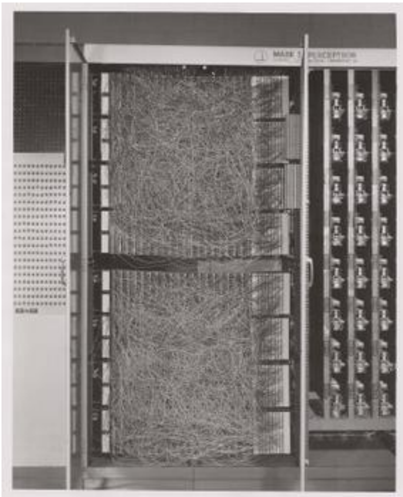
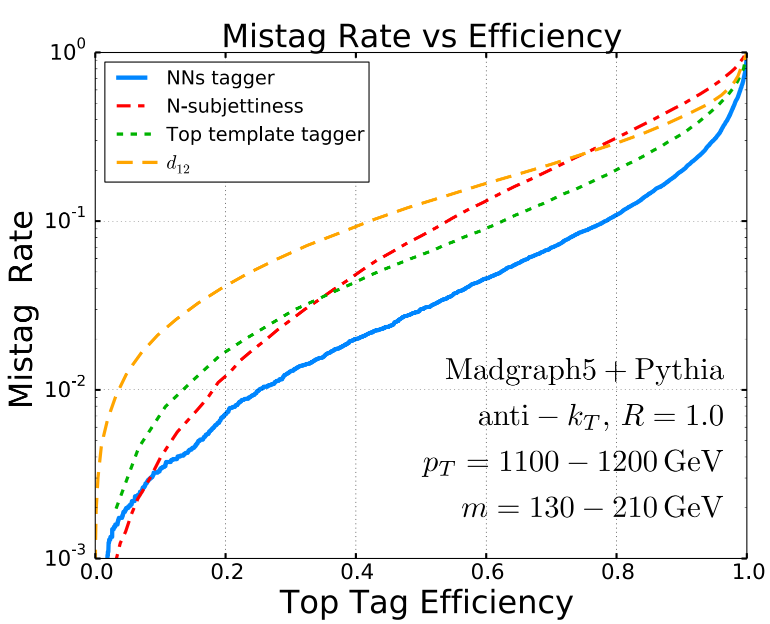
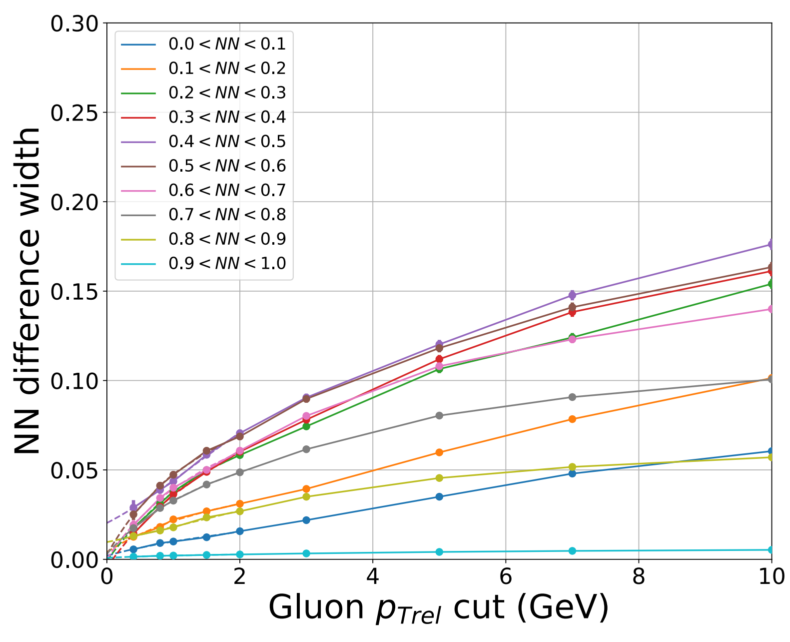


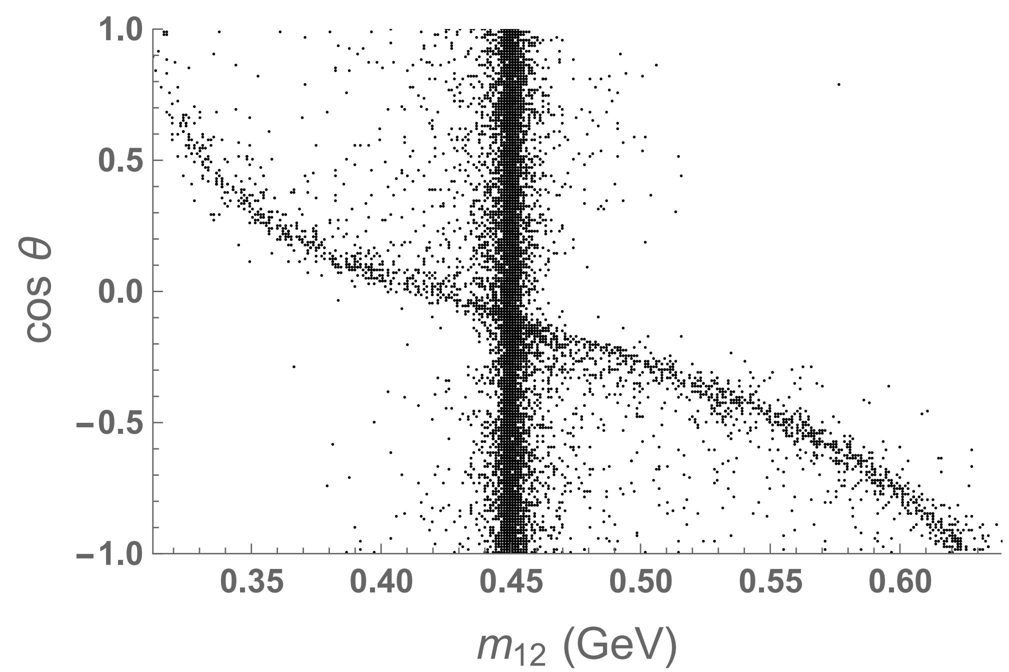
Adventures in Machine Learning

Maxim Perelstein, Cornell

BNL/Stony Brook Joint Seminar, November 7 2018







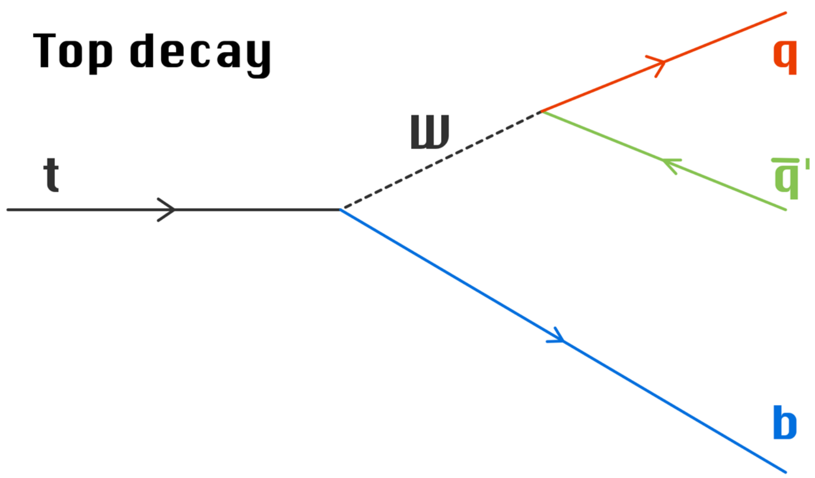
Talk 1: Boosted Top Tagging with Neural Networks

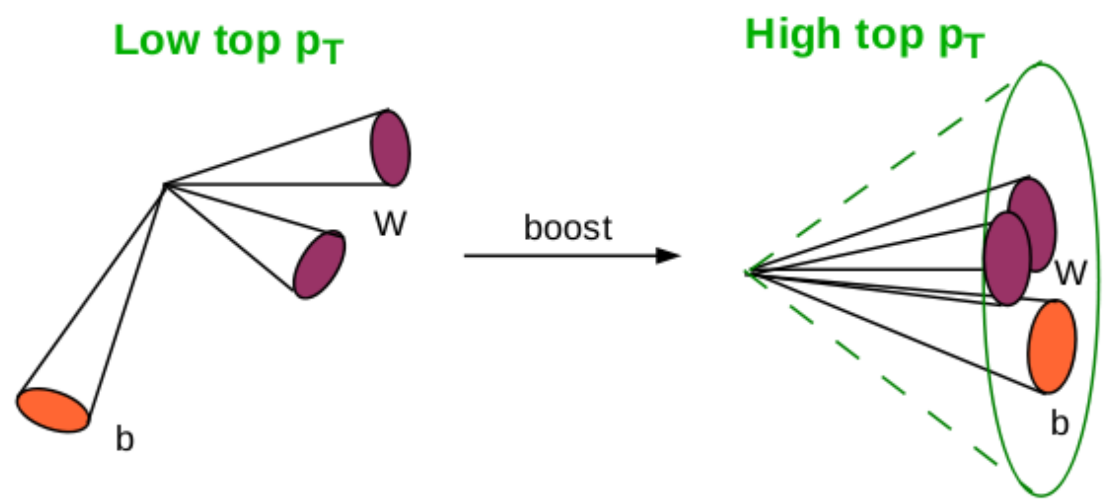
Almeida, Backovic, Cliche, Seung Lee, MP, 1501.05968
S. Choi, S. Lee, MP, 1806.01263

Talk 2: Monte Carlo Simulations with Neural Networks

Matthew Klimek, MP, 1810.11509

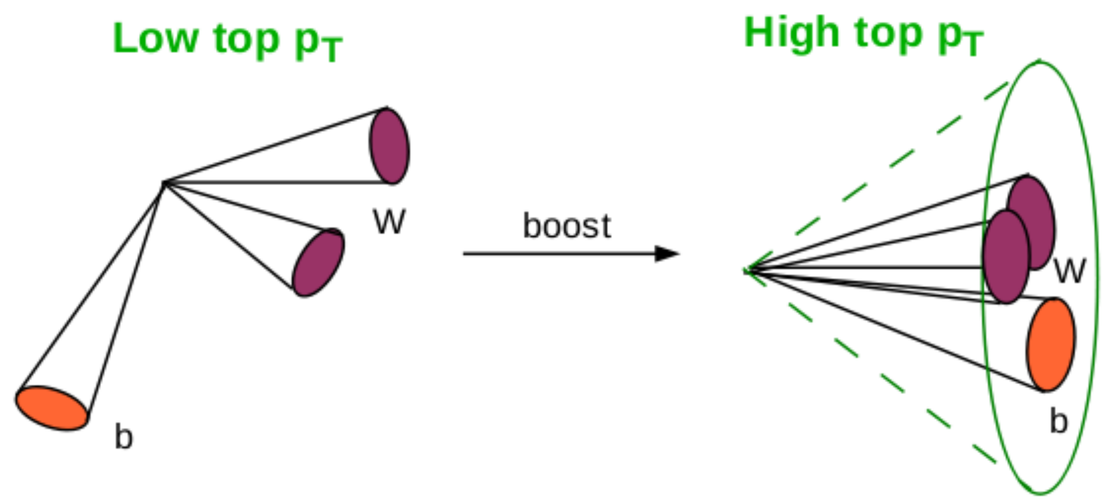
Hadronic Boosted Top





- Sources of boosted tops:
  - High-pT tail of SM t-tbar
  - Extra Dimensions: KK gluon decays $G^1 \to t\bar{t}$
  - SUSY: e.g. gluino decays $\tilde{g} \to t\bar{t}\tilde{\chi}^0$
  - Spin-1/2 top partners: $T \to tZ, th$
- As interesting new physics scale is pushed higher by LHC bounds, boosted tops become ever more important in searches for BSM

Boosted Top ID

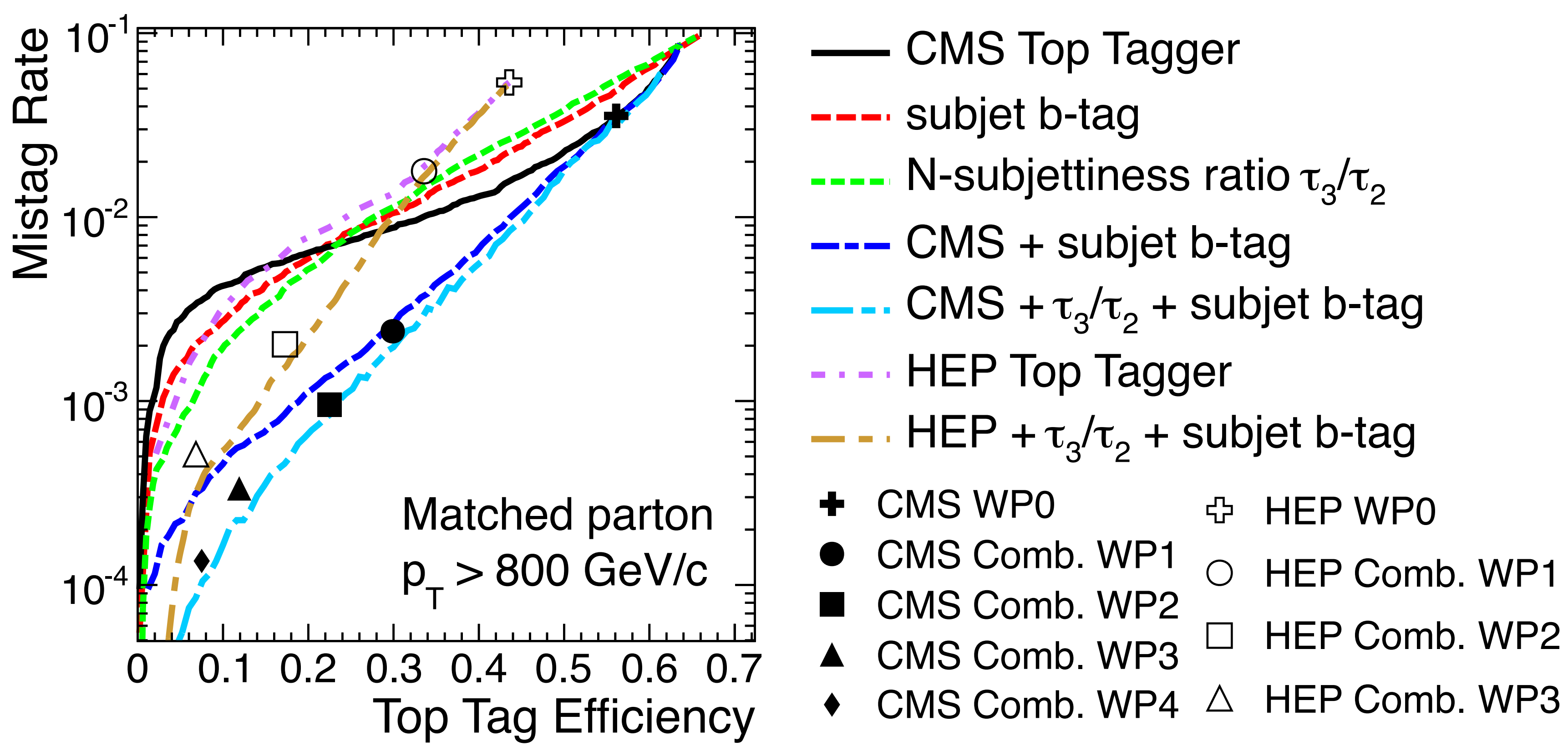


- Cluster jets with a large cone, typically $\Delta R = 1.0$ ("fat jets")
- Each boosted top appears as one fat jet
- Challenge: distinguish "QCD jets" (light quark/gluon-initiated) from "boosted tops", based on "jet substructure"
- QCD rates are >> top rates, so need high efficiency and good rejection power (i.e. small mis-tag rate)

Efficiency = Prob(top-tag|top)
Mis-tag = Prob(top-tag|QCD)

