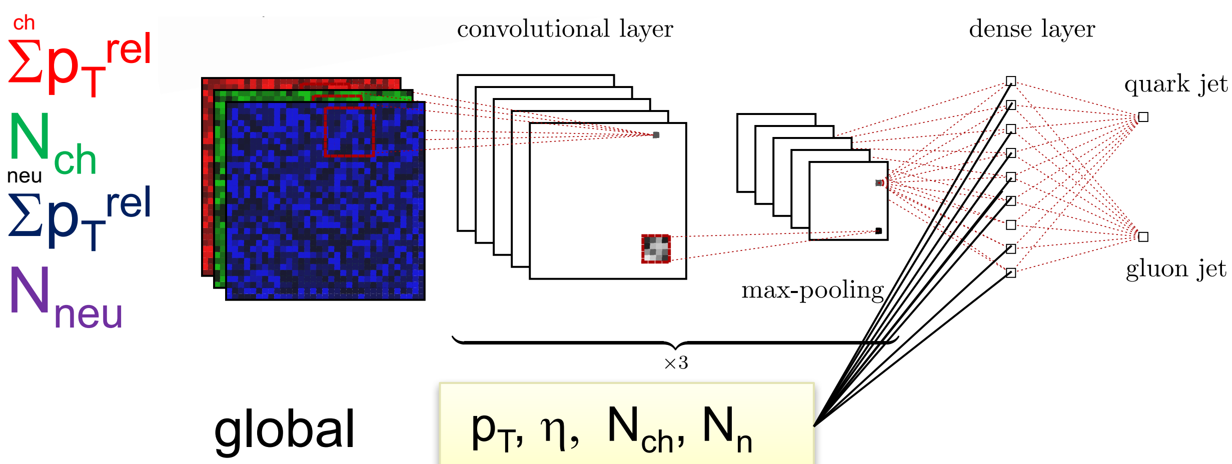
up to 25 charged: p_T^{rel}, $\Delta\eta$, $\Delta\phi$, p_W — 100

