Robust Anomaly Detection with NuRD

Abhijith Gandrakota[1], Lily Zhang[2], Aahlad Puli[2], Nhan Tran[1], Jennifer Ngadiuba[1]

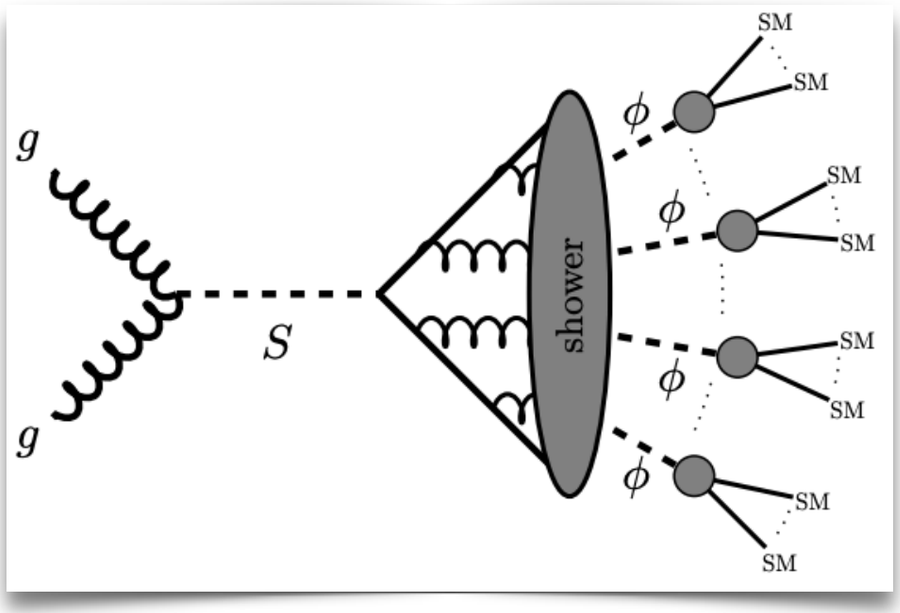
1: Fermilab
2: New York University

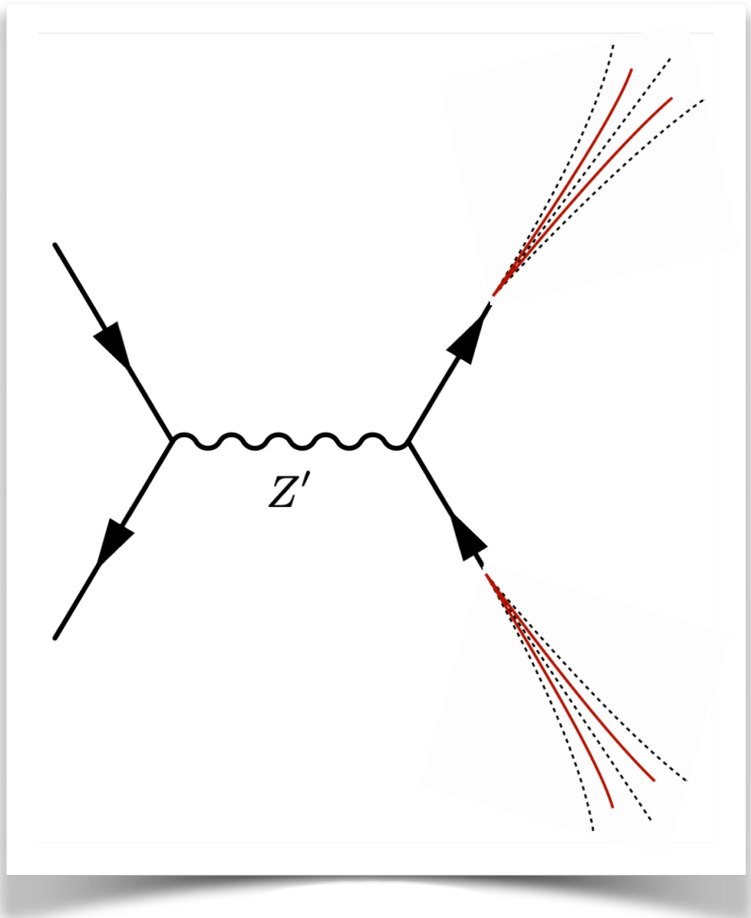
BOOST 2023, LBNL

Arxiv: 2308.SOON

Introductions

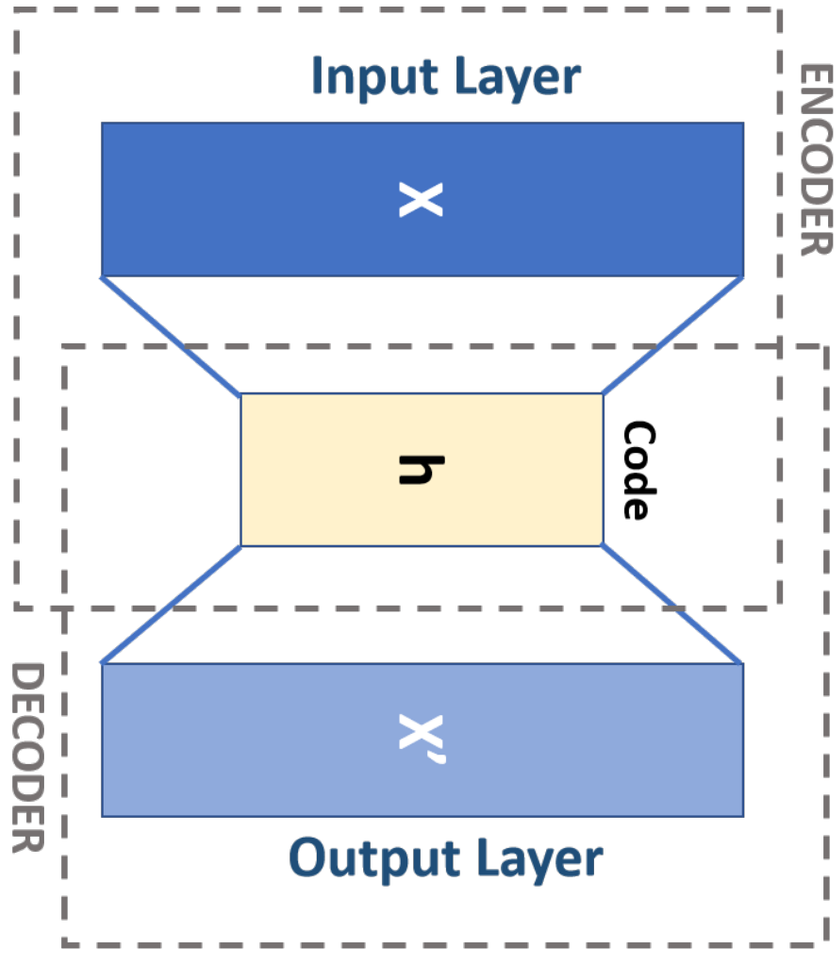
- Anomaly detection is a promising approach for detecting BSM physics without a BSM model prior
  - It is particularly helpful in detecting jets with *non-standard* substructure