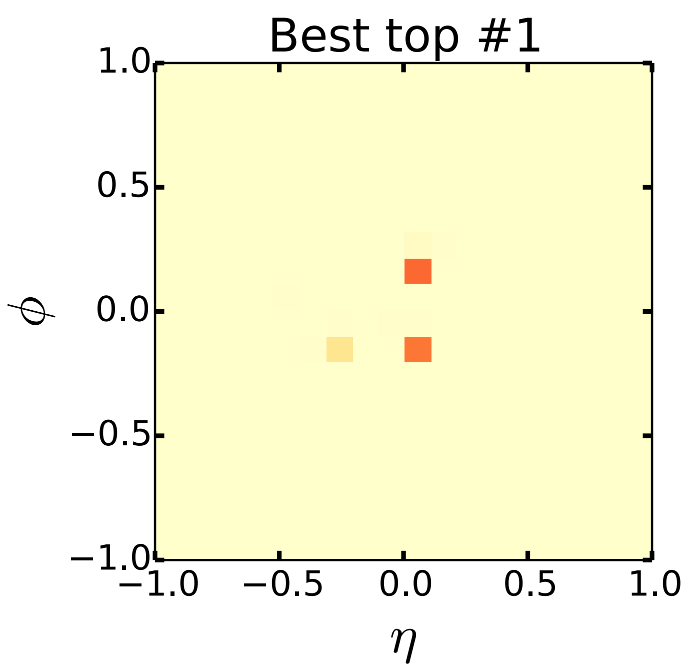
Top Taggers

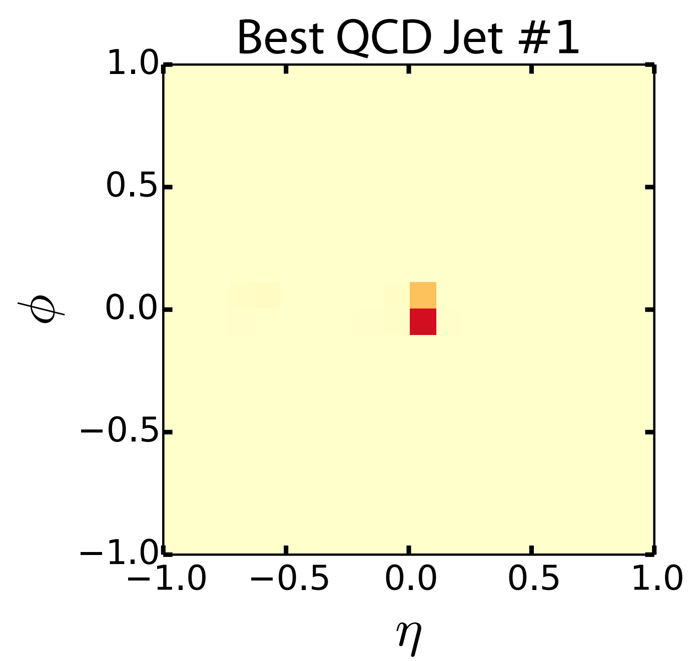
- Since the subject became popular (circa 2009), many jet-substructure observables and "tagging algorithms" have been proposed
- Simplest observable is the jet invariant mass (corrected to remove effects of pile-up, by "pruning", "trimming", etc.)
- Other methods include "N-subjettiness", template algorithms, etc.



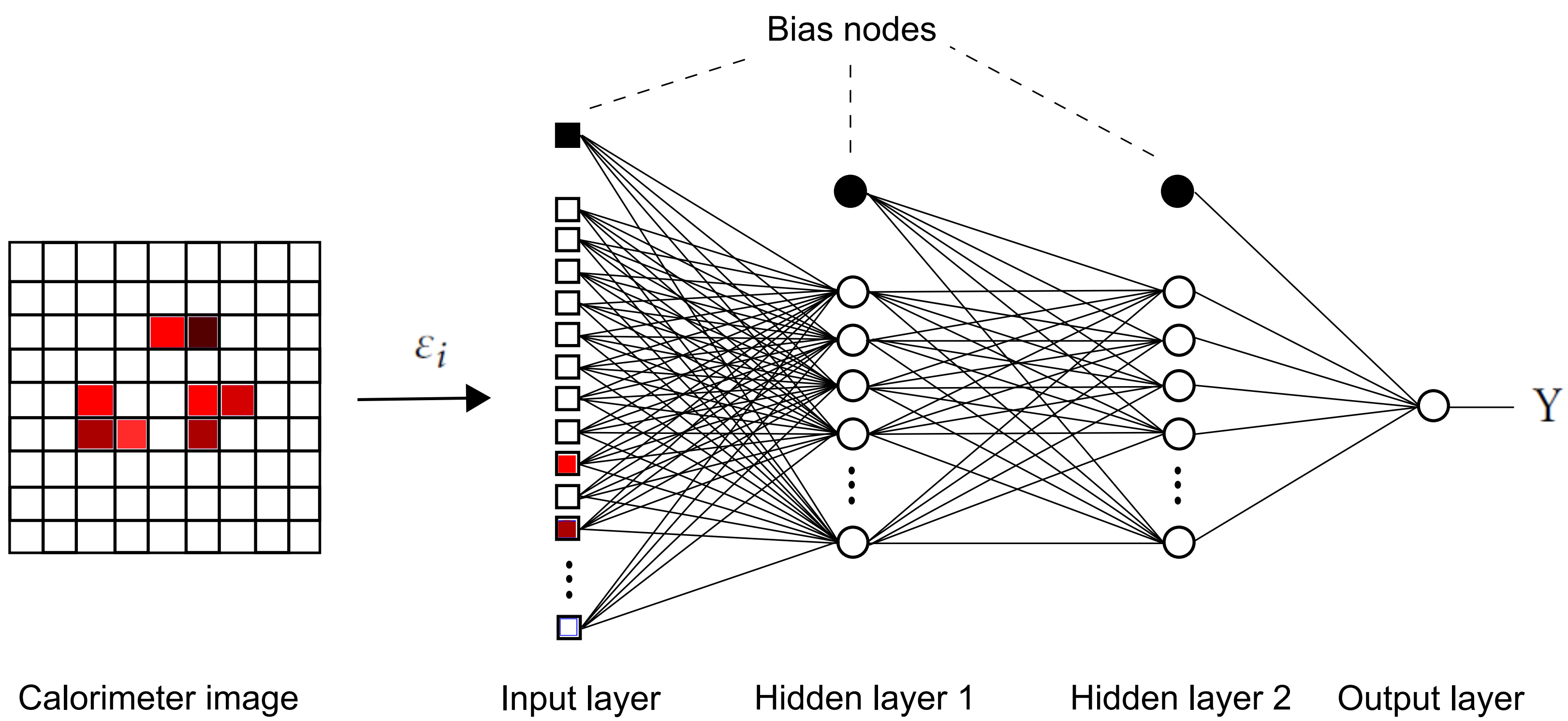
Jet as an Image

- We propose a new algorithm to distinguish top-jets from QCD-jets
- We only use HCAL information
- HCAL output = digital image of the jet: each cell=pixel, energy deposit in each cell = grayscale color/intensity [Cogan, Kagan, Strauss, Schwarzmann, '14]
- Top-jets and QCD-jets make different patterns - apply techniques from pattern recognition (a.k.a. computer vision)! Our algorithm uses Artificial Neural Network (ANN) approach





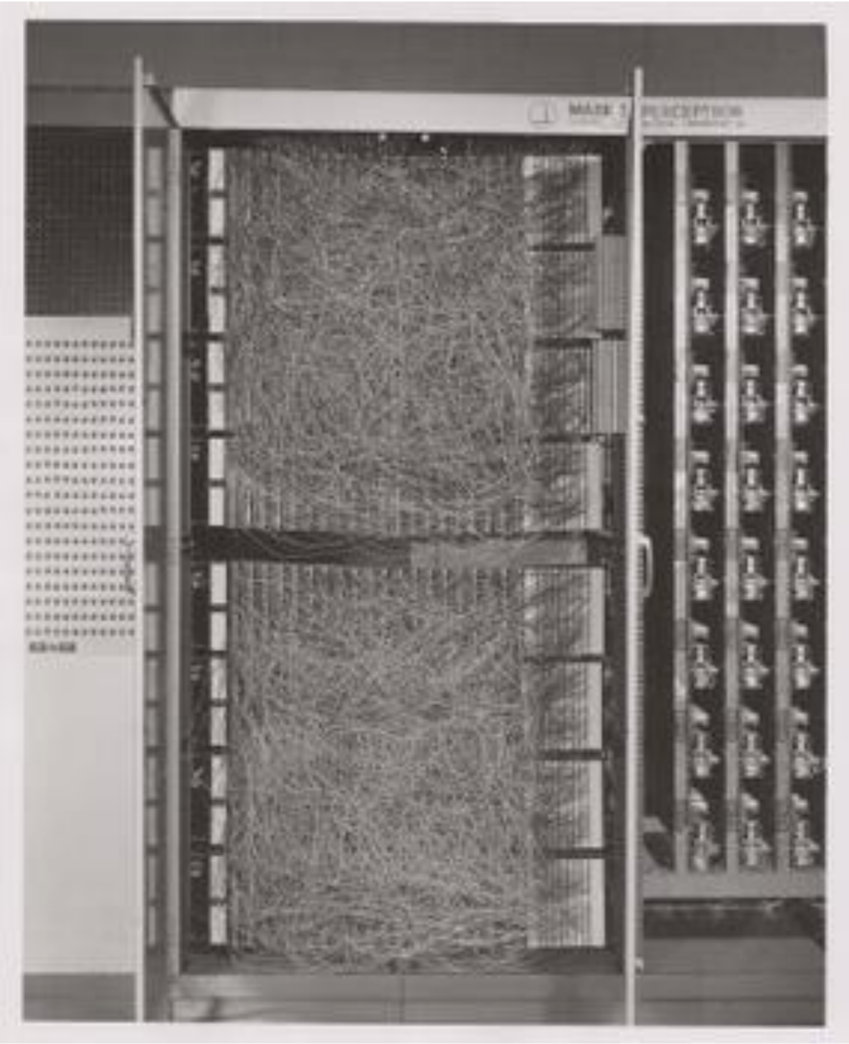
Neural Network Basics



$$\epsilon_i \to h_i^{(1)} = f(W_{ij}^{(1)}\epsilon_j + b_i^{(1)}) \to \cdots \to h_i^{(l)} = f(W_{ij}^{(l)}h_j^{(l-1)} + b_i^{(l)}) \to Y = f(W_j^{(O)}h_j^{(l)} + b^{(O)}), \in [0,1]$$

"Activation Function": $f(z) = \frac{1}{1+e^{-z}}$. (sigmoid)

- ANN is a highly non-linear (but fully deterministic) map from N inputs to 1 output
- Our ANN has 30x30=900 inputs (~0.1x0.1 HCAL cells); 2 hidden layers of 100 nodes each; and 1 output node
- There are ~100,000 "neurons" (connections), each with its own "weight" W



First NN: "Perceptron"
Frank Rosenblatt, Cornell, 1957

Network Training

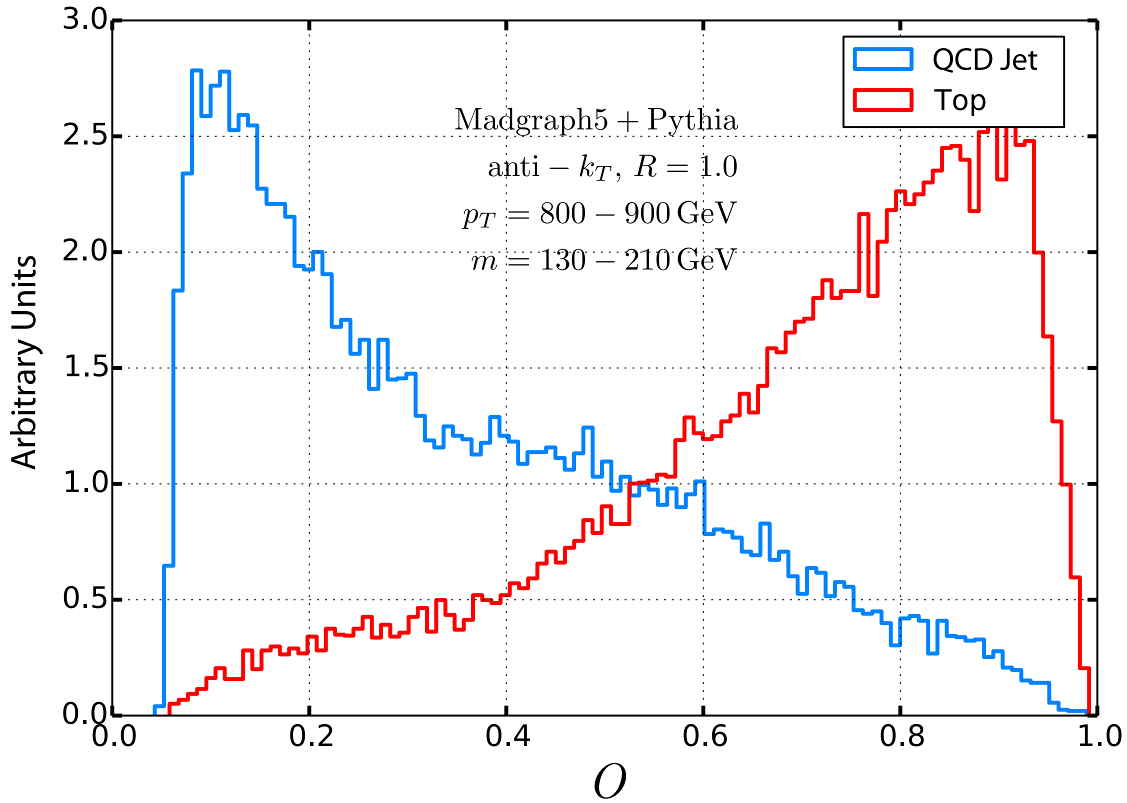
- The weights W are determined through a "training" procedure:
  - Generate large MC samples of top-jets (SM ttbar) and QCD jets (dijet)
  - "Feed" these samples to ANN, record output Y_i for each jet
  - Compute the "error function" (desired outputs: y_i=1 for top, y_i=0 for QCD):

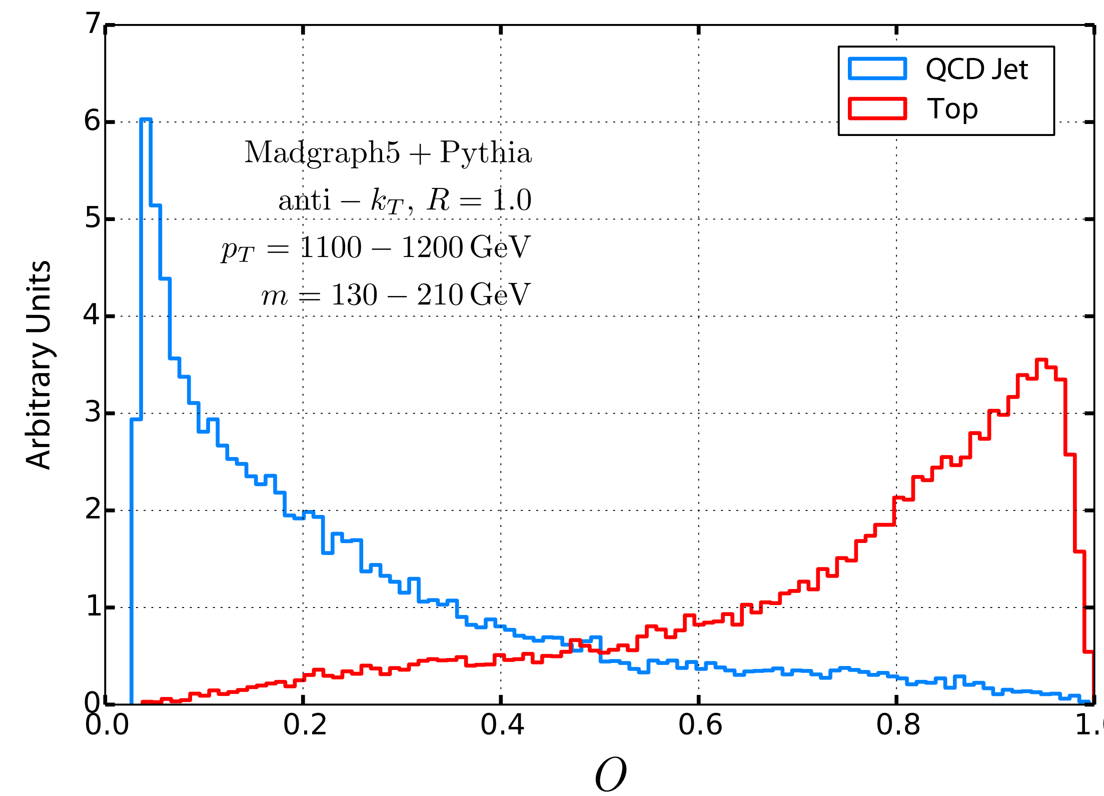
$$\text{Log-loss} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(Y_i) + (1-y_i)\log(1-Y_i)\right].$$