up to 25 neutrals: p_T^{rel}, $\Delta\eta$, $\Delta\phi$, p_W — 100

global: p_T, η, N_{ch}, N_{neu}

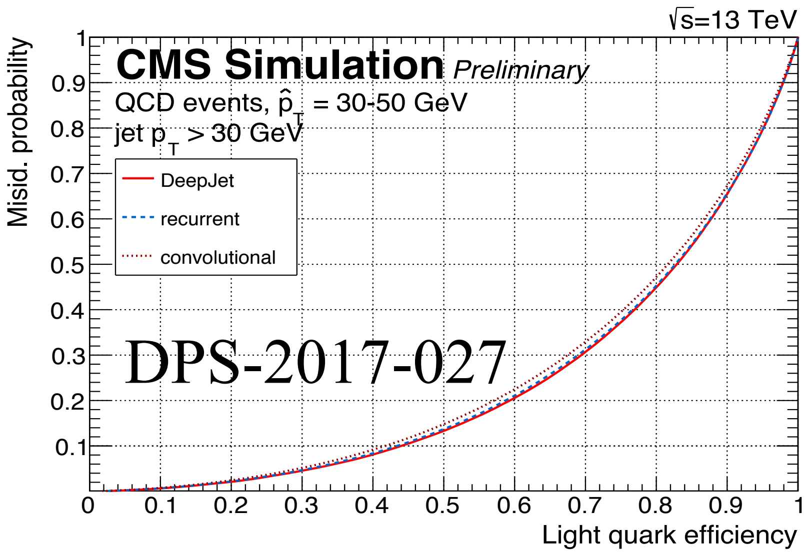
Dense: 200, 5x100

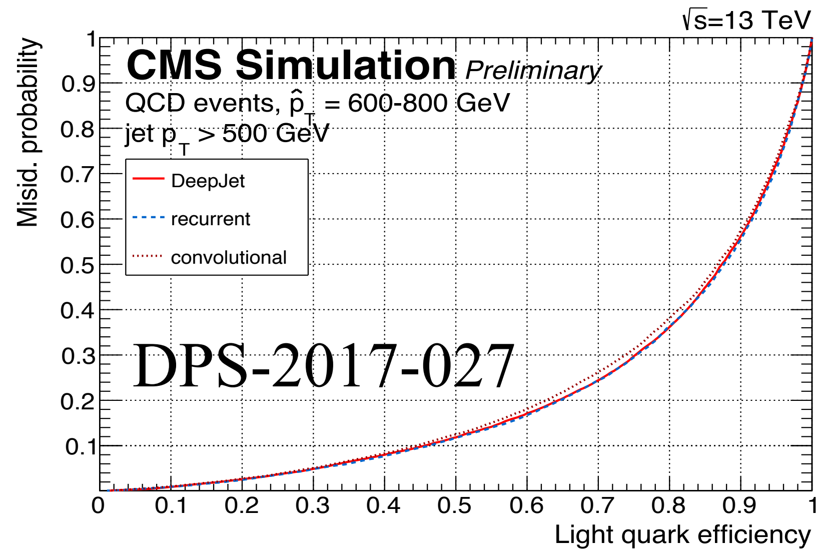
2D **convolutional**, four channels (CNN as in 1612.01551):



Comparisons of DNNs

- We filter on ***generator*** level only light quarks and gluons that did **NOT** split to heavy flavour.
- All DNN used in binary mode





→ Generic DeepFlavour and custom q/g gave very similar results!

→ **Data is multi-class, without heavy flavour removel DeepJet was clearly best**

Next steps

What is the target (loss)?

Simulation: huge well labeled dataset at all phase-spaces
→ Ideal to train big DNN!

We train&aim at MC with labels

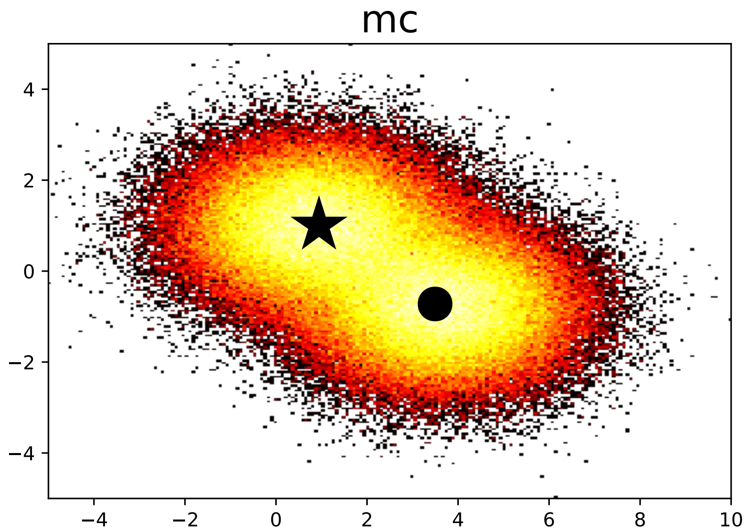
What is the target (loss)?

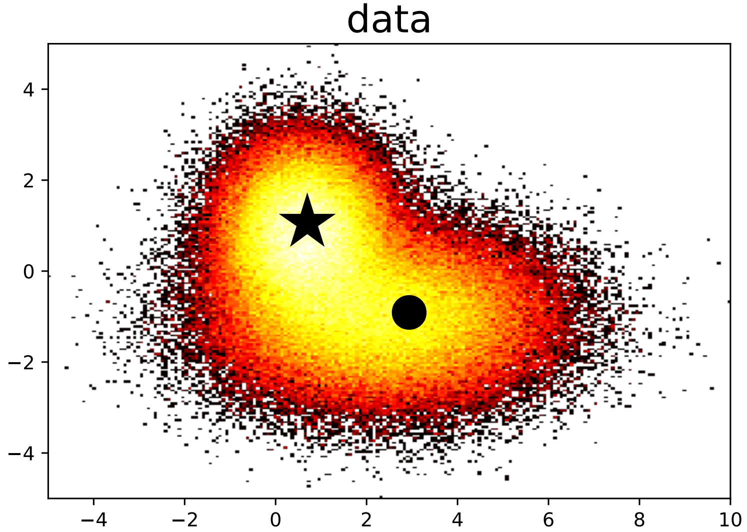
But **data** might be somewhere else

We train in simulation

- Need to move also the target!
- Ideally maintain advantages of well labeled MC!

