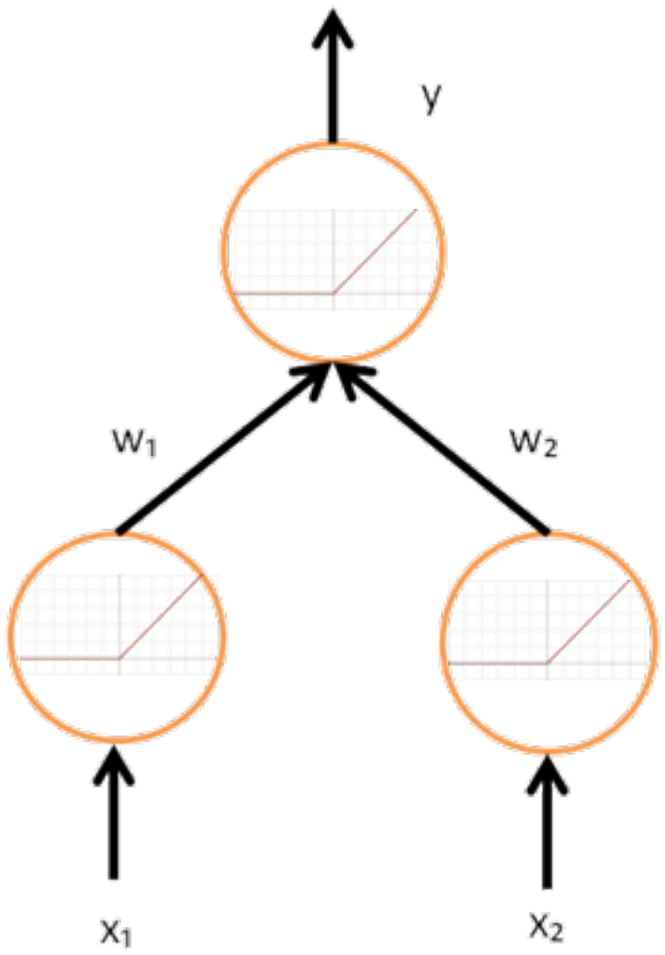
Deep Learning Top Tagger - or the end of QCD?

Gregor Kasieczka

CERN IML Workshop

2017-03-22

A Very Simple Network



$$y = f(f(x_1)w_1 + f(x_2)w_2)$$

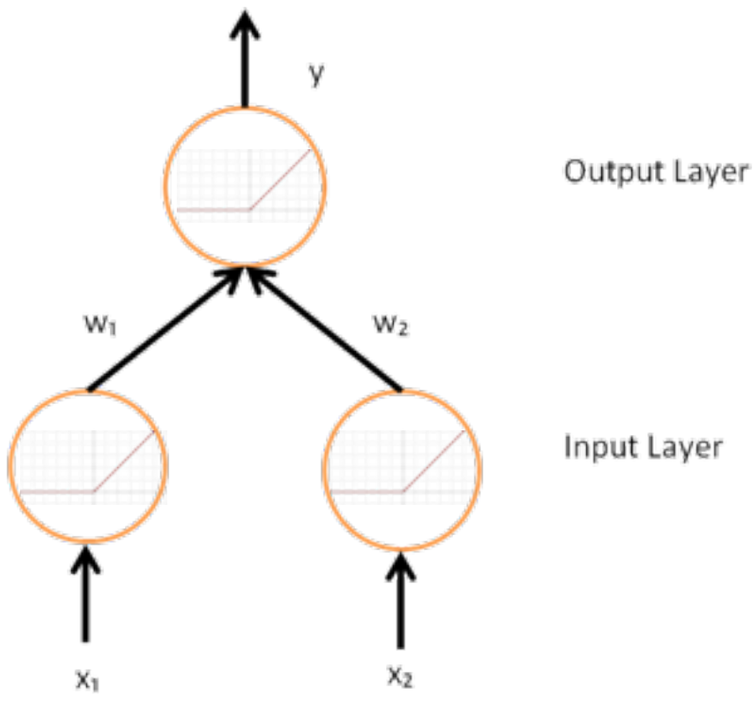
$$f(x) = \Theta(x) \cdot x$$

How do networks learn?

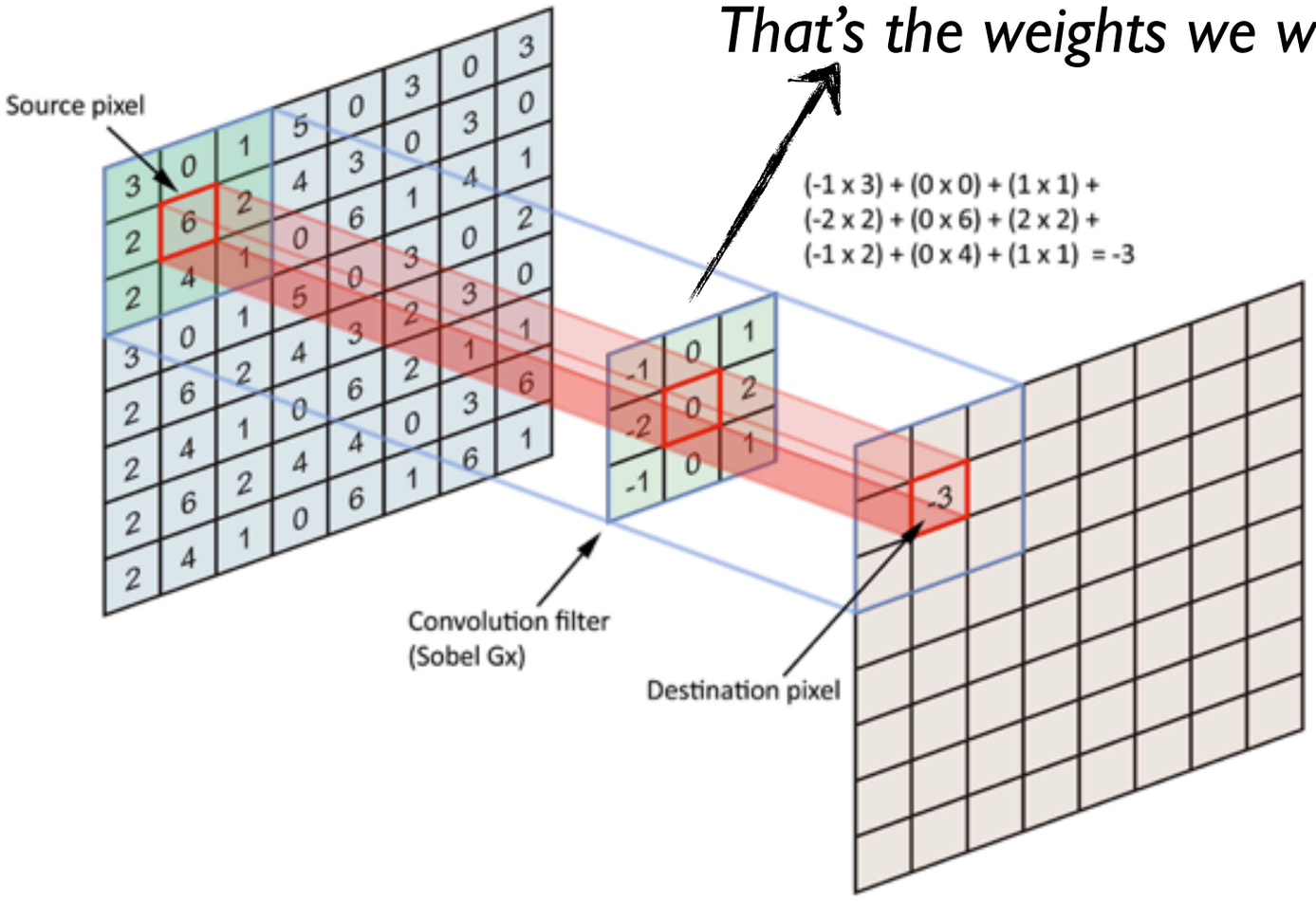
- *Backpropagation + Gradient descent*
- Pass input (x_1, x_2) to ANN
- Calculate output (y) and difference to true value ($\hat{y}$)
  This is the loss function L
- Find gradient of loss function with respect to weights
- Use gradient to find new weights

