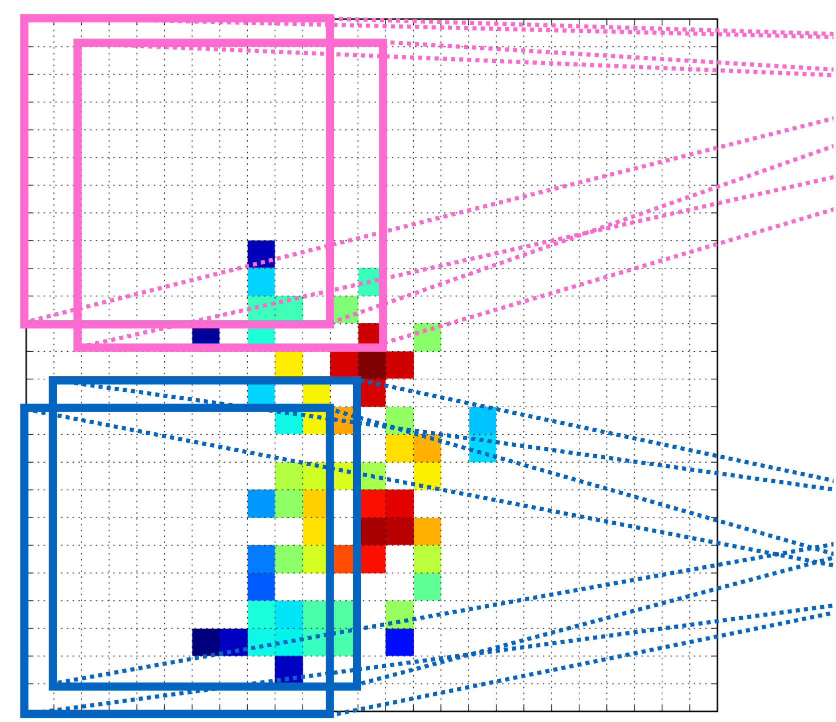
Machine Learning with Less or no Simulation Dependence

Benjamin Nachman

Lawrence Berkeley National Laboratory

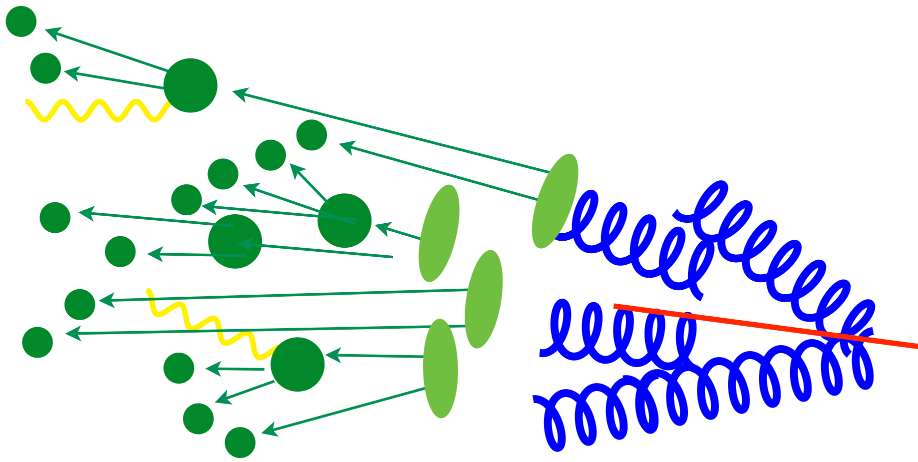
BERKELEY LAB

BERKELEY EXPERIMENTAL PARTICLE PHYSICS



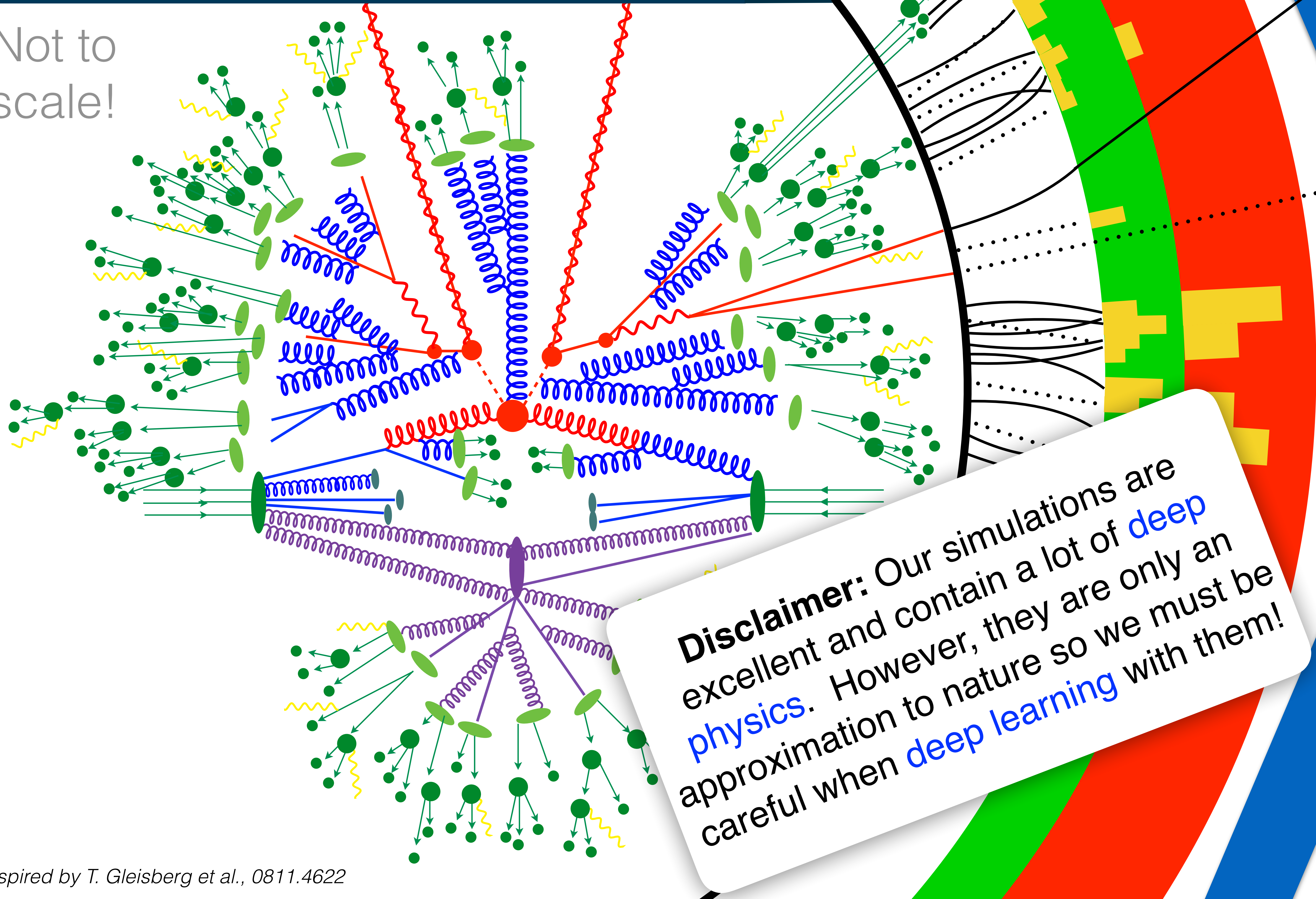
Outline

- Simulation dependence in traditional ML4HEP
- Classification
  - ✦ Adversarial approaches
  - ✦ Weak supervision
- Regression
- Anomaly Detection



HEP Simulations

Not to scale!

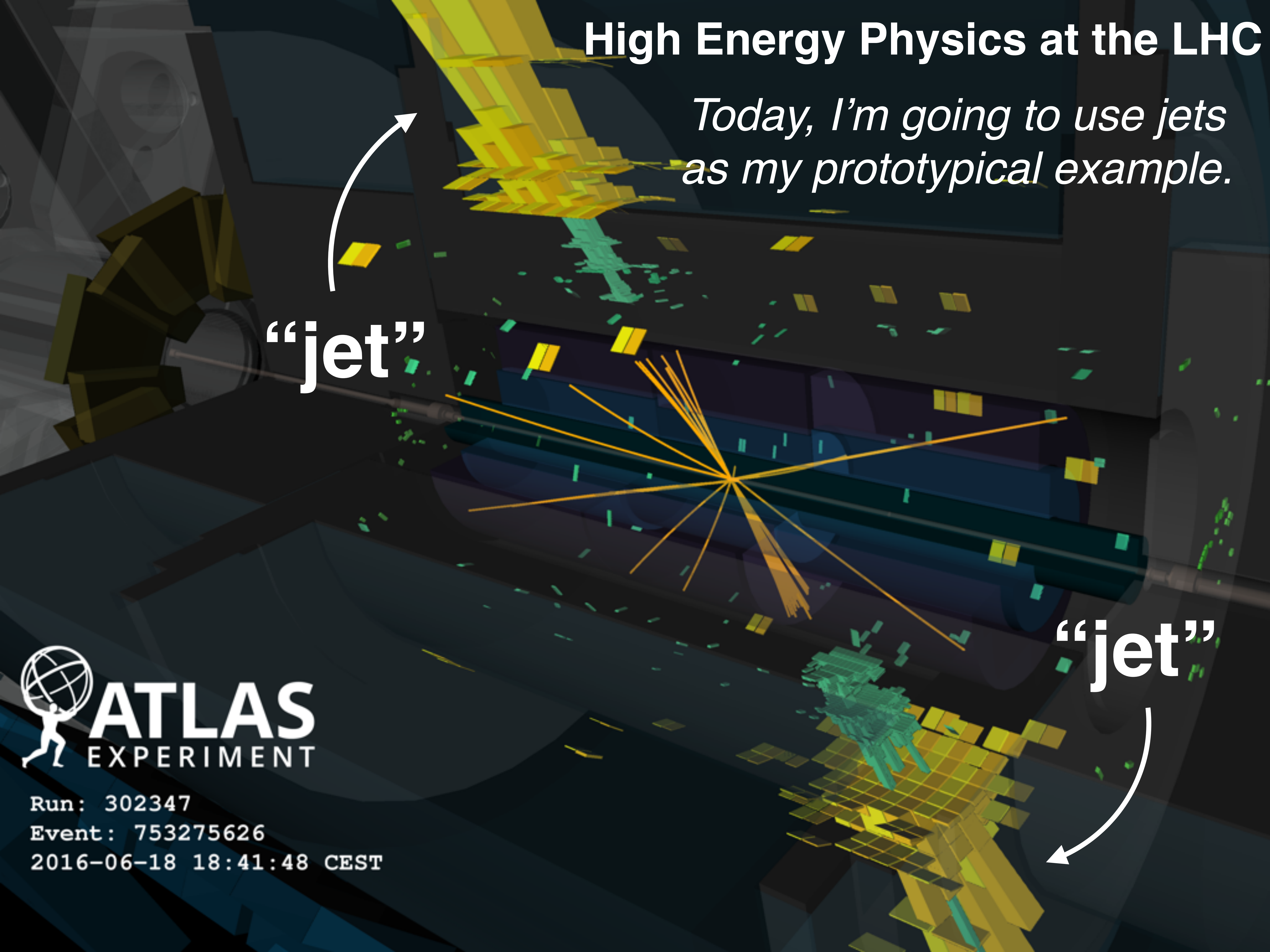


Disclaimer: Our simulations are excellent and contain a lot of deep physics. However, they are only an approximation to nature so we must be careful when deep learning with them!

Inspired by T. Gleisberg et al., 0811.4622

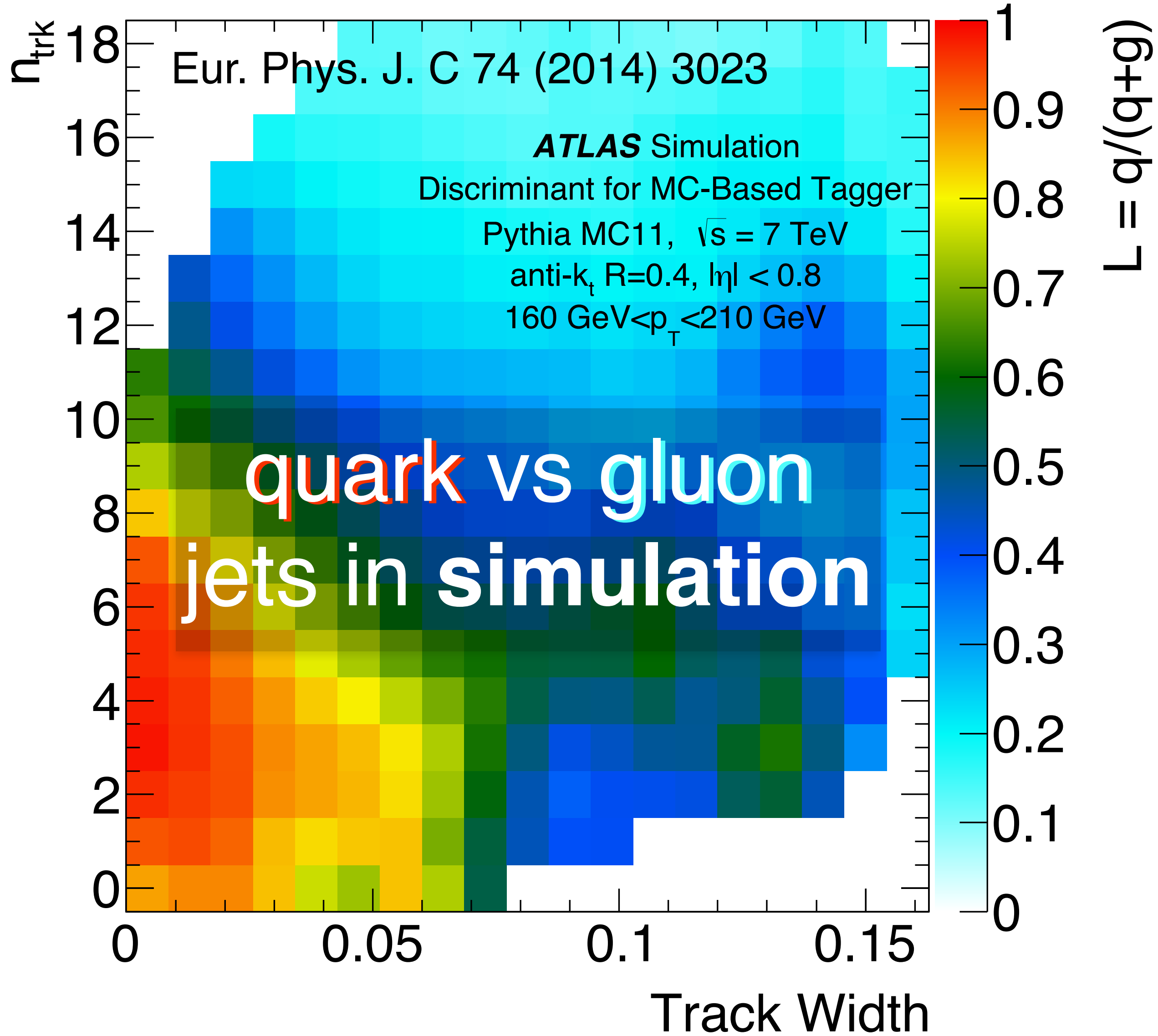
High Energy Physics at the LHC

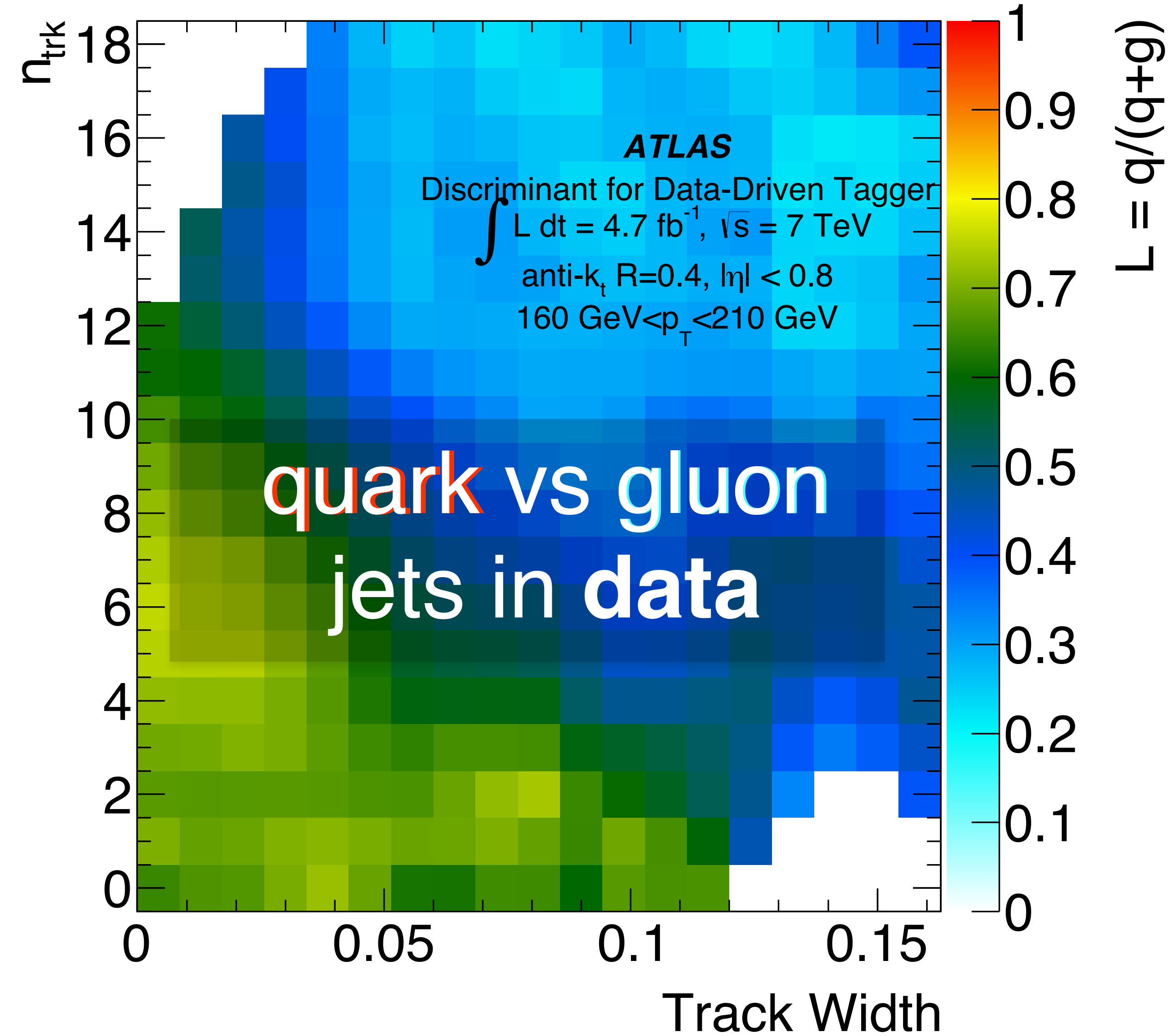
Today, I'm going to use jets as my prototypical example.



Background and Motivation

Usual paradigm: train in simulation, validate on data, test on data.

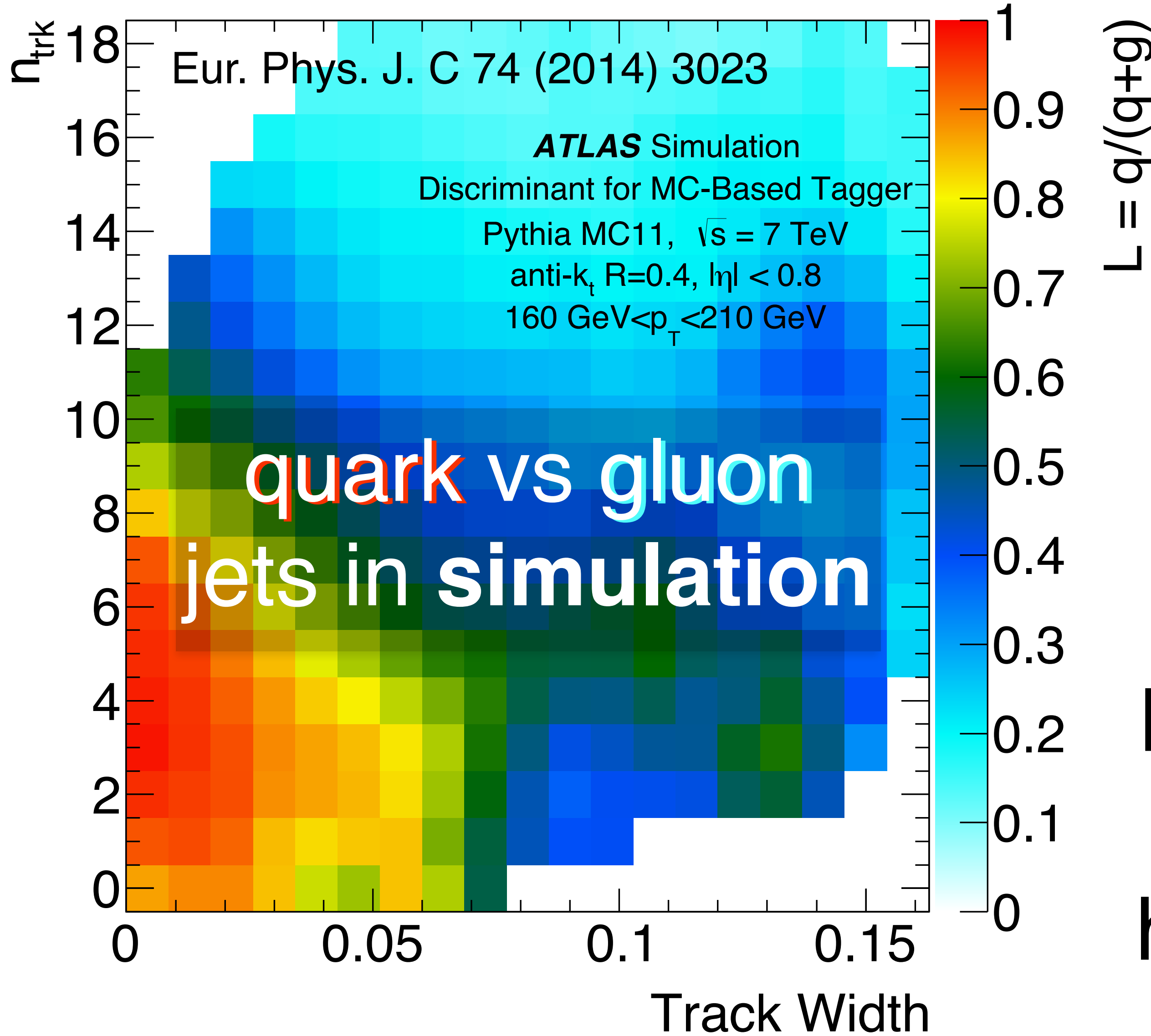




If data and simulation differ, this is sub-optimal!

Background and Motivation

Usual paradigm: **train in simulation**, validate on data, test on data.



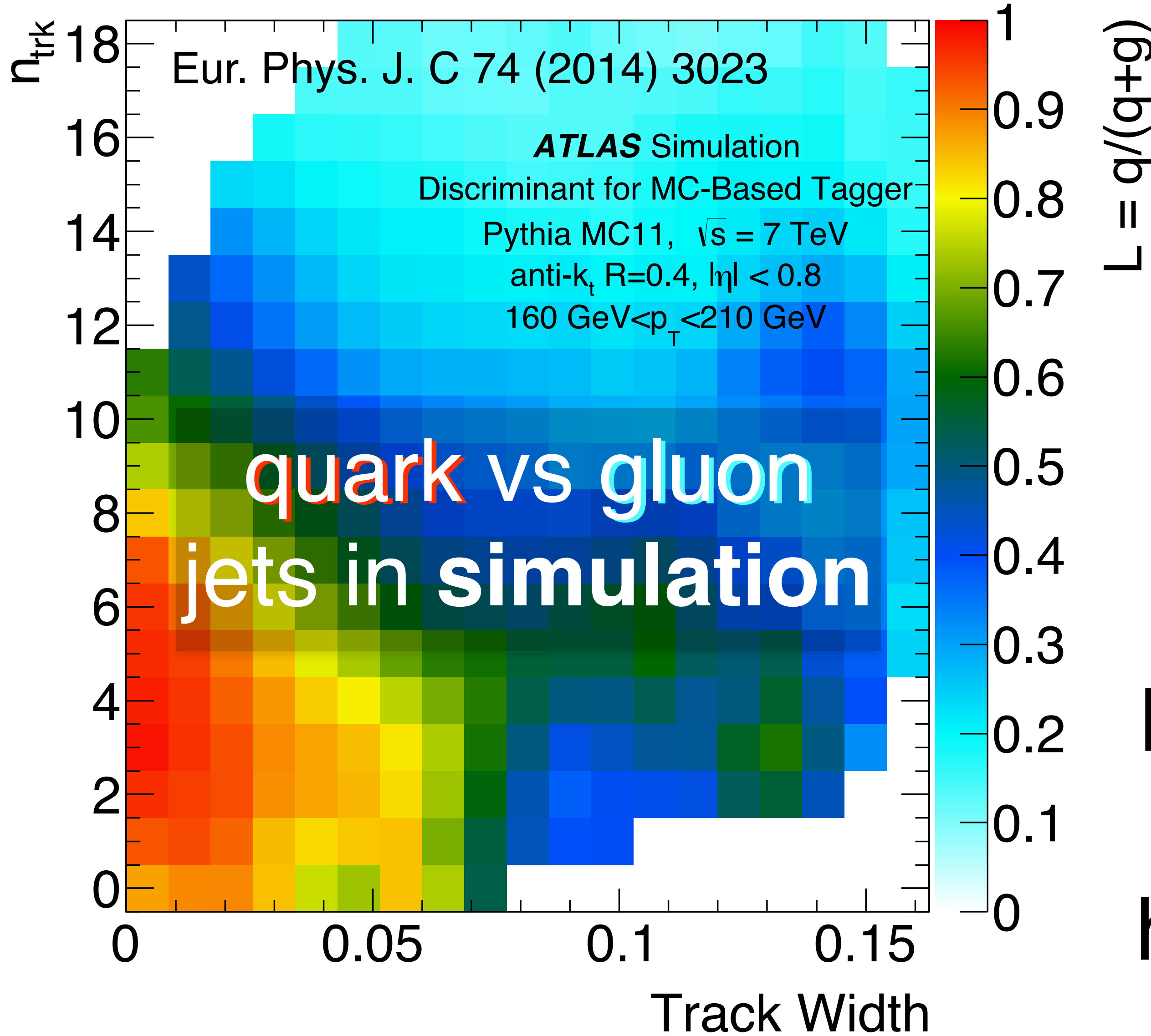
Recall: optimal classifier (by Neyman-Pearson NP) is a threshold cut on the likelihood ratio.