  - Adjust weights iteratively to minimize the error function
  - Minimizing a function of 100,000 variables is not trivial, but there are well-know numerical techniques for this; we use the back-propagation algorithm, with "batch gradient descent with momentum" minimization
- Outcome: a set of weights such that Y_i close to 1 for top jets, close to 0 for QCD jets
- ANN "learns" how to tell them apart, using all available info! (or: it just constructed a complicated but optimal - in some sense - observable)

Network Testing

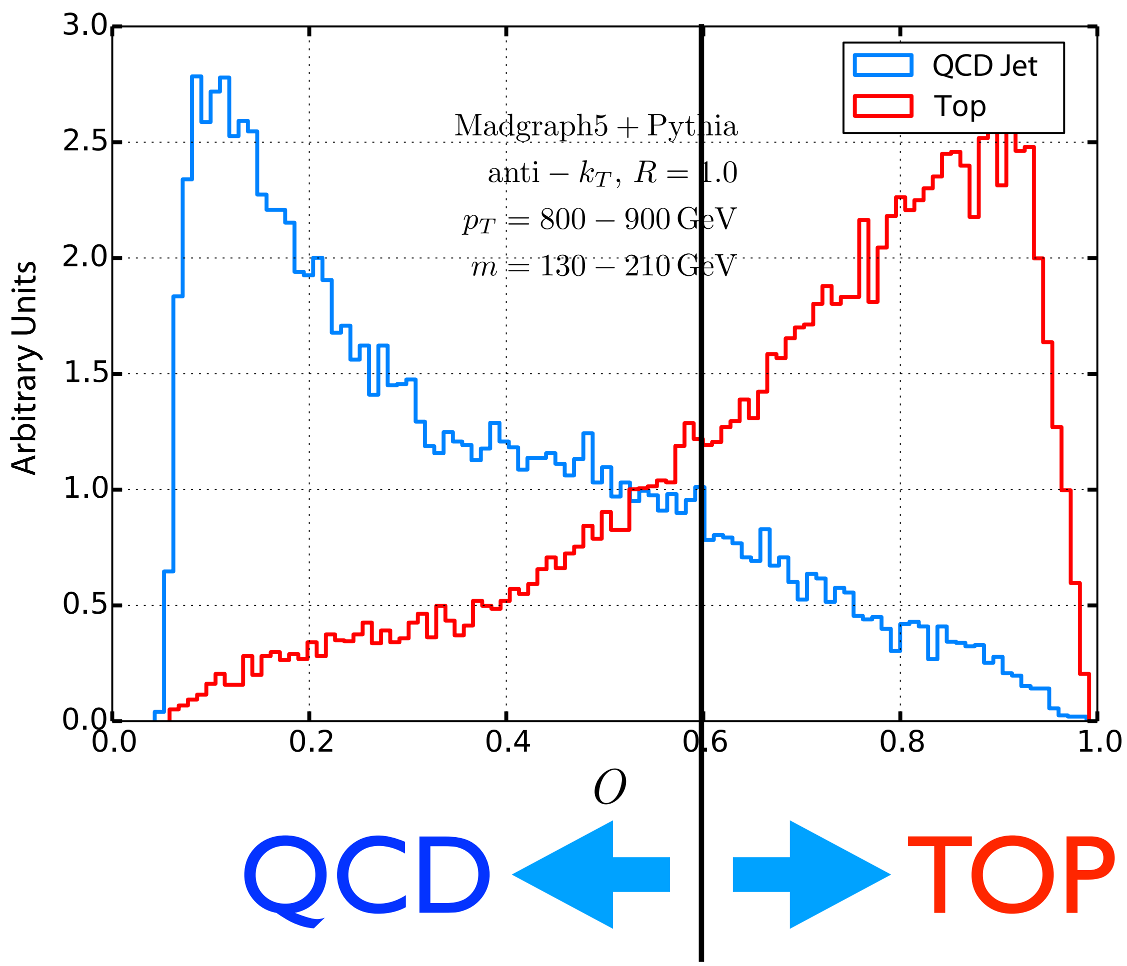
- Once training is complete, all weights and biases are fixed
- Generate a new, independent large MC sample of top and QCD jets
- Feed these jets to ANN and see how well it can tell them apart

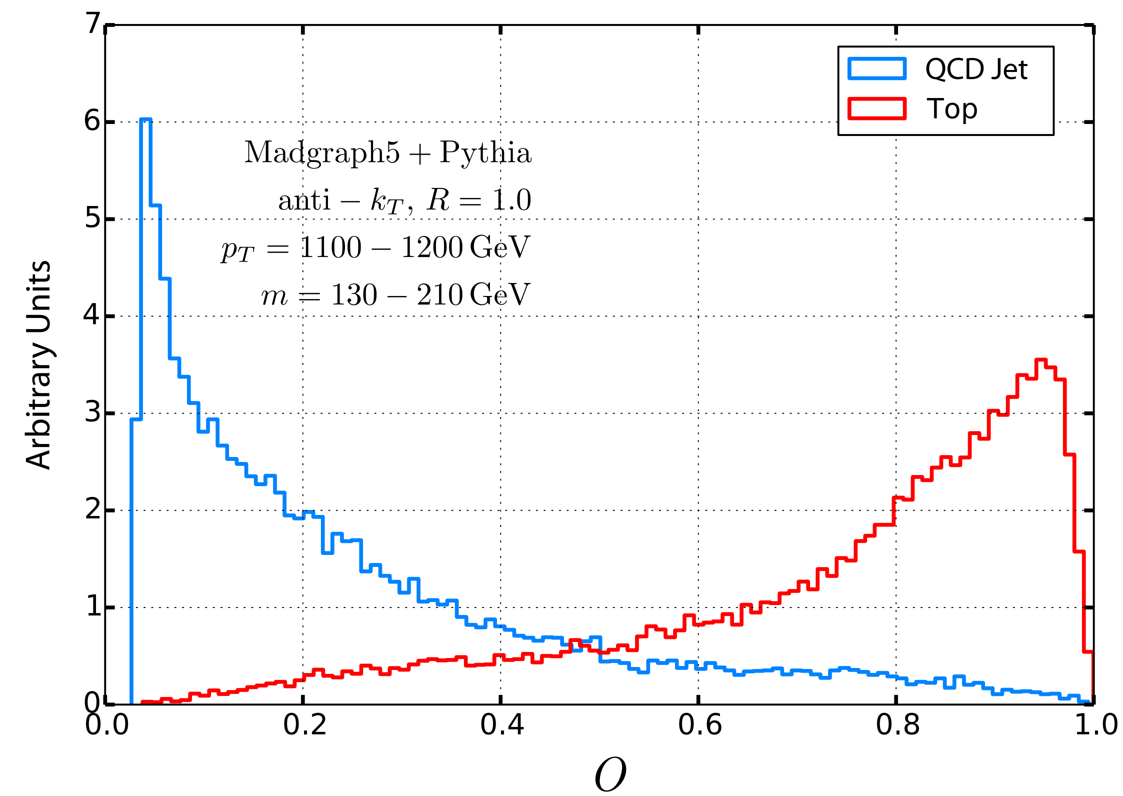




Network Testing

- Once training is complete, all W's are fixed
- Generate a new, independent large MC sample of top and QCD jets
- Feed these jets to ANN and see how well it can tell them apart

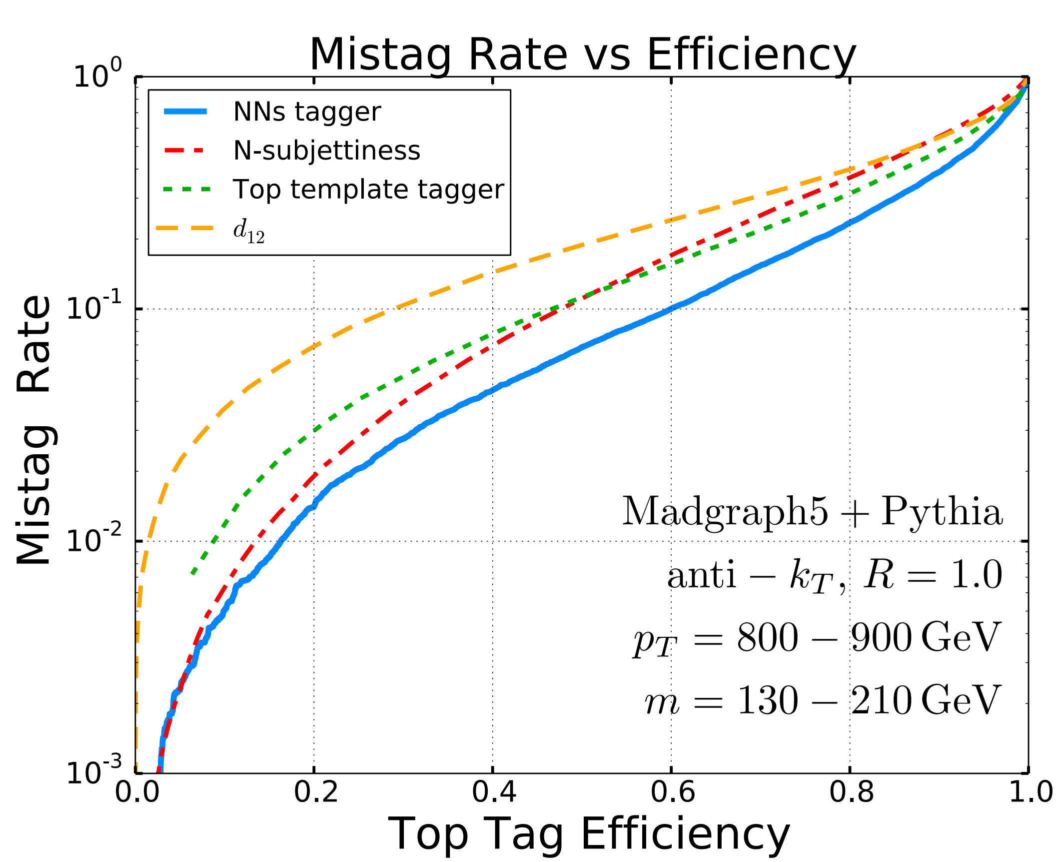


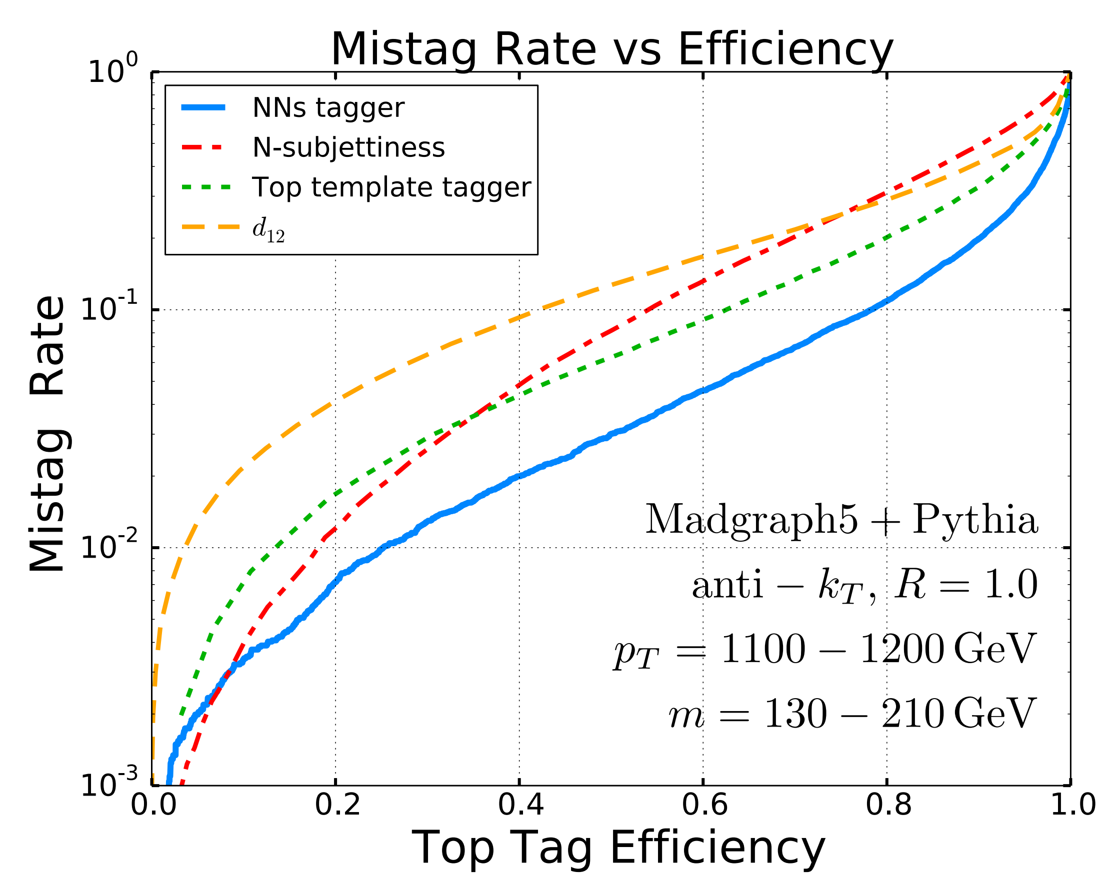


Set tagging threshold, compute efficiency and mistag rate

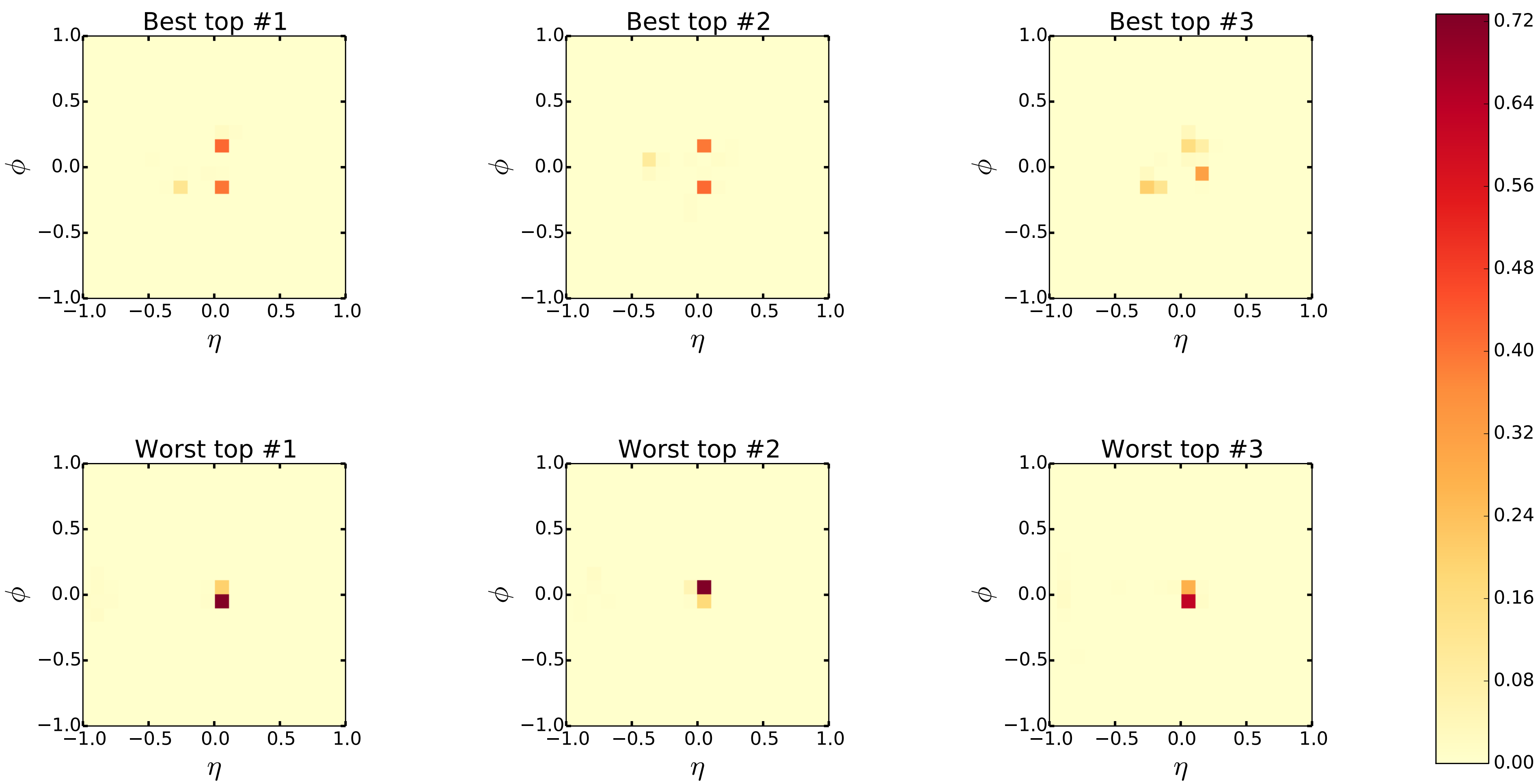
ANN Tagger Performance

- ANN tagger outperforms the "standard" algorithms applied to the same MC samples, especially for high-pT tops