$$L(y, \hat{y}) = (y - \hat{y})^2$$

$$w_i' = w_i + \alpha \cdot \frac{\partial L}{\partial w_i}$$



Convolutional Network



Fold a mask with the input to get output
What is learned are the parameters of the mask

Convolutional (conv) layer

- ***How to build a convolutional network***
  - Chain multiple conv layers
  - Reduce image resolution in between (optional)
  - Use multiple masks per layer
  - Add linear ANN in the end (optional)

(This is still a network. We just use a fancy idea to decide which nodes to connect to each other)

arXiv: *1701.08784*
submitted to JHEP

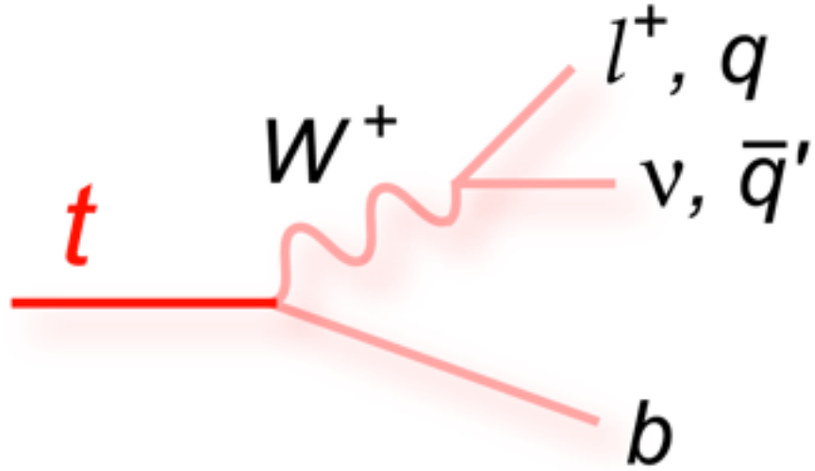
Deep-learning Top Taggers or The End of QCD?

Gregor Kasieczka,[1] Tilman Plehn,[2] Michael Russell,[3] and Torben Schell[2]

[1] *Institute for Particle Physics, ETH Zürich, Switzerland*
[2] *Institut für Theoretische Physik, Universität Heidelberg, Germany*
[3] *School of Physics and Astronomy, University of Glasgow, United Kingdom*

Problem

- Identify hadronically decaying top quarks from QCD jets
- p_T large enough so that the top quark is contained in one jet (boosted topology)
- Many QCD inspired variables available
  - jet mass = top mass
  - also find W mass inside jet
  - n-subjettiness (3-prong structure of the top)
- → *Top tagging is a well understood problem to test DNNs on*

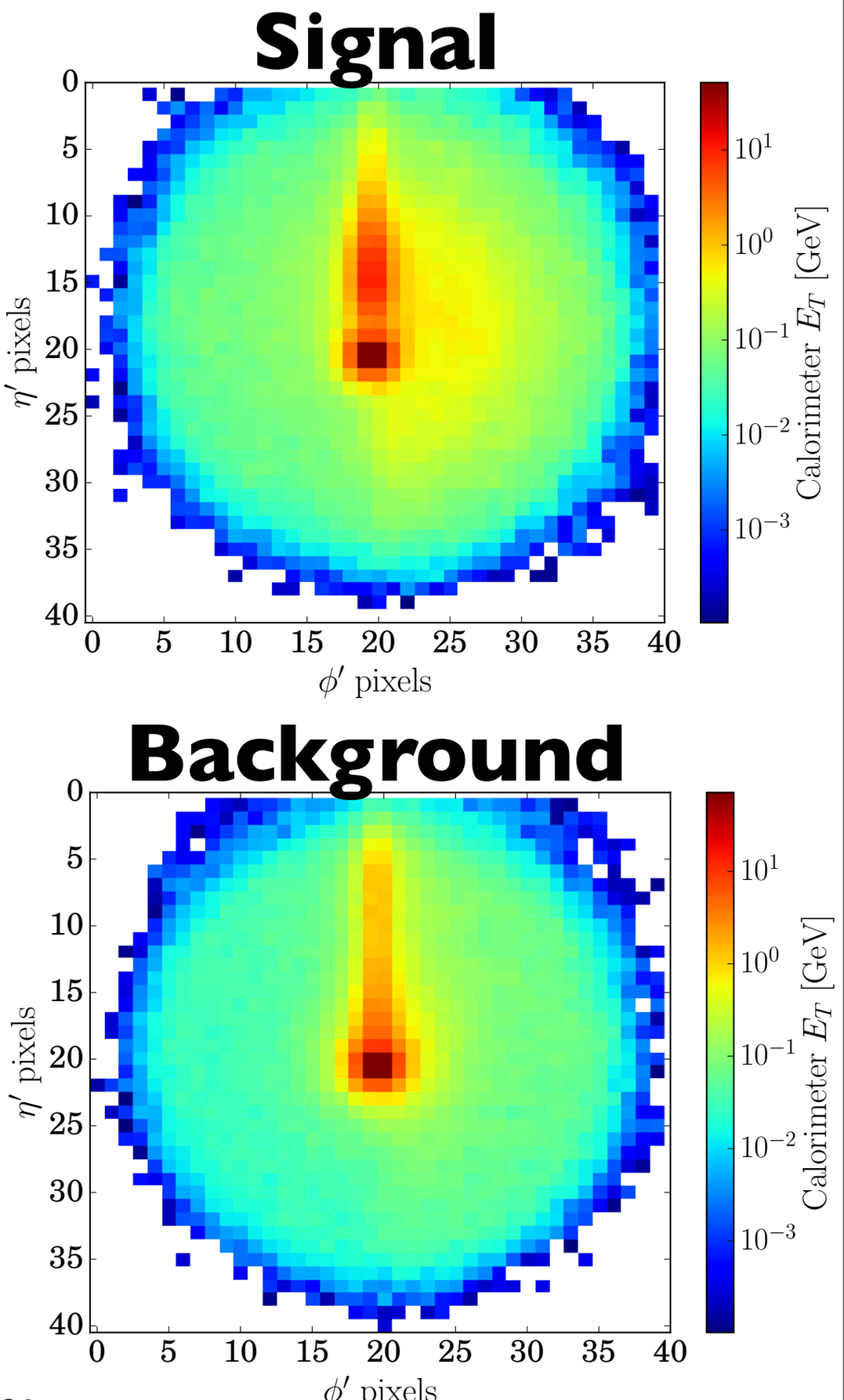


Technical interlude

- 14 TeV hadronic ttbar vs QCD, both simulated with Pythia 8
- Delphes 3 detector simulation
- Cluster with Anti-kT(R=1.5), recluster with Cambridge/Aachen (R=1.5)
- Jet pT: 350..450 GeV, |eta| < 1.0
- Signal is truth matched to top within Delta R < 1.2
- Samples (signal+background):
  - 150k+150k for training
  - 150k+150k for checks during training
  - 300k+300k for final test