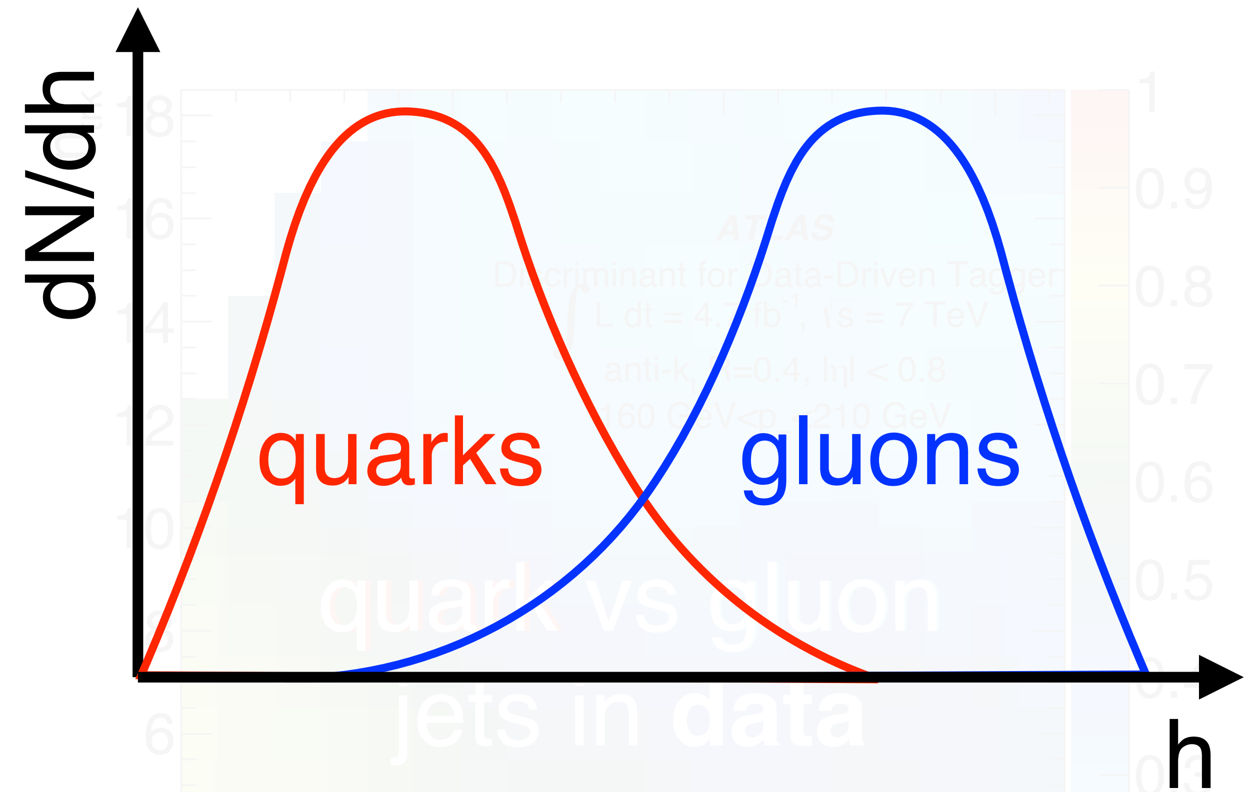
For a 2D feature space, no need for a NN or BDT - can use a histogram to “train” the classifier.

$h(n_{trk}, \text{Track Width}) \rightarrow [0,1]$

Background and Motivation

Usual paradigm: **train in simulation**, validate on data, test on data.



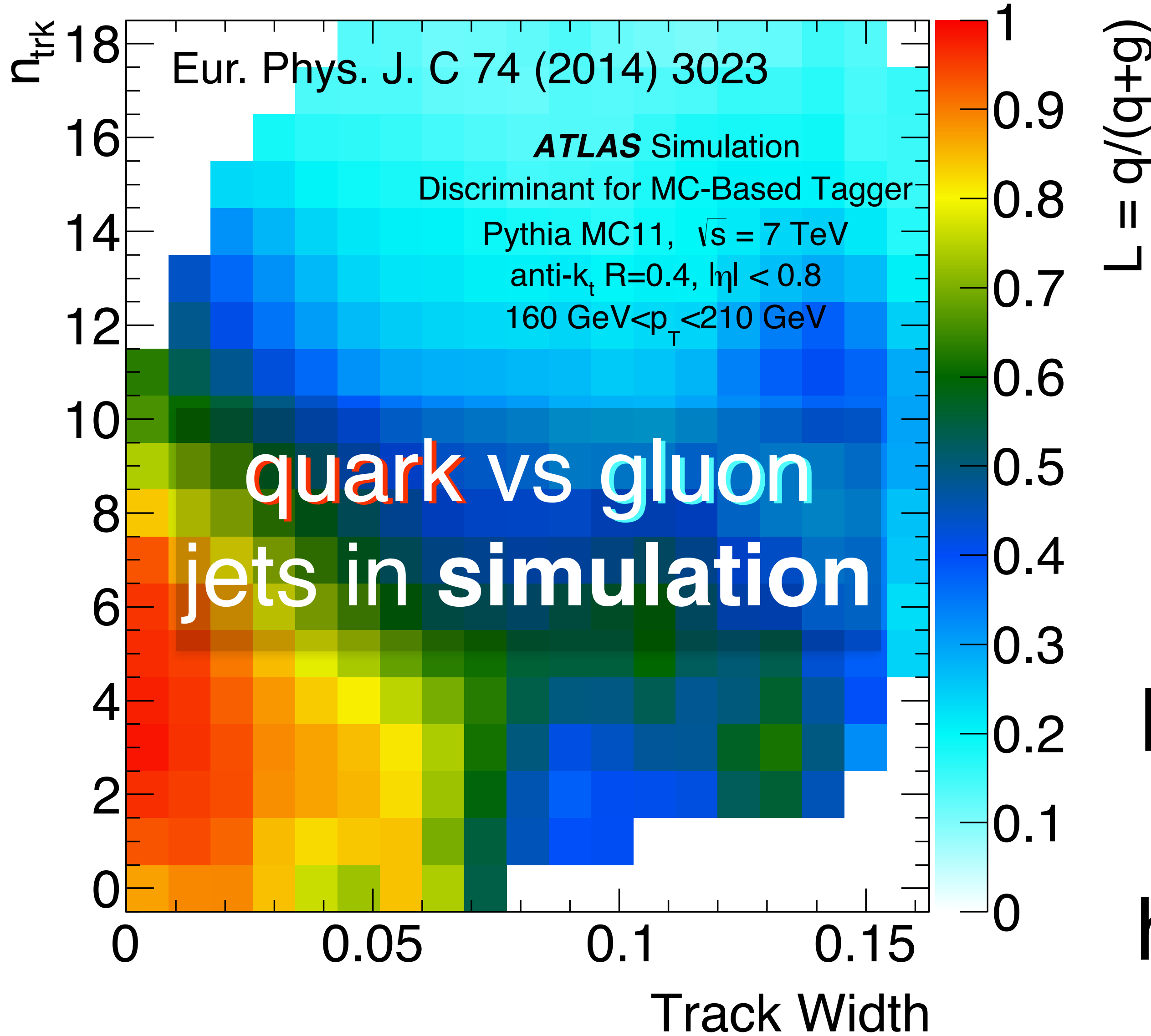


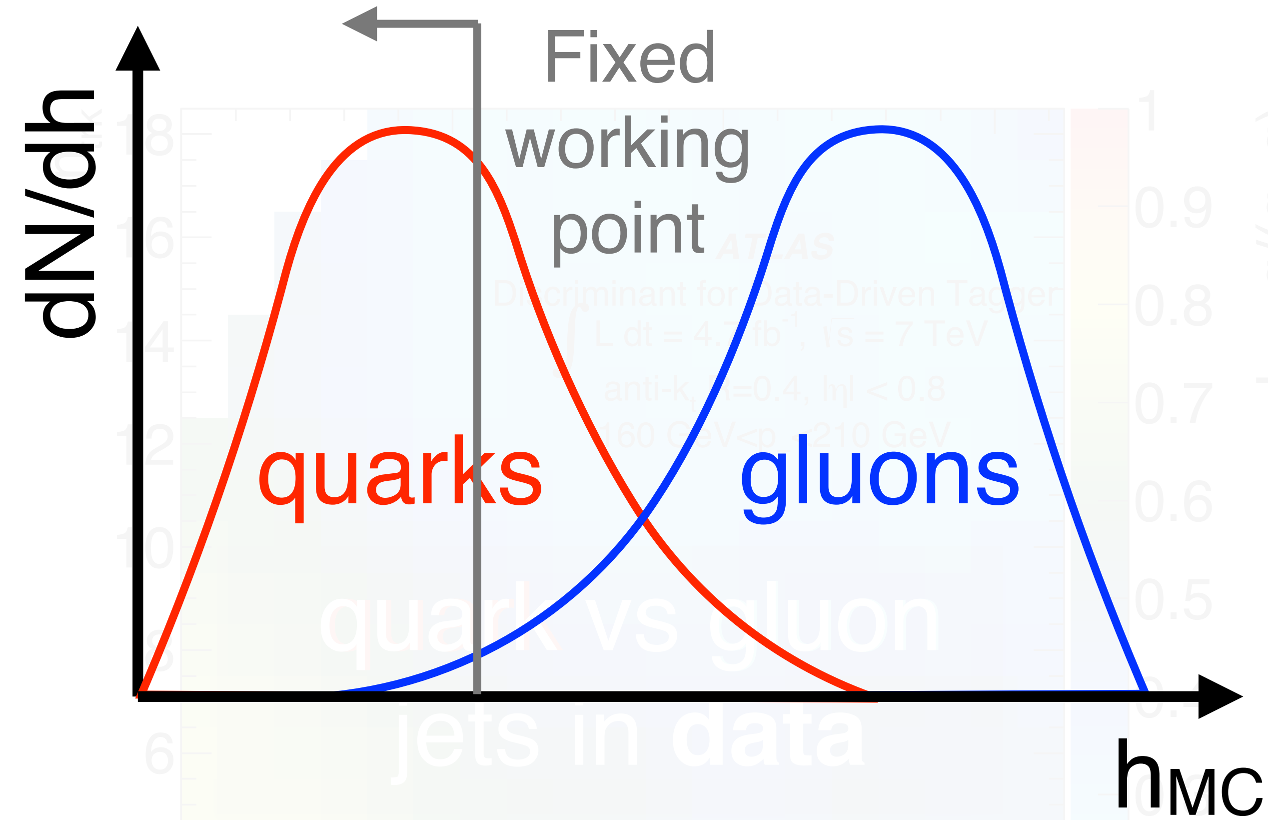
For a 2D feature space, no need for a NN or BDT - can use a histogram to “train” the classifier.

$h(n_{trk}, \text{Track Width}) \rightarrow [0,1]$

Background and Motivation

Usual paradigm: **train in simulation**, validate on data, test on data.



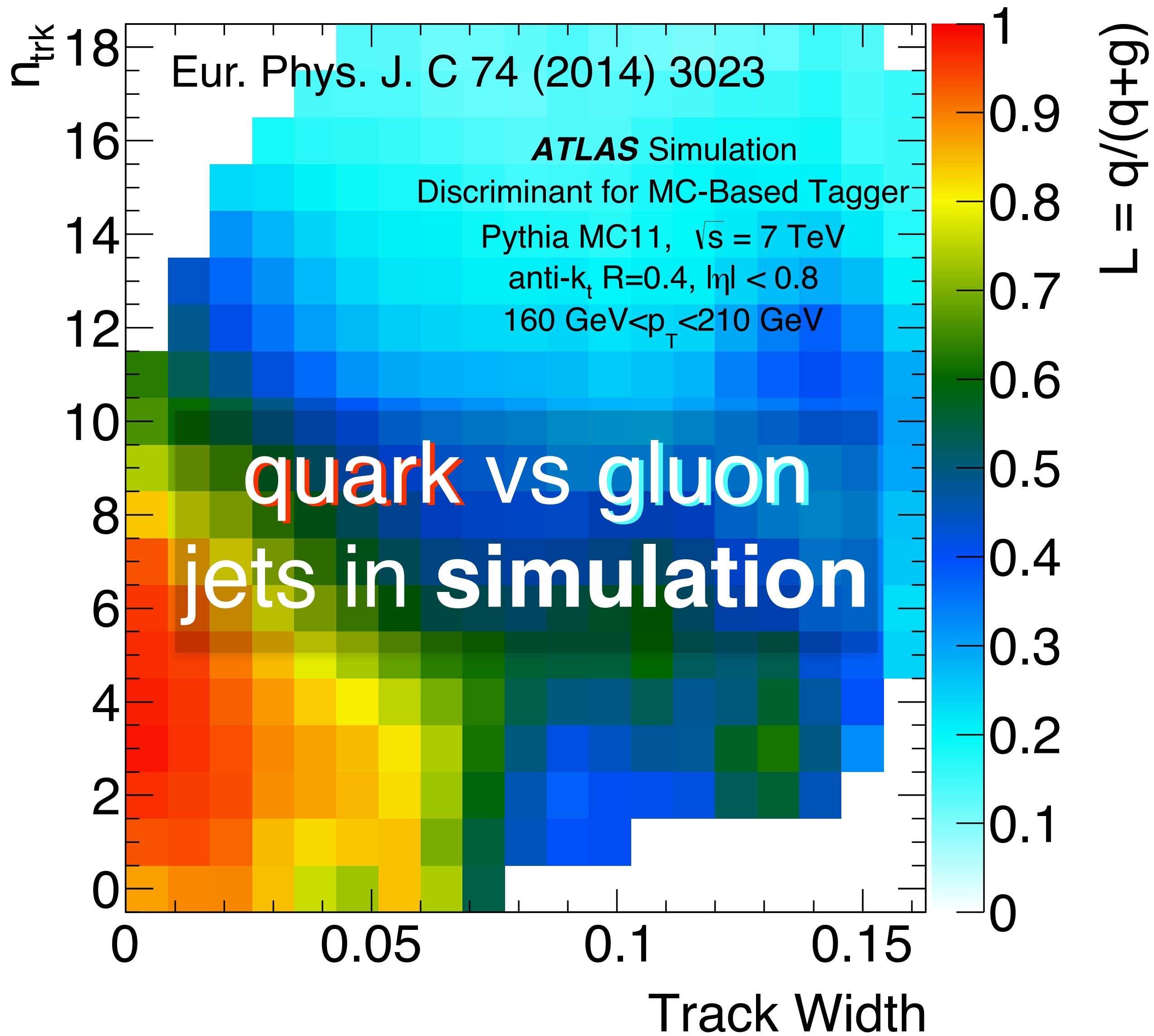


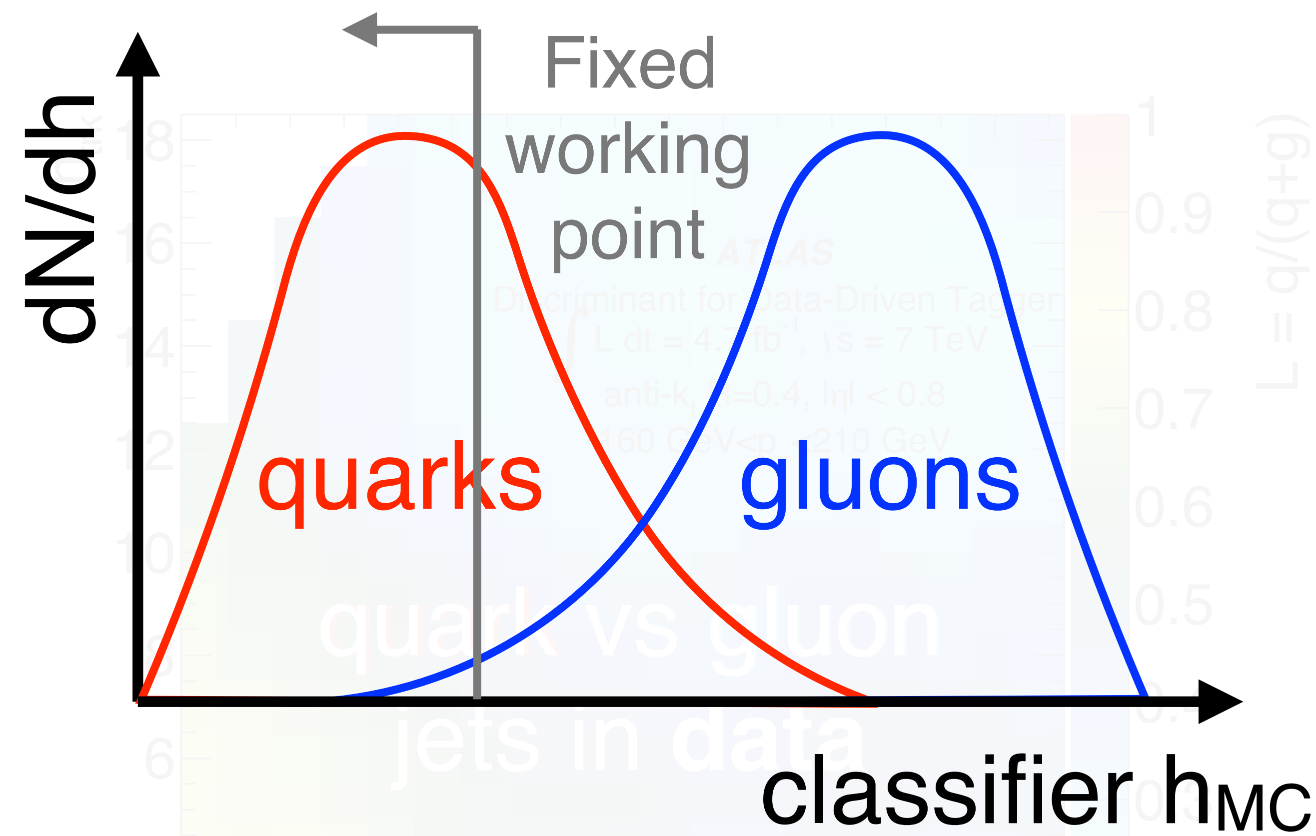
For a 2D feature space, no need for a NN or BDT - can use a histogram to “train” the classifier.

$h_{MC}(n_{trk}, \text{Track Width}) \rightarrow [0,1]$

Background and Motivation

Usual paradigm: **train in simulation**, validate on data, test on data.





WP in simulation:

$\varepsilon_{\text{signal,MC}}$, $\varepsilon_{\text{back,MC}}$

Background and Motivation

Usual paradigm: train in simulation, **validate on data**, test on data.

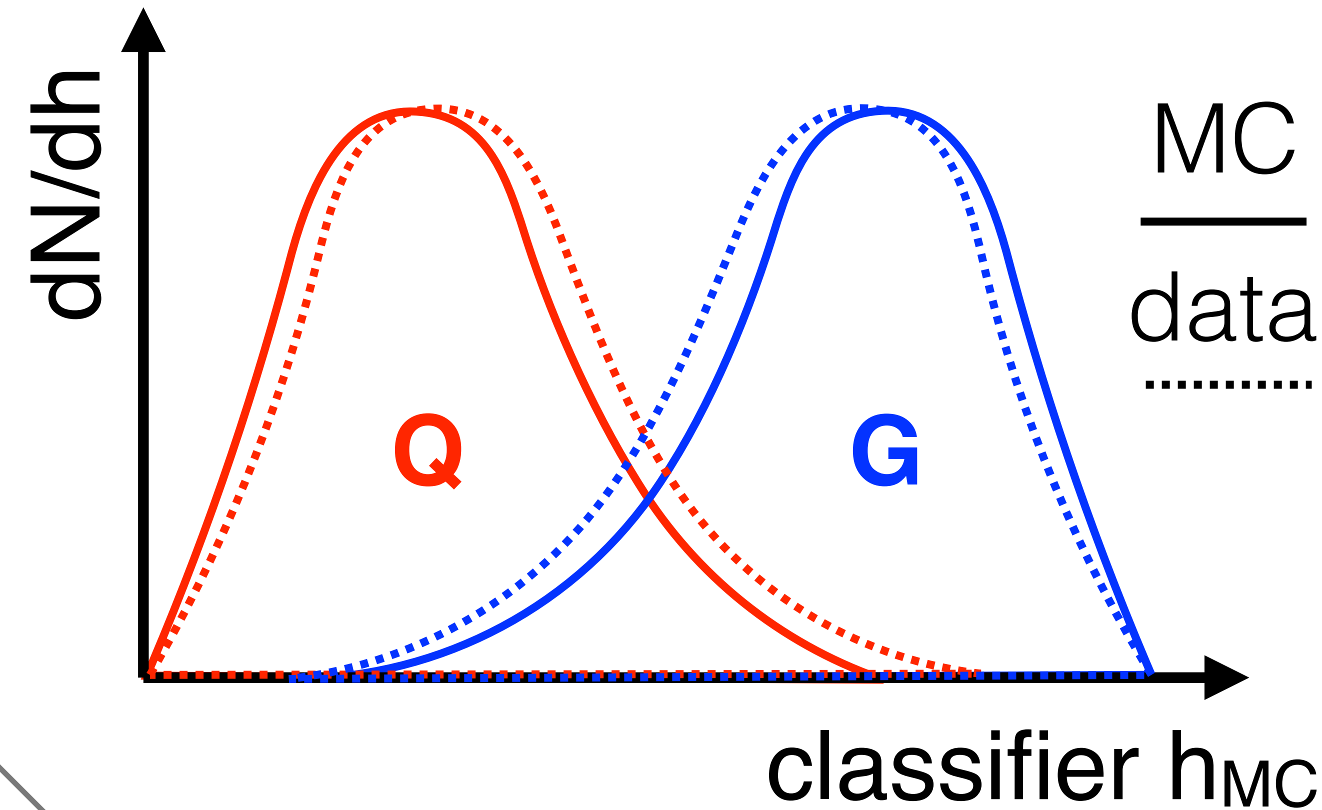
Determine the performance of the WP in data.

How did we get this?

dijets = f_q x **Q** + (1-f_q) x **G**

Z+jets = g_q x **Q** + (1-g_q) x **G**

2 equations, 2 unknowns (**Q**, **G**)



two event samples with different q/g fractions

(N.B. f & g from simulation and selection can't bias Q and G - more on that later)

Background and Motivation

Usual paradigm: train in simulation, **validate on data**, test on data.

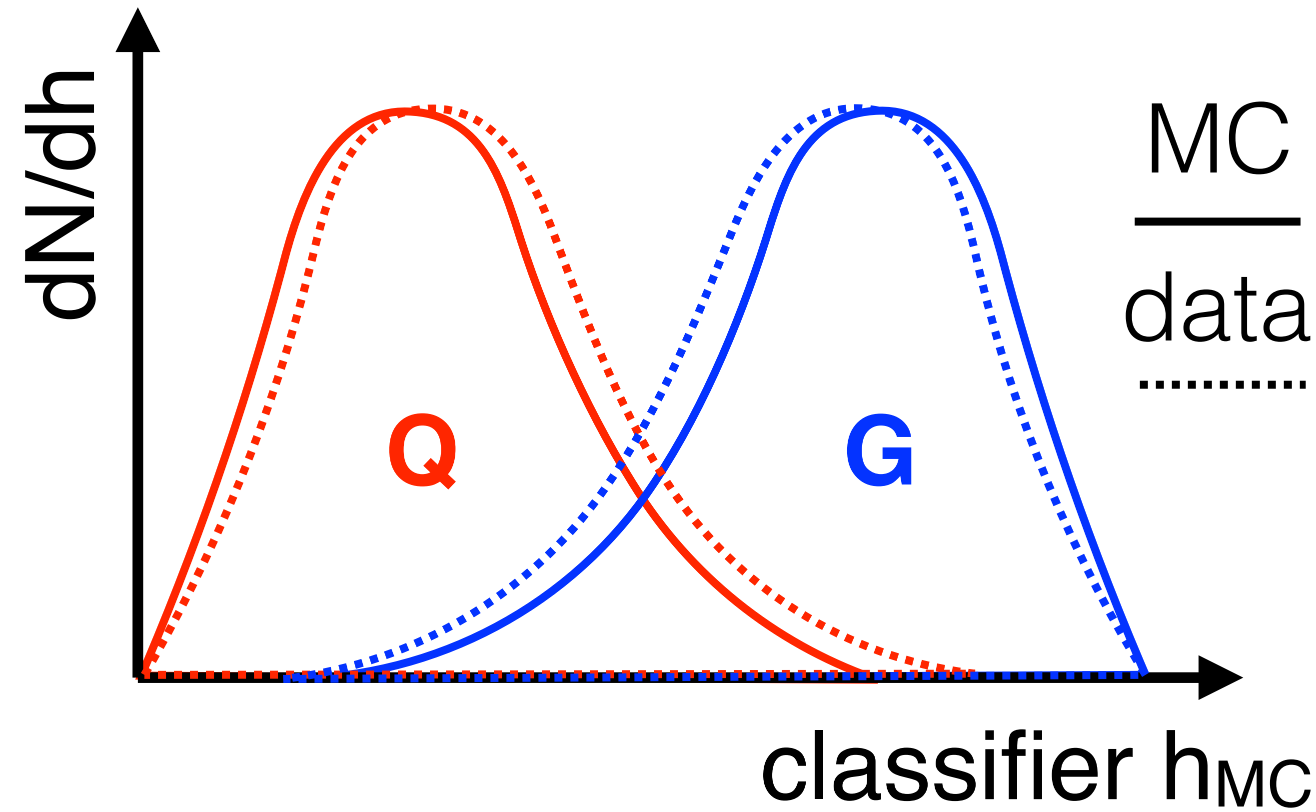
Determine the performance of the WP in data.

How did we get this?

dijets = f_q x **Q** + (1-f_q) x **G**

Z+jets = g_q x **Q** + (1-g_q) x **G**

2 equations, 2 unknowns (**Q**, **G**)



WP in data:

$\varepsilon_{signal,data}$, $\varepsilon_{back,data}$

Can correct the MC to have the same performance as data.

Background and Motivation

Usual paradigm: train in simulation, validate on data, **test on data**.

Once we have scale factors (& their uncertainty), we can ensure that our analysis will be accurate.

…so what is the problem?

remember my claim from earlier:

If data and simulation differ, this is sub-optimal!

This is an accuracy versus precision problem. It is "easy" to achieve accuracy through calibration, but the results may not be the best one possible.

Background and Motivation

In this 2D feature space, we can actually derive h_{data}.