Toy example





- Data (worse) half the horizontal resolution around circle
- We might know start/circle ratio in data, but not have individual labels

New target

- Classify the huge well labeled MC optimally
- Make it impossible to tell from the output that is was MC (given you know start/circle ration in data)

New proposal: **Hybrid loss**, add term to loss that enhances independence of output from ***data*** and ***mc***

Enhance Independence

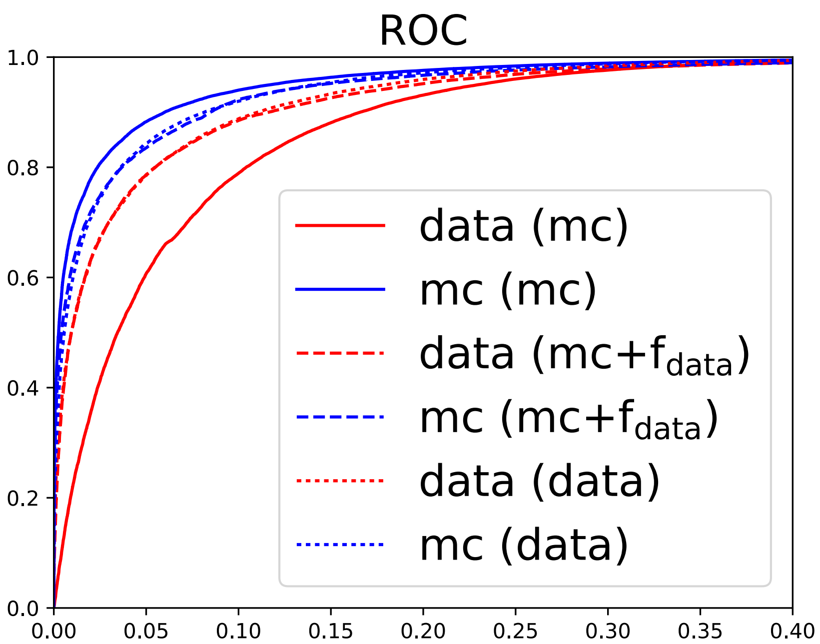
→ Approximate independence test: any slice of a 2D histogram looks the same, but better use moments of output for technical reasons (efficient in loss calculation)

$$\text{Loss} = \text{Loss} + \lambda \sum_{i=0}^{i=n} \frac{1}{m} \sum_{j=1}^{j=m} \sqrt{\left(\frac{M_{ij}}{M_{aj}} - 1\right)^2}$$

- $\acute{\imath}$ is the index of the classes (data or MC) or e.g. bin of a histogram with n bins, a stands for all
- M_j is the moment j of the output

New proposal: Use ***analytical*** stringent necessary condition for independence (e.g. moments of different bins) as penalty term in loss to enhance independence.

Toy Results



- Train with mc (mc)
- Retrain with mc adding the *moment* loss to enhance data and MC output agreement (mc+f^{data}). Assume to know star/circle ratio in data!
- Train using data ***labels*** (data)

Hybrid Loss achieved same performance as individually **labeled** data

Proposal: Pre-train with MC (high stats, good labels) and fine-tune with data (lower stats, average or approximate labels)

Summary

DeepCSV:

- Deep learned multi-label flavour tagger, which is the recommended CMS tagger 2017
- Used with success in 2016 analysis in data

DeepJet:

- Developed new ***generic*** DNN structure for jets reconstruction, that incorporates ***extensive*** information from ***all*** jet constituents
- Performs best CMS simulation in all categories
- Working other cone sizes and regression

Data/MC:

- Working on ***new*** methods to incorporate data/mc differences in overall strategy