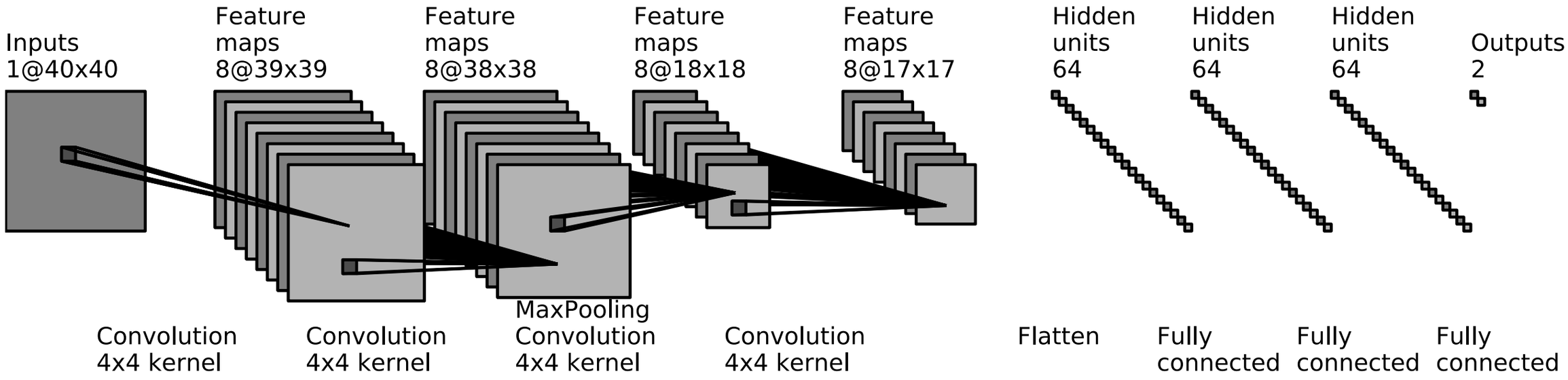
Image approach

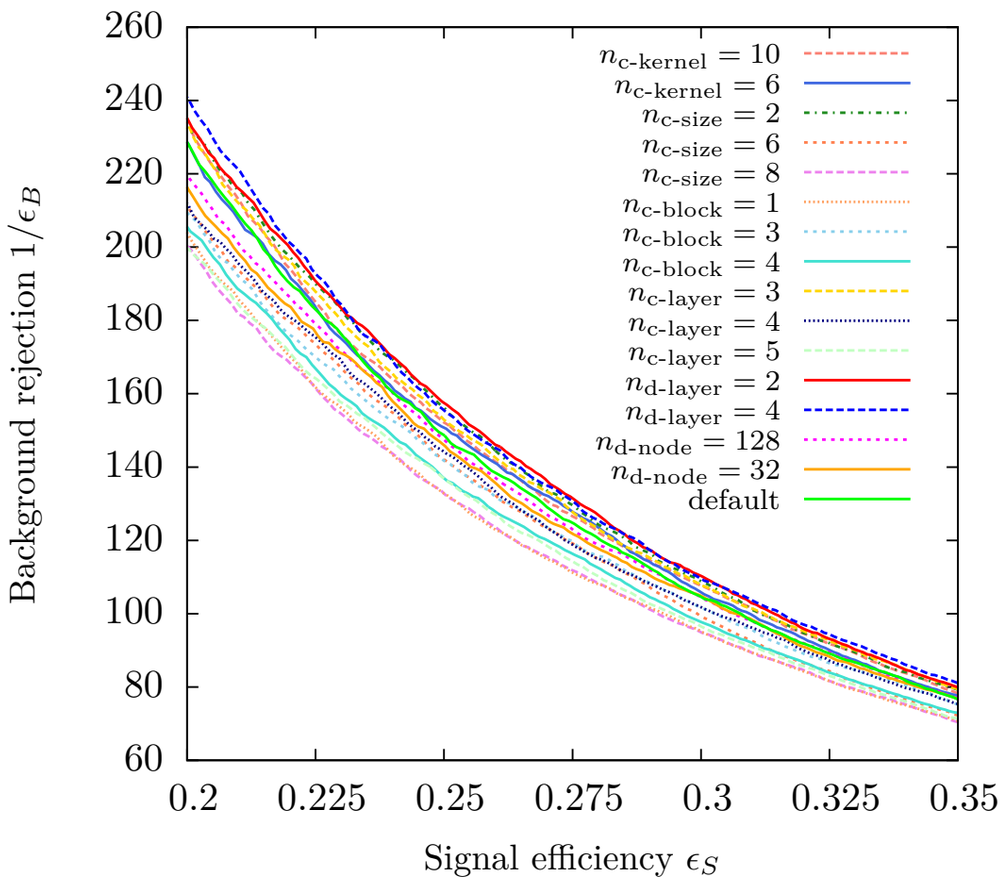
- Jets = 2d grayscale images:
  - 1 pixel = 0.1 in eta, 5 degree in phi
  - pixel energy: calorimeter ET
- Preprocessing
  - Center maximum
  - Rotate so that second maximum is 12 o'clock
  - Flip so that third maximum is on the right side
  - Crop to 40x40 pixels



Overlay of 10k images

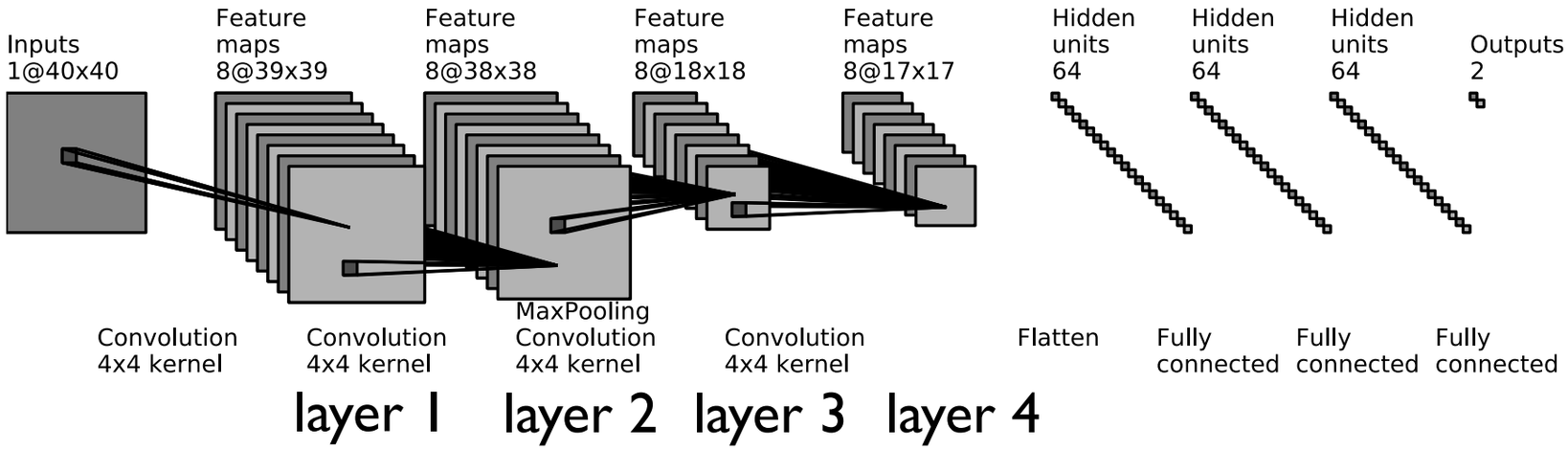
Network architecture





Disadvantage of DNN approach: choosing network architecture is a bit of voodoo
This seems to work though