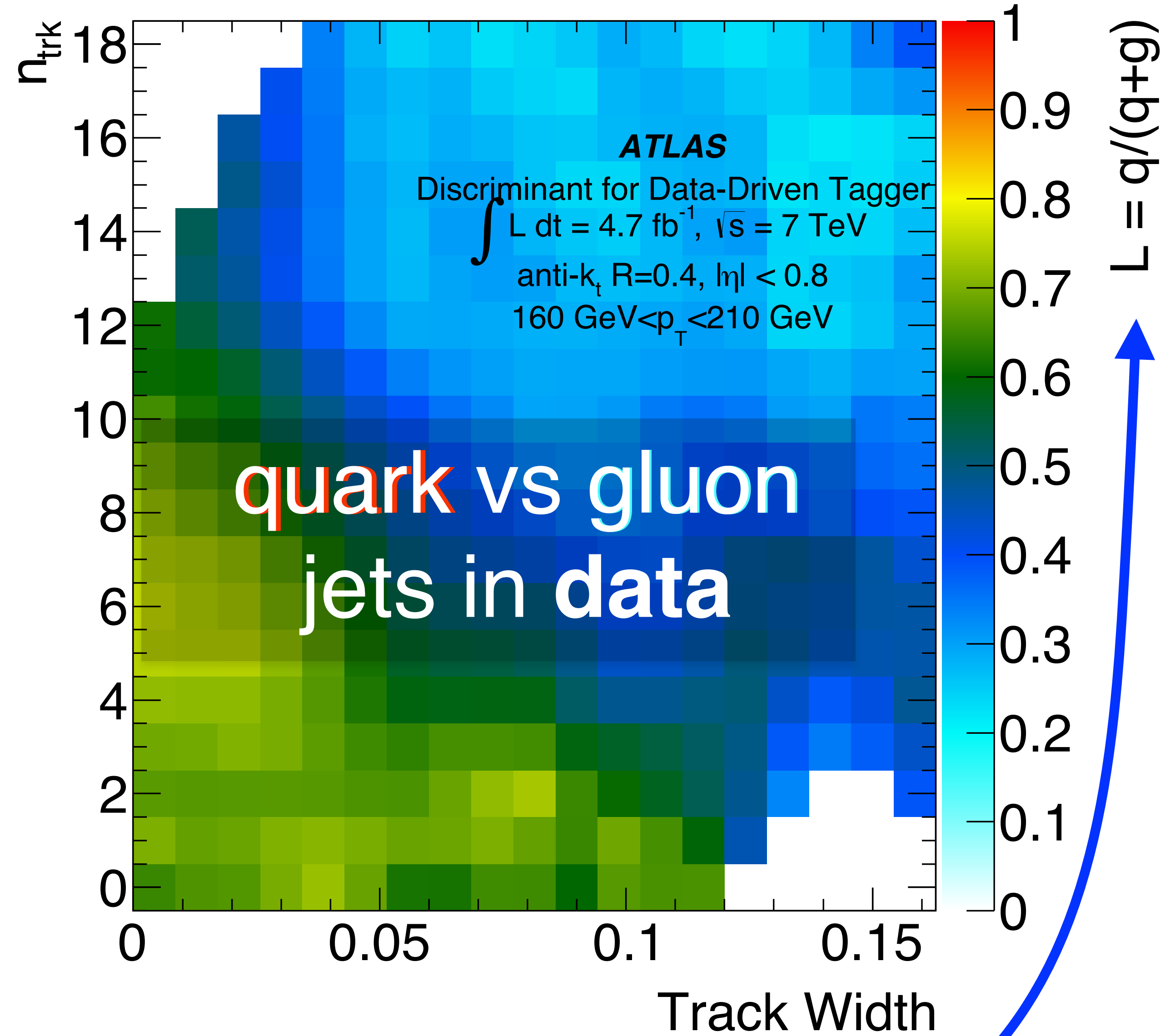
Using the same trick as earlier:

$$\text{dijets} = f_q \times Q + (1-f_q) \times G$$

$$\text{Z+jets} = g_q \times Q + (1-g_q) \times G$$

2 equations, 2 unknowns (Q, G)

(now Q and G are 2D histograms)



in general:

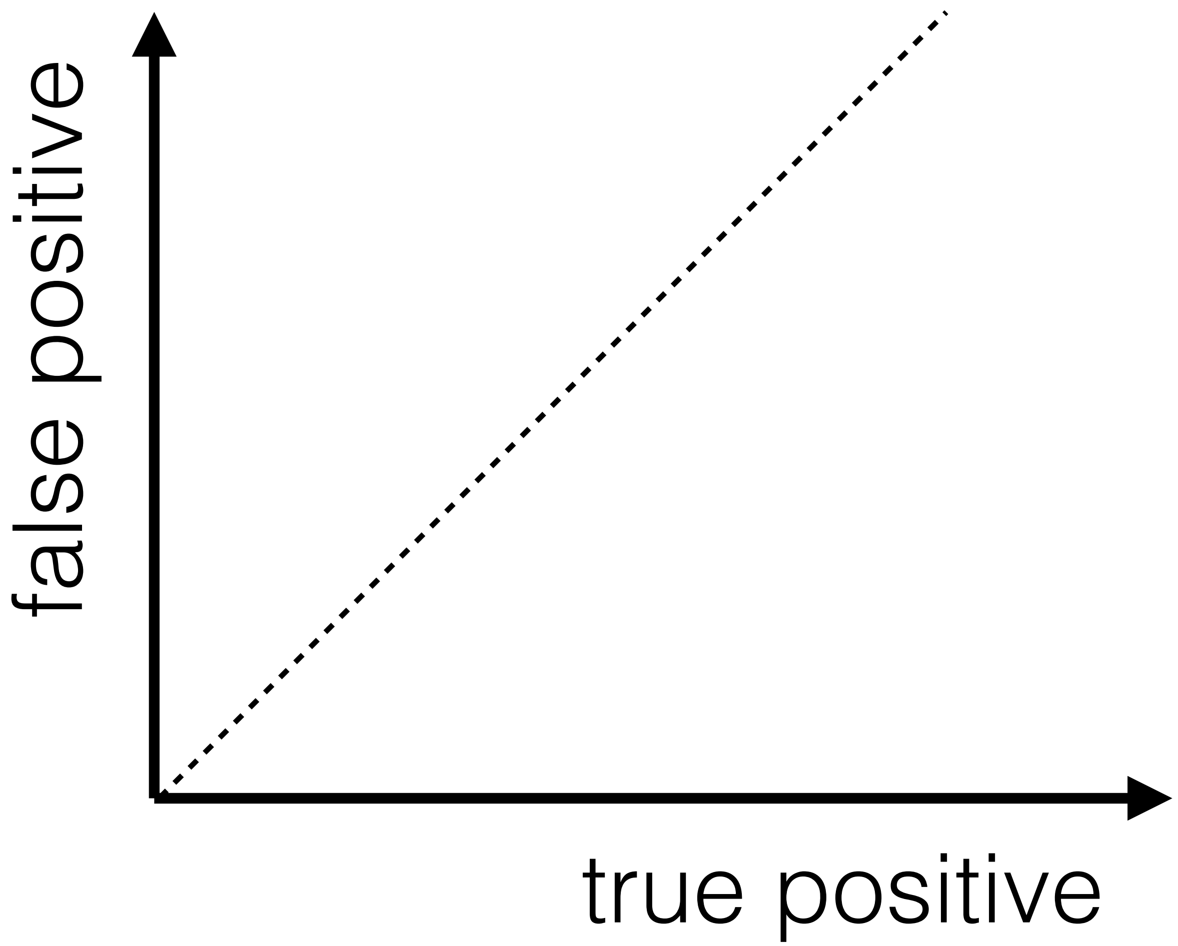
$h_{MC}(n_{trk}, \text{Track Width}) \neq h_{data}(n_{trk}, \text{Track Width})$

Take it to the extreme

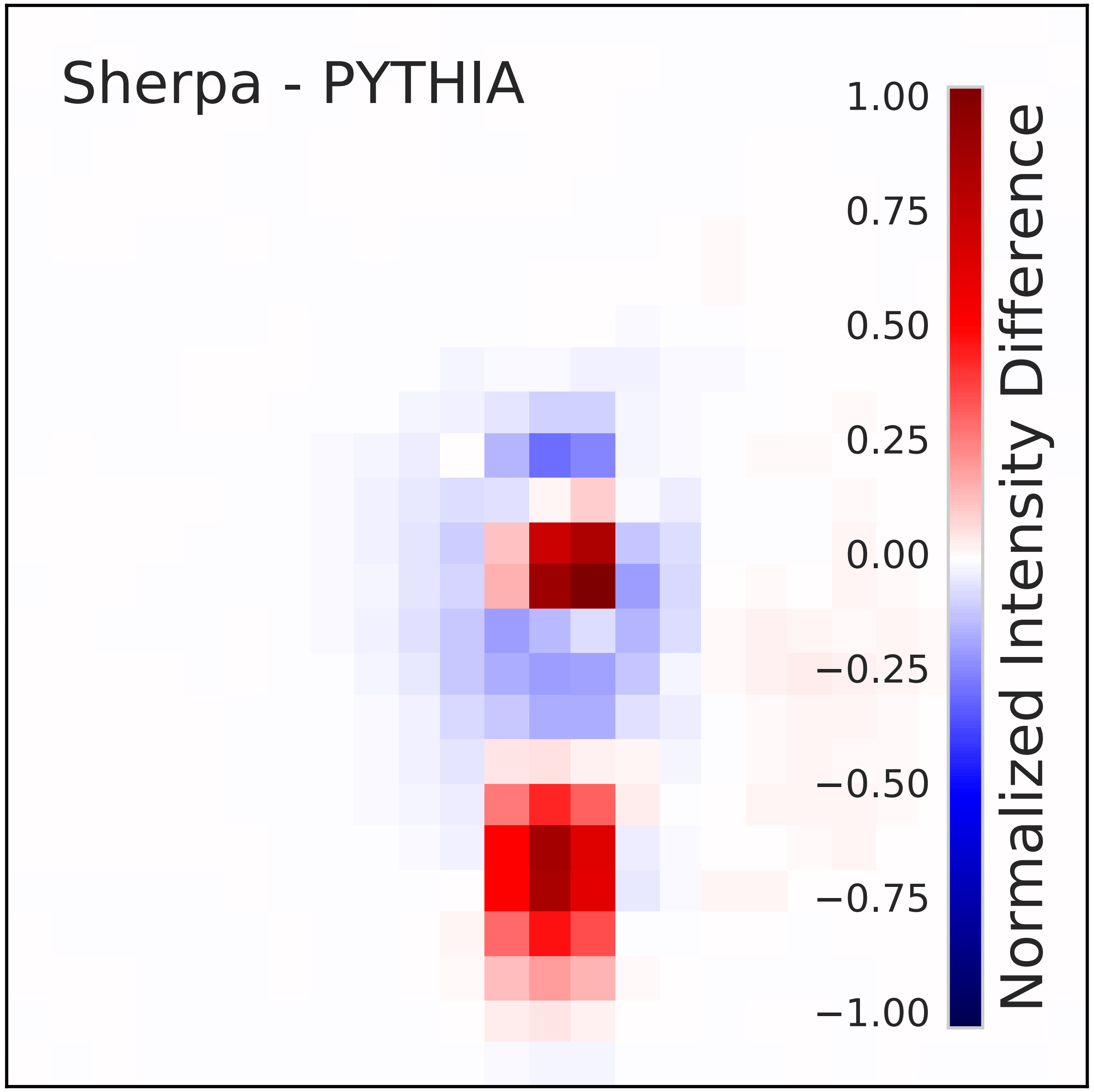
To stress this point, suppose that h_{MC} is the random classifier:

h_{MC} = 0 if you pick a random number x in [0,1] and $x < \varepsilon$
1 otherwise

We can calibrate this classifier in data, but clearly, it is sub-optimal !!



One more slide about why it matters



J. Barnard, E. Dawe, M. Dolan, N. Rajcic, Phys. Rev. D 95 (2017) 014018

Especially important for **deep learning** using subtle features → hard to model!

W boson radiation pattern - same physics, different simulators!

Achieving the Optimal Classifier

Two ways around the problems mentioned earlier:

(1) Derive the classifier in MC, but don't let it use information that is not present in data.

"Learning to pivot"

G. Louppe, M. Kagan, K. Cranmer, 1611.01406

(2) Train on unlabeled data.

"Weak supervision"

*L. Dery, **BPN**, F. Rubbo, A. Schwartzman, JHEP 05 (2017) 145*
*E. Metodiev, **BPN**, J. Thaler, JHEP 10 (2017) 174*

Achieving the Optimal Classifier

Two ways around the problems mentioned earlier:

(1) Derive the classifier in MC, but don't let it use inf… ation that is not present in data.

Ask Gilles if you have questions about pivoting!

"Learning to pivot"

G. Louppe, M. Kagan, K. Cranmer, 1611.01406

(2) Train on unlabeled data.

"Weak supervision"

*L. Dery, **BPN**, F. Rubbo, A. Schwartzman, JH…*
*E. Metodiev, **BPN**, J. Thaler, JHEP 10 (2…*

Disclaimer: I'll spend most of my time discussing this

Pivoting

A clever idea is to build in robustness to the loss function:

hyperparmeter

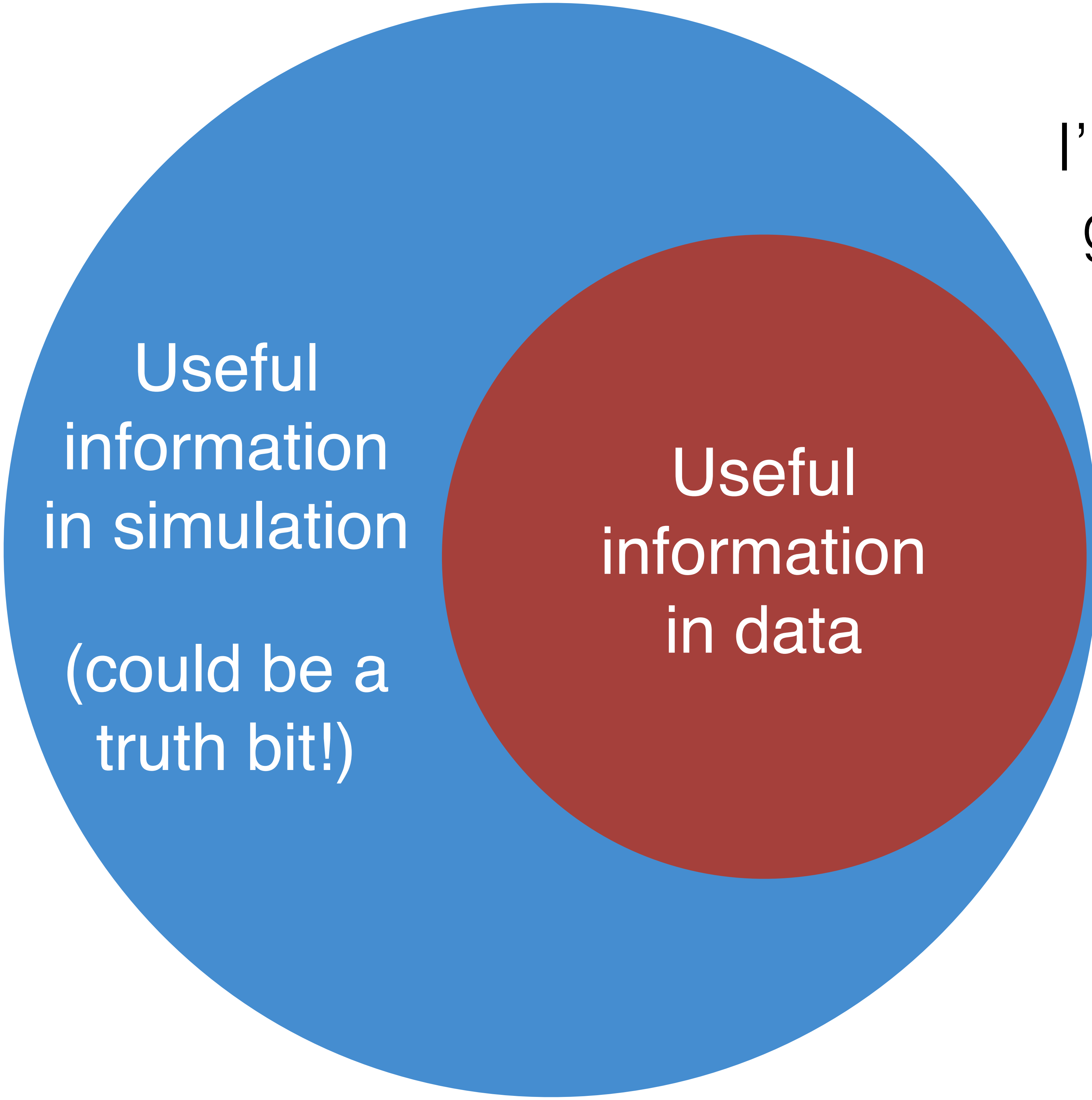
Loss = usual loss - λ x adversarial loss

e.g. binary cross-entropy using per-instance labels from simulation.

can the output of the classifier tell if it is looking at data or MC?

i.e. if h is the classifier, using h(x) as a feature, try to classify data versus MC.

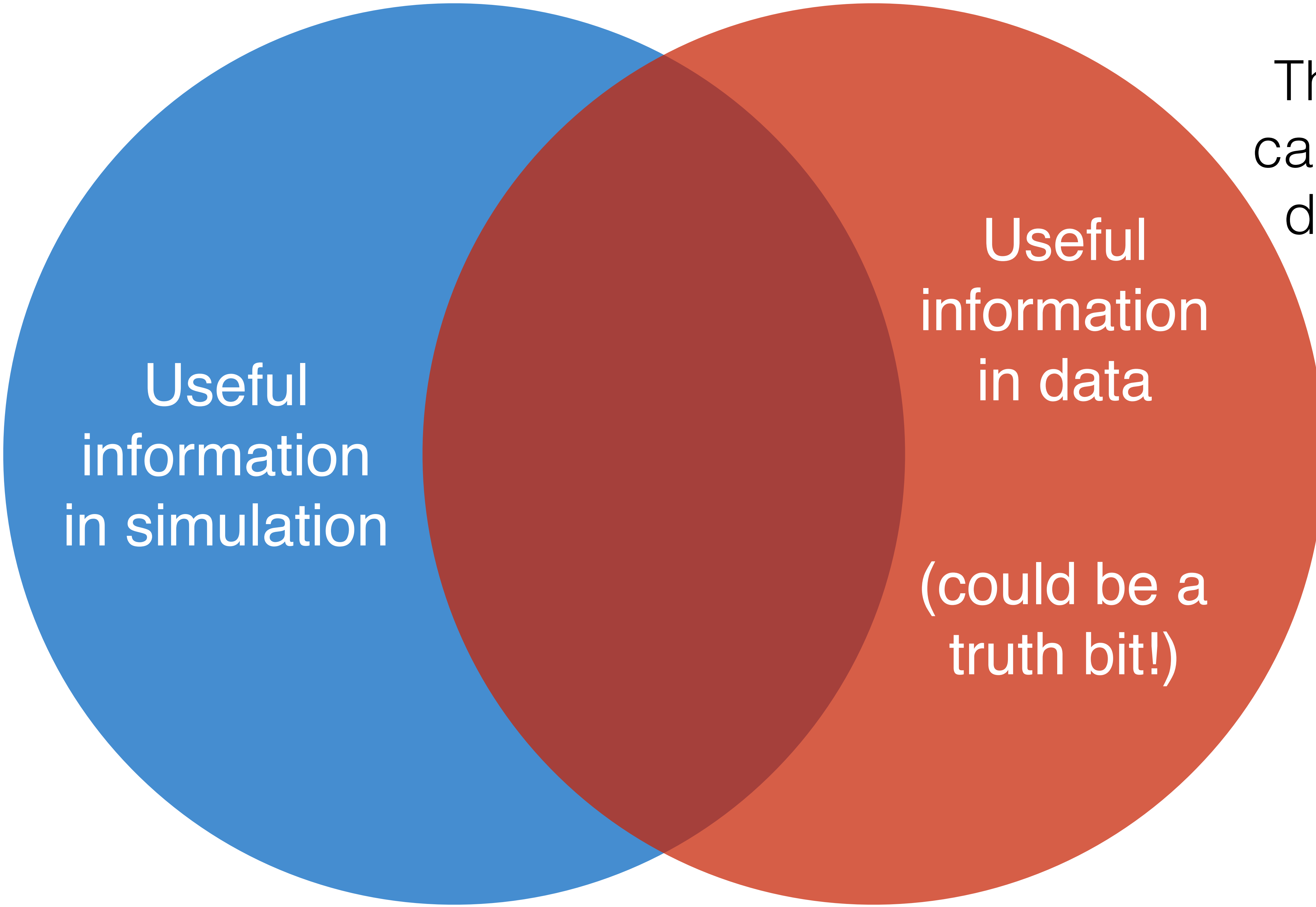
When pivoting is “optimal” in data



I’ll show some pictures to give you some intuition.

In this case, the adversary ensures that the classifier can’t use information from simulation that is actually not useful in data.

When pivoting is “suboptimal” in data



The simulation can’t use what it doesn’t know.

…many other applications of this approach, such as reducing sensitivity to systematic uncertainties, unwanted correlations between features, etc.

Another possibility: Learn from data!

One of the biggest challenges with any MC-based method is that it can't use information that the MC doesn't know about.

One solution is to train directly on data !

In general, this is not possible since data are unlabeled. However, in a wide range of cases, it is possible to work with less.

There is an interesting connection between what I'm calling "weak supervision" and the topic of "label noise".

Weak supervision, caveats up front

The setup: suppose you have (at least) two mixed samples, each composed of two classes (say q and g).

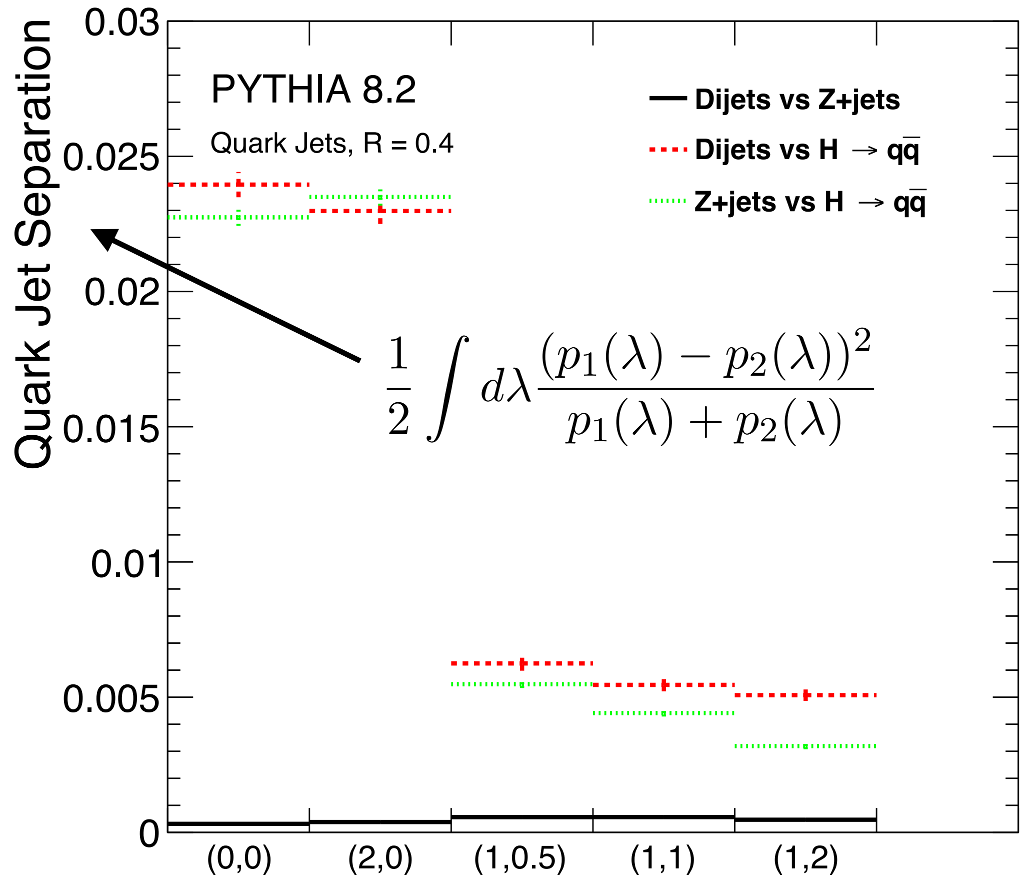
Requirement:

The two classes are well-defined i.e. q in sample 1 is statistically identical to q in sample 2).

Weak supervision, caveats up front

The two classes are well-defined (i.e. q in sample 1 is statistically identical to q in sample 2).

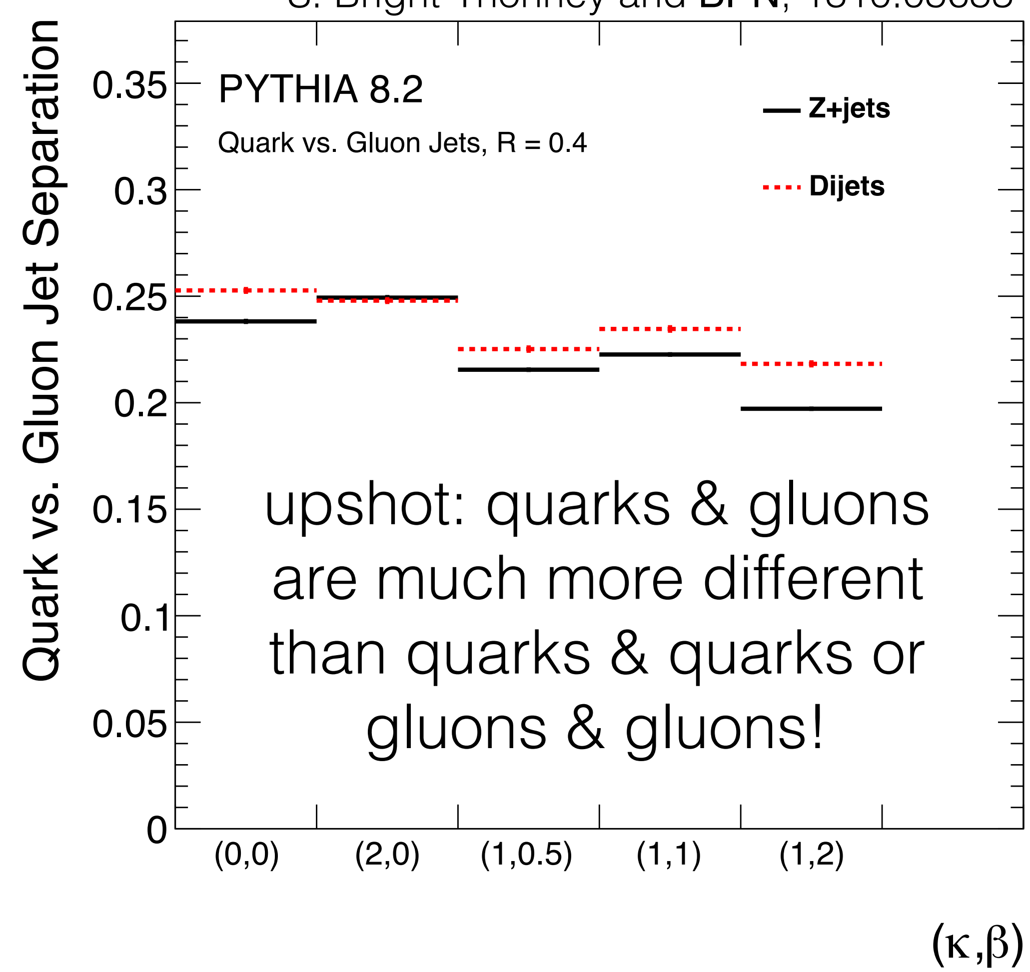
This is often not exactly true, but is often nearly true.



$$\frac{1}{2}\int d\lambda \frac{(p_1(\lambda) - p_2(\lambda))^2}{p_1(\lambda) + p_2(\lambda)}$$

parameterizes jet features →

S. Bright-Thonney and **BPN**, 1810.05653



upshot: quarks & gluons are much more different than quarks & quarks or gluons & gluons!