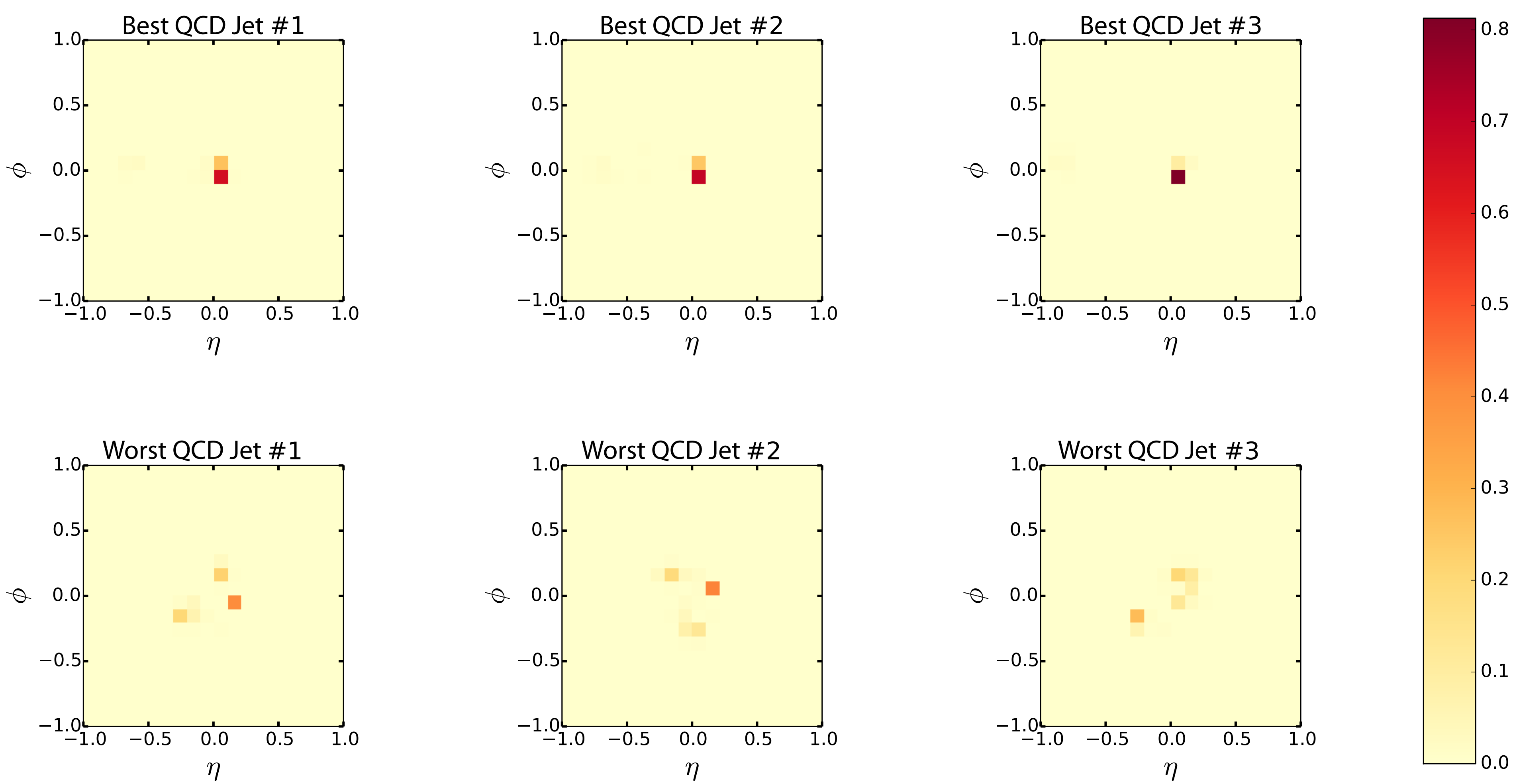


Some Images



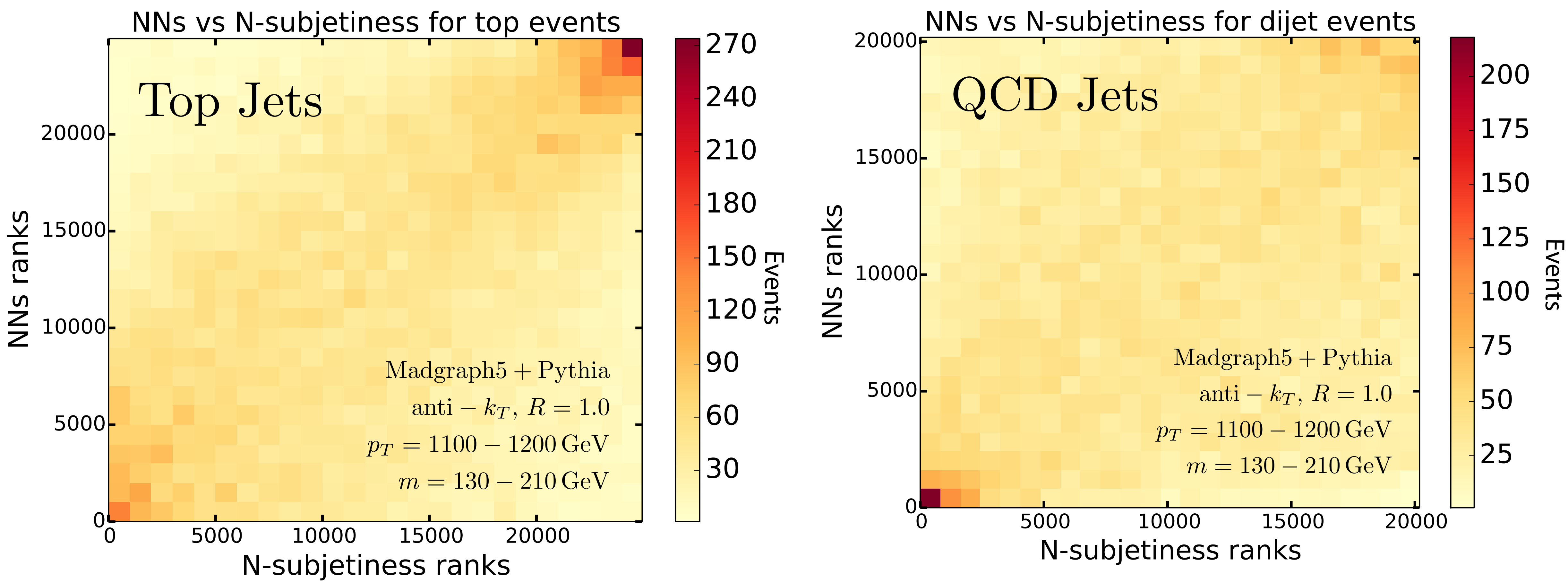
Best top #1
Best top #2
Best top #3
Worst top #1
Worst top #2
Worst top #3
ϕ
η
1.0
0.5
0.0
−0.5
−1.0
0.72
0.64
0.56
0.48
0.40
0.32
0.24
0.16
0.08
0.00

Some More Images



Suggests that the # of "prongs" (subjets) and/or angular size are the dominant discriminants

Correlation with Other Taggers



Tagger	Top		Dijet	
	$p_T \in [500, 600]$	$p_T \in [1100, 1200]$	$p_T \in [500, 600]$	$p_T \in [1100, 1200]$
TOM	0.50	0.52	0.52	0.65
N-sub.	0.59	0.52	0.48	0.31
ATLAS	0.33	0.44	0.42	0.72

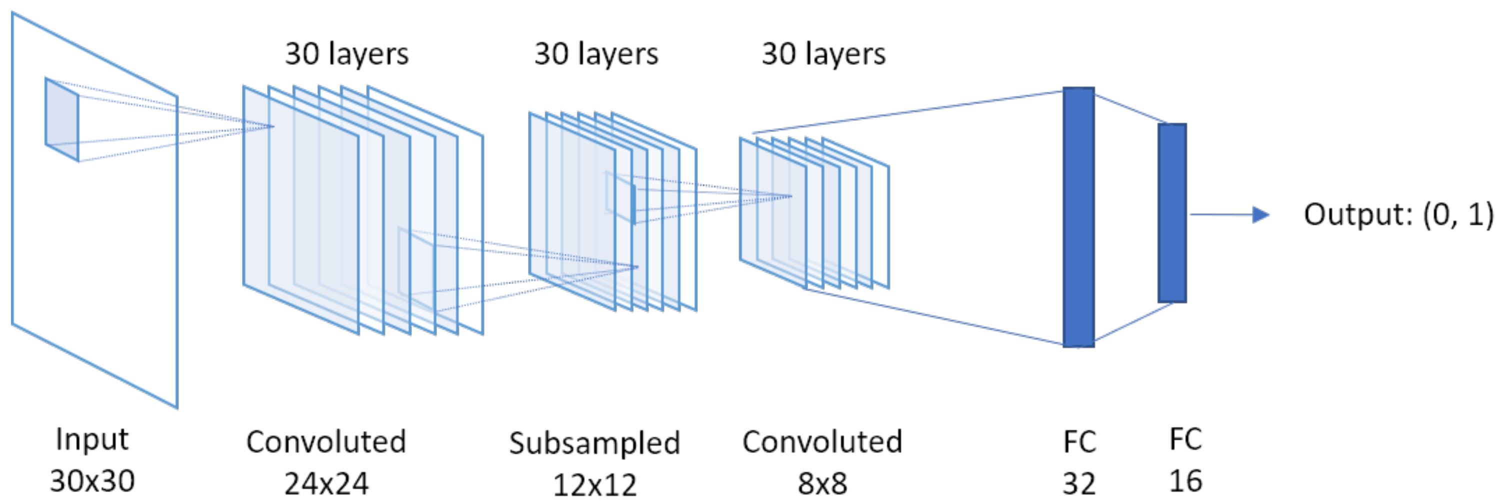
Table 1. Correlation coefficients between the ANN score and the output of alternative taggers, in a variety of samples.

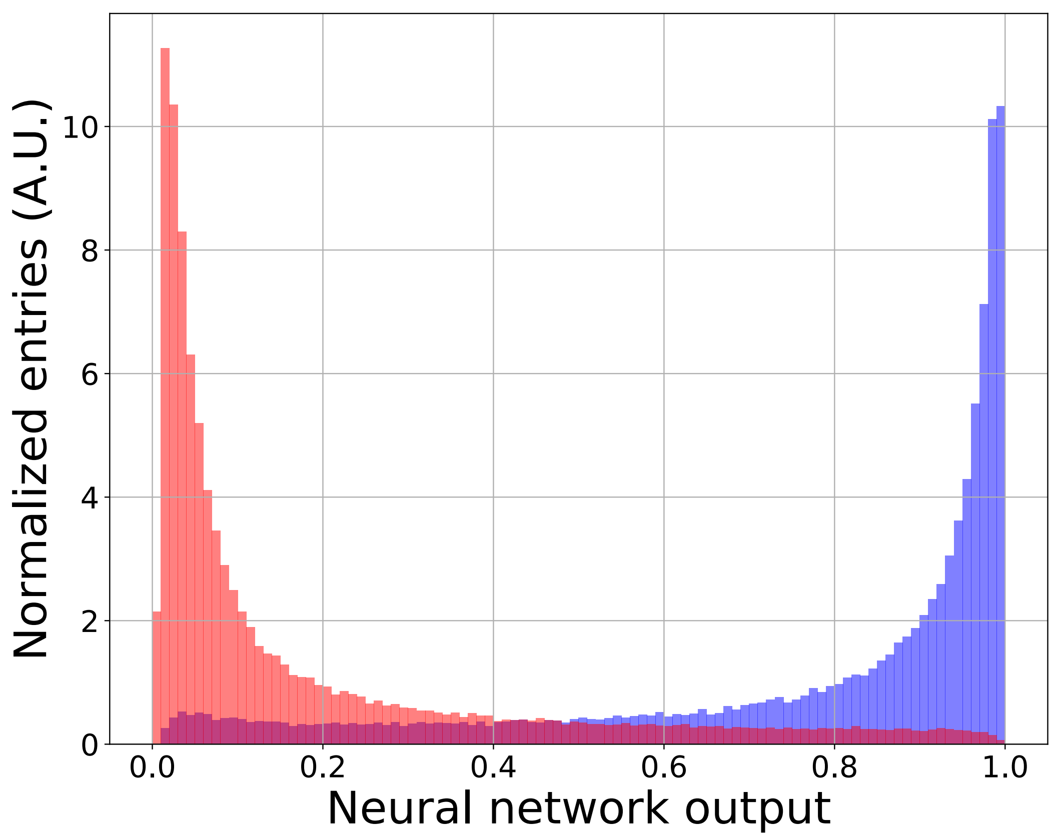
Fairly well correlated… but NN found some additional information not captured by others

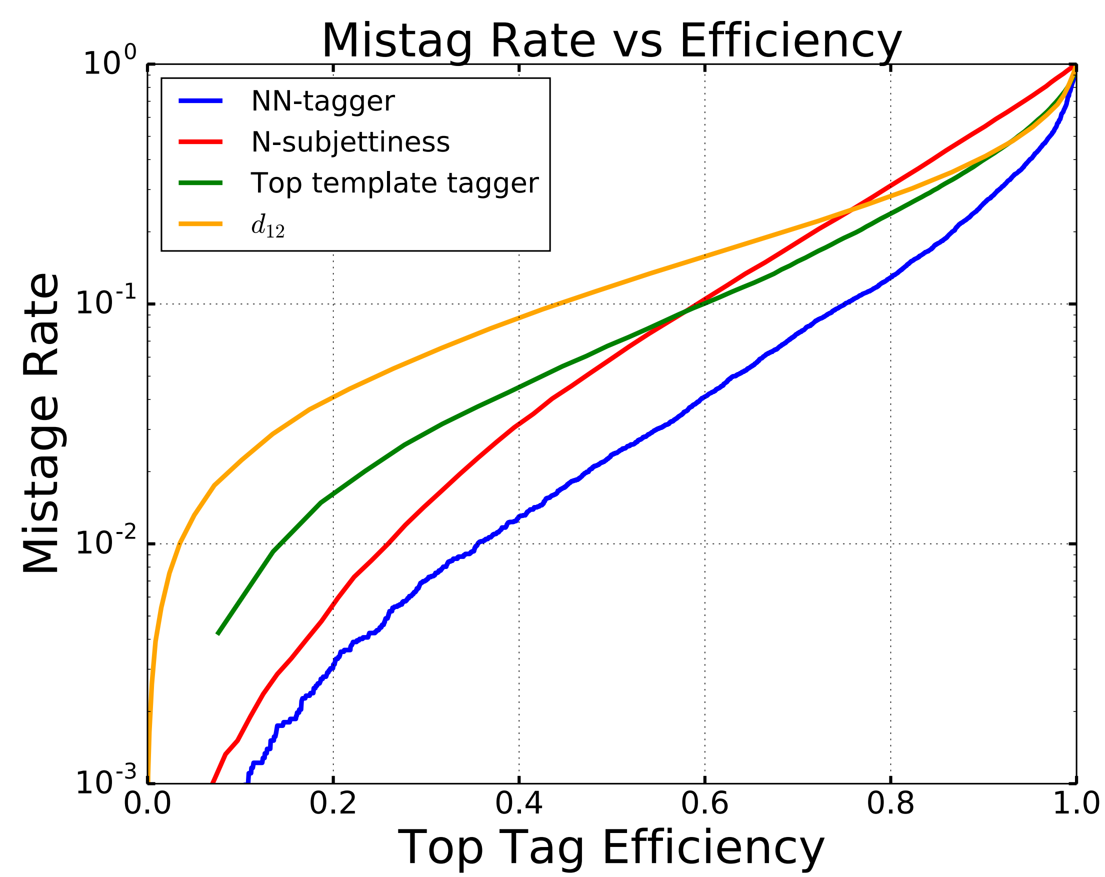
2018 Update: Convolutional NN

[Choi, Lee, MP, '18]