Intermediate output

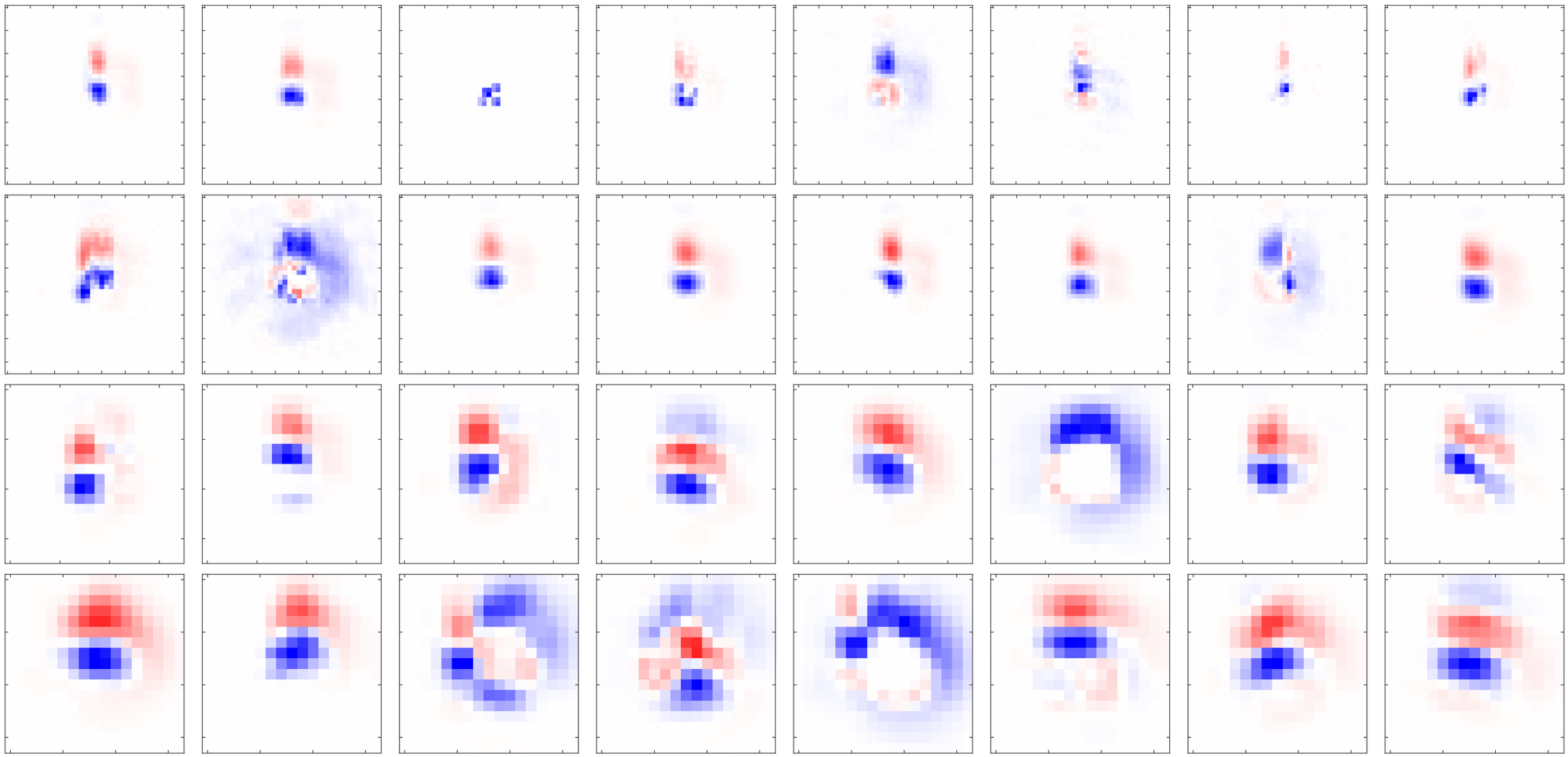


layer 1

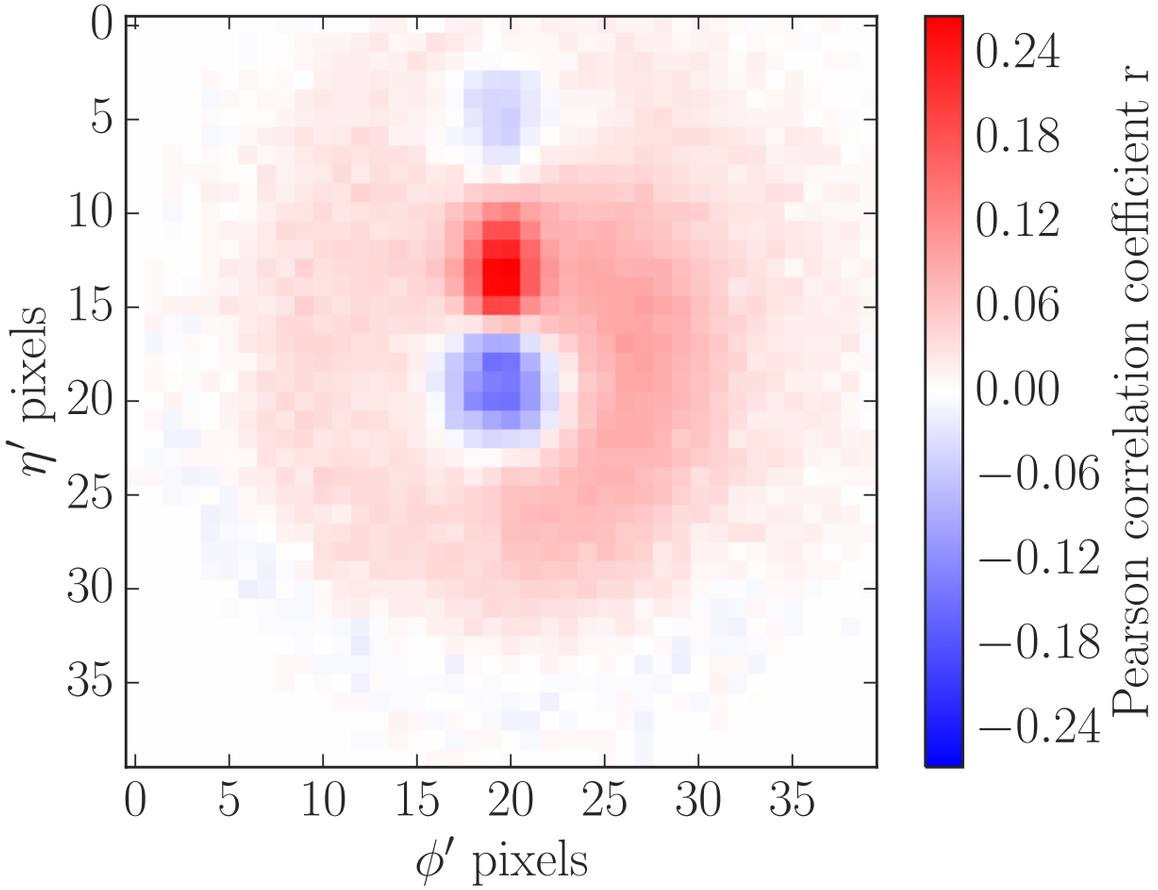
layer 2

layer 3

layer 4



Pearson correlation coefficient



$$r_{ij} = \frac{\sum_{\text{images}} \left(x_{ij} - \bar{x}_{ij}\right)\left(y - \bar{y}\right)}{\sqrt{\sum_{\text{images}} \left(x_{ij} - \bar{x}_{ij}\right)^2} \sqrt{\sum_{\text{images}} \left(y - \bar{y}\right)^2}}$$

Performance

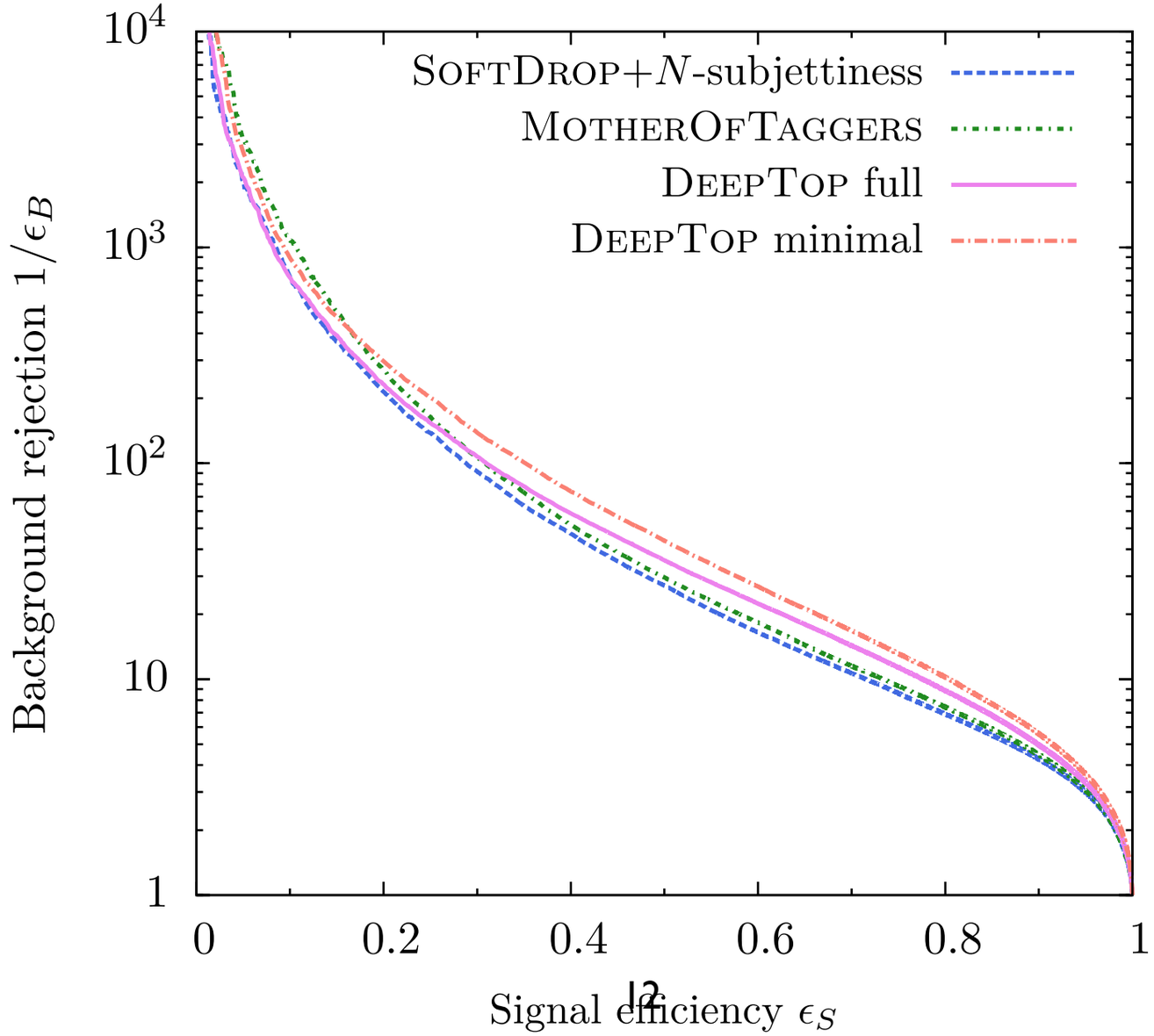
- Train a BDT on a set of standard tagging variables

SoftDrop + n-subjettiness:

$$\{ m_{\mathrm{sd}}, m_{\mathrm{fat}}, \tau_2, \tau_3, \tau_2^{\mathrm{sd}}, \tau_3^{\mathrm{sd}} \}$$

MotherOfTaggers:

$$\{ m_{\mathrm{sd}}, m_{\mathrm{fat}}, m_{\mathrm{rec}}, f_{\mathrm{rec}}, \Delta R_{\mathrm{opt}}, \tau_2, \tau_3, \tau_2^{\mathrm{sd}}, \tau_3^{\mathrm{sd}} \}$$