Weak sup. option 1: Use class proportions

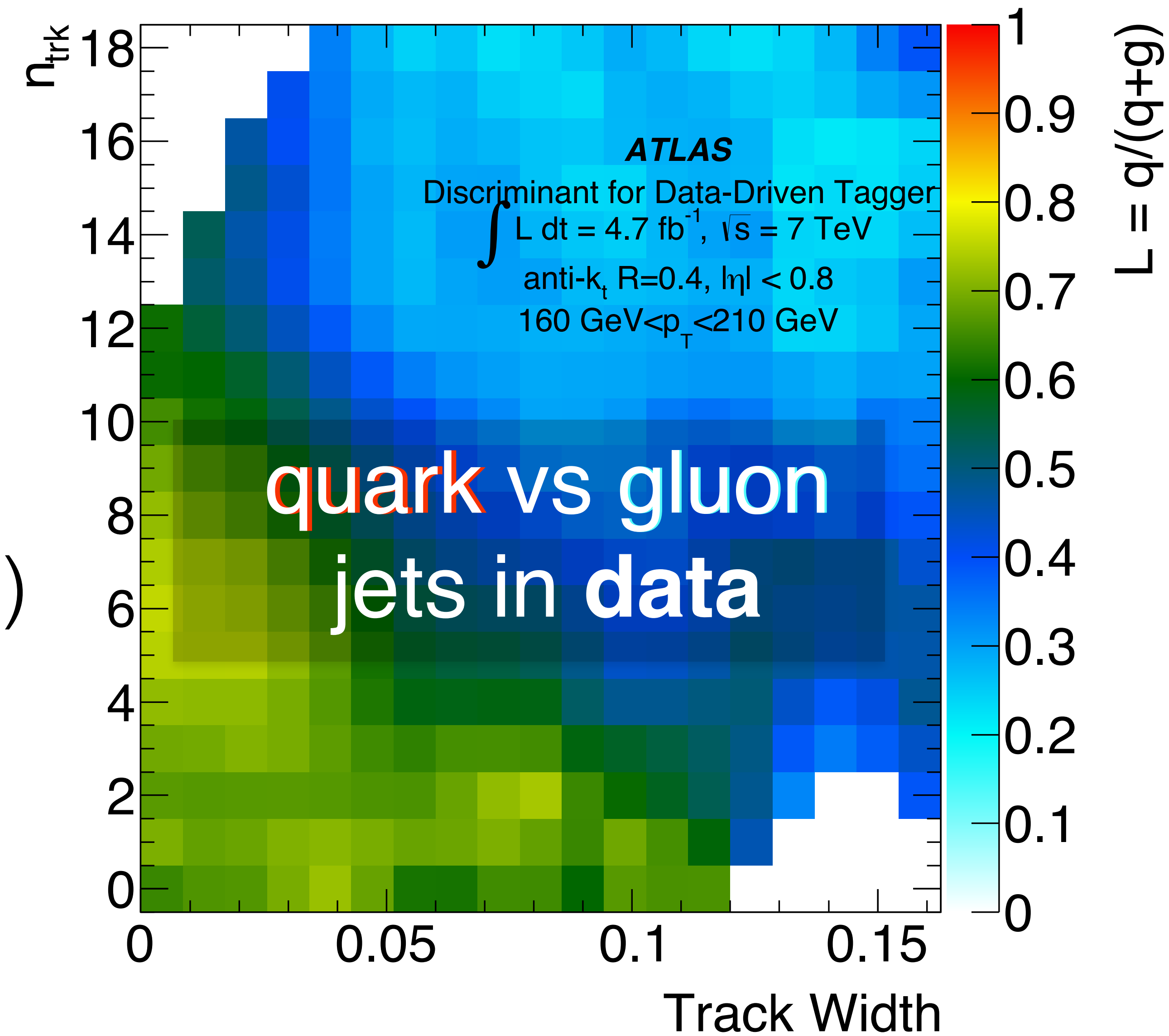
Remember this plot?

dijets = f_q x Q + (1-f_q) x G

Z+jets = g_q x Q + (1-g_q) x G

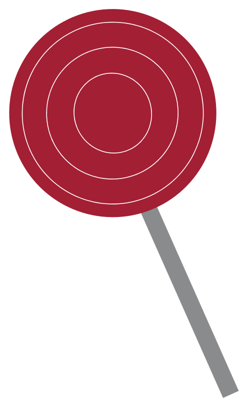
two equations, two unknowns (Q, G)

We often know f, g (from ME + PDF) much better than full radiation pattern inside jets.



This doesn't work well when you have more than 2 observables because the templates become sparse.

Method 1: Learn from Proportions



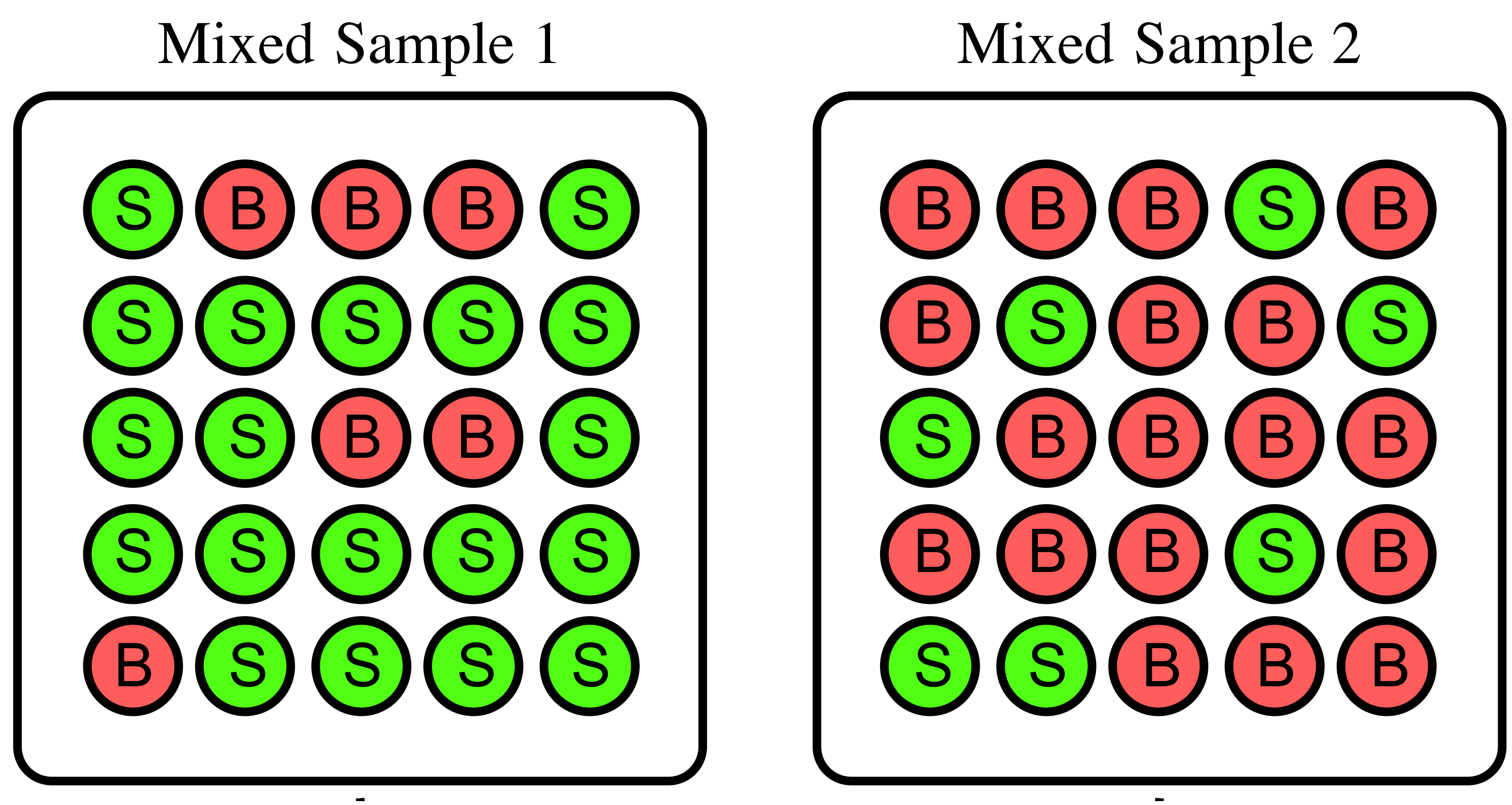
LoLiProp

Learning from Label Proportions

Solution: Train using class proportions. Work “on average”

loss fcn. — labels

$$f_{\text{full}} = \text{argmin}_{f':\mathbb{R}^n \to \{0,1\}} \sum_{i=1}^{N} \ell\left(f'(x_i) - t_i\right)$$



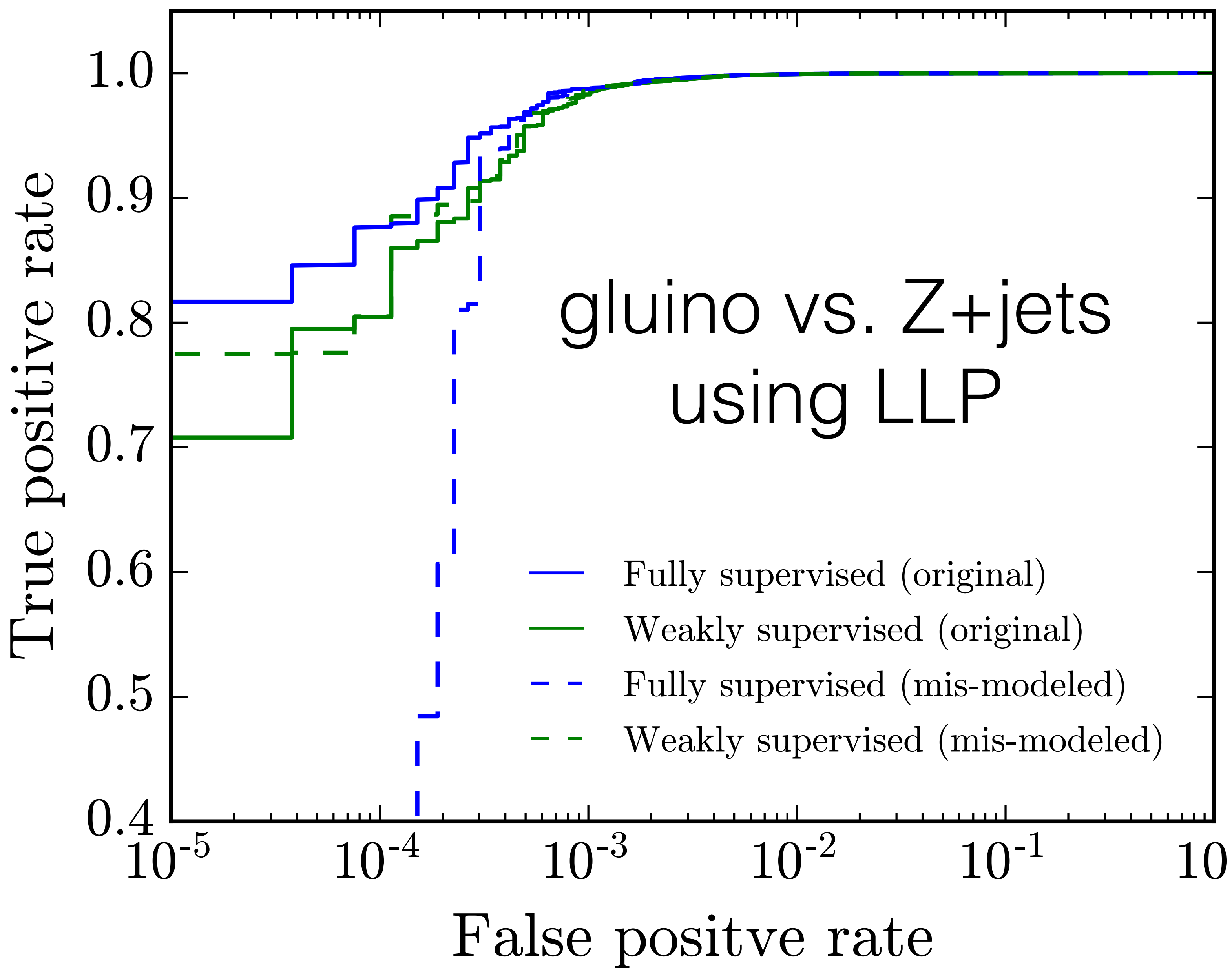
$$f_{\text{weak}} = \text{argmin}_{f':\mathbb{R}^n \to [0,1]} \ell\left(\sum_{i=1}^{N} \frac{f'(x_i)}{N} - y\right)$$

proportions

L. Dery, **BPN**, F. Rubbo, A. Schwartzman, JHEP 05 (2017) 145

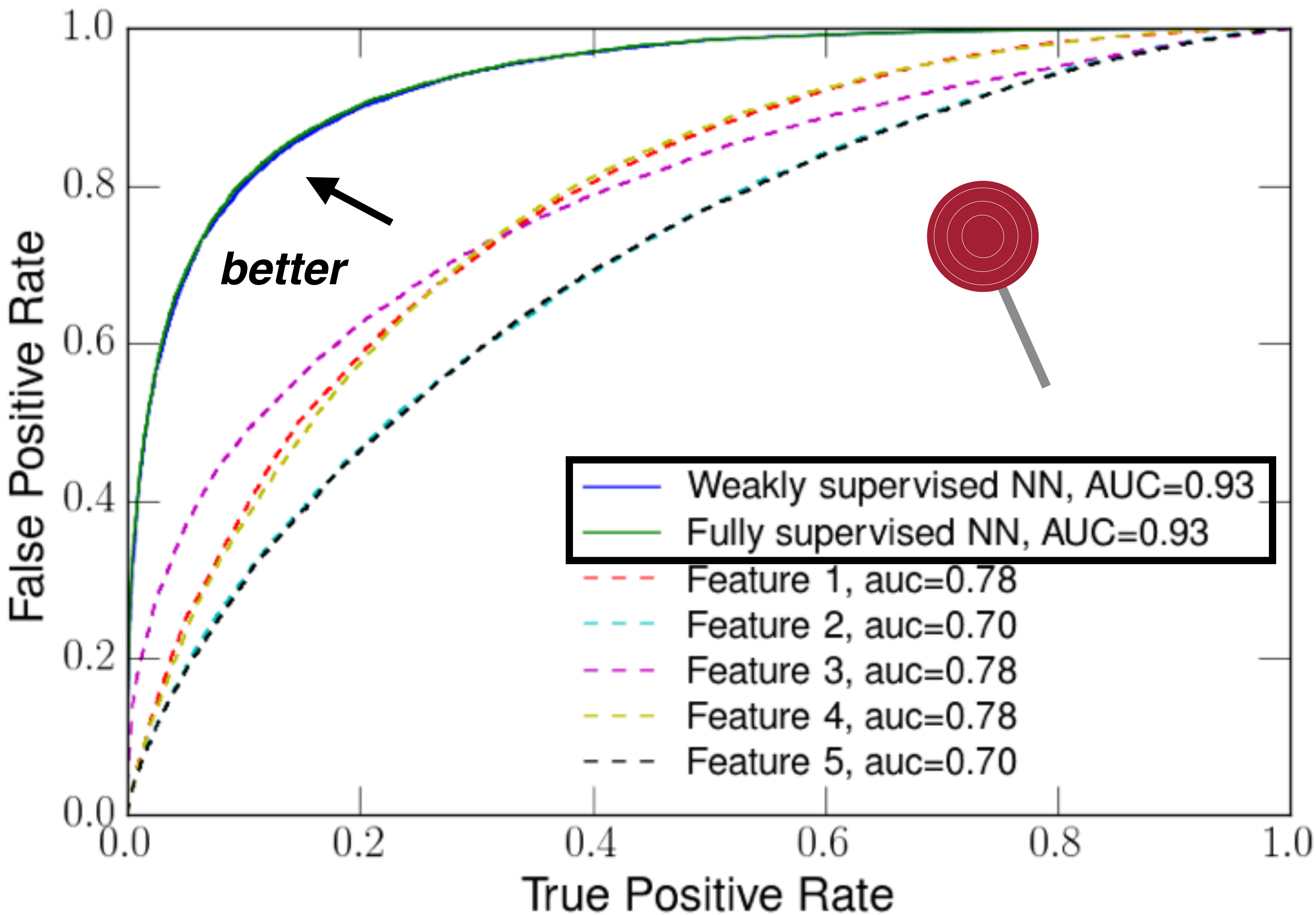
N.B. Don't need 100% fraction accuracy

Even though the proportions are required as input, if they are slightly wrong, you can end up with the correct classifier.



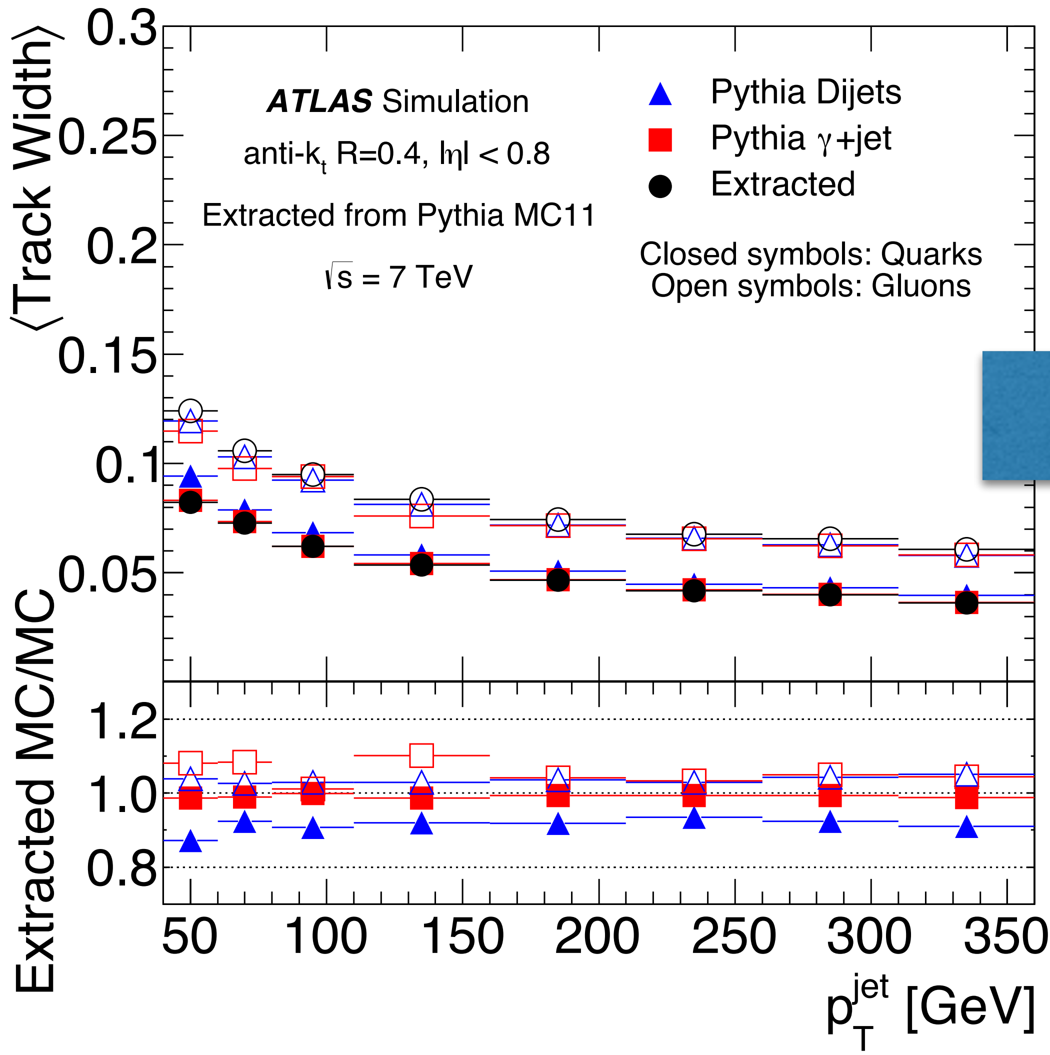
T. Cohen, M. Freytsis, B. Ostdiek, JHEP 02 (2018) 034

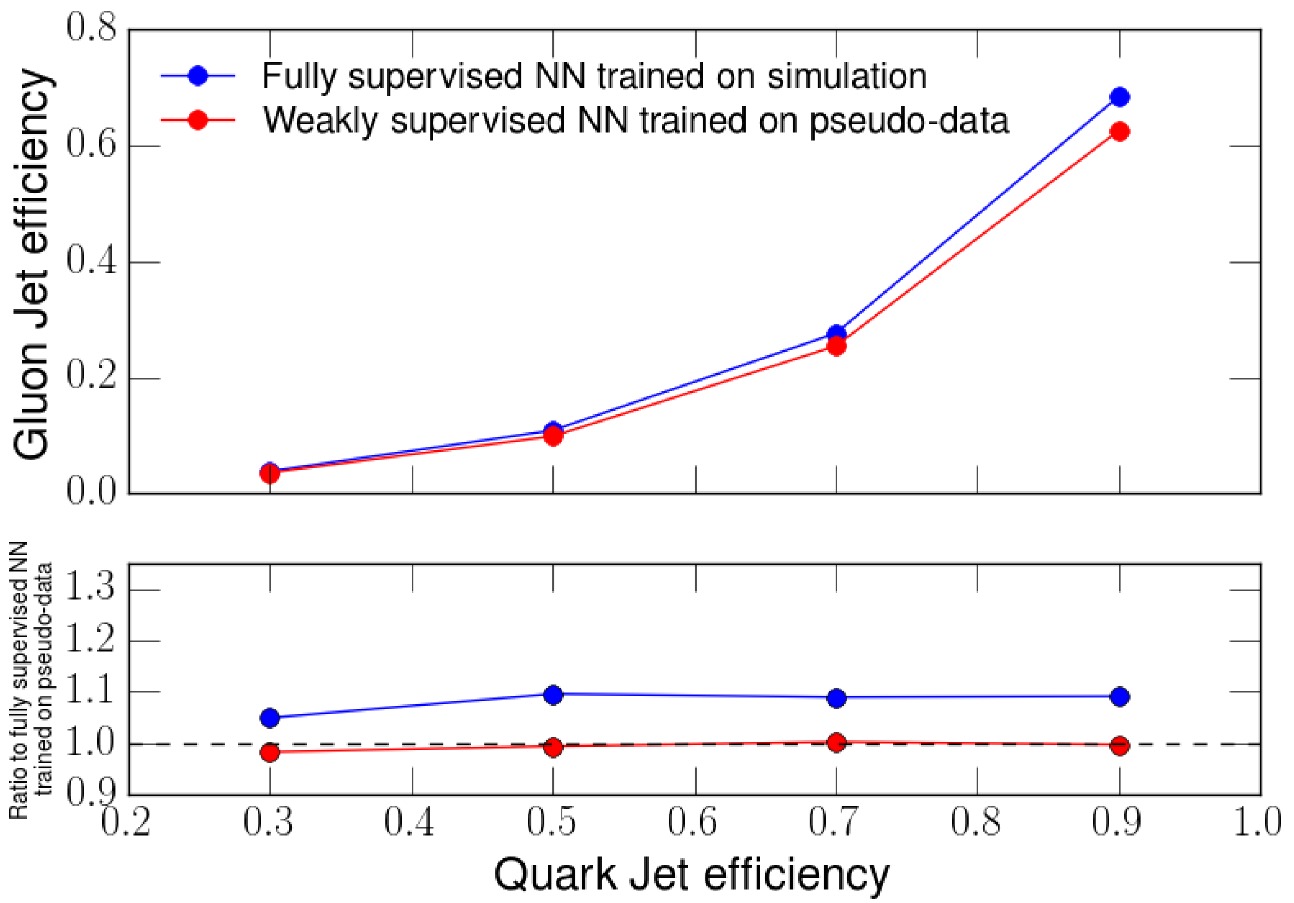
Works in low-dimensions



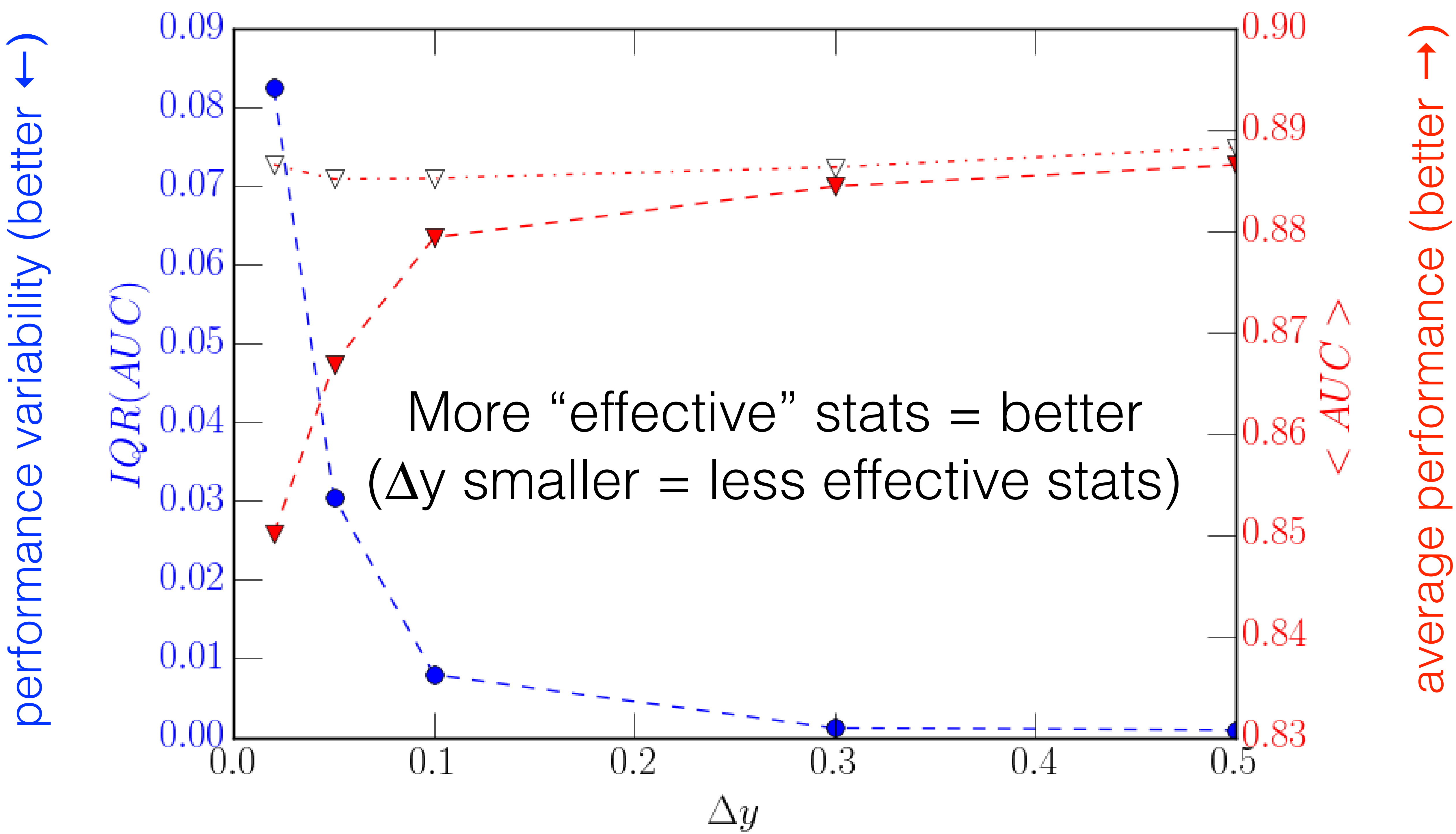
Works in low-dimensions … for q/g

Given the data/MC disagreement from the first slide, this is what you might expect in terms of the performance difference.