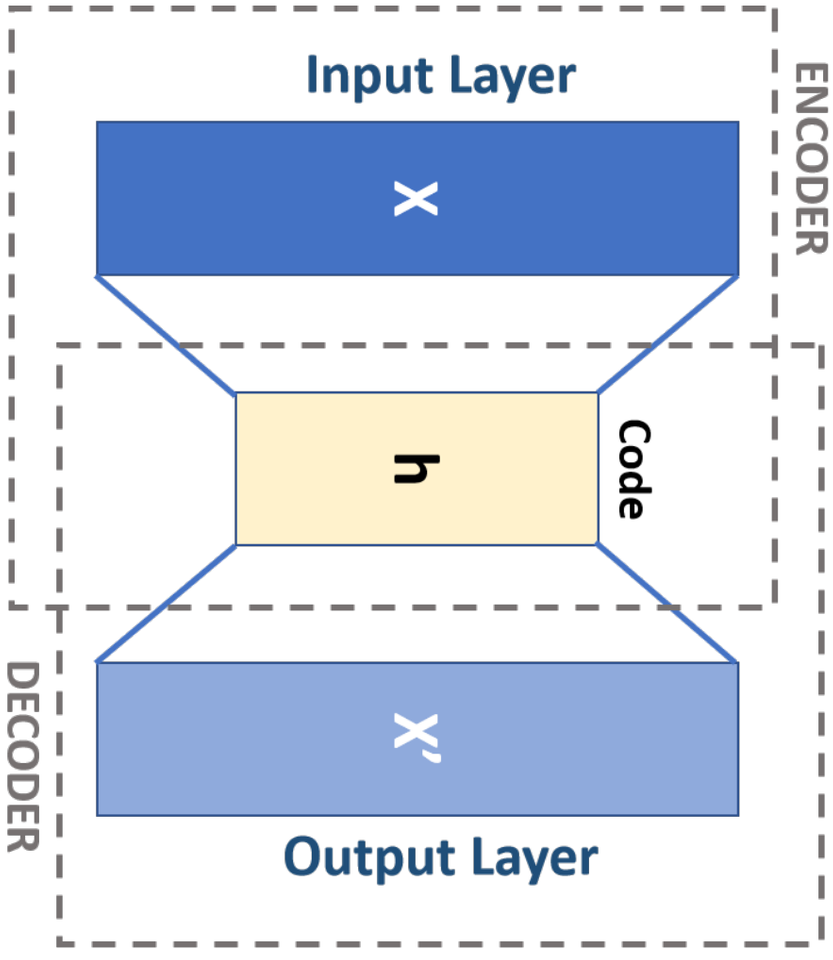
Introduction

- A standard approach for anomaly detection in High Energy Physics (@ LHC)
  - Look for "deviations" from expected (dominant) background physics
  - Encode the input into a smaller latent representation
  - Decode the representation back to initial input, examine reconstruction loss (~MSE)
  - Use this reconstruction loss to find anomalies



Introduction

- A standard approach for anomaly detection in High Energy Physics (@ LHC)
  - Look for "deviations" from expected (dominant) background physics
  - Encode the input into a smaller latent representation
  - Decode the representation back to initial input, examine reconstruction loss (~MSE)
  - Use this reconstruction loss to find anomalies
- Primary concerns
  - Is algorithm modeling the desired physics correctly?
    - Is it learning anything we don't want it focus on ?
    - AEs model everything, even the unimportant features
- Different take in approaching this challenge using NuRD



Robust anomaly detection

- *More importantly, is it learning anything we don't want it to know ?*

- Objective: Detect animal other than cow

Our Training data:

Cows in a typical
Grass background

Robust anomaly detection

- *More importantly, is it learning anything we don't want it to know ?*

- Objective: Distinguish between the animals ?

Our Training data:

Cows in a grassland backdrop

Sure, we may detect penguins in show
Expected anomaly

Robust anomaly detection

- *More importantly, is it learning anything we don't want it to know ?*

- Objective: Distinguish between the animals ?

Our Training data:

Cows in a grassland backdrop

Sure, we may detect penguins in show
Expected anomaly

This ?
Actual Anomaly

Robust anomaly detection

- *More importantly, is it learning anything we don't want it to know ?*

- Objective: Detect animal other than cow

Our Training data:

Cows in a grassland backdrop

Sure, we may detect penguins in snow
Expected anomaly

This ?
Actual Anomaly

How about this ?
Atypical BKG in data

Robust anomaly detection

- *More importantly, is it learning anything we don't want it to know ?*

- Objective: Detect animal other than cow

Our Training data:

Cows in a grassland backdrop

Needs to learn this !

What if it learnt this ?

Sure, we may detect penguins in show
Expected anomaly

This ?
Actual Anomaly

How about this ?
Typical BKG in data

Robust anomaly detection

- *In the case of Anomaly detection on jets*

- Objective: Detect ~~animal~~ Jets other than ~~cow~~ SM

Jets in a grassland backdrop

Our Training data:

Substructure, etc . . . !

Mass, P_T etc . . .

From inputs to representations

- Issue : Density estimation on the inputs typically models ***everything*** about the data (e.g: Autoencoders)

  - We want to model semantic features (*like jet structure*), while being decorrelated with nuisances (*like mass, etc* . . .)

From inputs to representations

- Issue : Density estimation on the inputs typically models ***everything*** about the data (e.g: Autoencoders)

  - We want to model semantic features (*like jet structure*), while being decorrelated with nuisances (*like mass, etc* . . .)