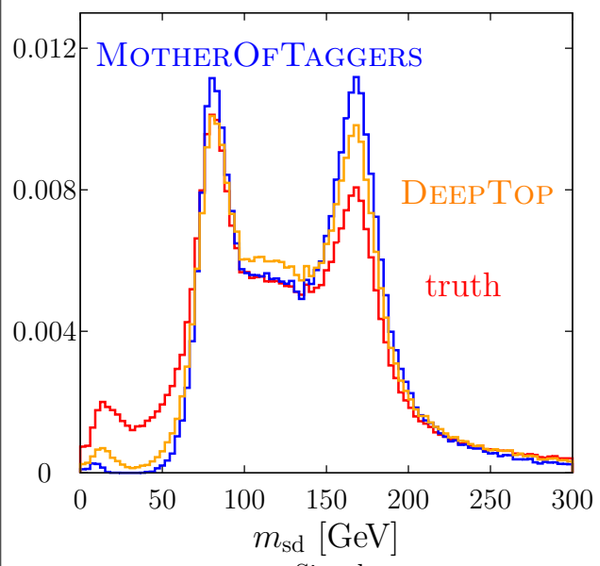


Technical Interlude II

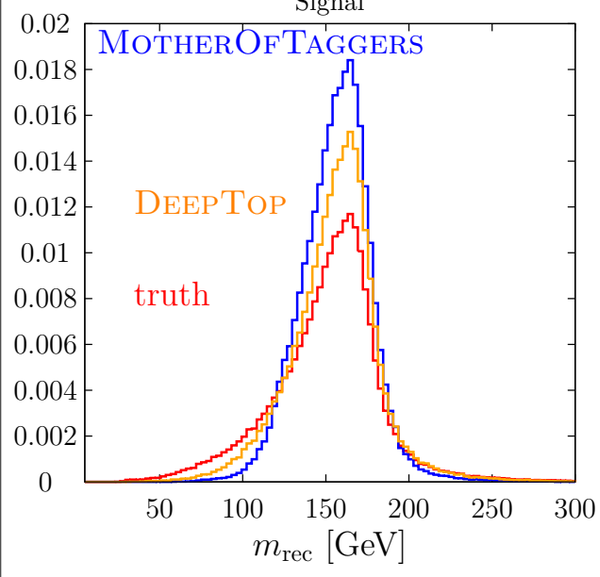
- Training done on Piz Daint nodes equipped with *Nvidia Tesla P100 GPUs* at CSCS
  - Applied for small development project of 25k node hours for a year
  - Also played with Amazon cloud (very easy and convenient, but they want money!)
- Network trains in O(some hours)
  - *Depends on N(events), network architecture, learning parameters. Can probably optimize a bit*
- Use
  - Keras with Theano backend for DNN
  - scikit-learn for BDT
  - NumPy and Pandas for processing & storage