A note about training statistics



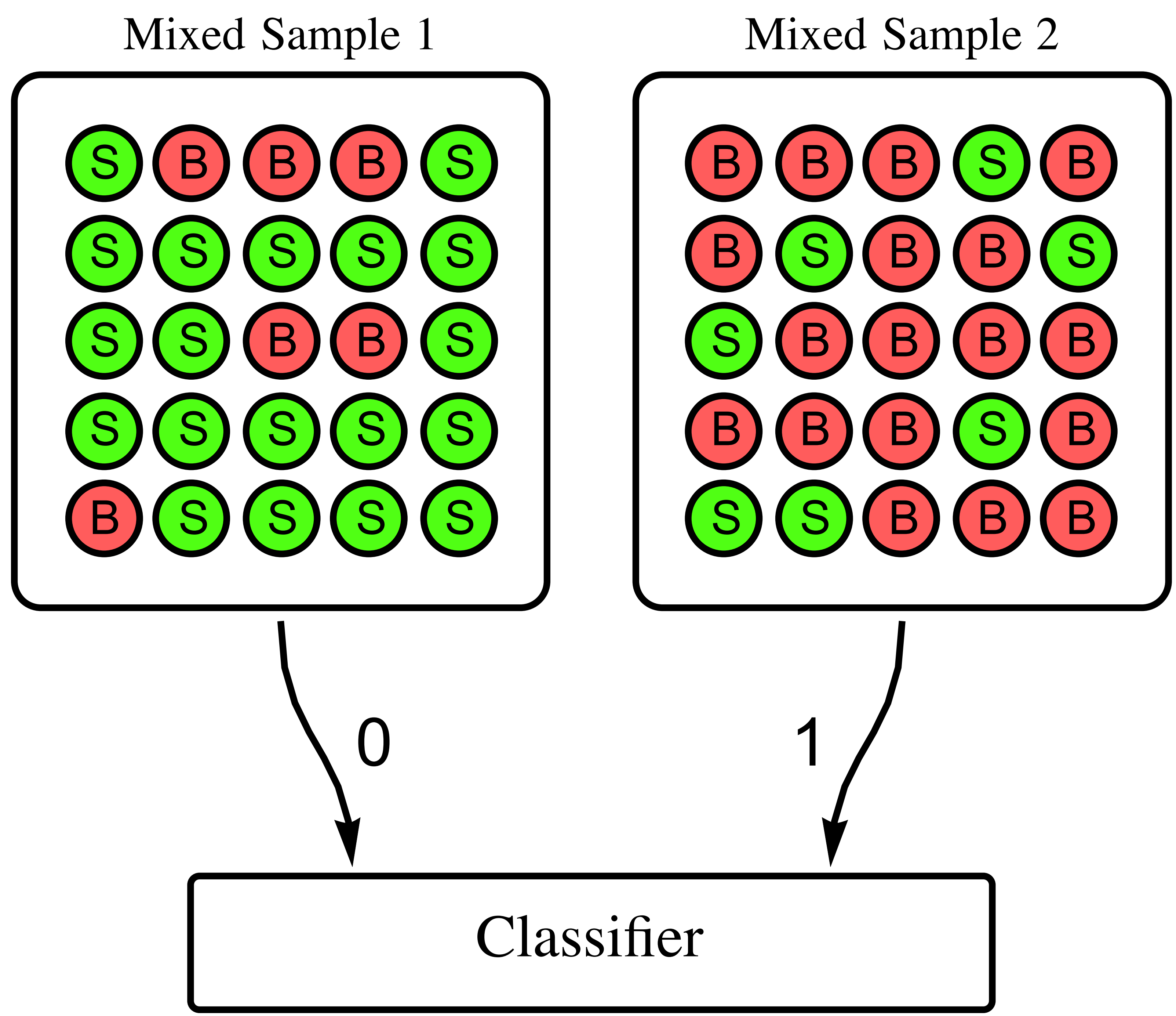
how different are the proportions for the two mixed samples

Method 2: Learning without Proportions

CWoLa

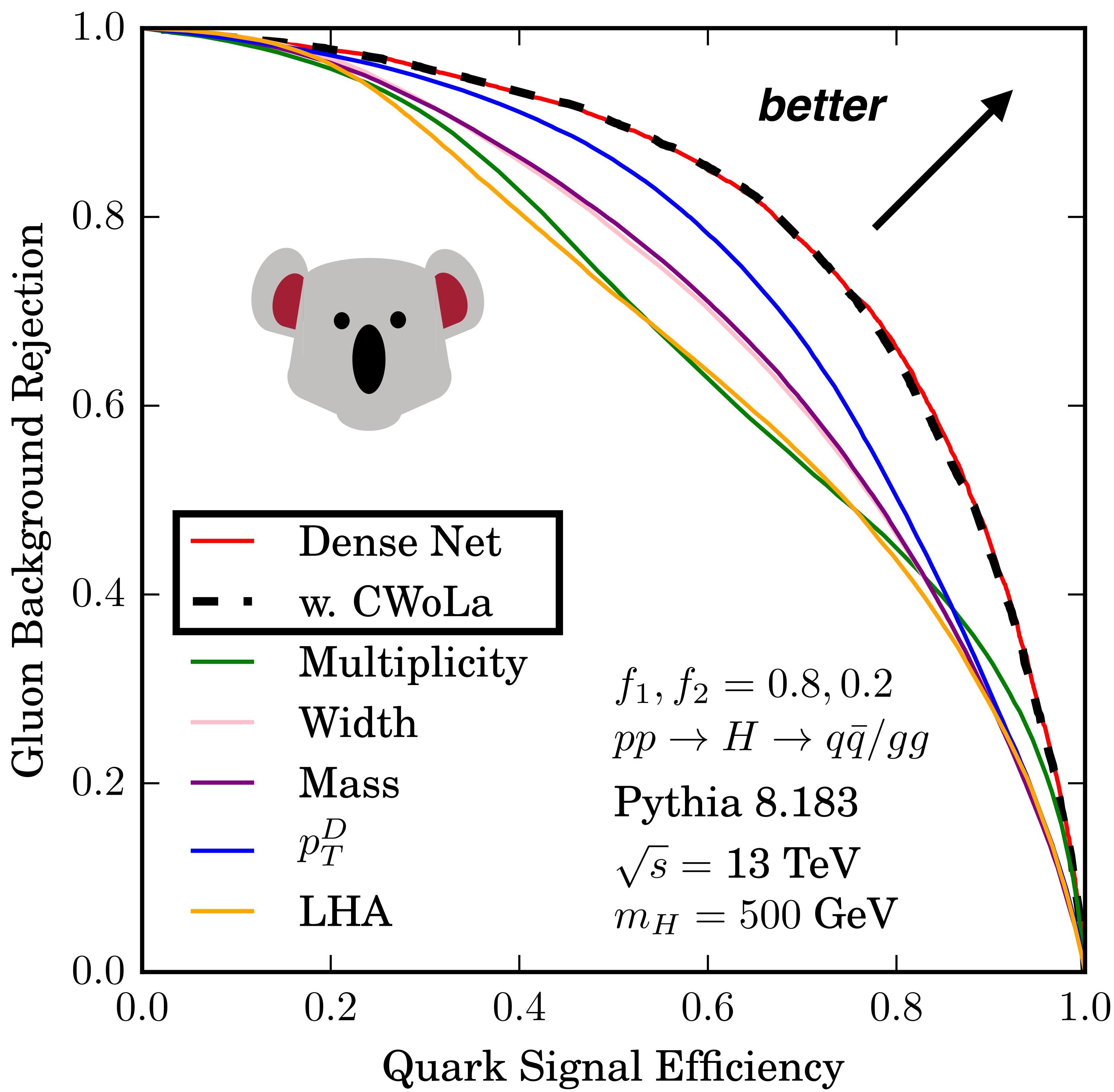
Classification Without Labels

Solution: Train **directly on data** using mixed samples

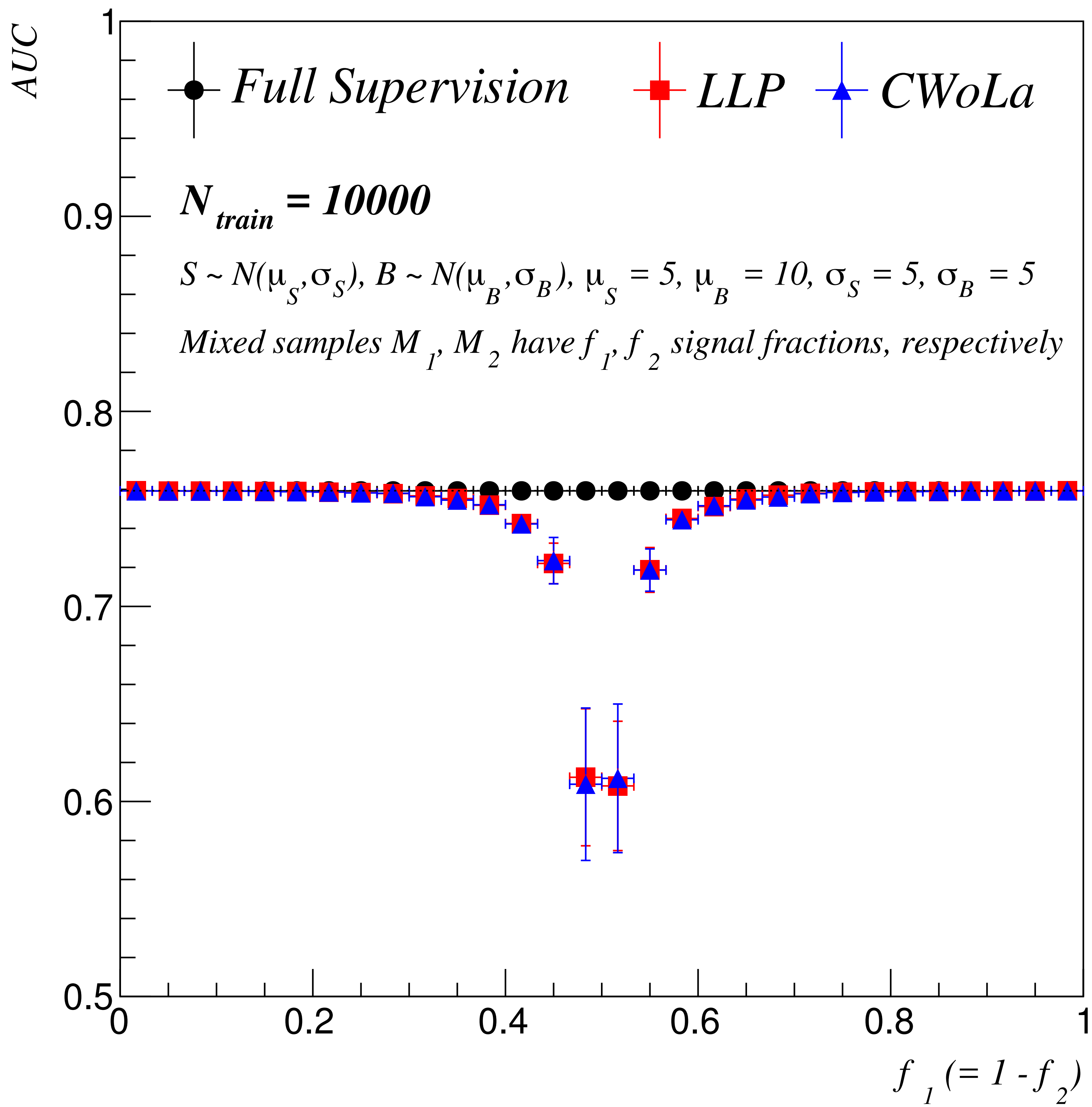


E. Metodiev, **BPN**, J. Thaler, JHEP 10 (2017) 51

Works in low-dimensions



A note about training statistics



As with LLP, need sufficient effective statistics

Can't learn when the two proportions are the same.

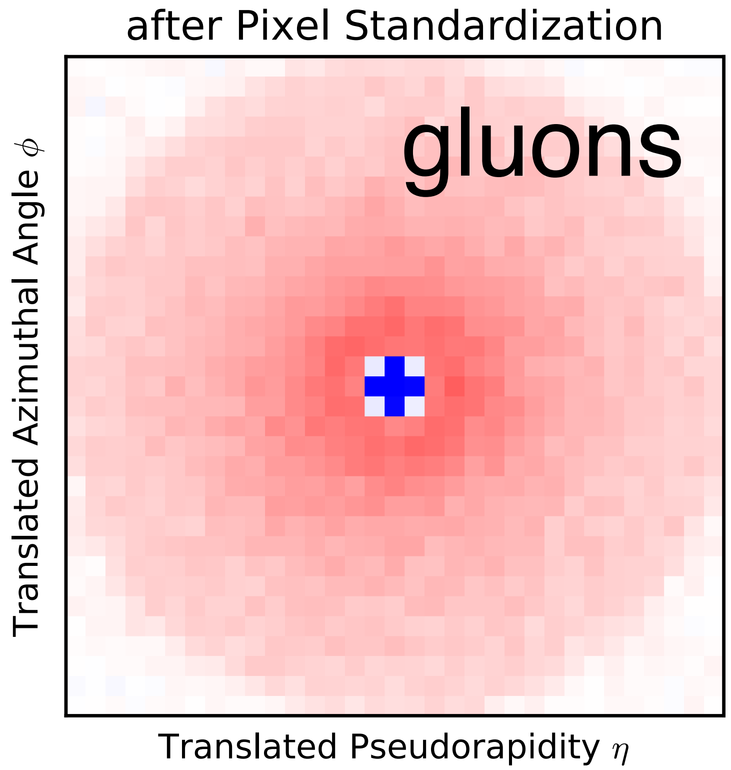
Methods Overview

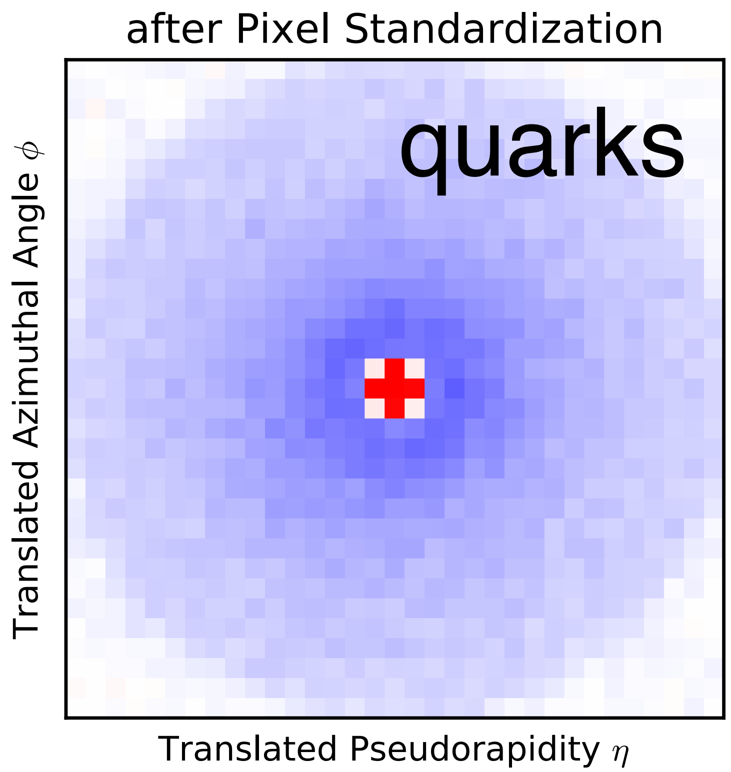
Property	**LLP**	**CWoLa**
Compatible with any trainable model	✓	✓
No training modifications needed	✗	✓
Training does not need fractions	✗	✓
Smooth limit to full supervision	✗	✓
Works for > 2 mixed samples	✓	?

Next step: what about high dim.?

There are many O(1)-dimensional ML problems for jets, but since the full radiation pattern is higher dimensional, need to go to bigger!

We'll use jet images as a testing ground, still focusing on quarks versus gluons.





P. Komiske, E. Metodiev, M. Schwartz, JHEP 01 (2017) 110

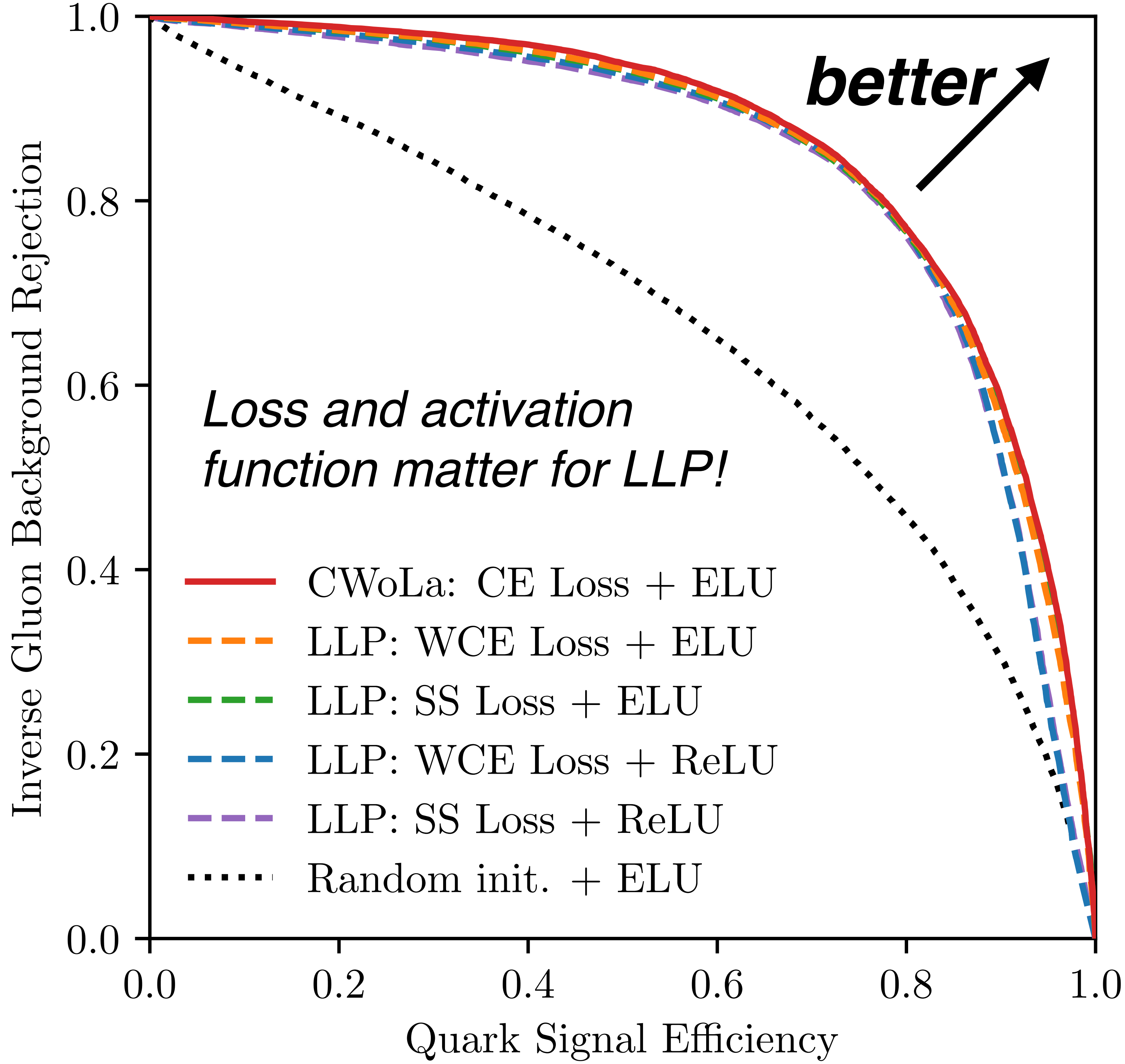
Some Technical Details

The CWoLa approach works out-of-the box - can use well-tested CNN architecture with usual cross-entropy loss.

On the other hand, LLP requires significant work on the technical implementation / optimization.

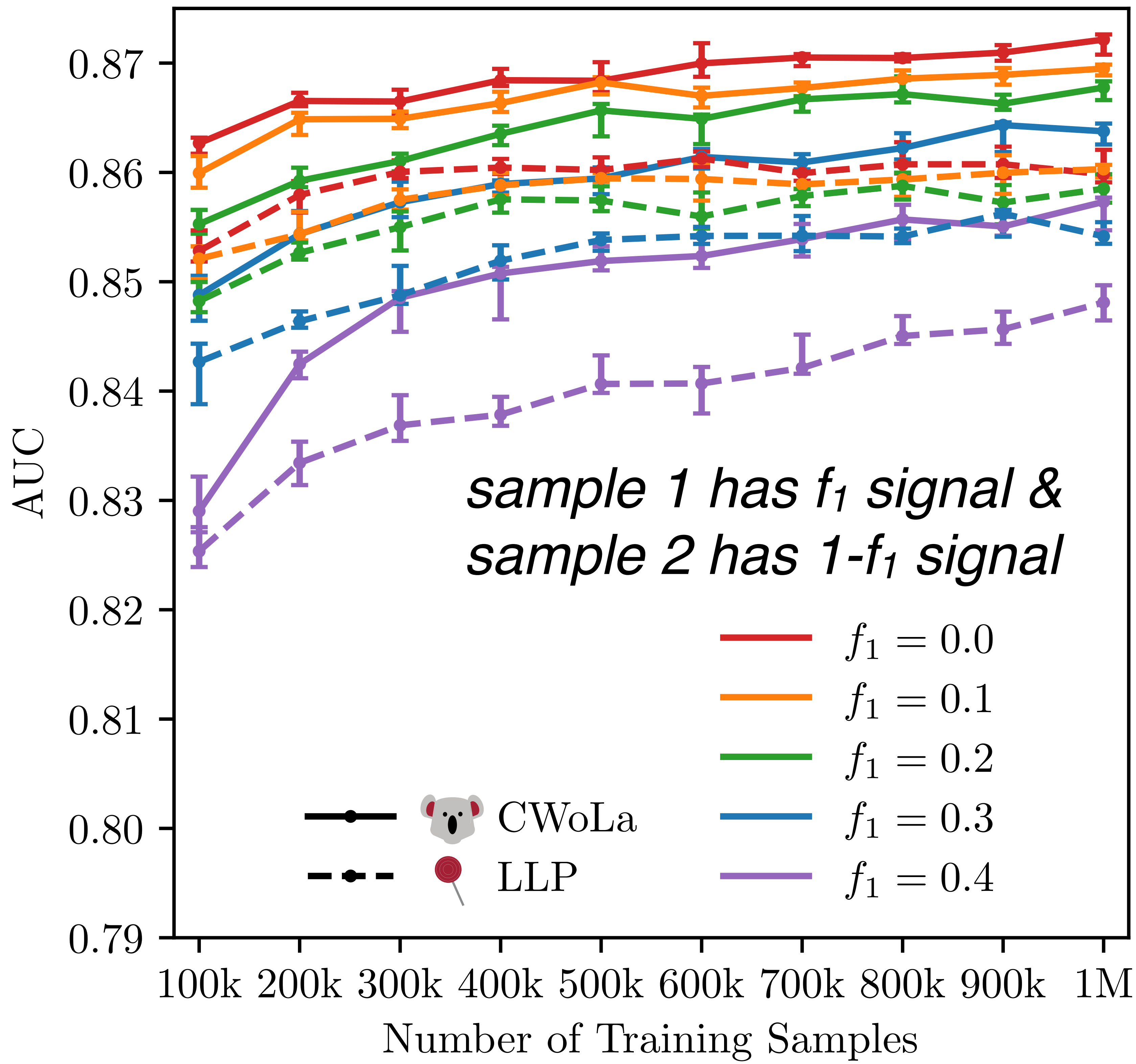
$$\ell_{\mathrm{WMSE}} = \sum_a \left(f_a - \frac{1}{N} \sum_{i=1}^{N} h(\mathbf{x}_i) \right)^2 \qquad \ell_{\mathrm{WCE}} = \sum_a \mathrm{CE}\left(f_a, \frac{1}{N} \sum_{i=1}^{N} h(\mathbf{x}_i) \right)$$

Works in many-dimensions!