- Idea: Use different backgrounds to learn what is *important*

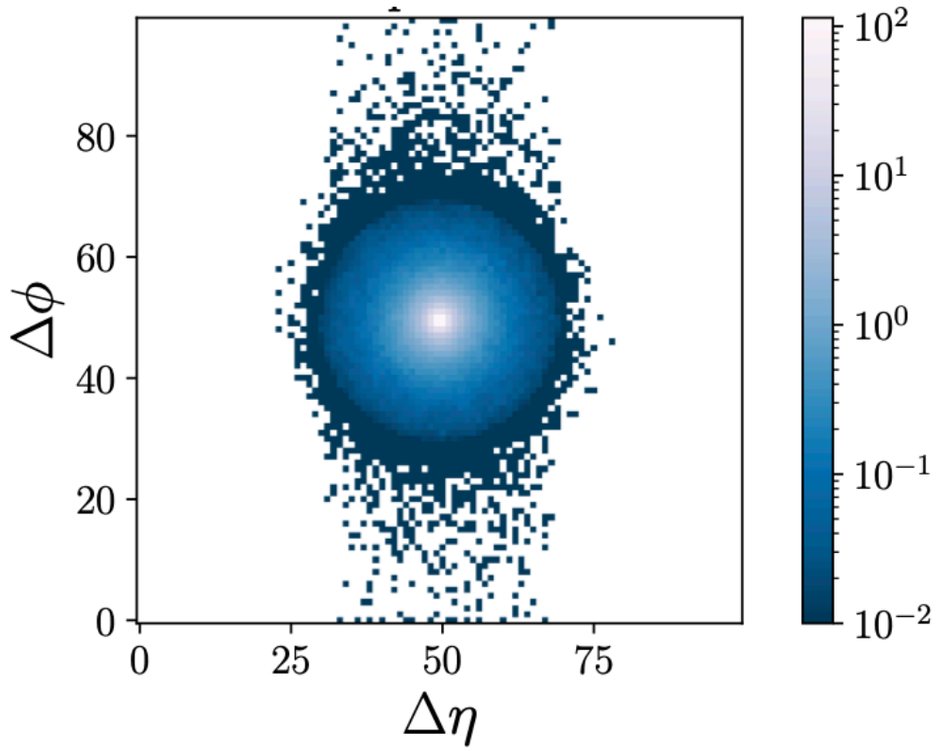
From inputs to representations

- Issue : Density estimation on the inputs typically models ***everything*** about the data (e.g: Autoencoders)
  - We want to model semantic features (*like jet structure*), while being decorrelated with nuisances (*like mass, etc* . . .)
- Idea: Use different backgrounds to learn what is *important*
- Solution: Use multiple known background labels (not just QCD)
  - Avenue to learn what's important [~ minimal hand holding]
  - Build representations to have maximum information with the labels
  - Ensure representations do not vary w/ nuisances (Zhang et al. 2022, Puli et al. 2022).
  - This way, we can maximize only the relevant information

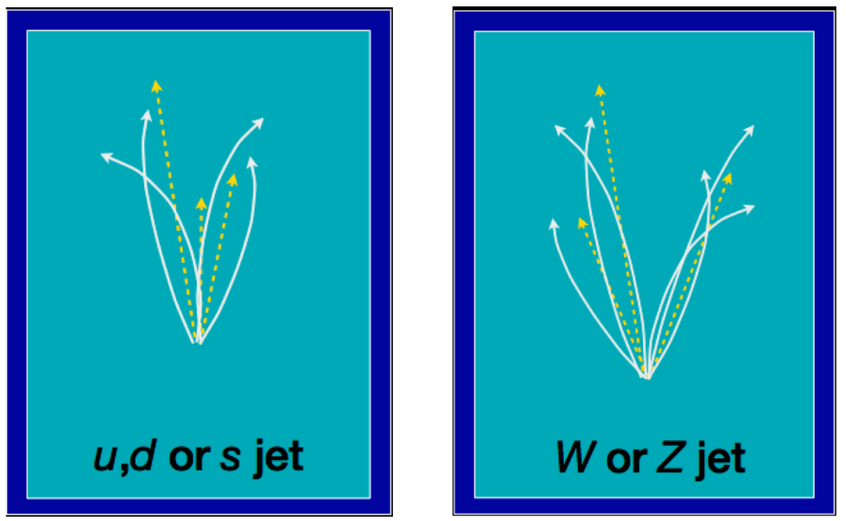
Input Dataset

- For out dataset we have input features (X), labels for BKG types (Y), and Nuisance (Z)
- Objective is to learn particles decays at LHC, specifically hadronic jet shower

  - Input: Energy deposits in the detectors
    - Images ~ 50 X 50 pixels
    - Images normalized individually



  - We have two background samples to learn semantics
    - We use QCD and W/Z jets w/ labels



  - We want the our representation to capture physics and not depend on the nuisance