Sliced Masses

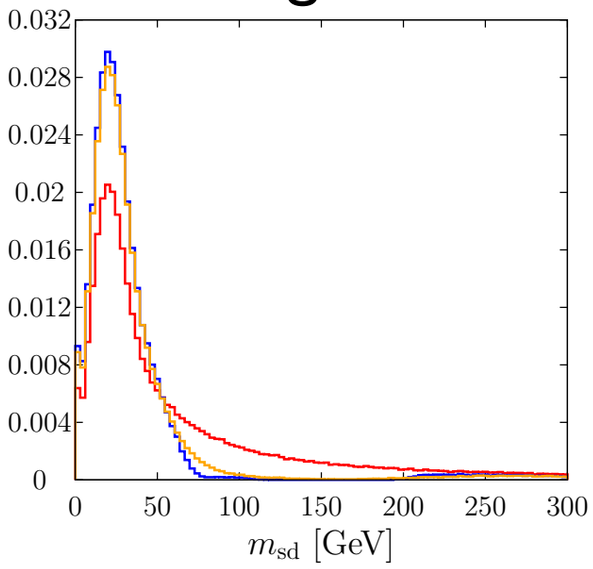
Signal



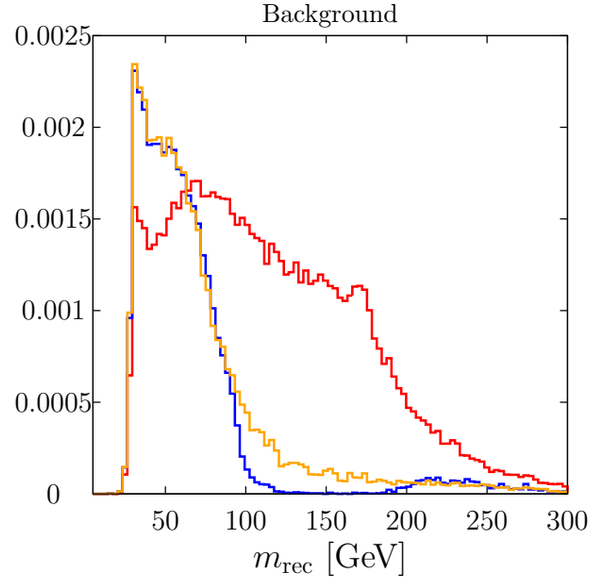
Signal



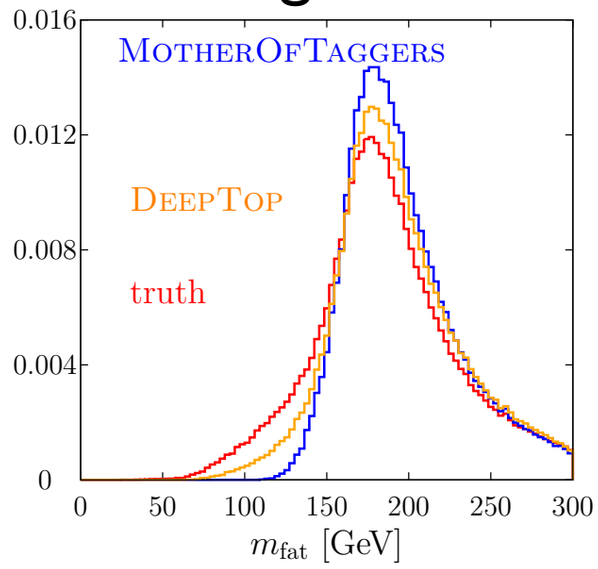
Background



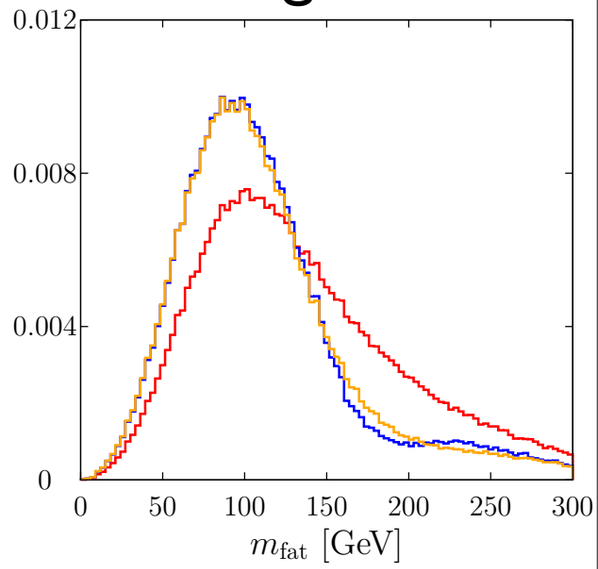
Background



Signal



Background



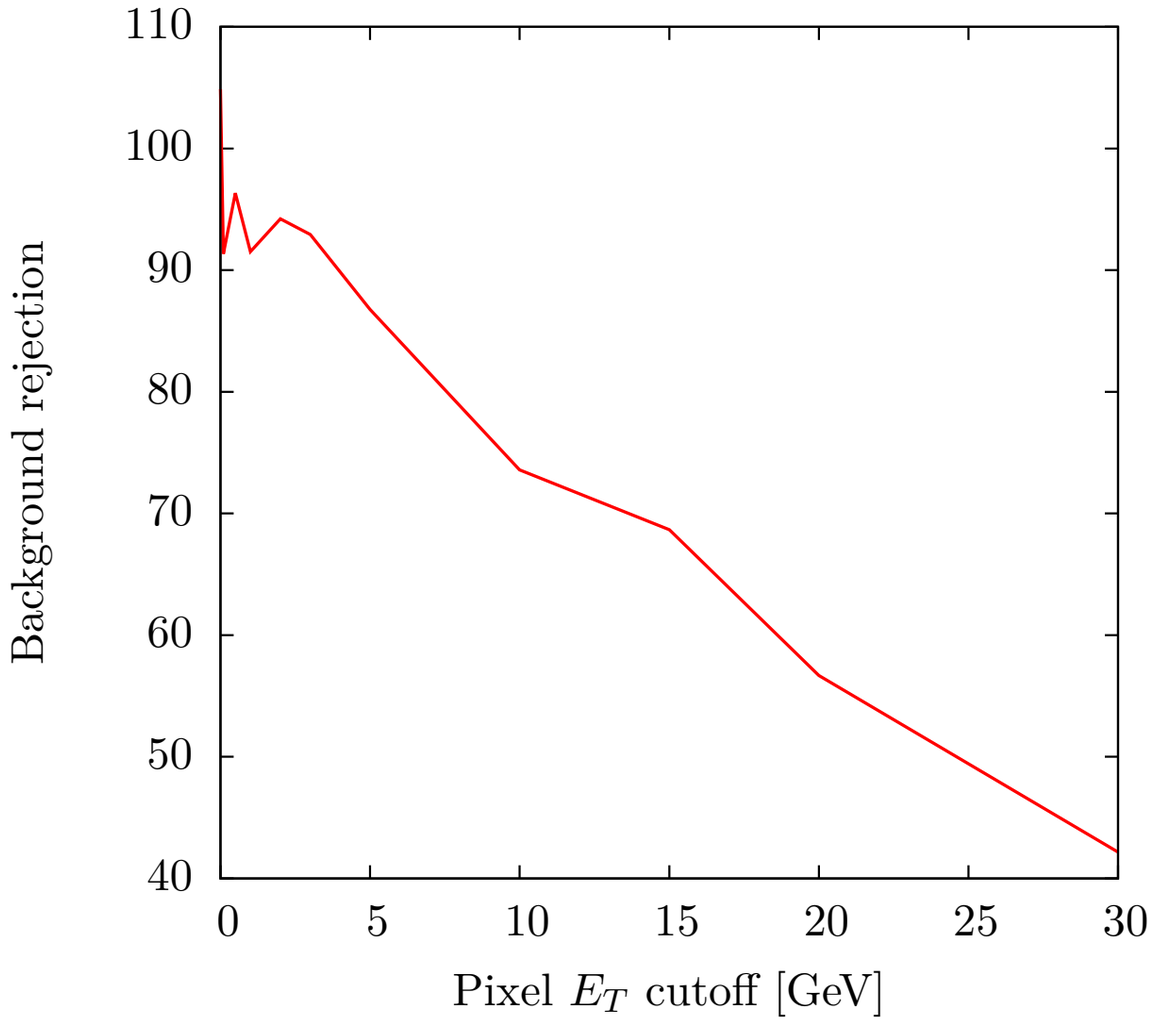
Compare:

Selection on truth

Selection using BDT (<0.2 / >0.8)

Selection using DNN ((<0.2 / >0.8)

Pixel threshold

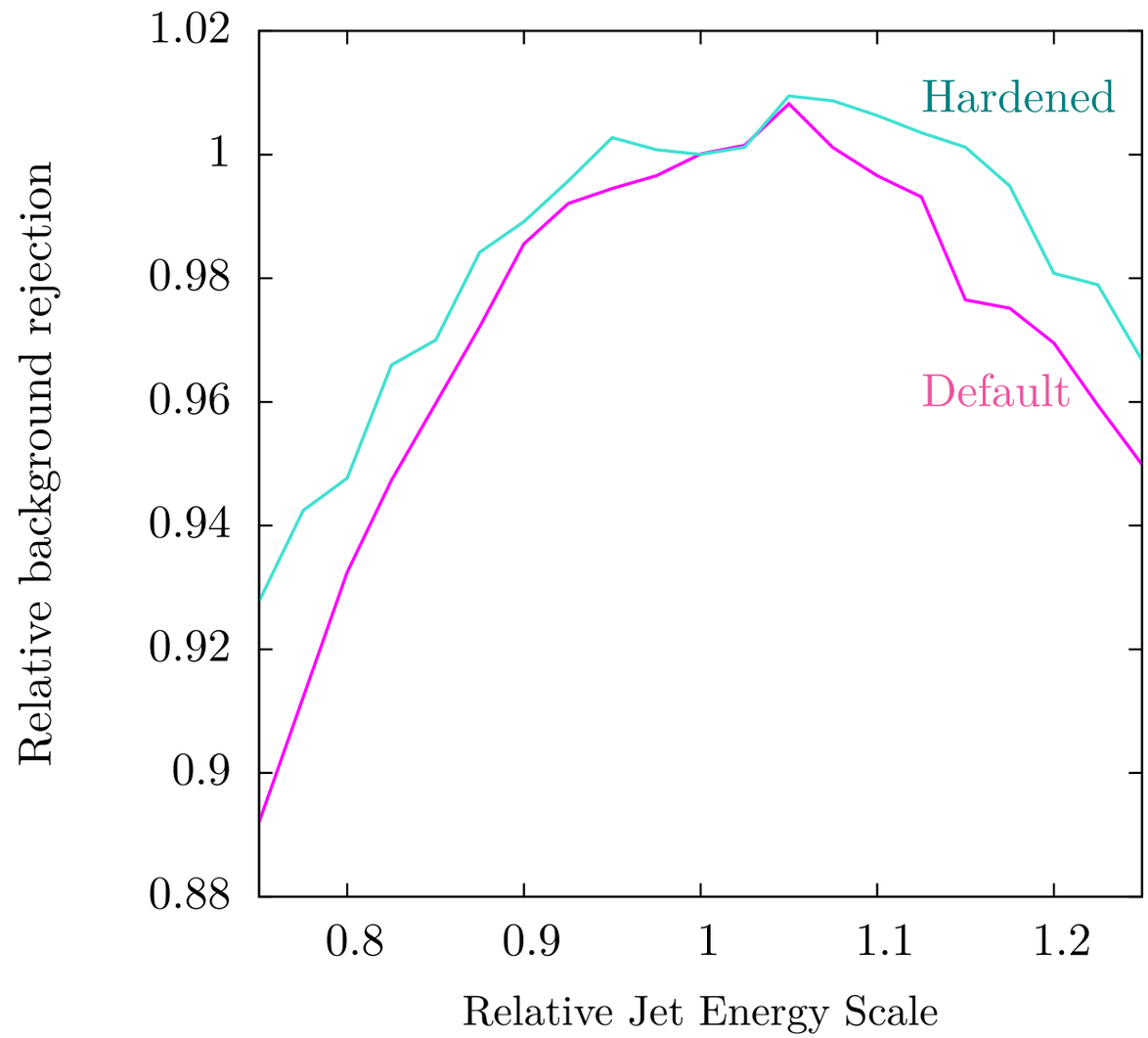


(BG rejection at 30% signal)

- Train and test network on images with higher pixel threshold
  (and set values below threshold to zero)