P. Komiske, E. Metodiev, **BPN**, M. Schwartz, Phys. Rev. D 98, 011502(R), arXiv:1801:10158

A note about training statistics

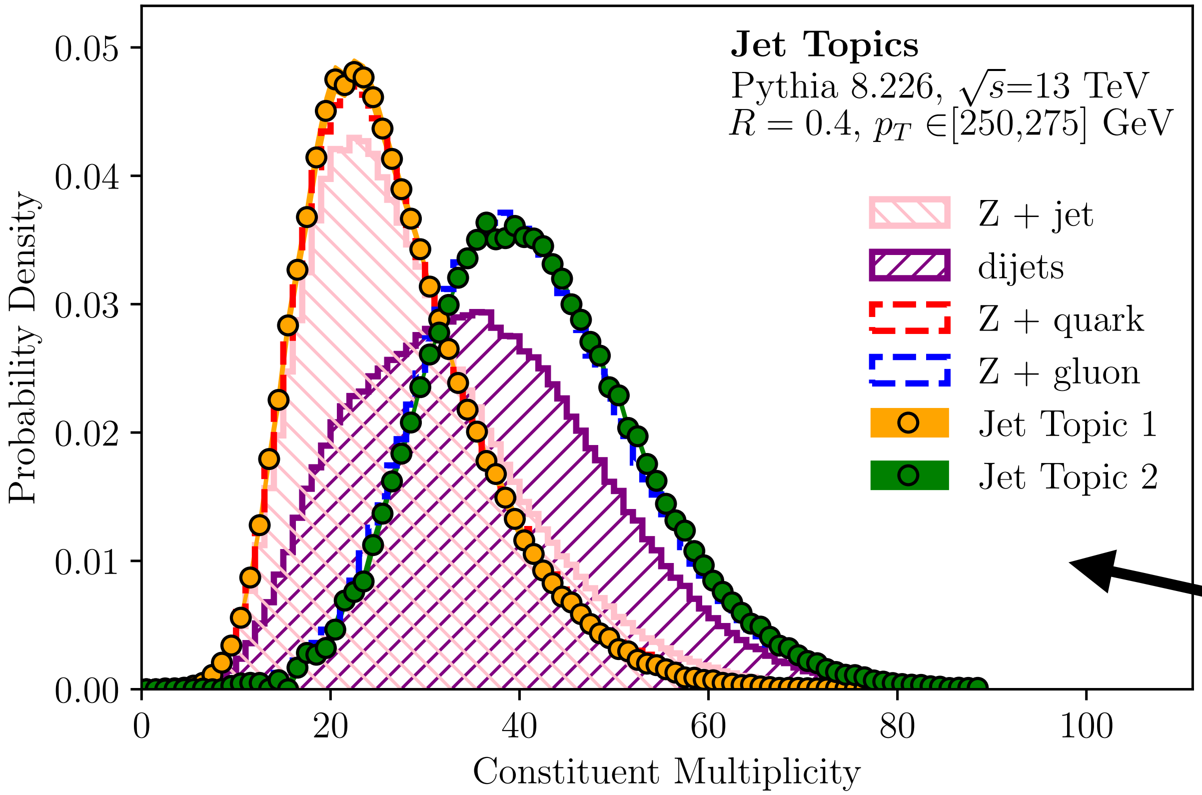


Hybrid Approaches

As usual, it is likely that the best approach will use all of the available information, including some input from simulation.

…this could be as simple as pre-training in MC and then running weak supervision or actually explicitly combining weak supervision and pivoting.

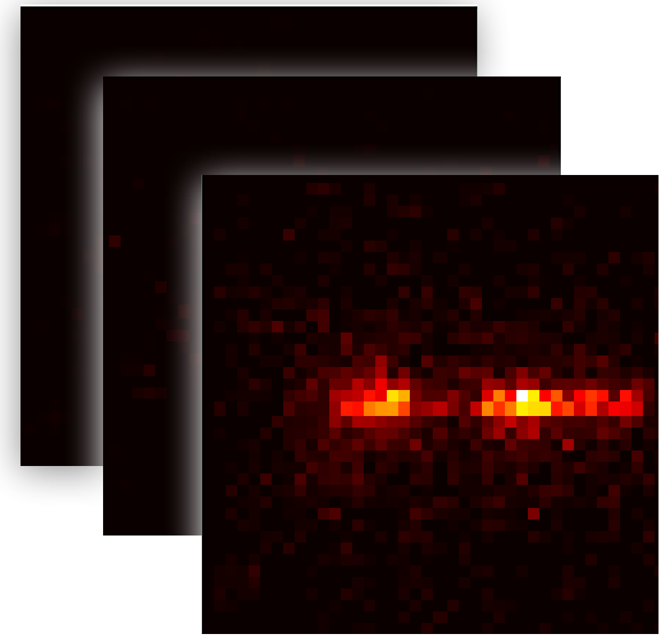
E. Metodiev and J. Thaler, Phys. Rev. Lett. 120 (2018) 241602



Another interesting direction is to push the weak supervision paradigm a step further and **define** the classes so that it works.

ML beyond classification

There is a lot more ML can do than classify examples!



Classification

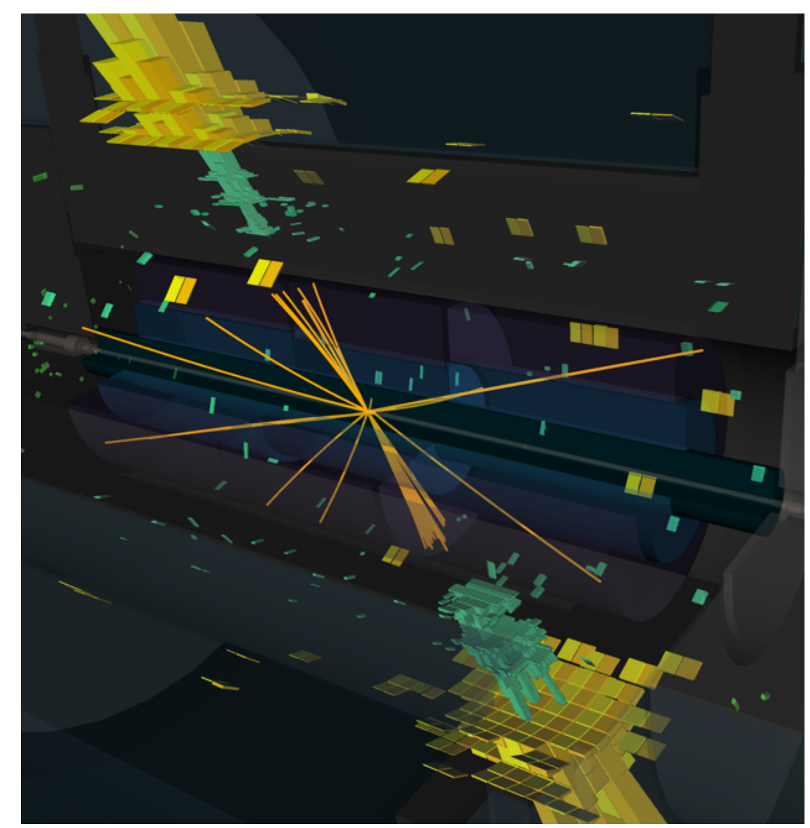
arbitrarily many categories

Regression

map noise to structure

Generation

provide examples for training



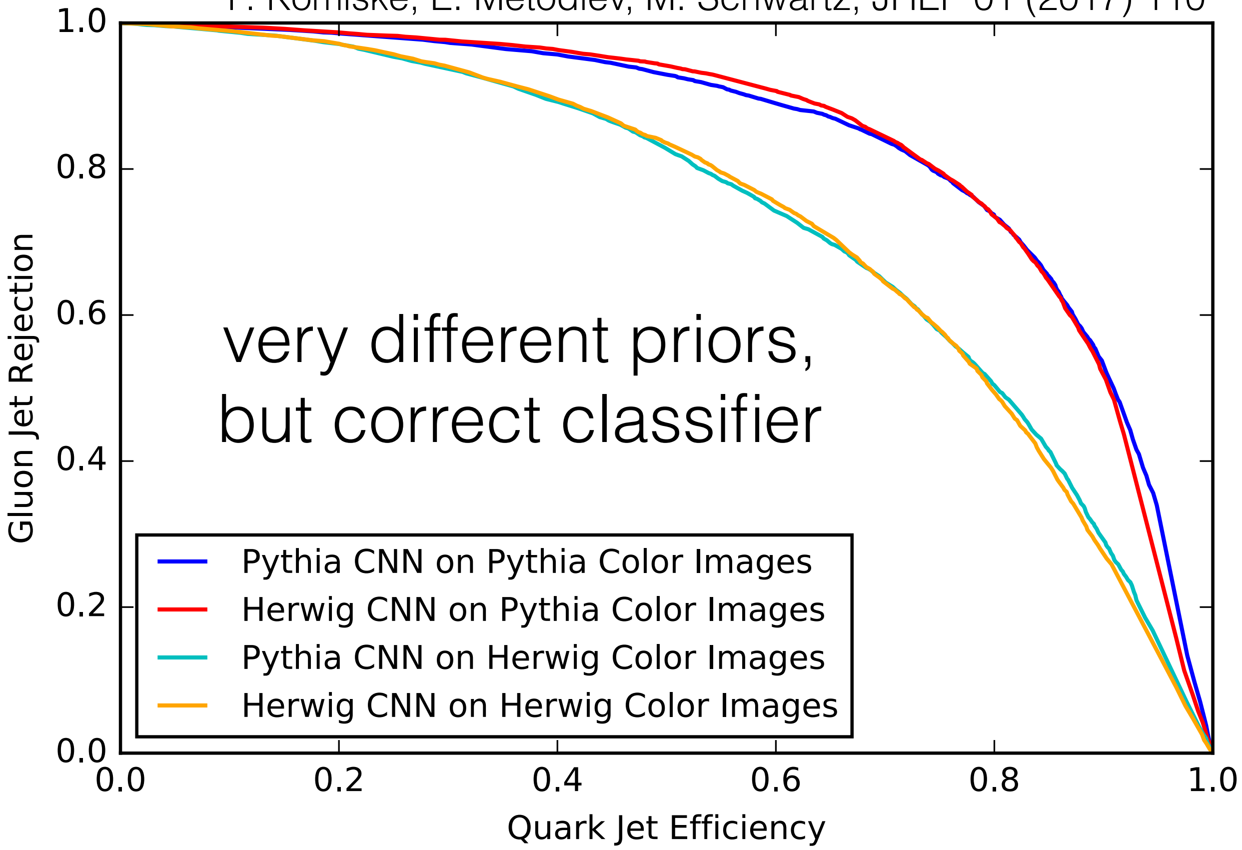
Simulation dependence + regression

One source of MC dependence is the same as classification:

→ mis-modeling dependencies between features

However, there is a new source:
dependence on the feature priors.

P. Komiske, E. Metodiev, M. Schwartz, JHEP 01 (2017) 110

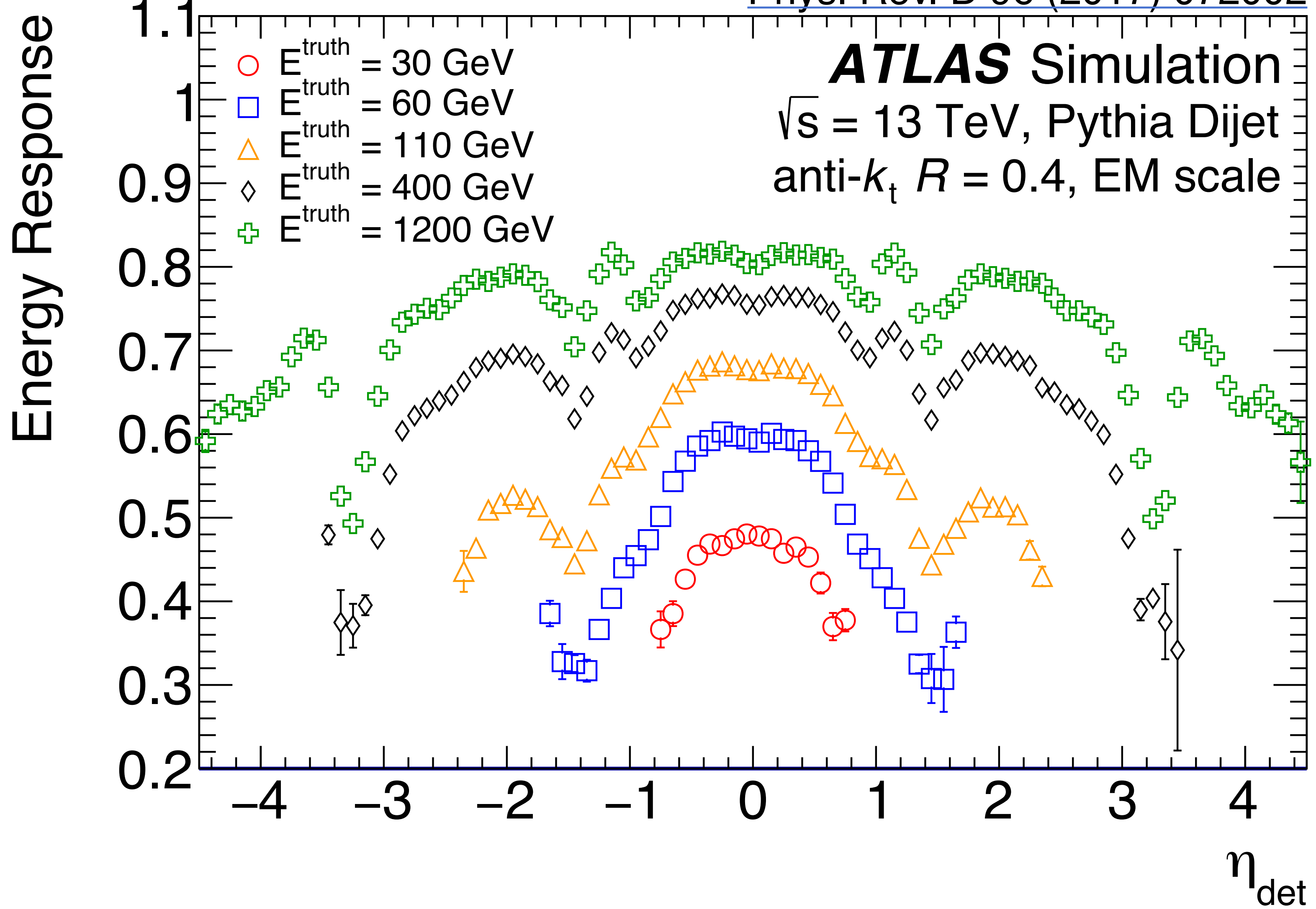


This is really new for regression - for classification, if p(x|signal) and p(x|background) are mis-modeled, you get the NP-optimal answer as long as their ratio is correct.

Simulation dependence + regression

An example that you can have in mind is jet energy calibration.

Phys. Rev. D 96 (2017) 072002



We want to predict the true energy given the measured energy

(and possibly other features - more on that soon)

…however what I'm about to say applied more generally (though the impact is biggest when the resolution is poorest)

What can go wrong?

Suppose you have some features x and you want to predict y.

detector energy *true energy*

One way to do this is to find an f that minimizes the mean squared error (MSE):

$$f = \text{argmin}_g \sum_i (g(x_i) - y_i)^2$$

Then, f(x) = E[y|x].

If you did not know this, prove it! For fun, you can also show that f is the median if instead you used the mean absolute error.

Why is this a problem?

What can go wrong?

$$f(x) = E[y|x] = \int \mathrm{d}y \, y \, p(y|x)$$

$$E[f(x)|y] = \int \mathrm{d}x \, \mathrm{d}y' \, y' \, p_{\mathrm{train}}(y'|x) \, p_{\mathrm{test}}(x|y)$$

this need not be y even if $p_{train} = p_{test}$ (!)

One solution: Numerical inversion

ATLAS and CMS use a trick to be prior-independent:

Numerical inversion *instead of predicting y from x, predict x from y and then invert the function*

… put another way:

learn $f: y \rightarrow x$ and then for a given x, predict $f^{-1}(x)$

by construction, f is independent of p(y) and thus f^{-1} also does not depend on p(y), as desired.

For math details, see A. Cukierman and B. Nachman, NIMA 858 (2017) 1

Caveats about numerical inversion

This procedure is independent of the prior p(y) but may not close exactly, i.e. $E[f^{-1}(x)|y]$ may not be y.

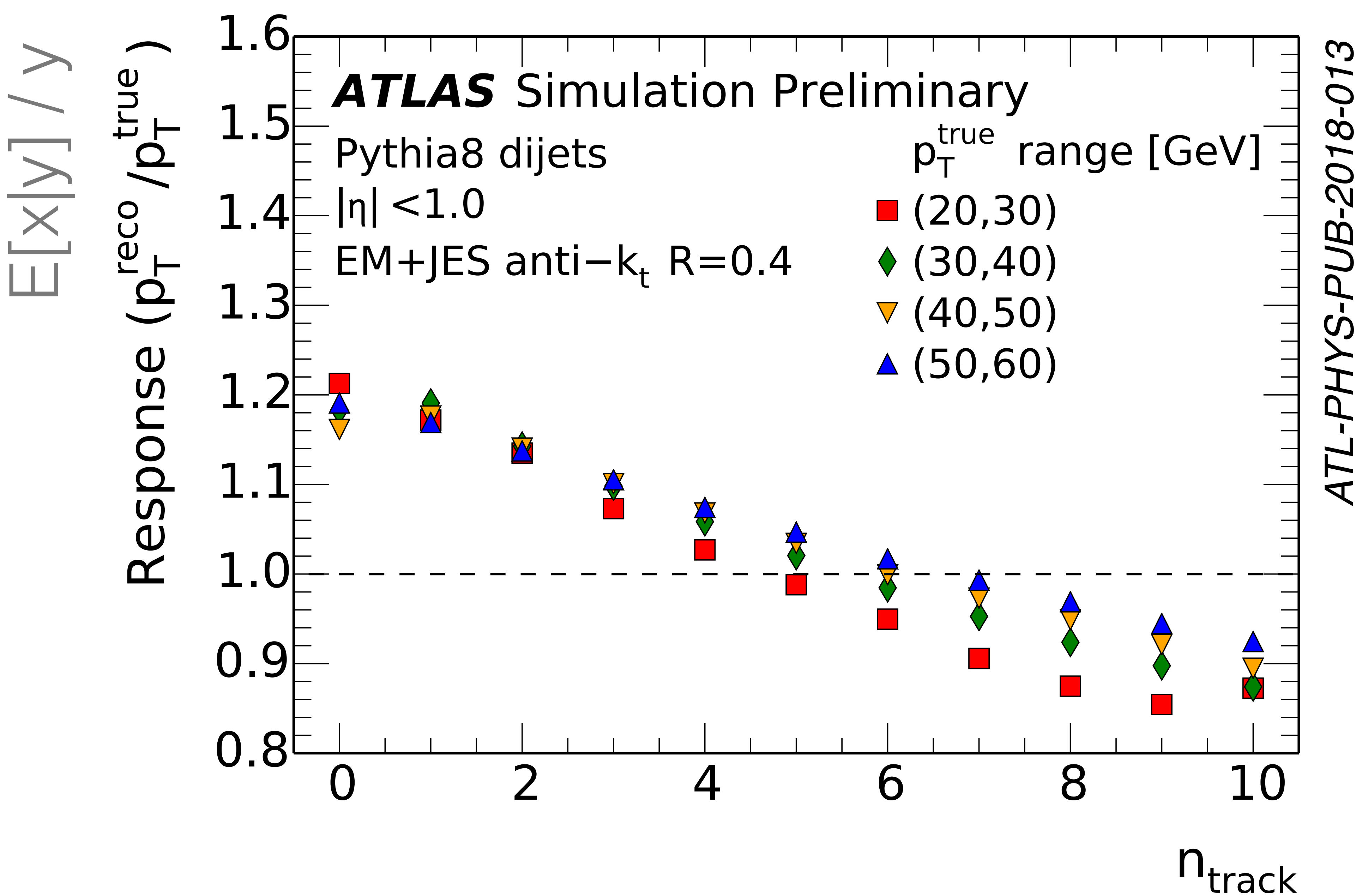
…under mild assumptions, it does close for the mean absolute error, but usually has some non-closure for the MSE.

Also, the calibration procedure can distort the underlying distribution, i.e. if you start with a Gaussian, you almost never end up with exactly a Gaussian.

For math details, see A. Cukierman and B. Nachman, NIMA 858 (2017) 1

+ more features

The detector response of jets depends on many properties of the jet. Ideally, the calibration can include this!

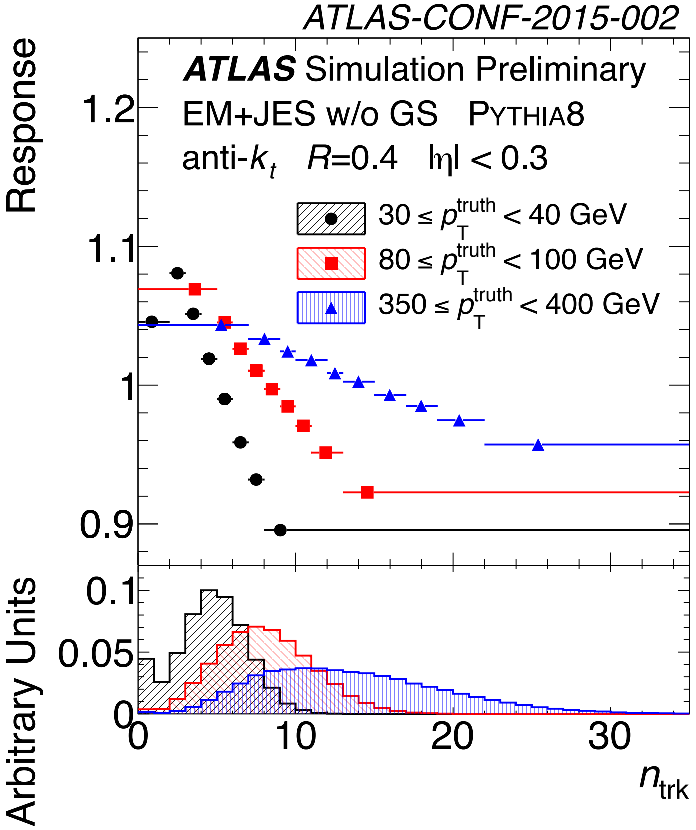


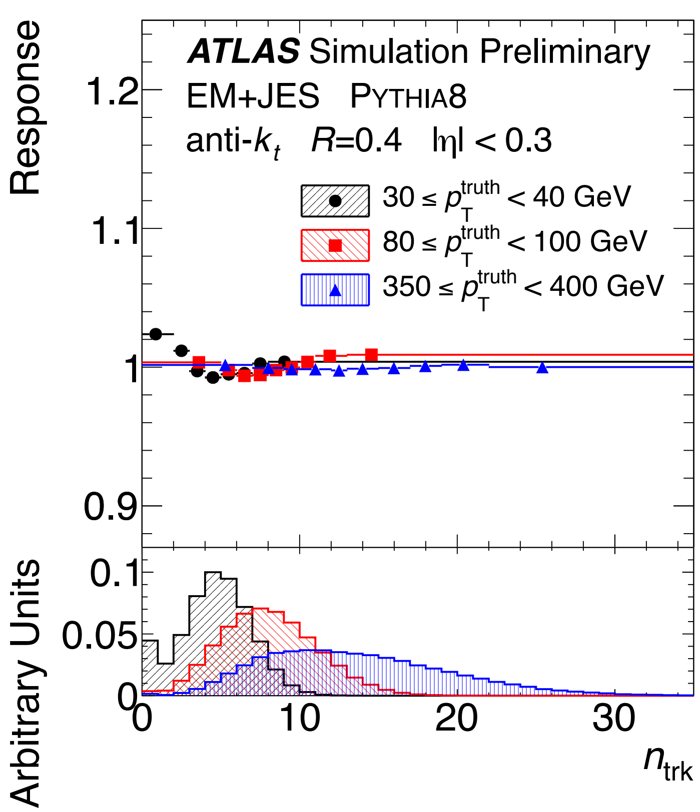
Global sequential calibration

The current ATLAS approach to including more features is to repeat NI sequentially:

$$p_T^{\text{reco}} \mapsto \hat{p}_T^{\text{reco}} = f_{\theta_n}^{-1}\left(\cdots f_{\theta_2}^{-1}\left(f_{\theta_1}^{-1}\left(p_T^{\text{reco}}\right)\right)\cdots\right)$$

ATLAS-CONF-2015-002





This works well when the jet response is independent of θ_i given θ_j.

Machine learning calibration

For reasons discussed earlier, we can't include correlations by learning y given x and all the θ's.

However, it would still be great to use machine learning to automatically and efficiently make use of correlated information.

We cannot use numerical inversion out-of-the-box because we now have a many-to-one function.

Generalized numerical inversion

Since we are not (necessarily) interested in calibrating the θ's, we can generalize NI as follows:

(1) Learn a function f to predict x given y and all the θ's.

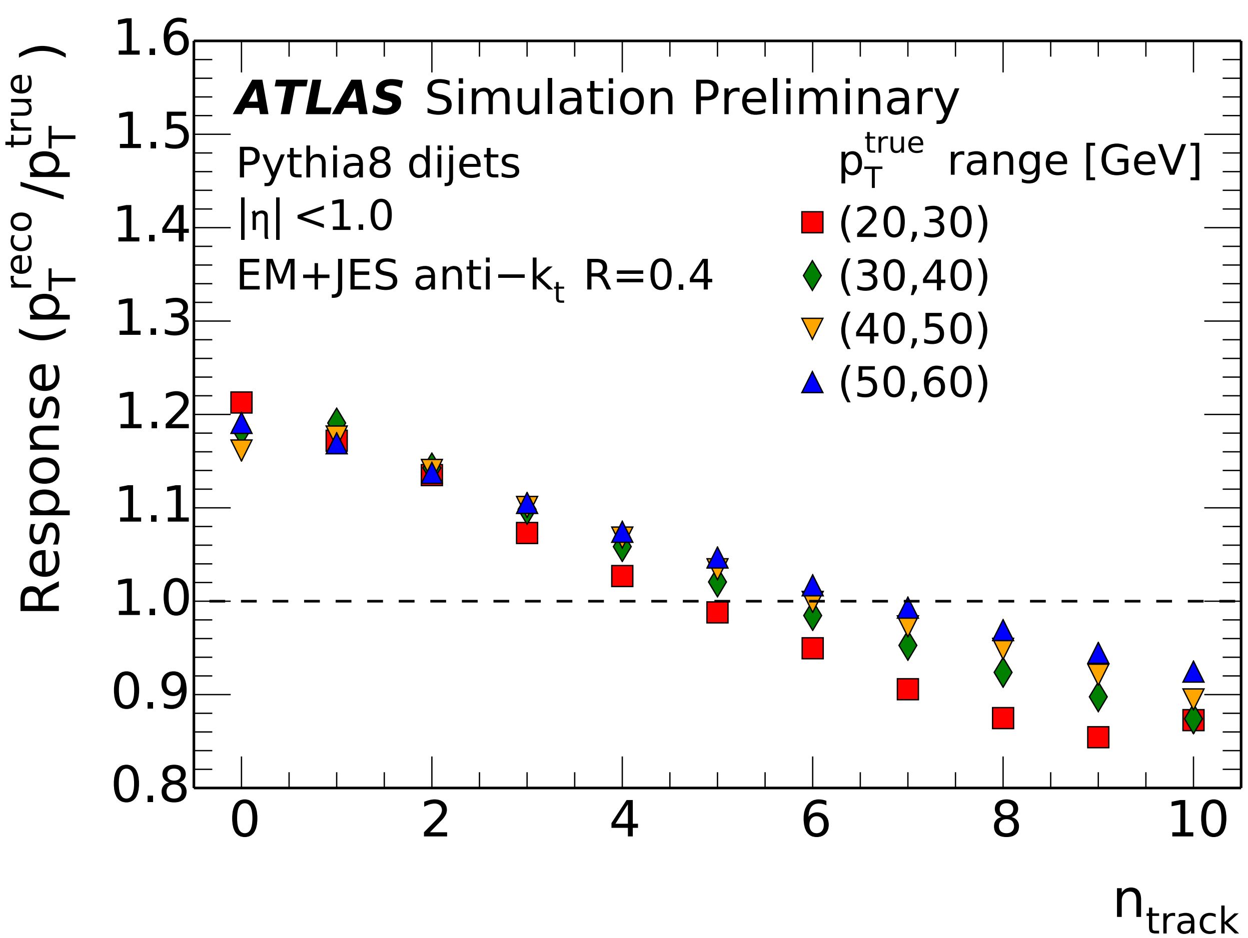
(2) For every combination of θ, invert f.