[1] JEDI-Net, Eric A. Moreno et al

Nuisance Randomized Distillation

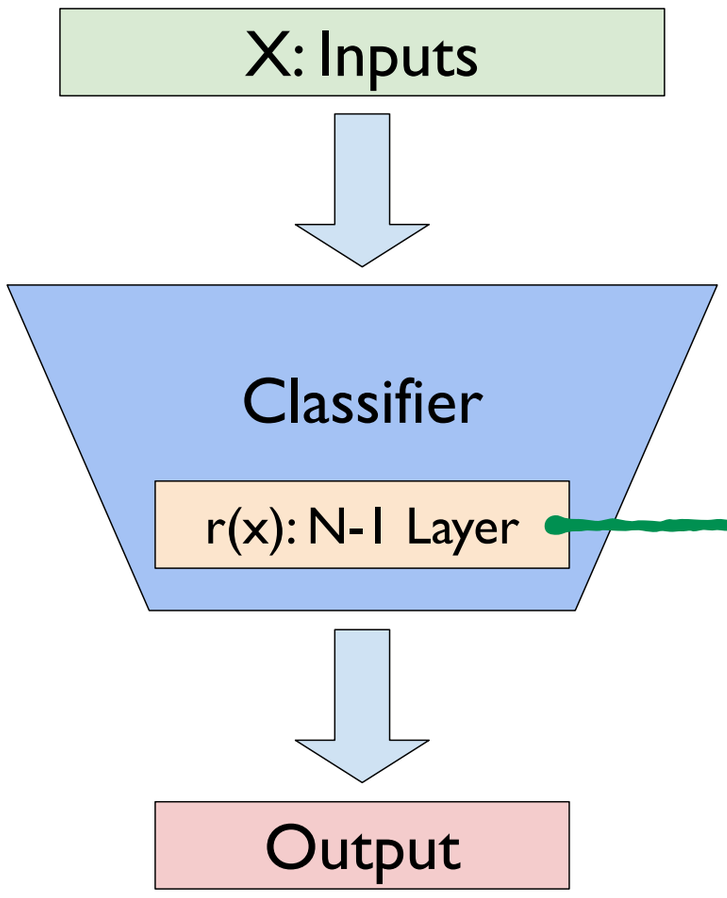
- For out dataset we have input features (X), labels for BKG types (Y), and Nuisance (Z)
- **Nu**isance **R**andomized **D**istillation:
  - I : Do not let model learn nuisance: break the dependence b/n label and nuisance.
    - Use importance weights w to break dependence.
  - II : Build informative representations that do not vary with the nuisance
    - Intuitively, it shouldn't be possible to distinguish b/n [Joint independence]
      - (r_X, Y, Z) vs (r_X, Y, randomized nuisance($\hat{Z}$))
      - If representations contain info about nuisance, penalize the mutual information
- Use the representations to detect anomalies.

[1] Puli et al. 2022

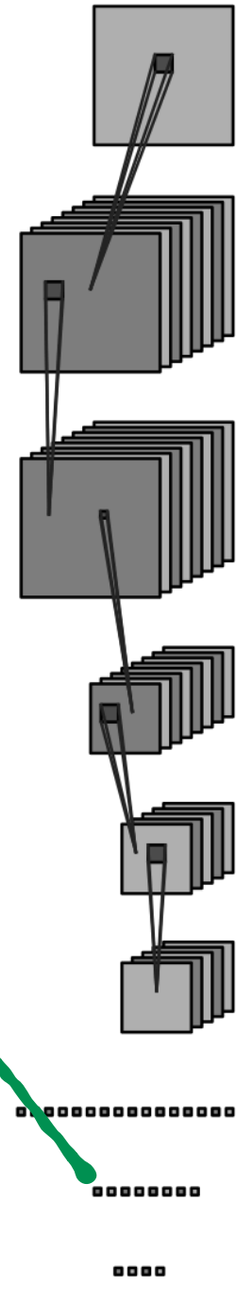
Nuisance Randomized Distillation

- Building out representation:
  - Start with a simple classifier b/n different background process
  - CNNs w/ final dense layers output to logits / softmax probabilities

    (Similar to the CNN Encoder architecture used in QCD AE)