- We are not sensitive to very soft information (this is good)

Detector effects

- *Default*
  - Train network on JES=1.0
  - Test network on images with JES scaled up/down
- *Hardened*
  - Train network while randomly smearing JES
  - Test network on images with JES scaled up/down



Control what the network is sensitive to

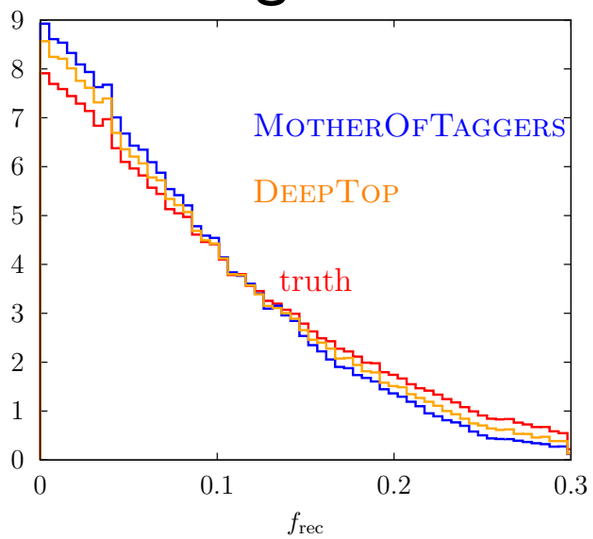
Conclusions

- DNNs for top jet tagging work, achieve decent performance and are surprisingly stable
- Techniques available to understand what it going on
- More things to try out (data, regression, RNNs, multi-object,..)
- Ready for physics analysis?