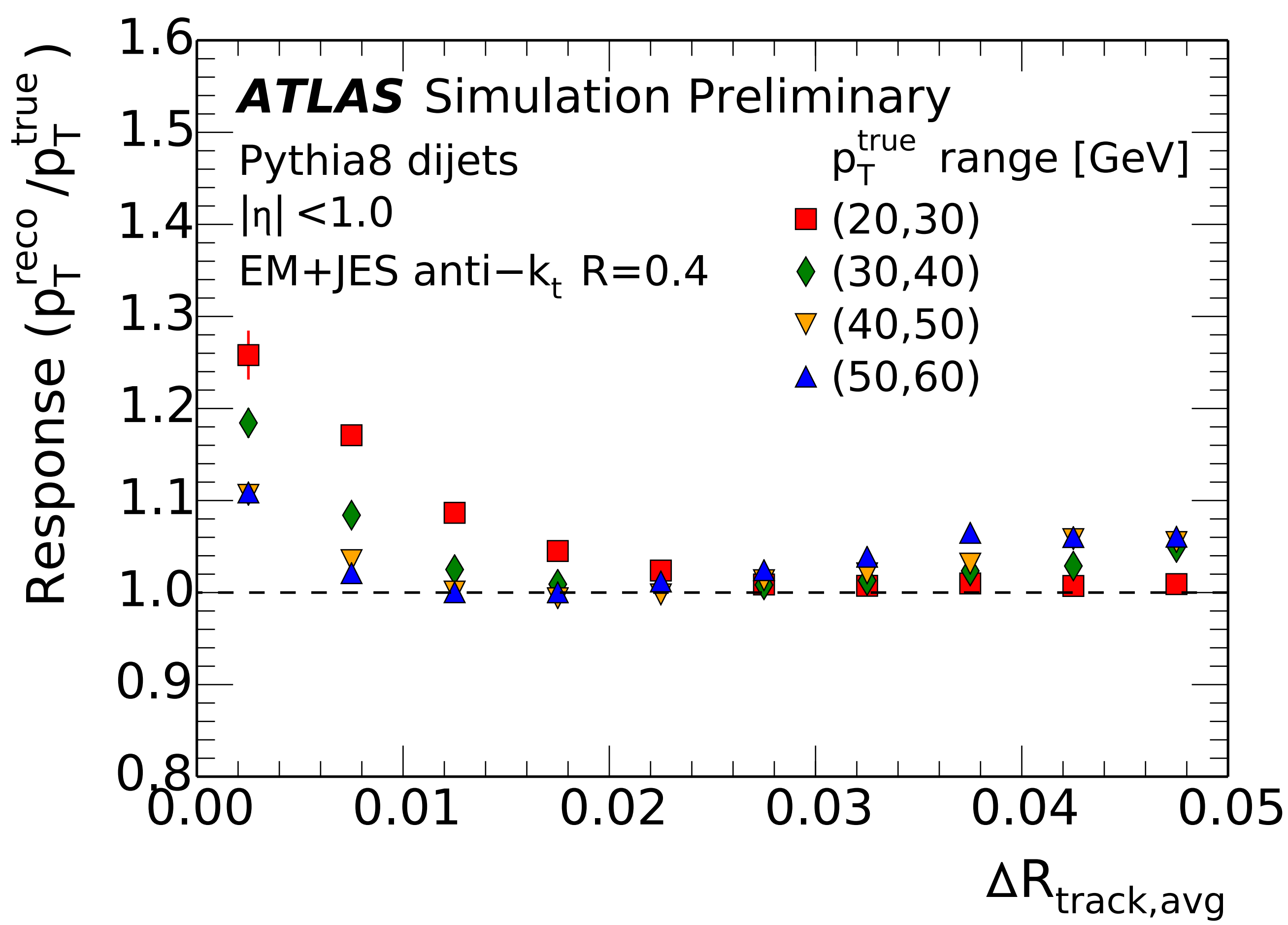
(3) Calibrate via $x \rightarrow f_\theta^{-1}(x)$

Step (2) is intractable, so replace it with another learning step: predict y given $f(y,\theta)$ and θ.

GNI in action

Consider two features:

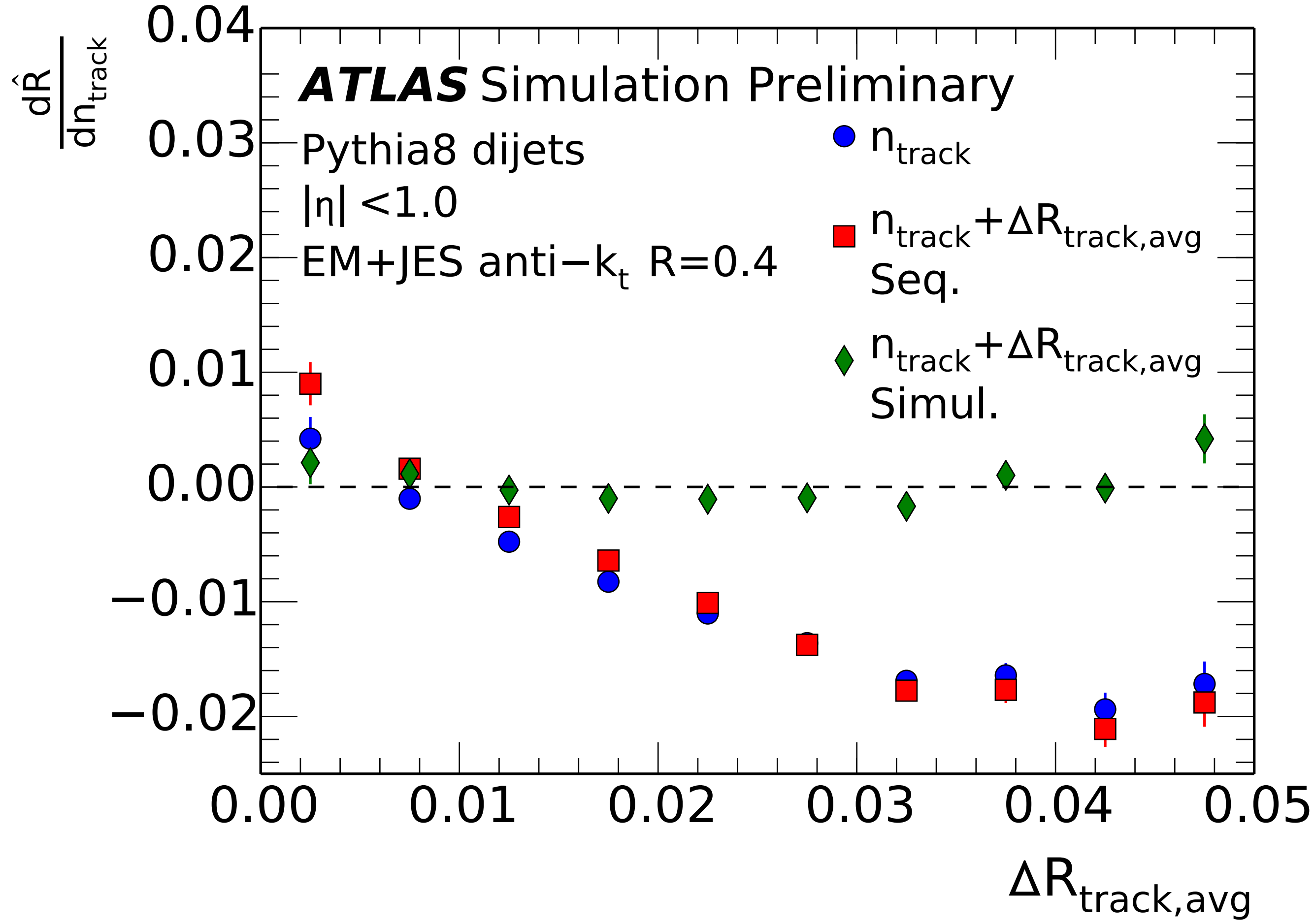


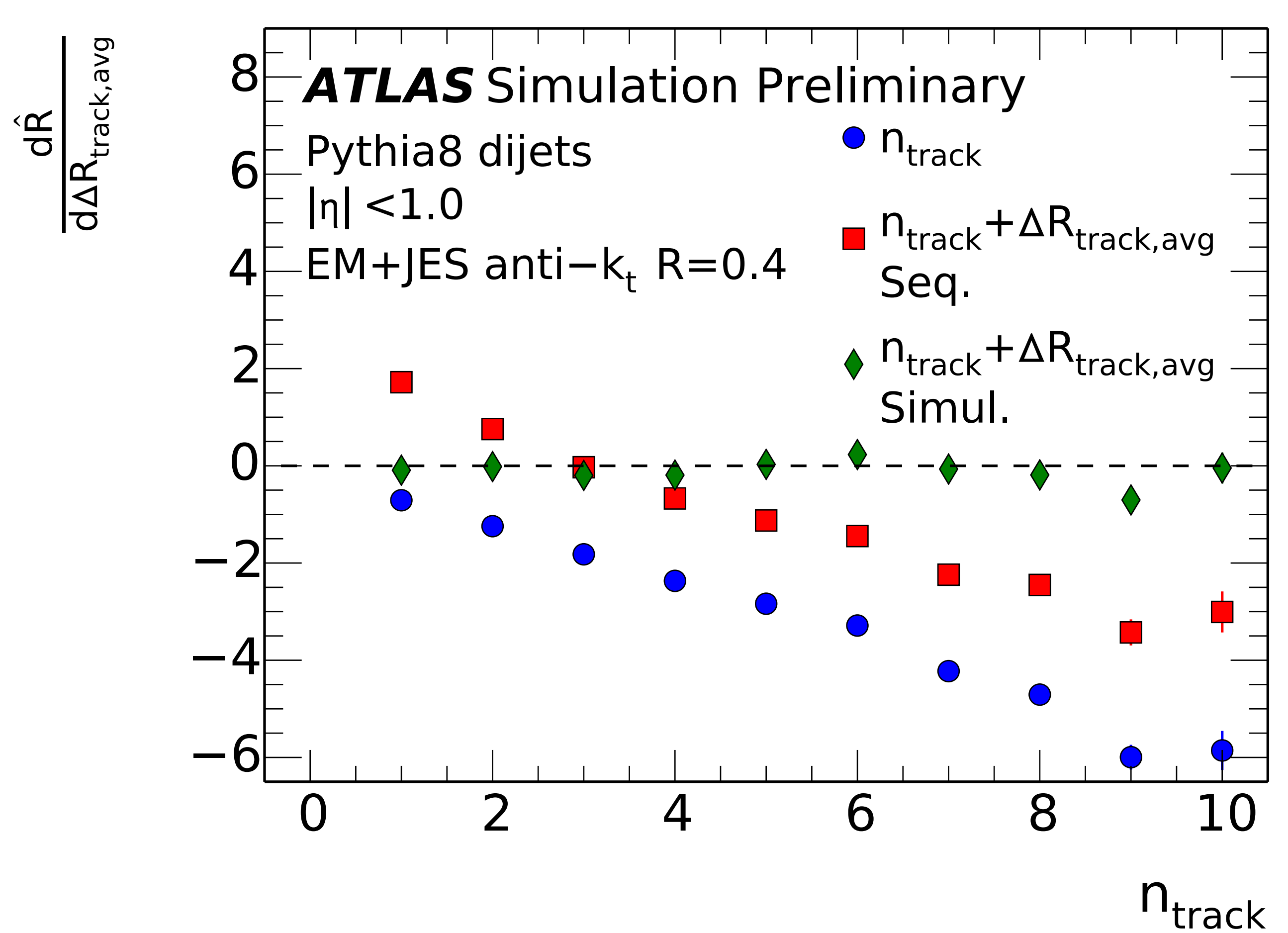


average track p_T-weighted distance from jet center

GNI in action

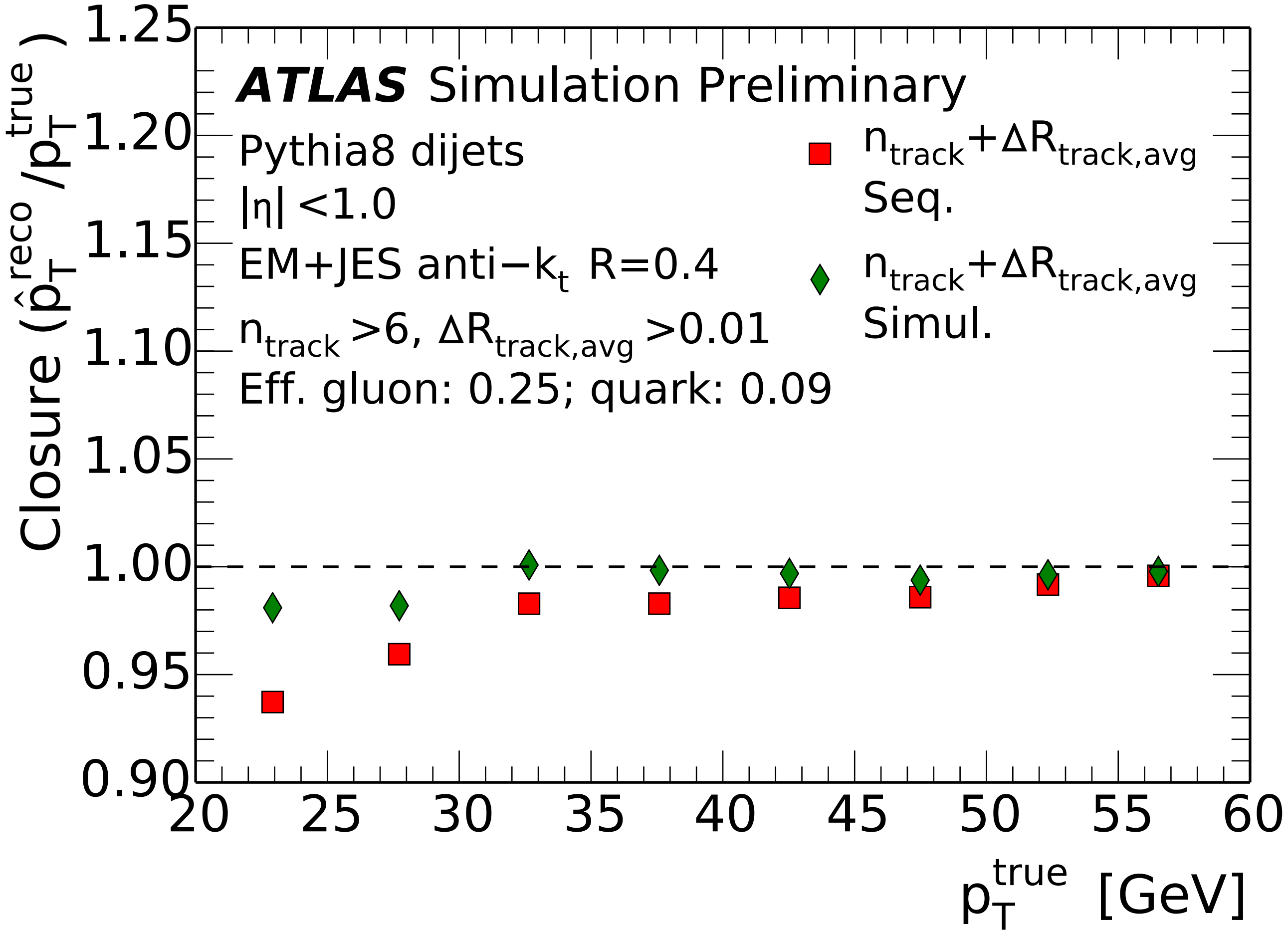
$\hat{R}$ is the calibrated E[x|y] / y

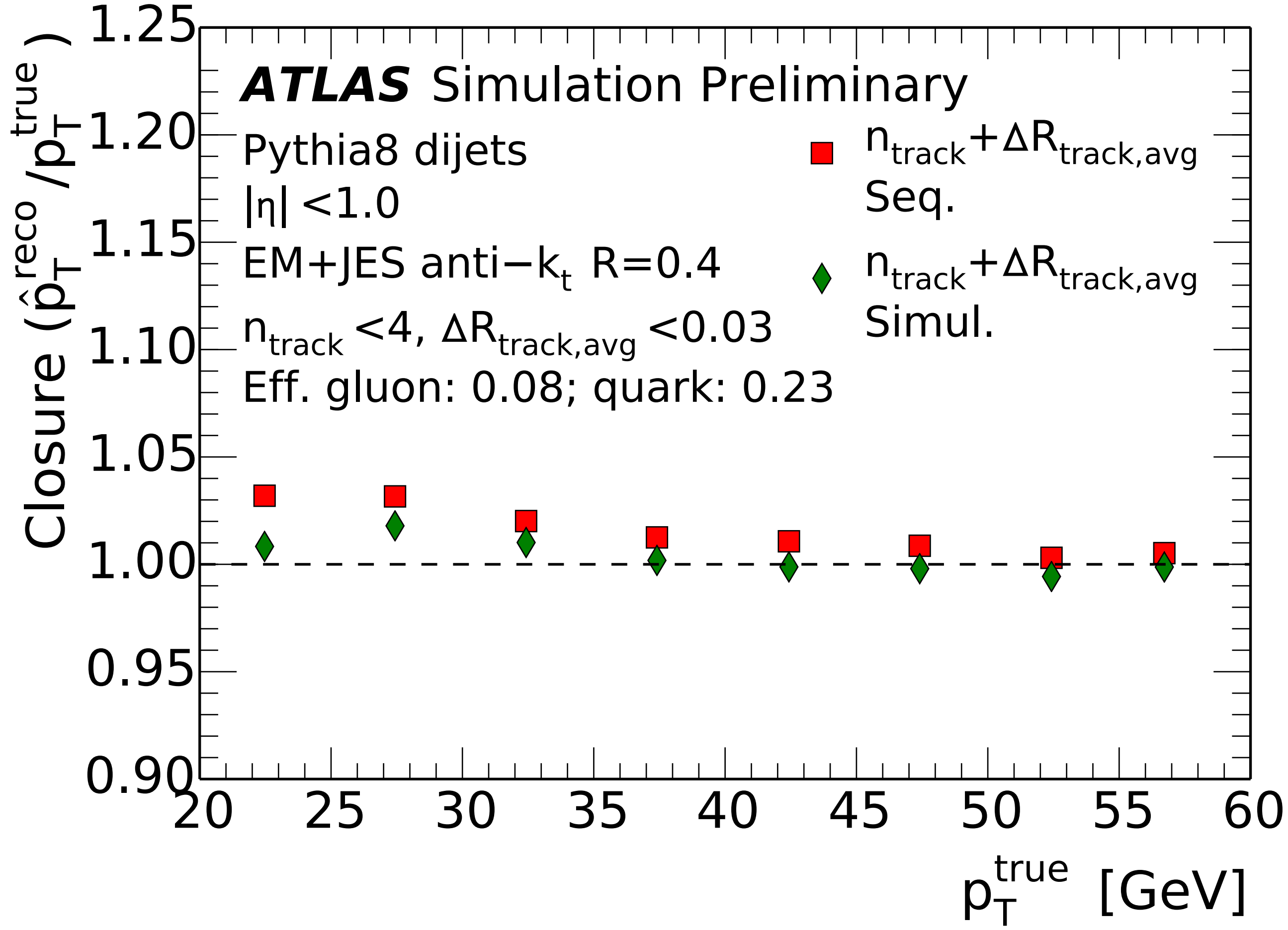




Only the simultaneous approach removes the full residual dependence!

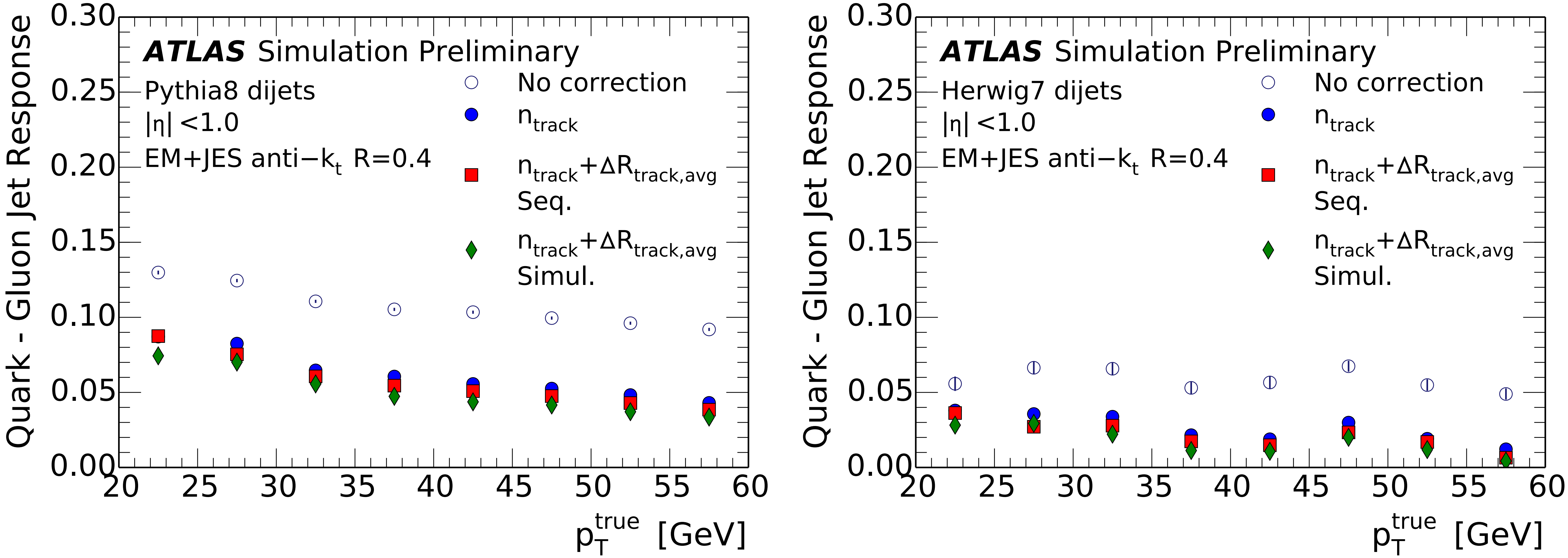
GNI in action





Slightly better closure for the simultaneous calibration.

GNI in action



Slightly less dependence on the origin using the simultaneous approach.

The future

Adding more features (with more interdependencies) will lead to more dramatic improvements.

We can also extend this approach to calibrate other observables and even simultaneously calibrate some of the θ's (even more generalized NI!)

There is an interesting connection to unfolding (see e.g. A. Glazov, 1712.01814).

Anomaly Detection

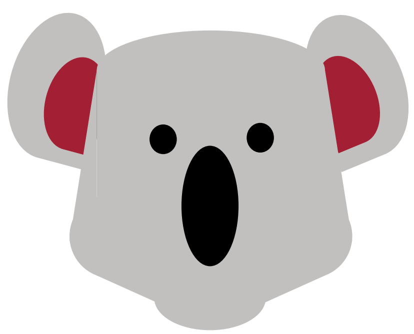
I will leave you with one last, but very exciting topic.

One of the most important goals of HEP is to search for new particles. However, we have not found anything (significantly) unexpected in a while … we need simulation-independent ways of searching for new particles !

anomalies, i.e. something unexpected

N.B. The approach discussed here is not the only one - see also M. Farina, Y. Nakai, D. Shih, 1808.08992 & T. Heimel, G. Kasieczka, T. Phlen, J. Thompson, 1808.08979 for an alternative approach based on auto-encoders.