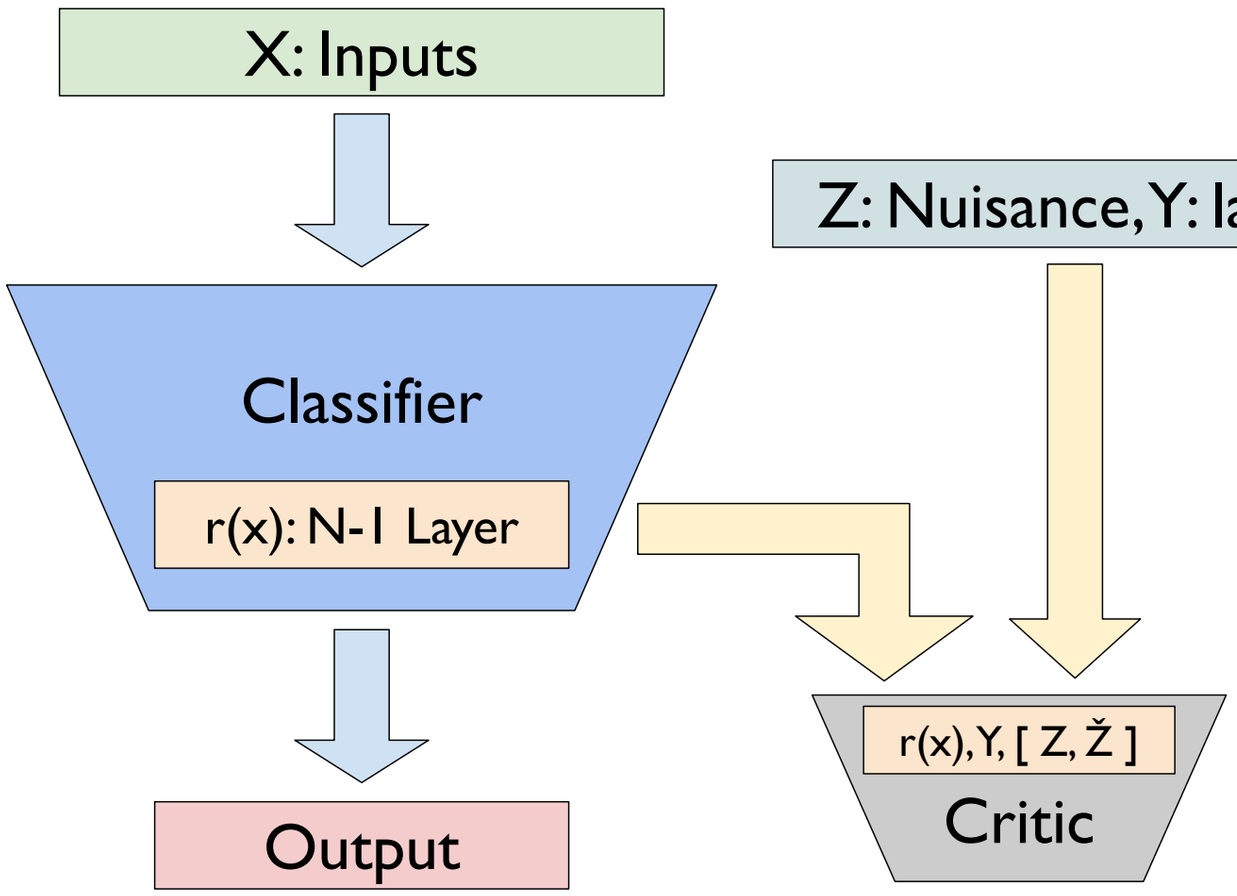


Representation (r_X)



Nuisance Randomized Distillation

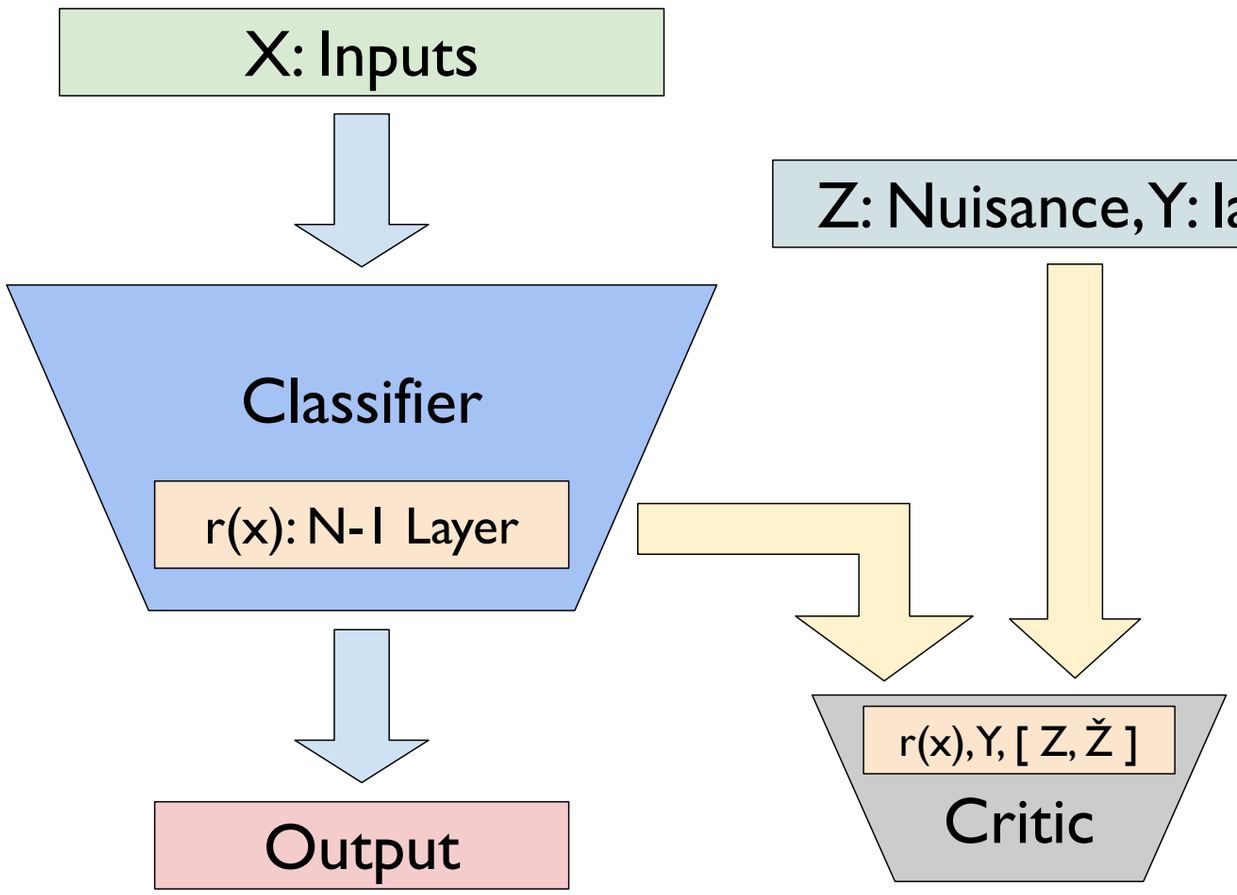
- Penalize mutual information
  - Input $(r_X, Y, [Z, \hat{Z}])$ to critic model (ϕ), a simple MLP
    - Approximates the mutual information, use this to penalize the loss