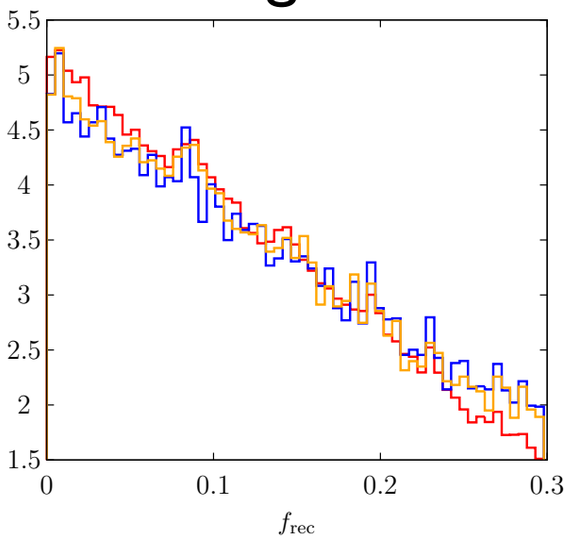
Bonus Slides

Other Slices

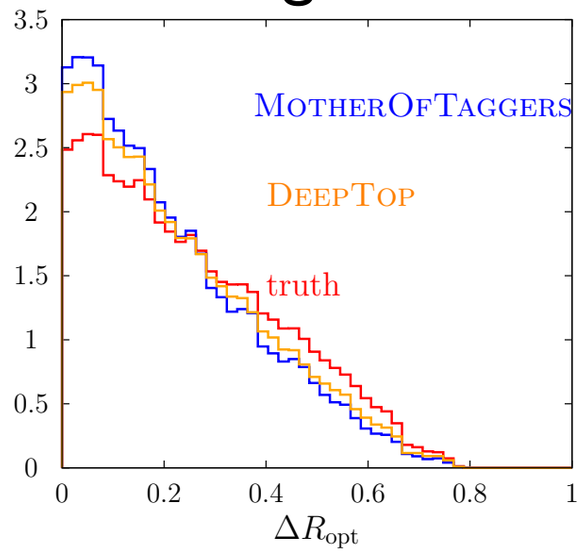
Signal



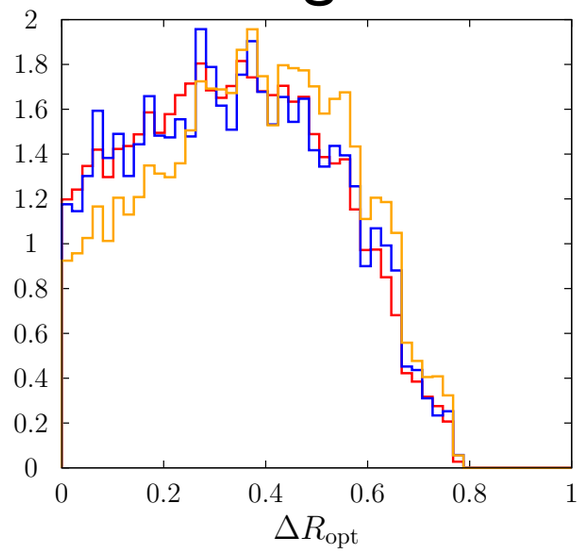
Background



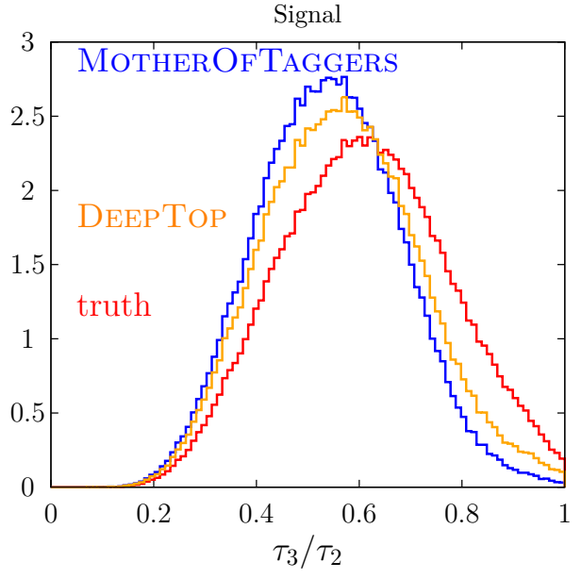
Signal



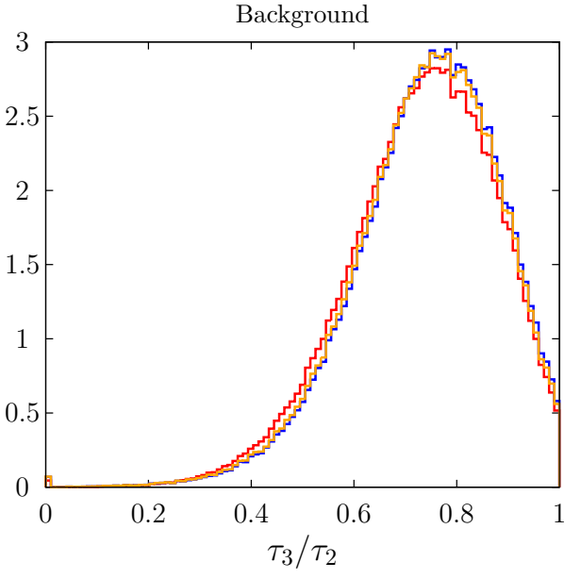
Background



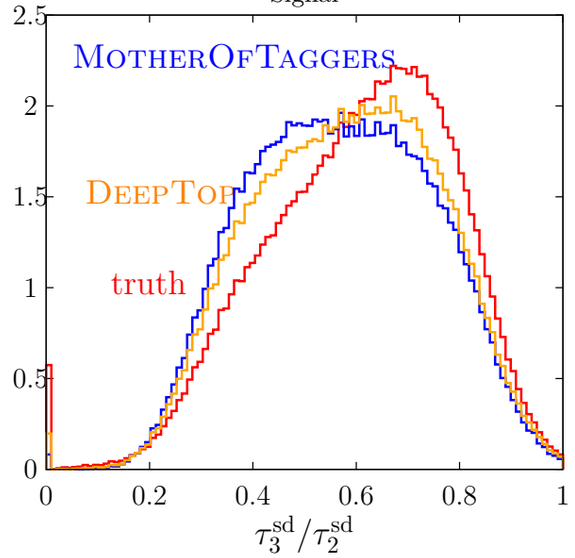
Signal



Background



Signal



Background