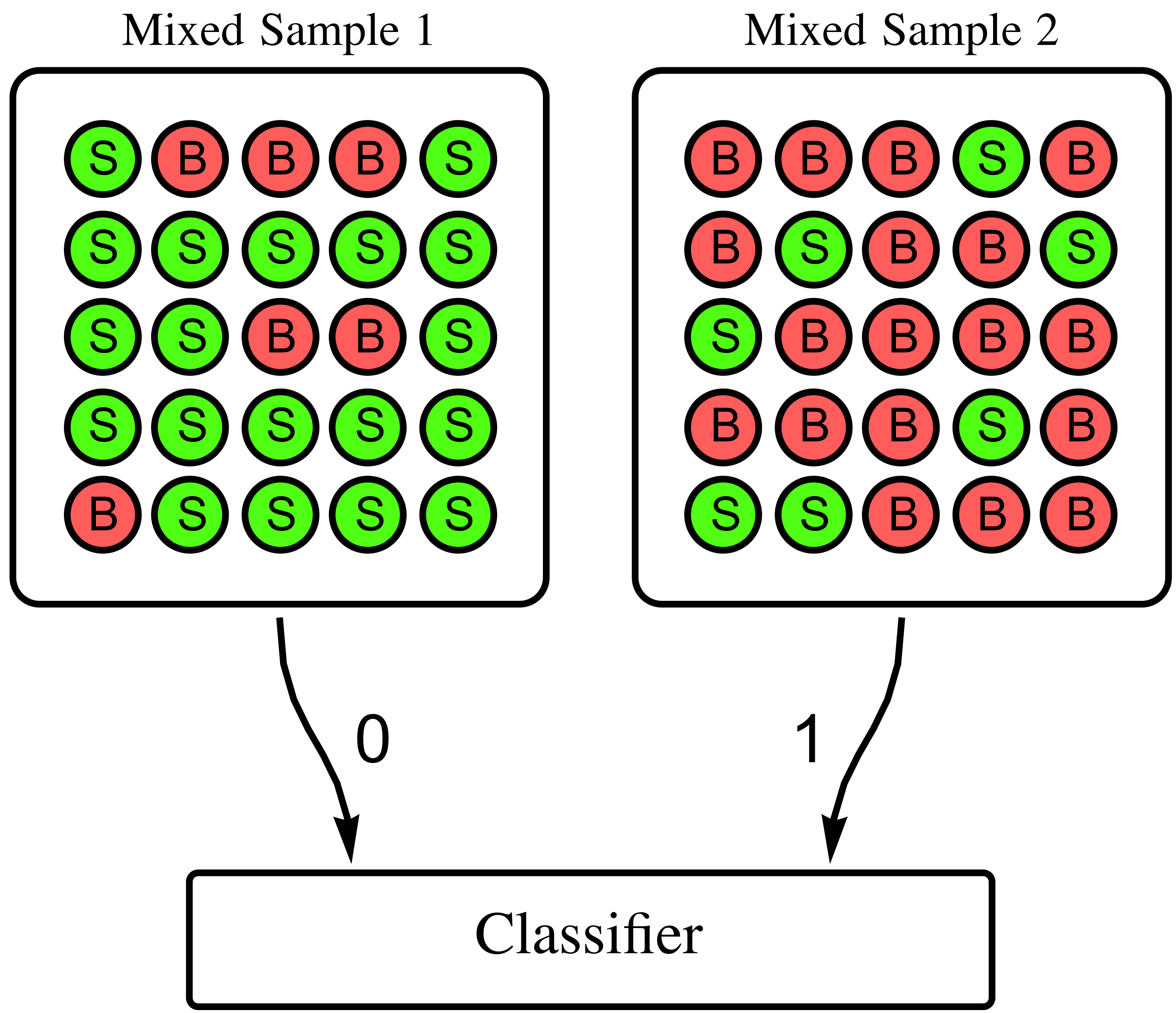
Remember CWoLa …



CWoLa

Classification Without Labels

Solution: Train **directly on data** using mixed samples



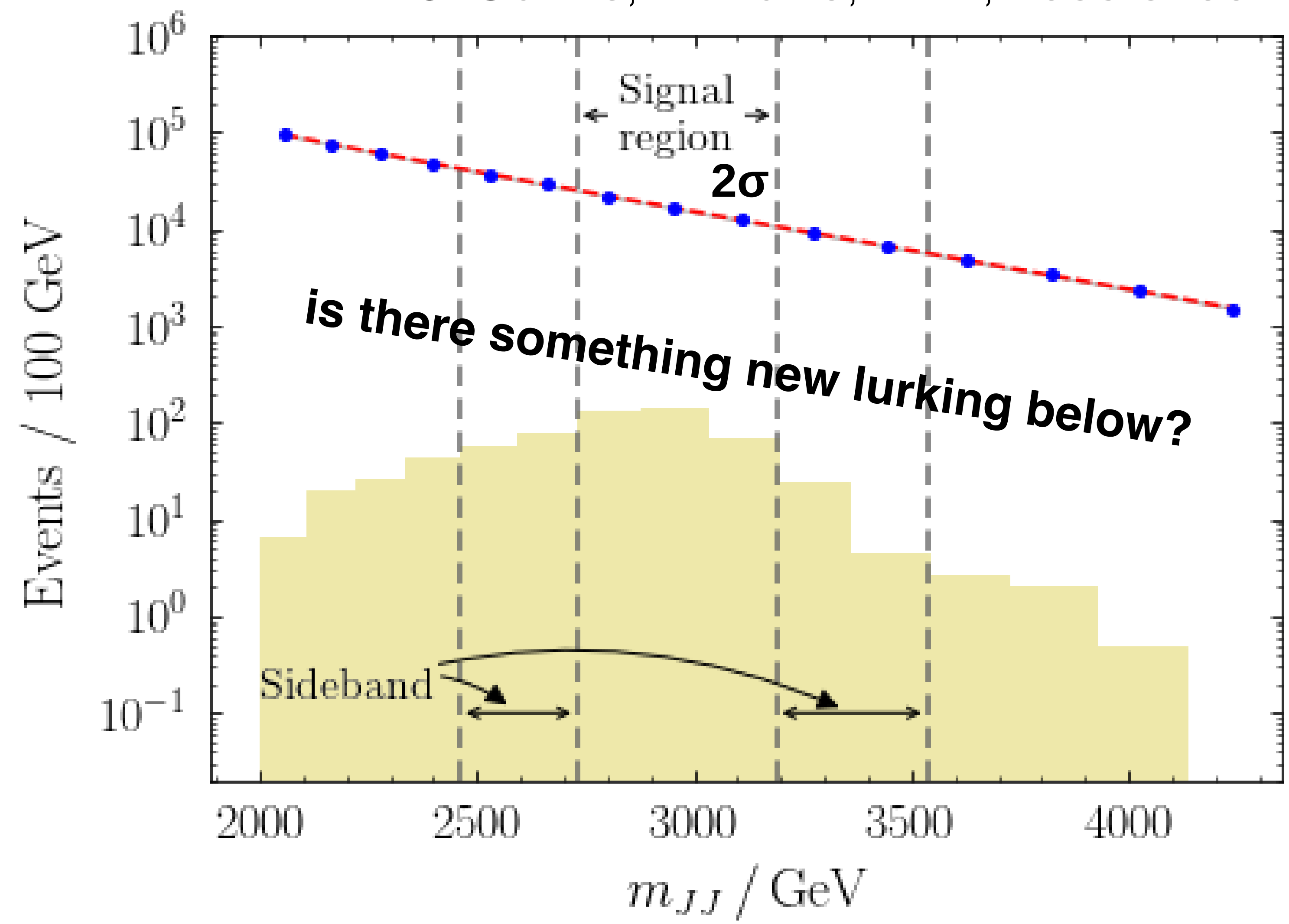
E. Metodiev, **BPN**, J. Thaler, JHEP 10 (2017) 51

Weak/unsupervised learning for anomalies

Can we take this idea one step further to look for something unexpected?

= CWoLa Hunting*

J. Collins, K. Howe, **BPN**, 1805.02664



Minimal assumption: if there is a signal, it is localized in one known dimension.

*Image from this article. This Koala is actually being freed - I do not condone violence against these animals!

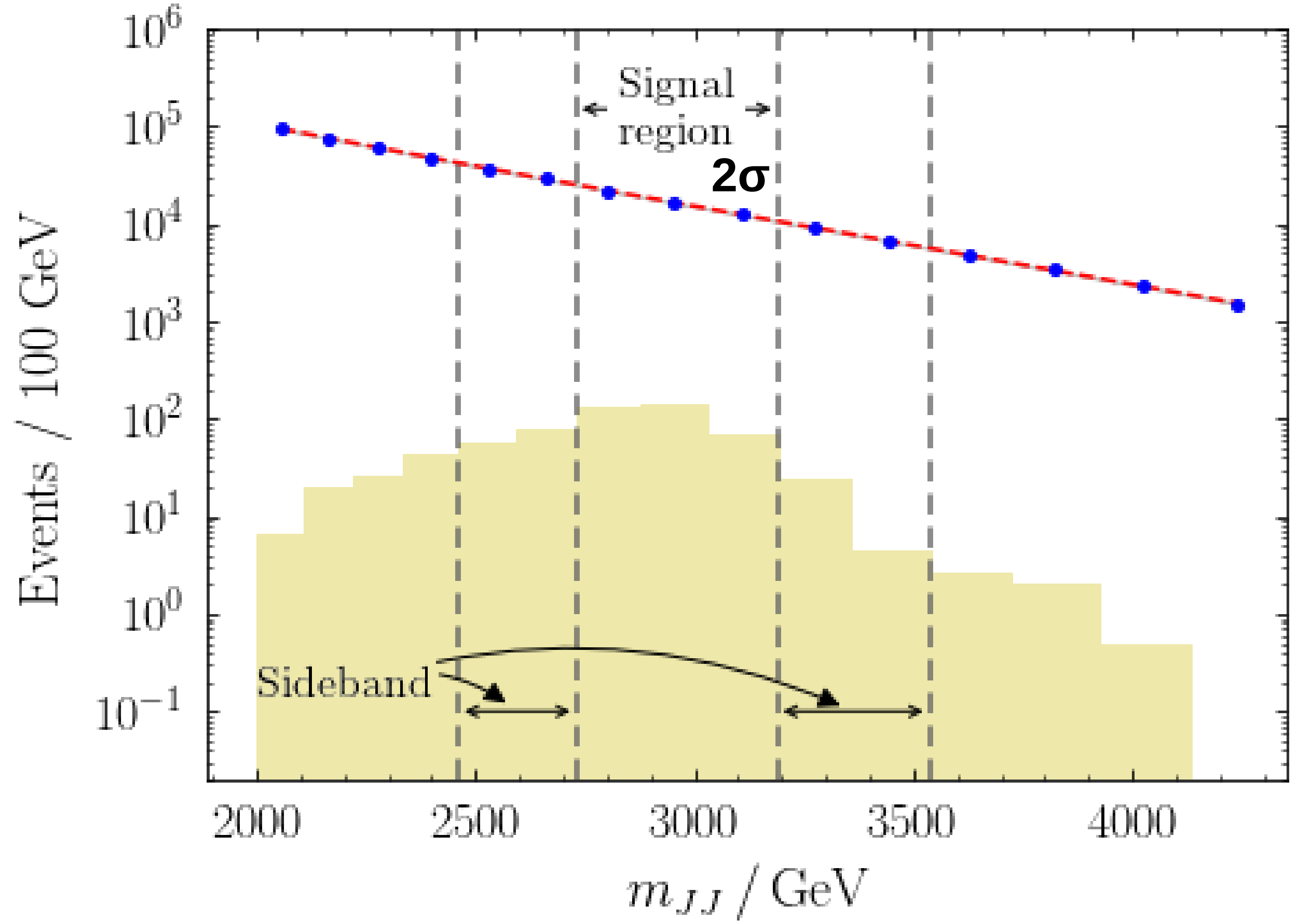
Weak/unsupervised learning for anomalies

Mixed sample 1:
signal region

Mixed sample 2:
sideband region

Train a classifier to distinguish the two mixed samples.

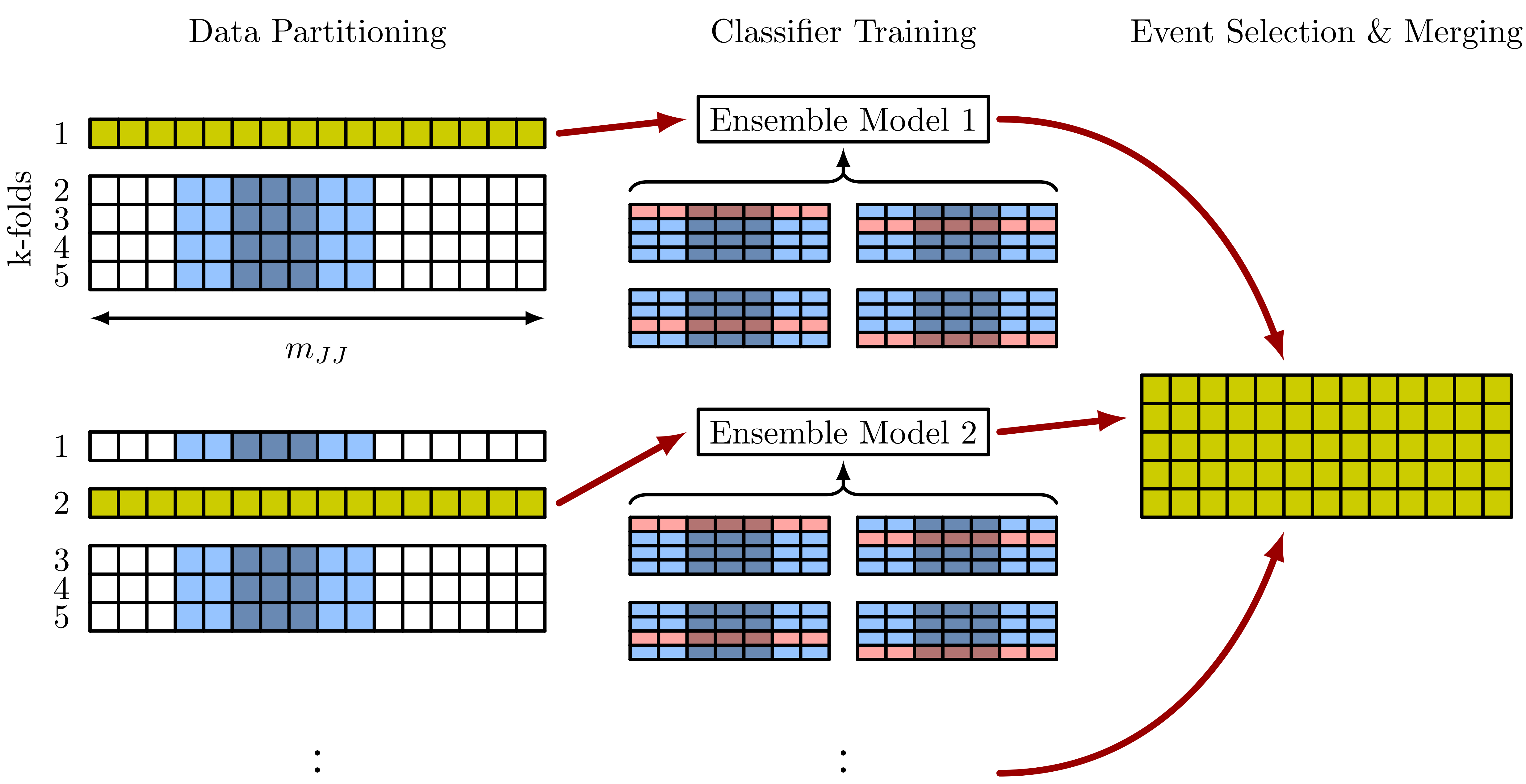
J. Collins, K. Howe, **BPN**, 1805.02664



If there is a signal, there will be something to learn and the signal will be enhanced. If no signal, nothing to learn.

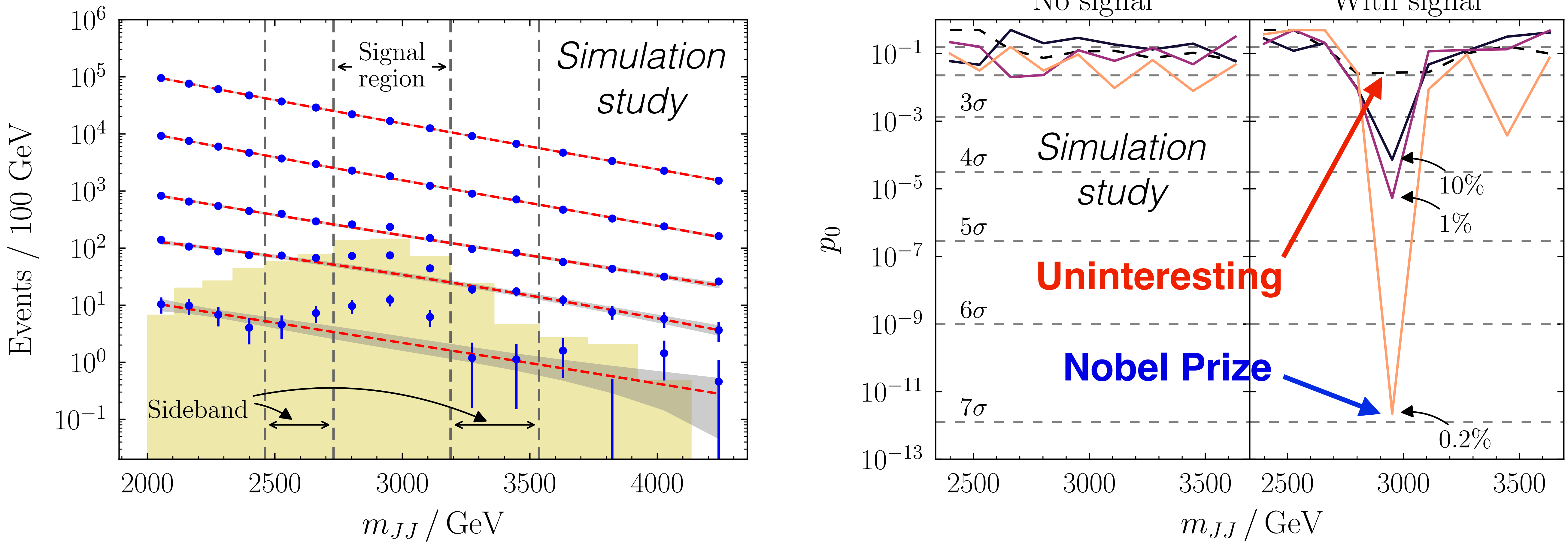
Weak/unsupervised learning for anomalies

Need to be careful about testing/training on the same data.



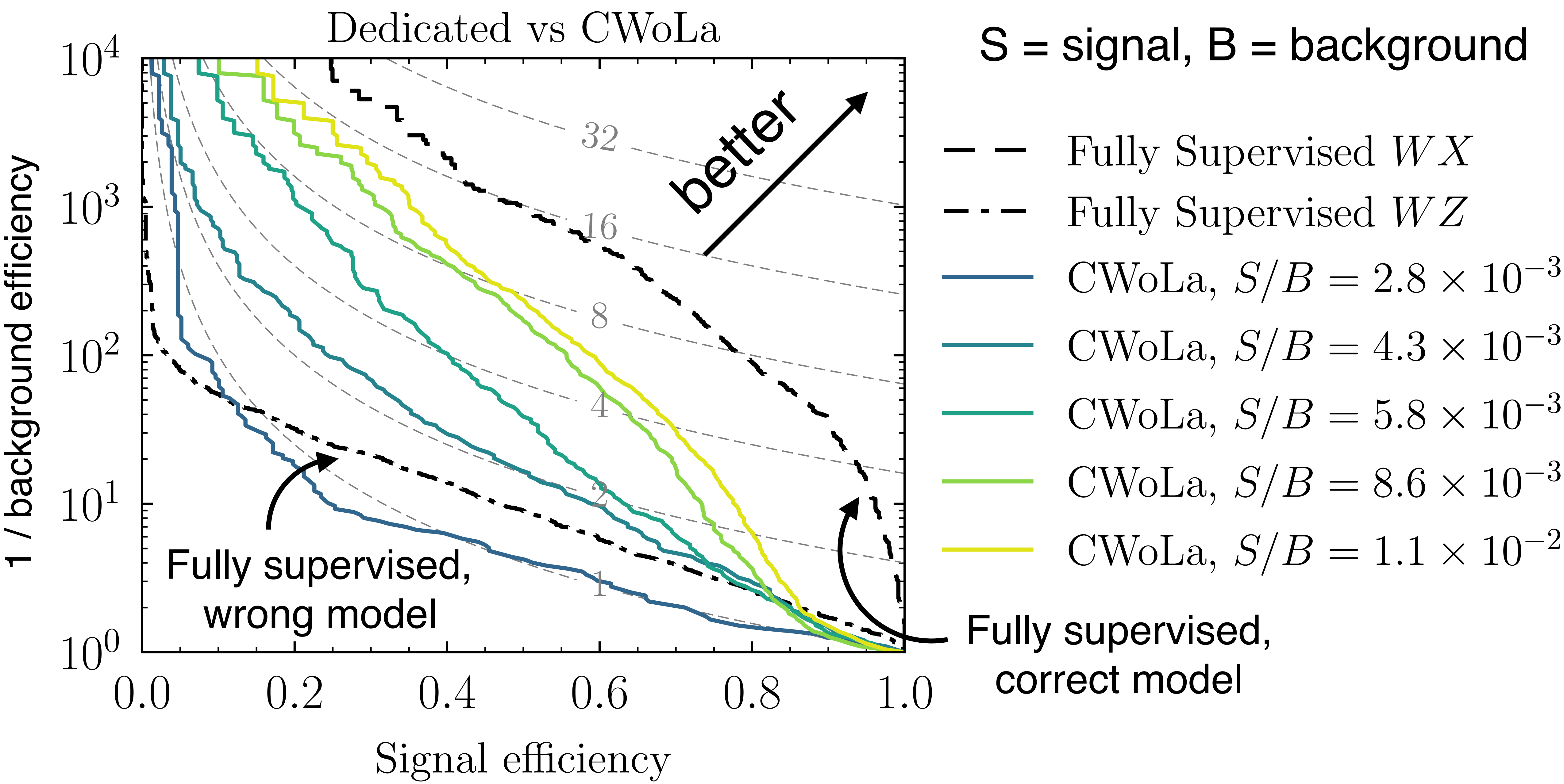
Weak/unsupervised learning for anomalies

J. Collins, K. Howe, **BPN**, 1805.02664



Using a classifier trained to distinguish a signal region from a sideband, make progressively harsher cuts on the NN output

CWoLa hunting vs. Full Supervision



If you know what you are looking for, you should look for it. If you don't know, then CWoLa hunting may be able to catch it!

Review of CWoLa hunting

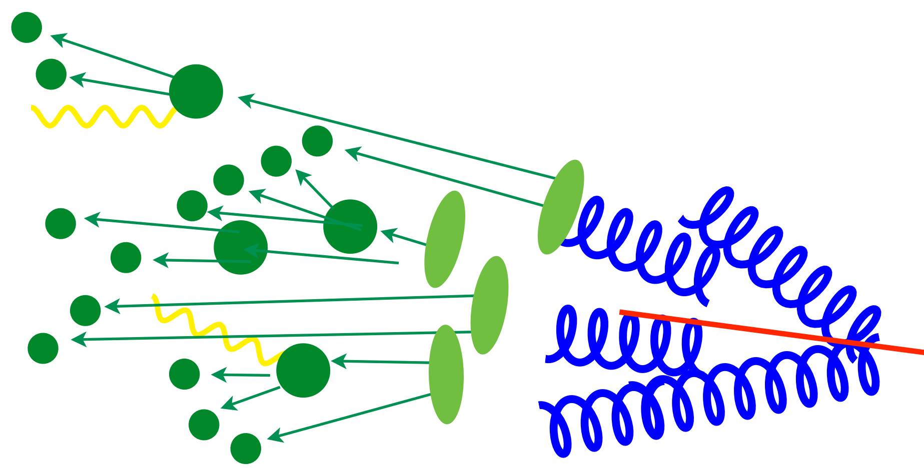
(1) Need an observable X (e.g. m_{JJ}) where the signal is localized and the background is not.

(2) Identify features Y (e.g. jet substructure) that are ~independent of X, but can be useful for identifying a broad range of new particles.

actually, we don't need independence, we just need them to not allow us to sculpt bumps.

Summary

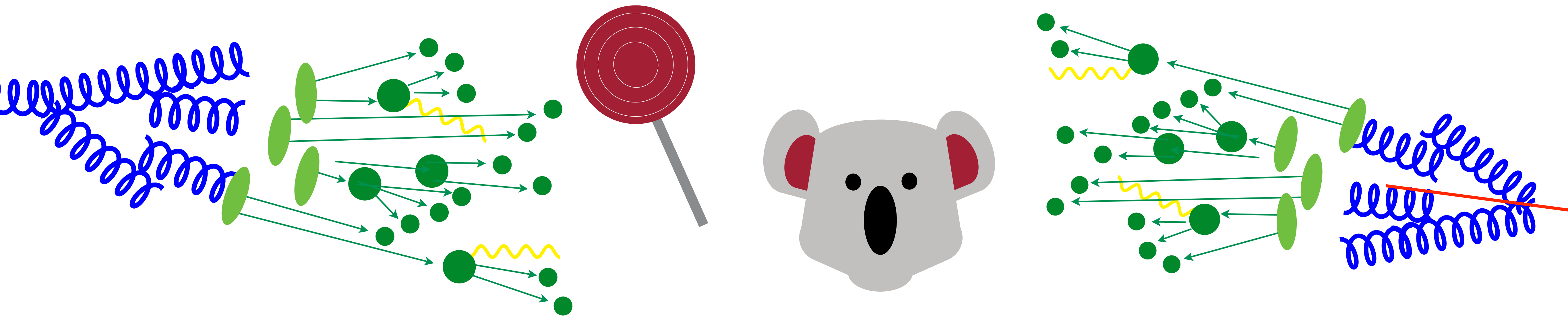
- Simulation dependence in traditional ML4HEP
- Classification
  - ✦ Adversarial approaches
  - ✦ Weak supervision
- Regression
- Anomaly Detection

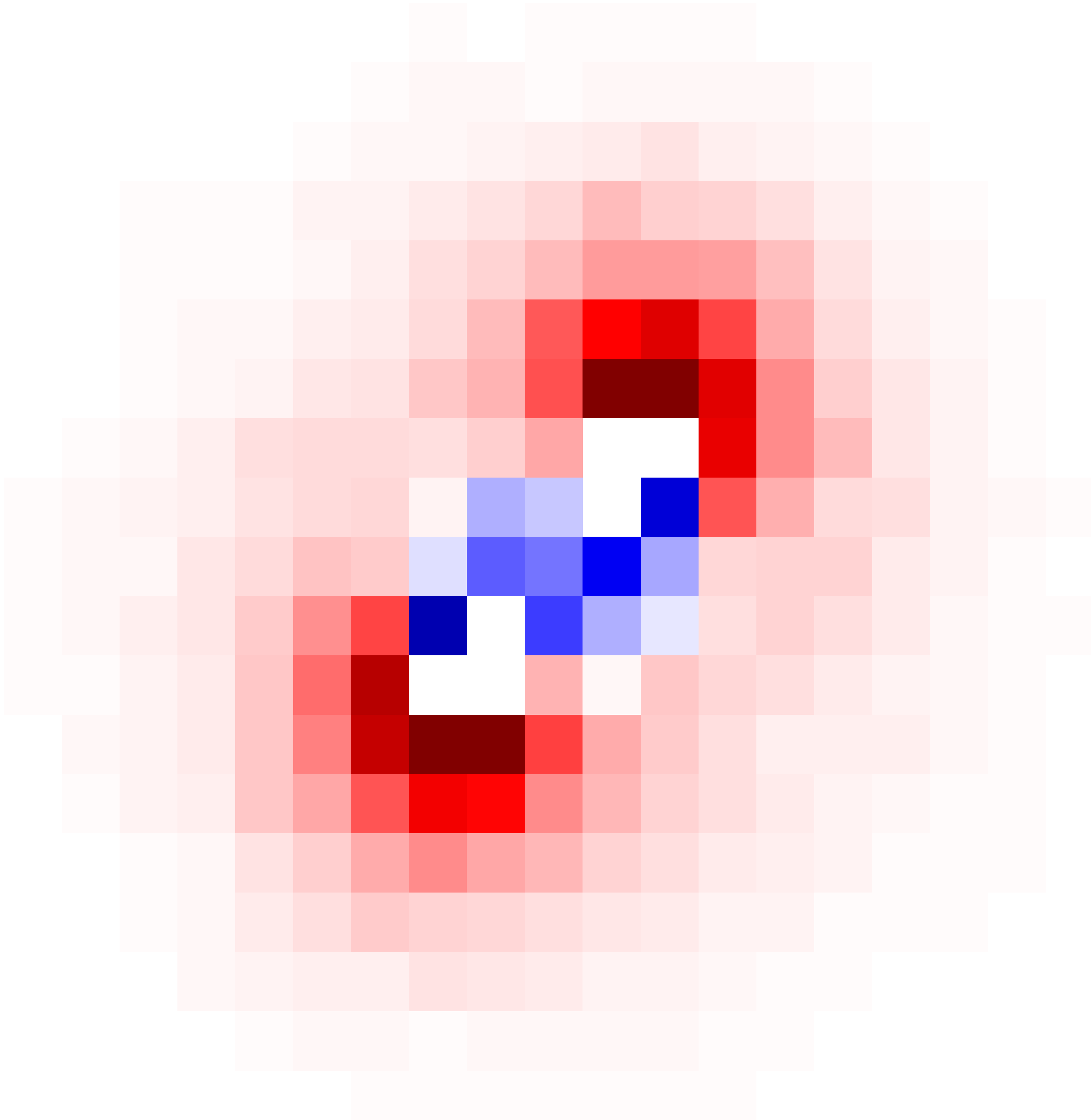


Conclusions and Outlook

Deep learning is a powerful tool for enhancing data analysis. However, it is crucial to know when and where we depend on prior knowledge.

Mitigating/reducing dependence on priors can improve performance and may even help us to understand something new and fundamental about nature!





Fin.