[Software: MXNet]

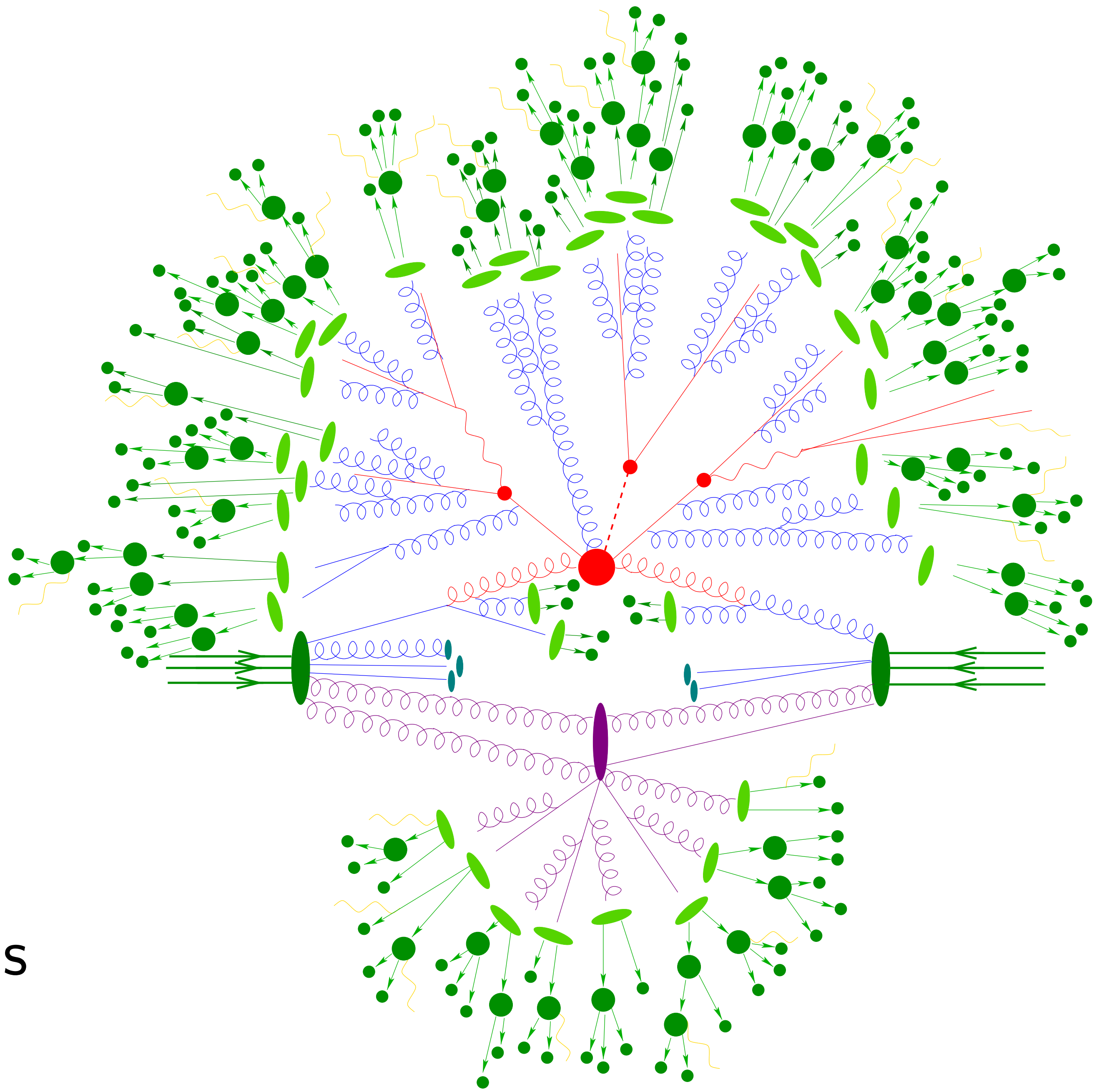






Advanced NN architecture yields improved performance

Scales in High-Energy Collision



[from S. Hoeche]

ls

- 10^-16 cm: core event (e.g. BSM production+decay)

  perturbative expansion
  in coupling constant

- 10^-16->10^-13 cm: parton shower (gluon emission/splitting)

  perturbative expansion in log(Q^2/s);
  independent of new physics

- 10^-13 cm: hadronization (form pions, kaons, etc.)

  non-perturbative QCD;
  requires non-first-principles modeling;
  independent of new physics

- 10^-1 - 10^3 cm: particles interact with detector

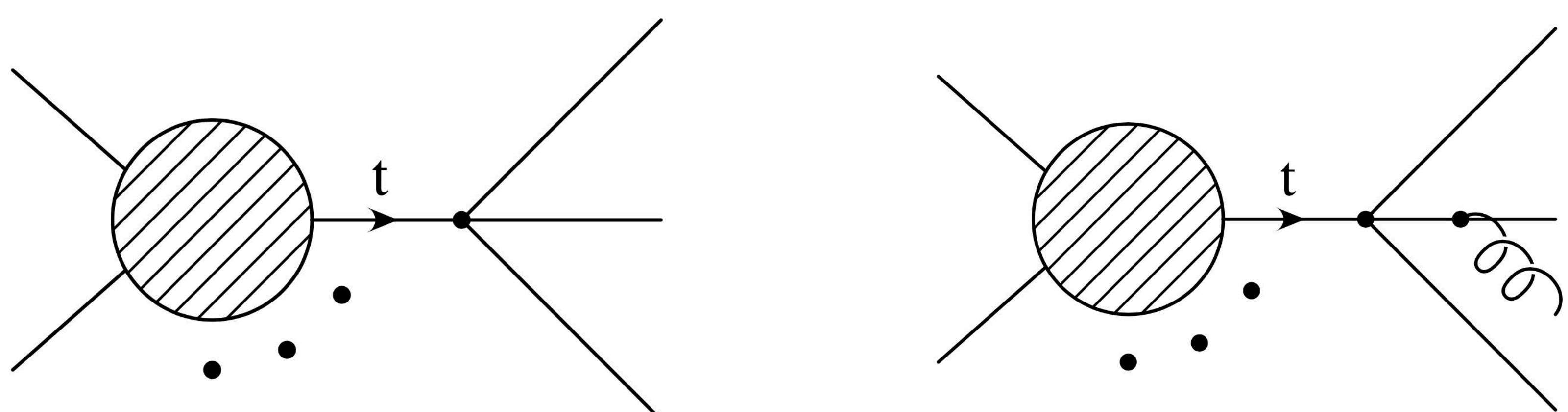
MC Challenge: simulate this multi-scale process

Is NN Learning MC Artifacts?

- NN training and validation used Monte Carlo samples of top/QCD jets
- Since NN map is complicated, it not clear what features are important for tagging, and whether these features are well-modeled by MC
- Data validation is needed (task for experimentalists)
- Necessary condition: NN output must be unaffected by soft/collinear splittings in the parton shower (“Infrared/Collinear Safety”)

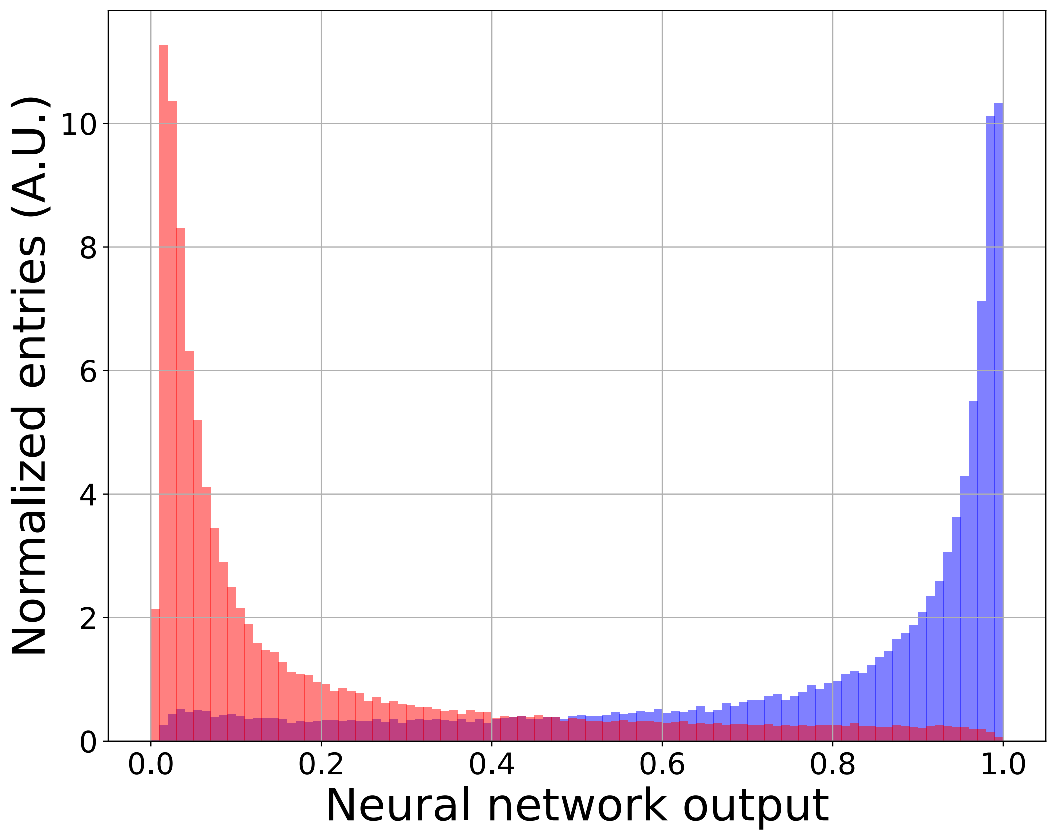
$$\mathcal{O}_n(p_1,\ldots,p_i,p_{i+1},\ldots,p_n) \to \mathcal{O}_{n-1}(p_1,\ldots,p_i+p_{i+1},\ldots,p_n) \quad \text{if} \quad p_i \cdot p_{i+1} \to 0$$

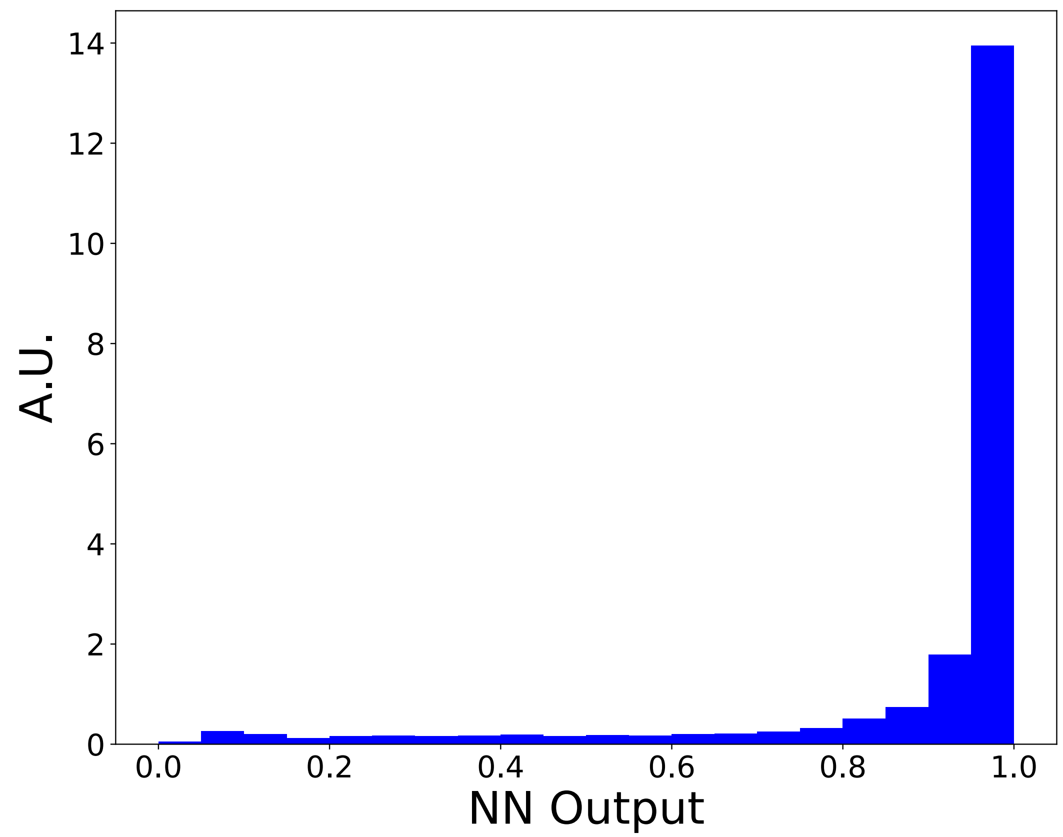
- To test IRC safety, we apply NN tagger to parton-level samples, compare output with and without an extra soft/collinear parton



(a) (b)

Tagging Parton-Level Events

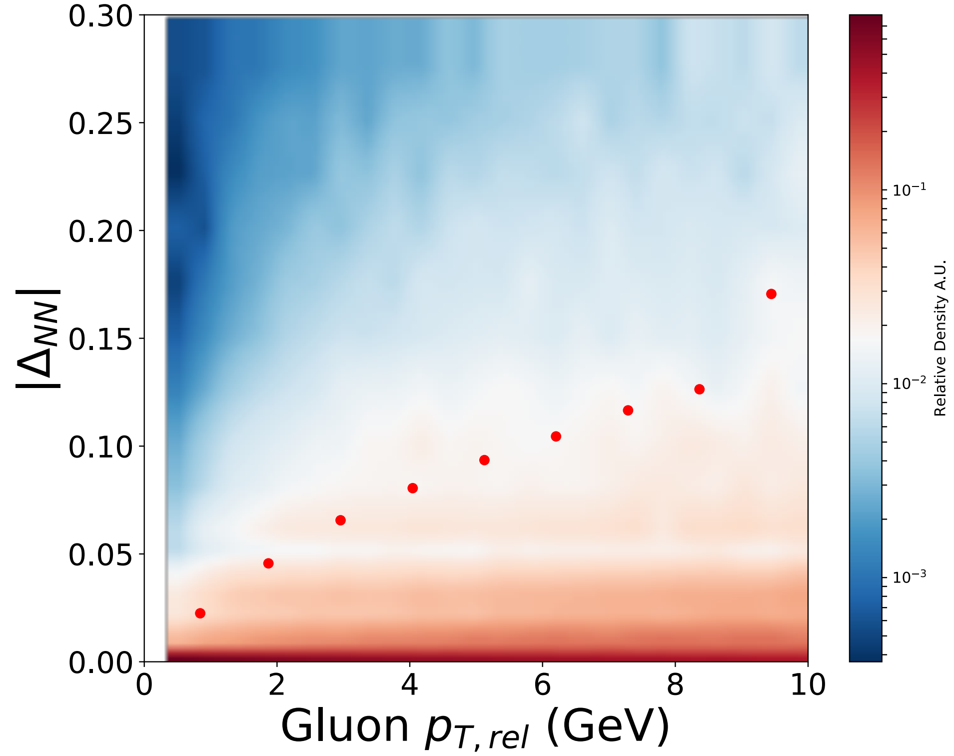


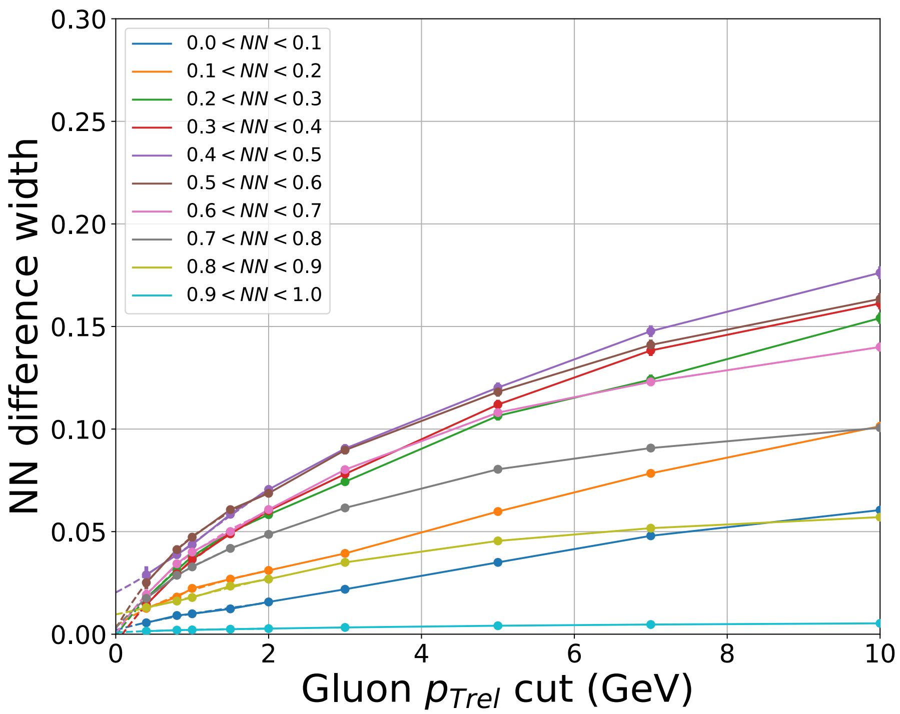


particle-level events — same NN — parton-level events

- CNN tagger trained on particle-level events was applied to parton-level top events
- Similar output distribution indicates that most of the important information is already present in parton-level events

Infrared/Collinear Safety





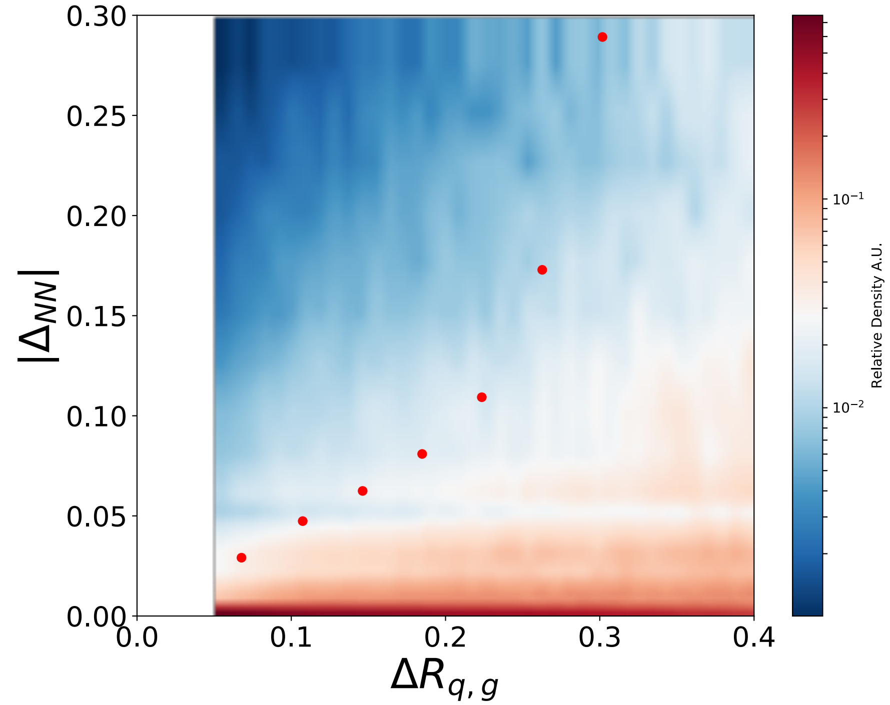
- Plot difference in NN output on parton-level events with/without extra gluon, as a function of the gluon's "relative pT":

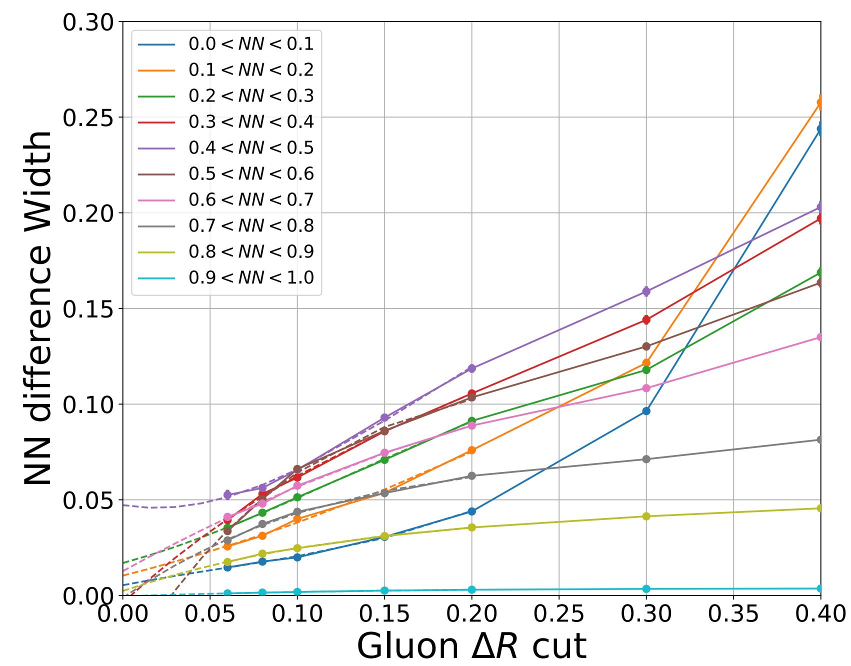
$$p_T^g = \left| \mathbf{p}_g - \frac{\mathbf{p}_g \cdot \mathbf{p}_q}{|\mathbf{p}_q|^2} \mathbf{p}_q \right|$$

- Observed convergence of the NN output with/without extra gluon in the IRC limit - numerical confirmation that the observable defined by the NN is IR-safe

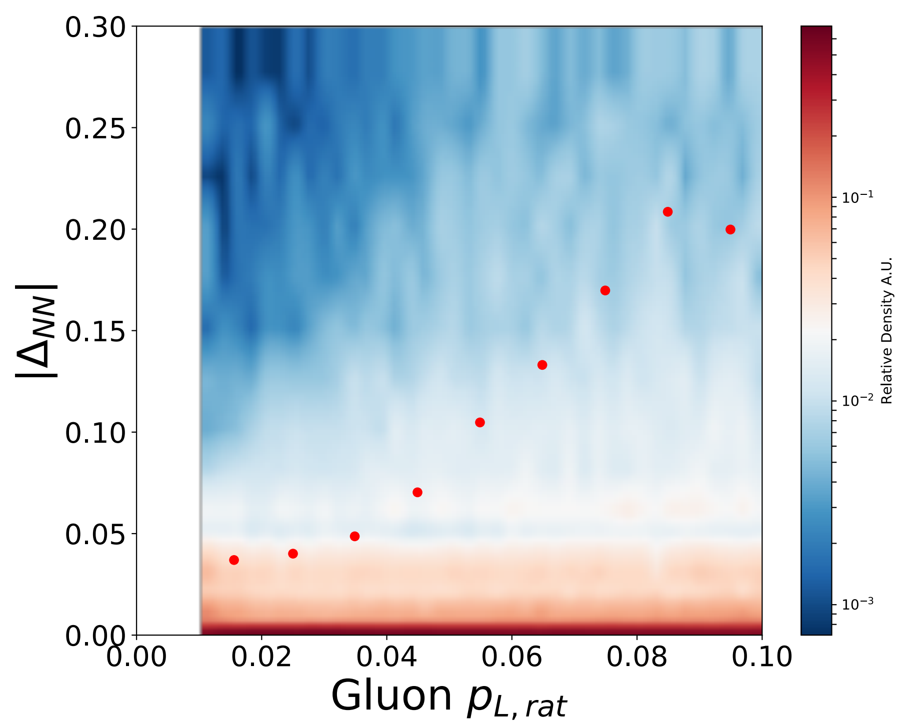
Infrared AND Collinear Safety

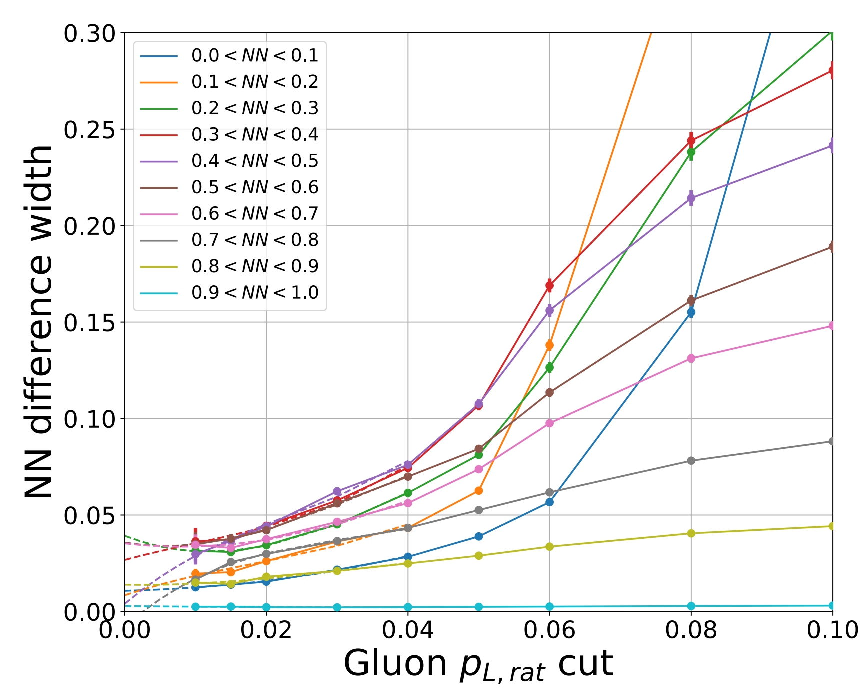
collinear
limit





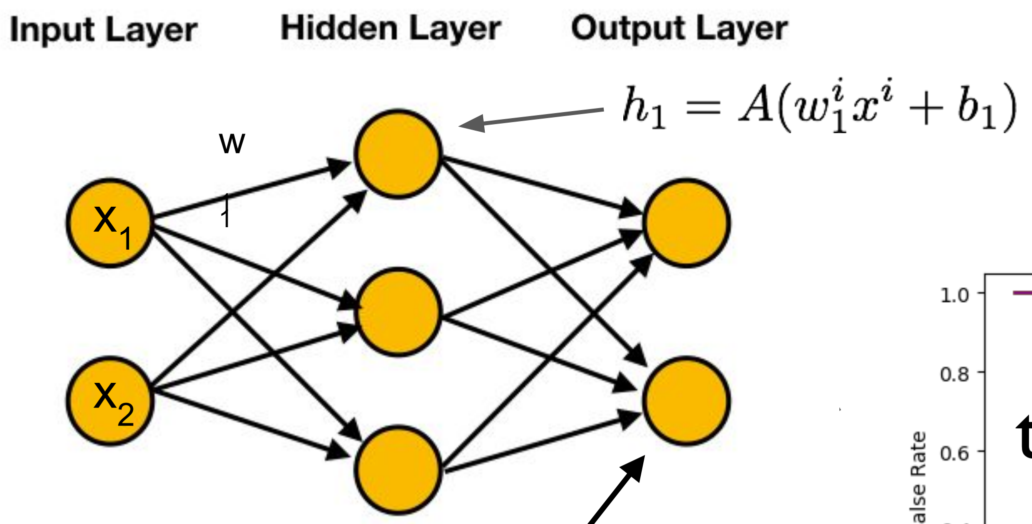
soft
limit



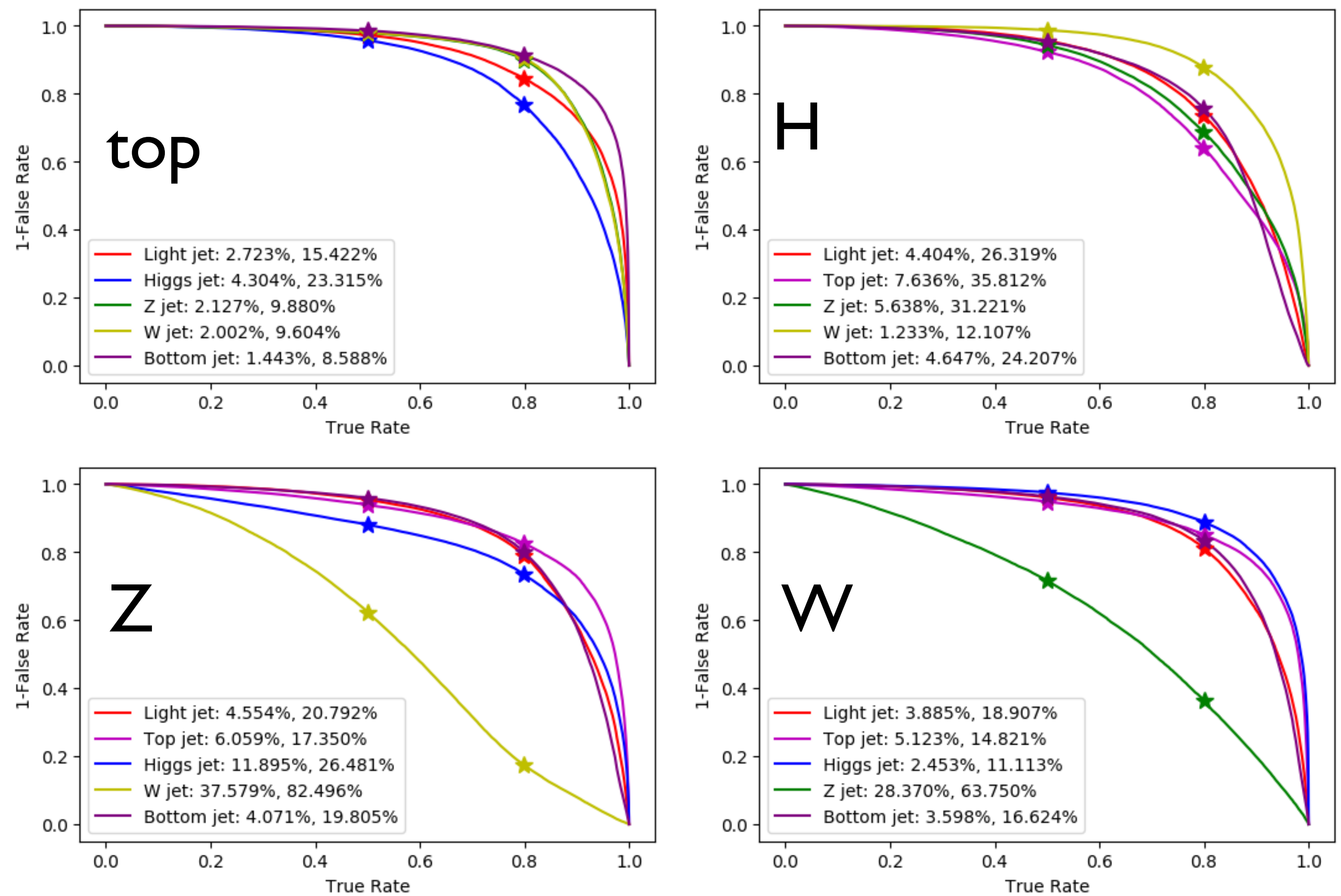


“Multi-Dimensional” Tagging

[Csaki, De Freitas, Li, Ma, MP, Shu, 1811.01961, Appendix C]



6 output nodes:
top, Higgs, W, Z, b, QCD jets



Talk 1: Conclusions/Outlook

- Jet substructure tagging (e.g. top vs. QCD jets) is essentially an image recognition problem
- Neural Network seems a natural candidate to tackle this
- Simple NN outperforms existing taggers/observables in MC studies, correlated with traditional observables but contain extra information (2015)
- Convolutional NN performs even better (2018)
- NN output seems "IR safe" to a good approximation - MC is probably not misleading
- Studies with real data in progress in ATLAS/CMS (e.g. B. Nachman et.al.)
- "Multi-dimensional tagging" (top/Higgs/W/Z/QCD jets) is also possible

Talk 1: Boosted Top Tagging with Neural Networks

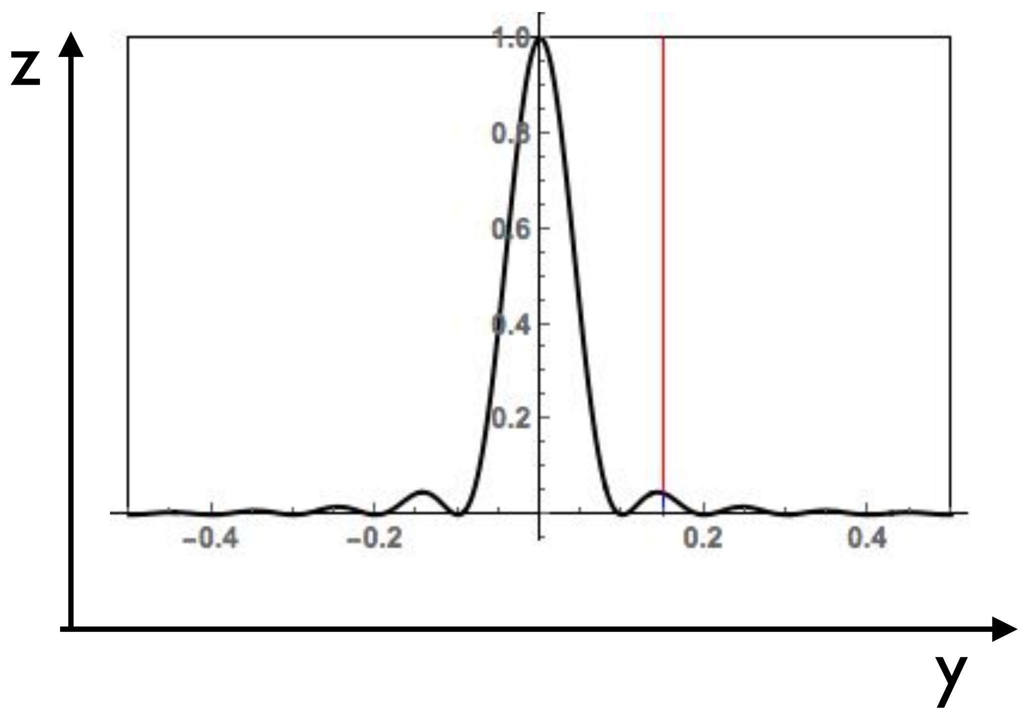
Almeida, Backovic, Cliche, Seung Lee, MP, 1501.05968
S. Choi, S. Lee, MP, 1806.01263

Talk 2: Monte Carlo Simulations with Neural Networks

Matthew Klimek, MP, 1810.11509

MC Simulation/Integration

- Monte Carlo Problem: Given a function f(y), such that $f(y) \geq 0$, generate a set of "random" points {y_i} with density proportional to f(y).
- In particle physics, typically y=phase space points, f(y)=differential cross section or decay rate, {y_i}=Monte Carlo sample ("pseudo-experiment")
- Most Naive MC algorithm: randomly select points in 2D box, discard the points with z > f(y).
- Fraction of points that are actually used = "unweighting efficiency": $\epsilon(y) = \frac{f(y)}{f_{\max}}$



integration: $\int f(y)dy = f_{\max} \int \epsilon(y)dy$

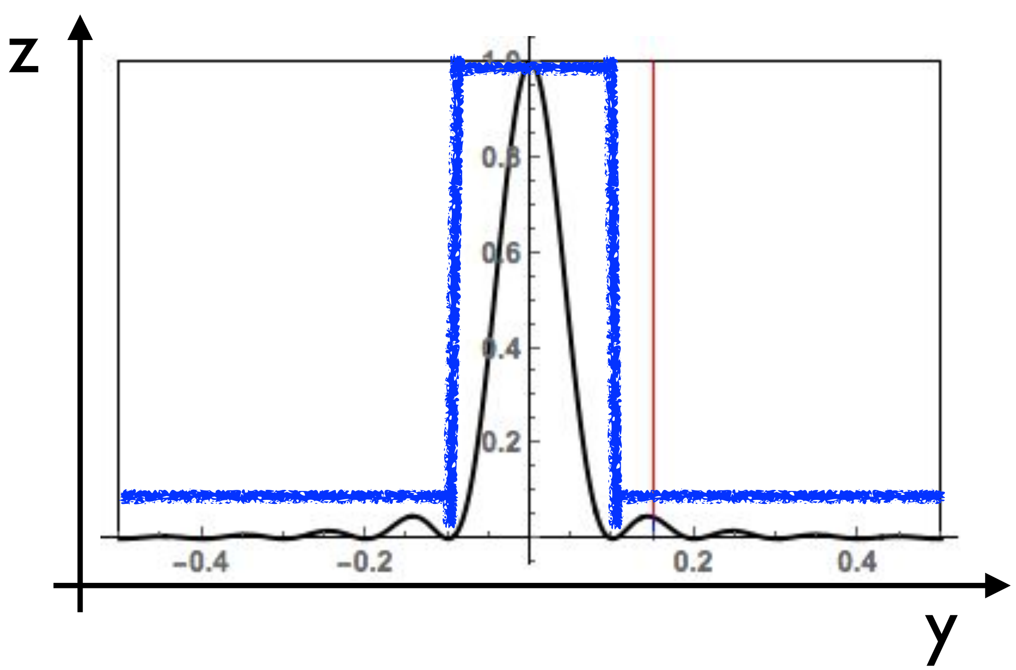
Problem: Resonances, Collinear/Infrared Singularities

$\rightarrow \epsilon \ll 1$

In modern applications, $f(y)$ is often numerically expensive to evaluate (e.g. NNLO - may require numerical integrations)

Importance Sampling

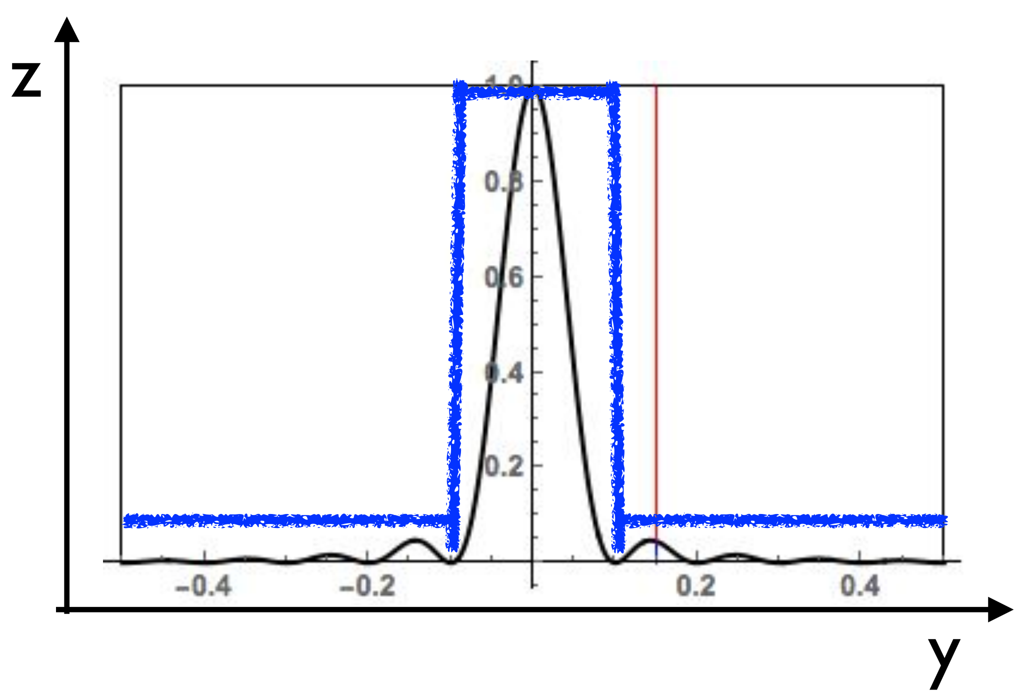
- Classic solution: construct a number of "bounding boxes" in yz plane, covering the function's domain, with heights adjusted to correspond to local values of f(y)
- Classic implementation: VEGAS [Lepage, 1978]
- Divide the domain into N bins, roughly compute "weight" = $\int_{\text{bin}} \epsilon(y)dy$ in each bin
- Iteratively adjust bin boundaries until each bin contains the same weight
- Simulation: choose a bin at random (equal probabilities), then follow Naive algorithm in that bin. Repeat.

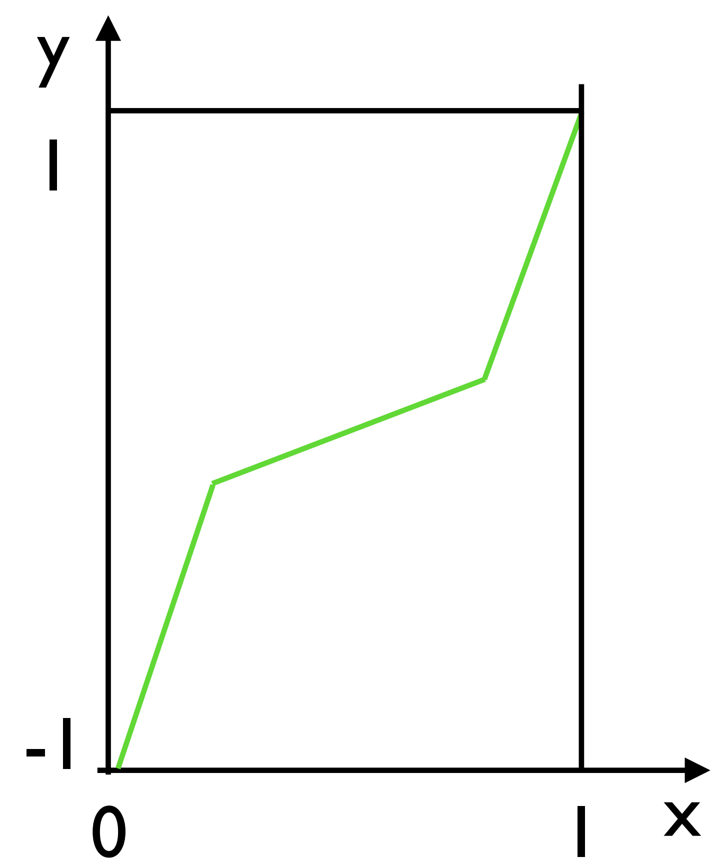


Construct a piecewise-constant approximation to f(y), then sample from that distribution

Importance Sampling as a Map

- Importance sampling can also be described as a map from "input space" x to "target space" y
- Randomly choose $x \in [0,1]$ (uniform distribution)
- Deterministic, piecewise-linear map $x \to y(x)$
- Equivalent to "pick a box + random point within the box"
- Unweighting: keep the point with probability $P(y) = f(y)\left|\frac{dy}{dx}\right|$





MC with Neural Networks

- Idea: Generalize importance sampling from piecewise-linear to nonlinear maps
- Simulation would be 100% efficient if we found a nonlinear map such that

$$\left|\frac{dy}{dx}\right|^{-1} = f(y)$$

- Generalization to functions in N dimensions (same dimensionality for input and target spaces, =dimensionality of phase space)

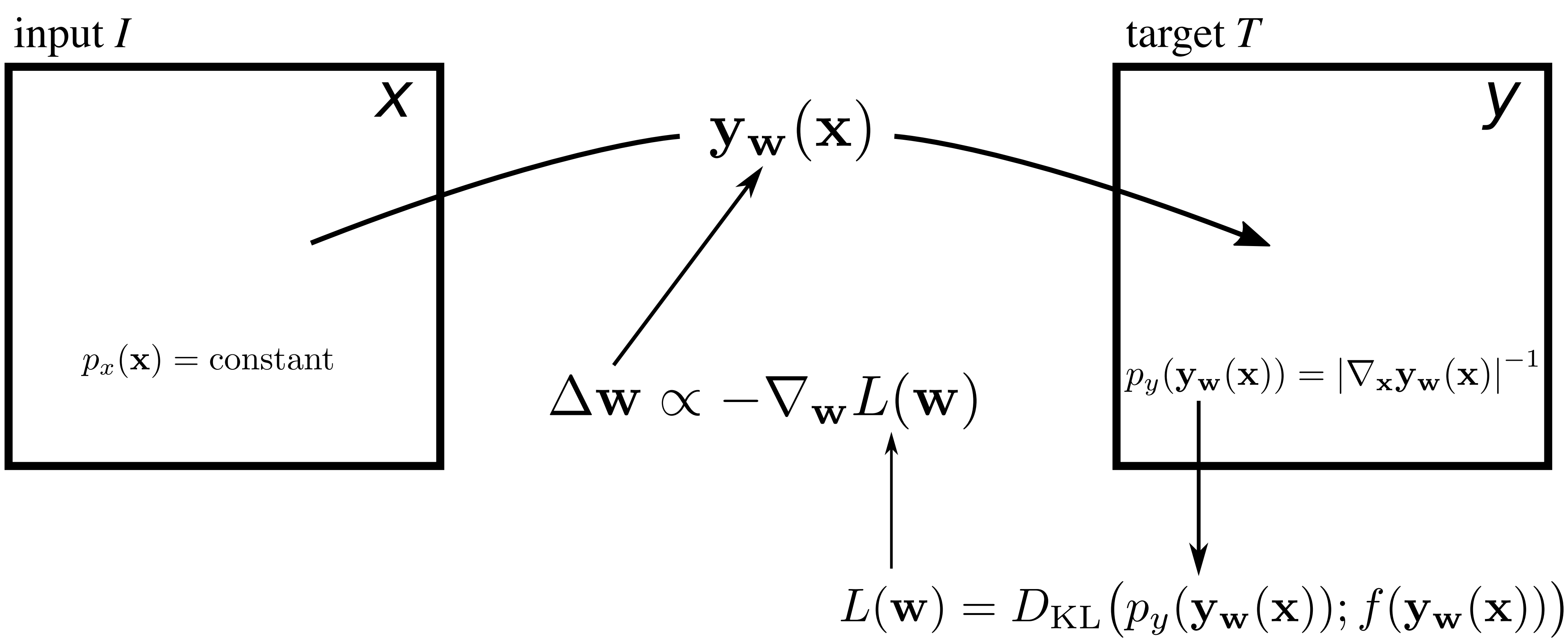
$$J = \det \frac{\partial y_i}{\partial x_j}, \qquad |J|^{-1} = f(y)$$

- Universal Approximation Theorem: under mild assumptions, a neural network can approximate any continuous functional map $\mathcal{I}_N \to \mathcal{I}_N$ (where $\mathcal{I}_N$ is an N-dimensional hypercube)

[Cybenko, '89; Hornik, '91]

- This makes a NN a natural choice to implement nonlinear importance sampling

MC with Neural Networks



- Error function: Kullbeck-Leibler divergence between $|J|^{-1}$ and $f(y)$

$$D_{\mathrm{KL}}[p_y(\mathbf{y}); f(\mathbf{y})] \equiv \int p_y(\mathbf{y}) \log \frac{p_y(\mathbf{y})}{f(\mathbf{y})}\, d\mathbf{y}$$

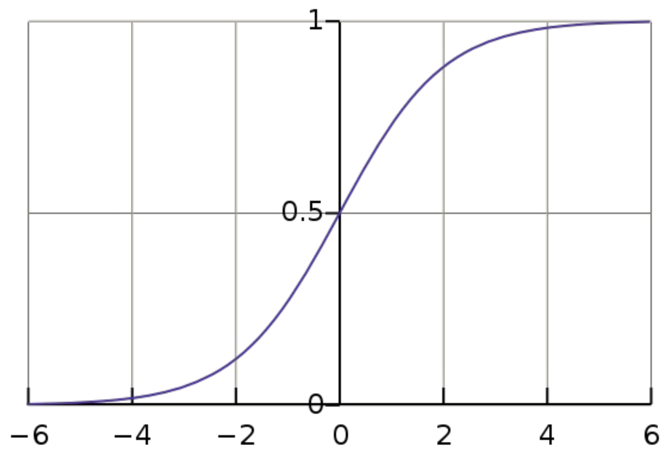
- Training: generate a batch of 100 points, compute D_{KL}, adjust weights, iterate

Output Functions

- An important subtlety is the choice of output function (=activation function for the last layer)

sigmoid:

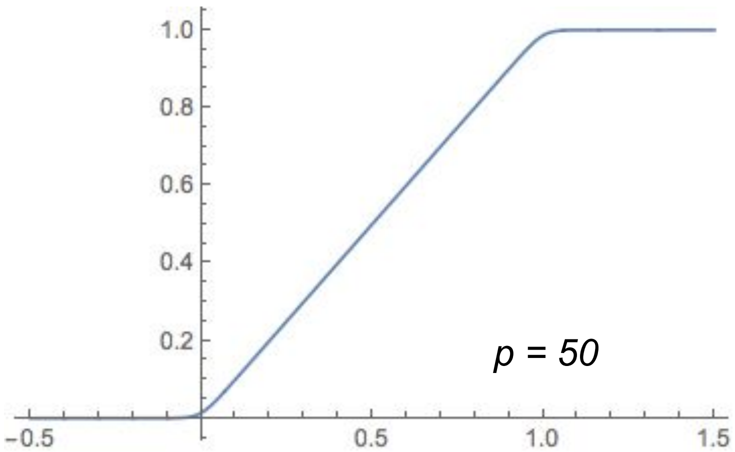
$$S(x) = \frac{1}{1+e^{-x}}$$

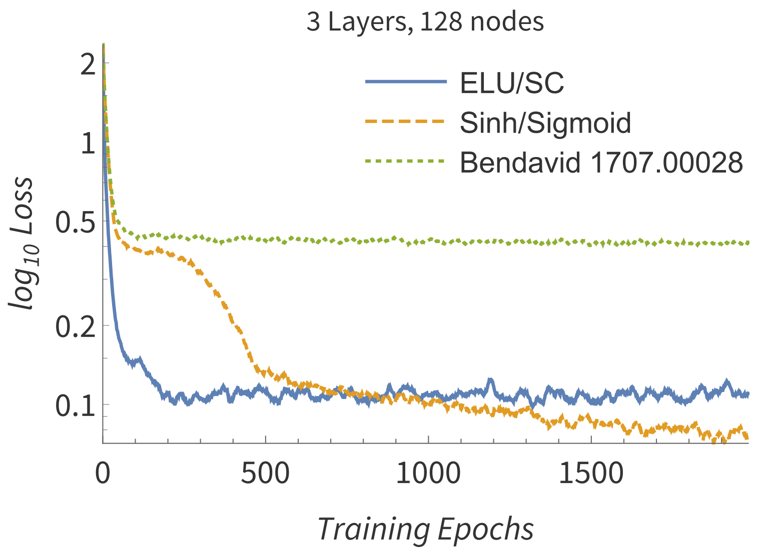


Approaches asymptotic values slowly → hard to populate the edges of phase space

"soft clipping function":

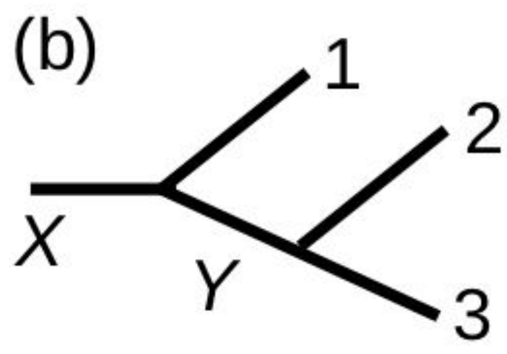
$$SC(x) = \frac{1}{p}\log\left(\frac{1+e^{px}}{1+e^{p(x-1)}}\right)$$

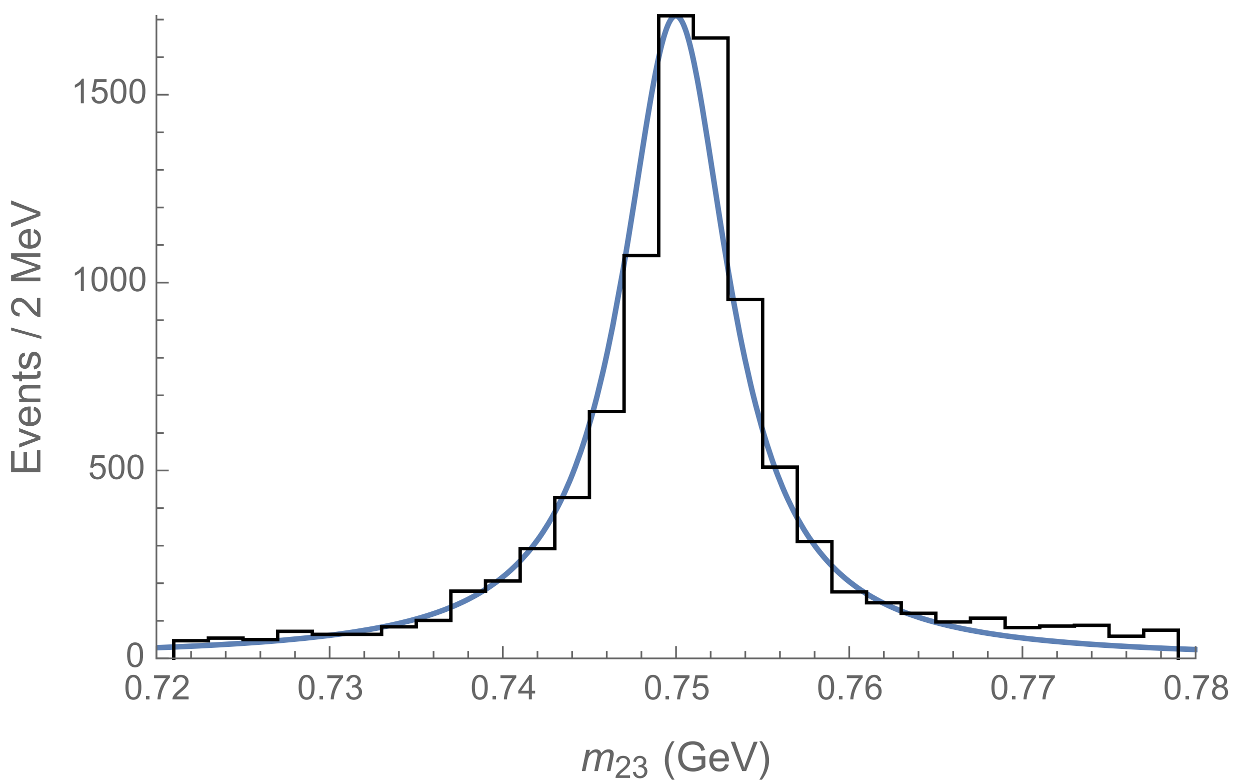




Sample Applications

- Simulate 3-body decay of a scalar X, with a resonance Y

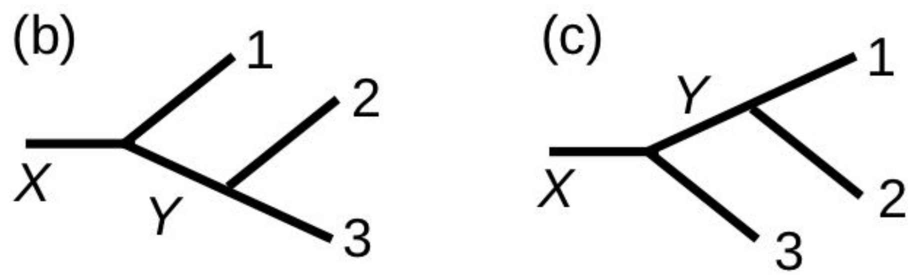




- Choose phase-space coordinates $m_{23}, \theta_{1(23)}$
- Simulated with $\Gamma_Y/m_Y = 10^{-2}, 10^{-3}, 10^{-4}$
- Achieved unweighting efficiency 30-70%, depending on resonance width
- MadGraph (off-the-shelf) efficiency: 6%

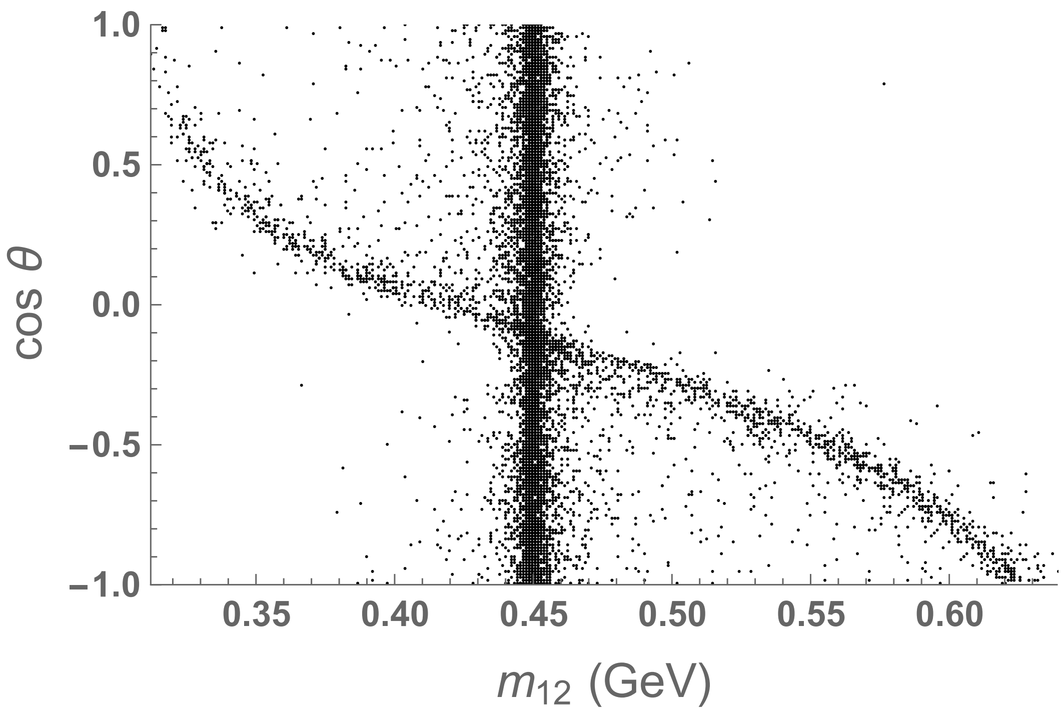
Sample Applications

- Simulate 3-body decay of a scalar X, with resonances in two channels

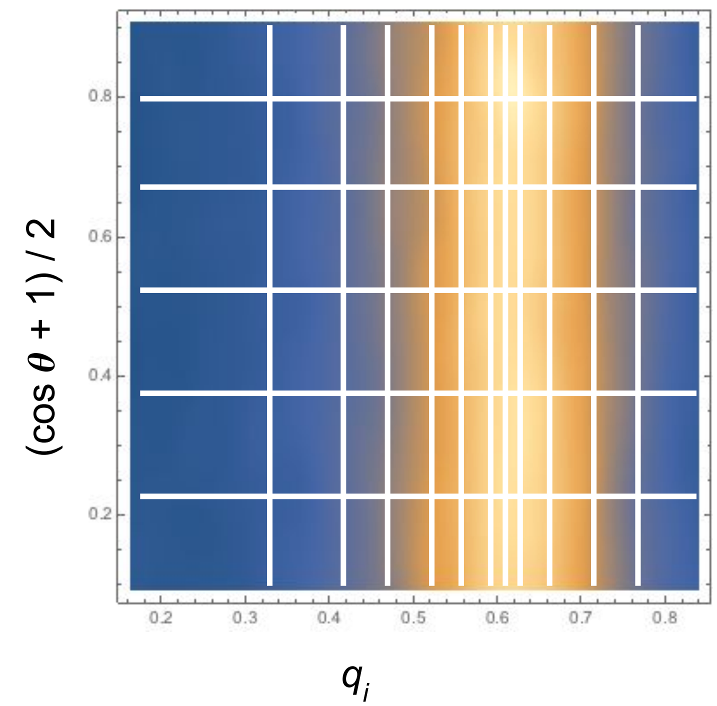


- NN was able to learn both the feature aligned with coordinate axis, and the feature with complicated shape in these coordinates

- In contrast, VEGAS needs each feature to be aligned with a coordinate axis (coordinate choice handled separately by "multi-channeling")



NN output



VEGAS grid/output

Sample Applications

- A more realistic example: $e^+e^- \to q\bar{q}g$

$$\frac{d\sigma}{dm^2_{qg}dm^2_{\bar{q}g}} \propto \frac{(s-m^2_{qg})^2 + (s-m^2_{\bar{q}g})^2}{m^2_{qg}m^2_{\bar{q}g}},$$

- Soft/collinear singularities $\Rightarrow$ need to impose kinematic cuts
- Simple rectangular cuts aligned with target-space coordinates can be simply handled by redefining the target space boundaries
- In practice we need to be able to handle more general cuts:

$$Y \geq Y_{cut} \quad \text{where} \quad Y = Y(y_1, \ldots, y_N)$$

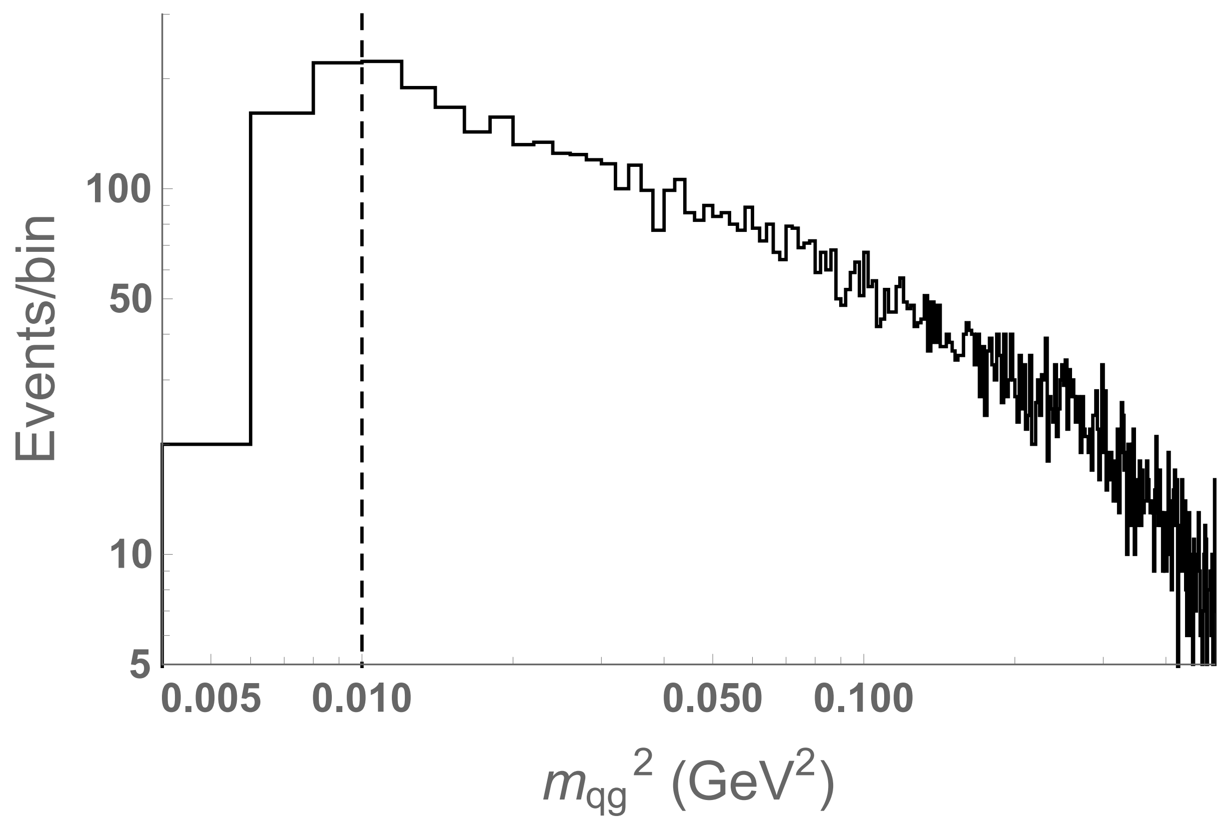
- Naively, we could just replace $f(\mathbf{y}) \to \theta(Y(\mathbf{y}) - Y_{cut})\, f(\mathbf{y})$
- However NN target function must be differentiable! So we opt for

$$f(\mathbf{y}) \to \kappa(Y(\mathbf{y}) - Y_{cut})\, f(\mathbf{y}) \qquad \text{with} \qquad \kappa(x) = \begin{cases} 1 & x > x_{\text{cut}} \\ (x/x_{\text{cut}})^n & x < x_{\text{cut}} \end{cases}$$

Sample Applications

- A more realistic example: $e^+e^- \to q\bar{q}g$

$$\frac{d\sigma}{dm_{qg}^2 dm_{\bar{q}g}^2} \propto \frac{(s - m_{qg}^2)^2 + (s - m_{\bar{q}g}^2)^2}{m_{qg}^2 m_{\bar{q}g}^2},$$



- In this example, we used n=8.
- Unweighting efficiency is 70% (vs. 4% for off-the-shelf MadGraph)

Talk 2: Conclusions/Outlook

- Neural Network seems a natural candidate to realize "nonlinear importance sampling"
- With a bit of tweaking (e.g. proprietary "soft clipping" output function), we got it to work
- Can handle resonances, in a nicely coordinate-choice-independent way
- Can handle soft/collinear enhancements, generic kinematic cuts
- High unweighting efficiency achieved in all examples
- This may be a crucial advantage in situations when matrix element is computationally expensive to evaluate
- Next: Integration with automated Matrix Element calculators
- Next-to-next: Parton showers? NLO?