- Critic is trained to differentiate (r_X, Y, Z) vs $(r_X, Y, \hat{Z})$
- Critic model is updated for every batch of the classifier training
- It is proxy as the likelihood approximator

Nuisance Randomized Distillation

- Training
  - Train and update critic model for every batch of classifier training

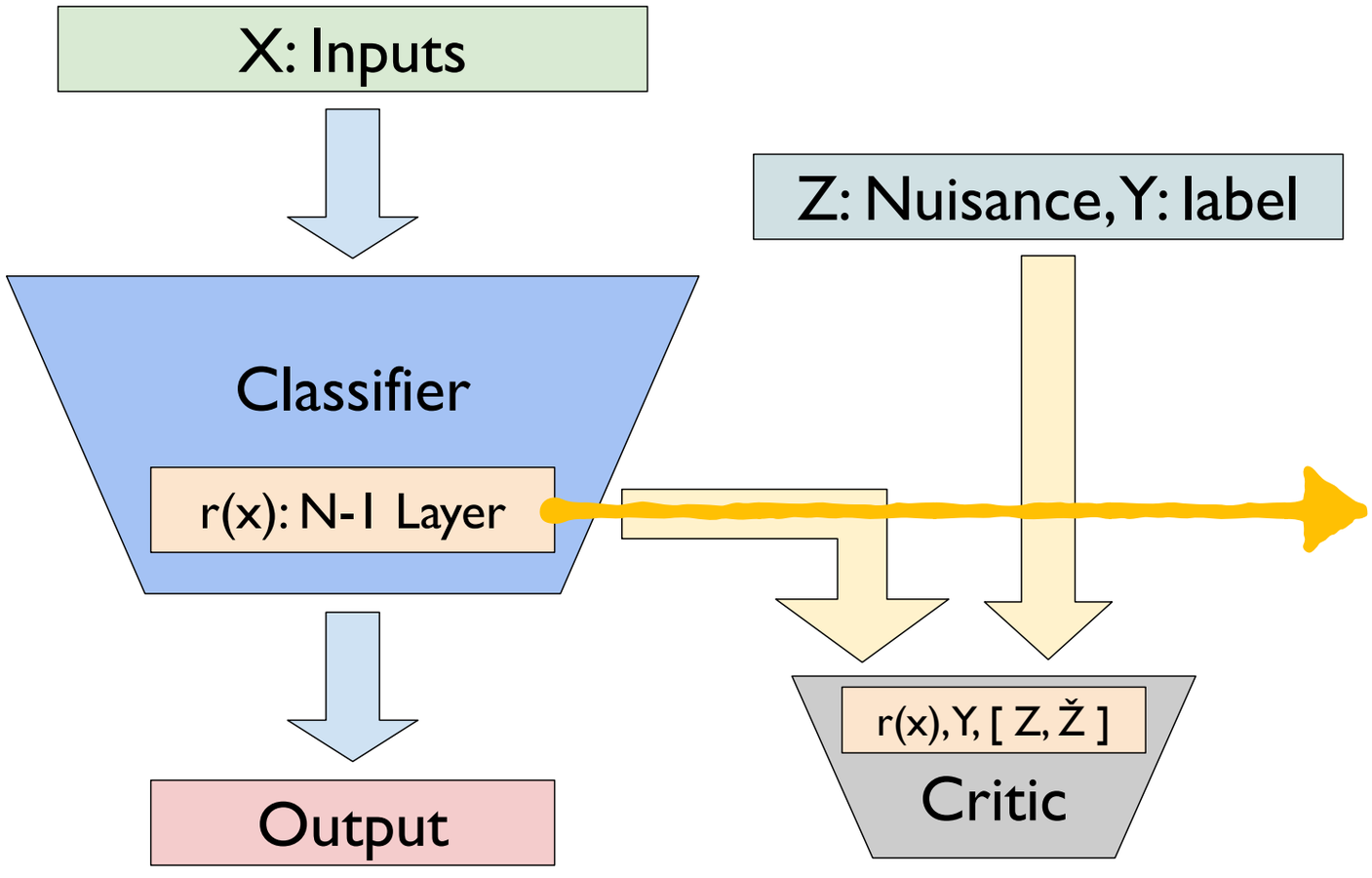


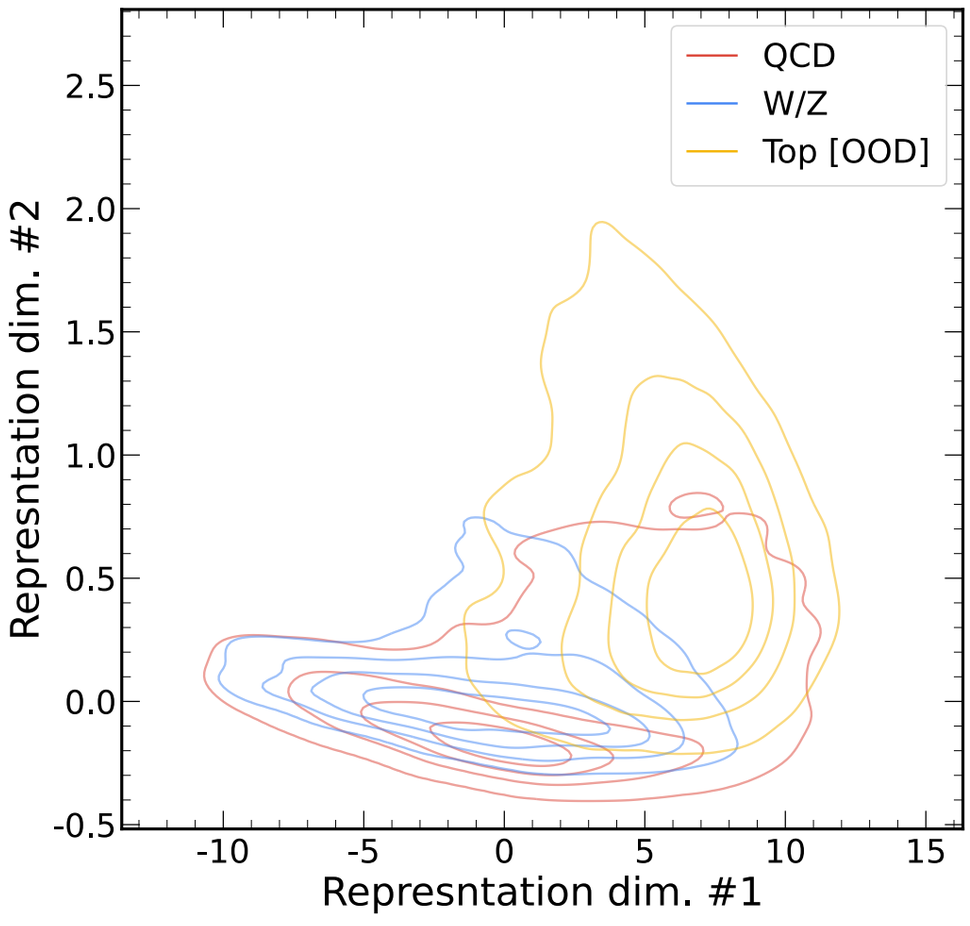
$$\mathscr{L} = w\left(CE(Y_{pred}, Y_{true}) - \lambda \log \frac{p_\phi(r_X, Y, [Z, \hat{Z}])}{1 - p_\phi} \right)$$

- Critic is trained to differentiate (r_X, Y, Z) vs $(r_X, Y, \hat{Z})$
- Critic model is updated for every batch of the classifier training
- It is proxy as the likelihood approximator

Nuisance Randomized Distillation

- Training
  - Train and update critic model for every batch of classifier training

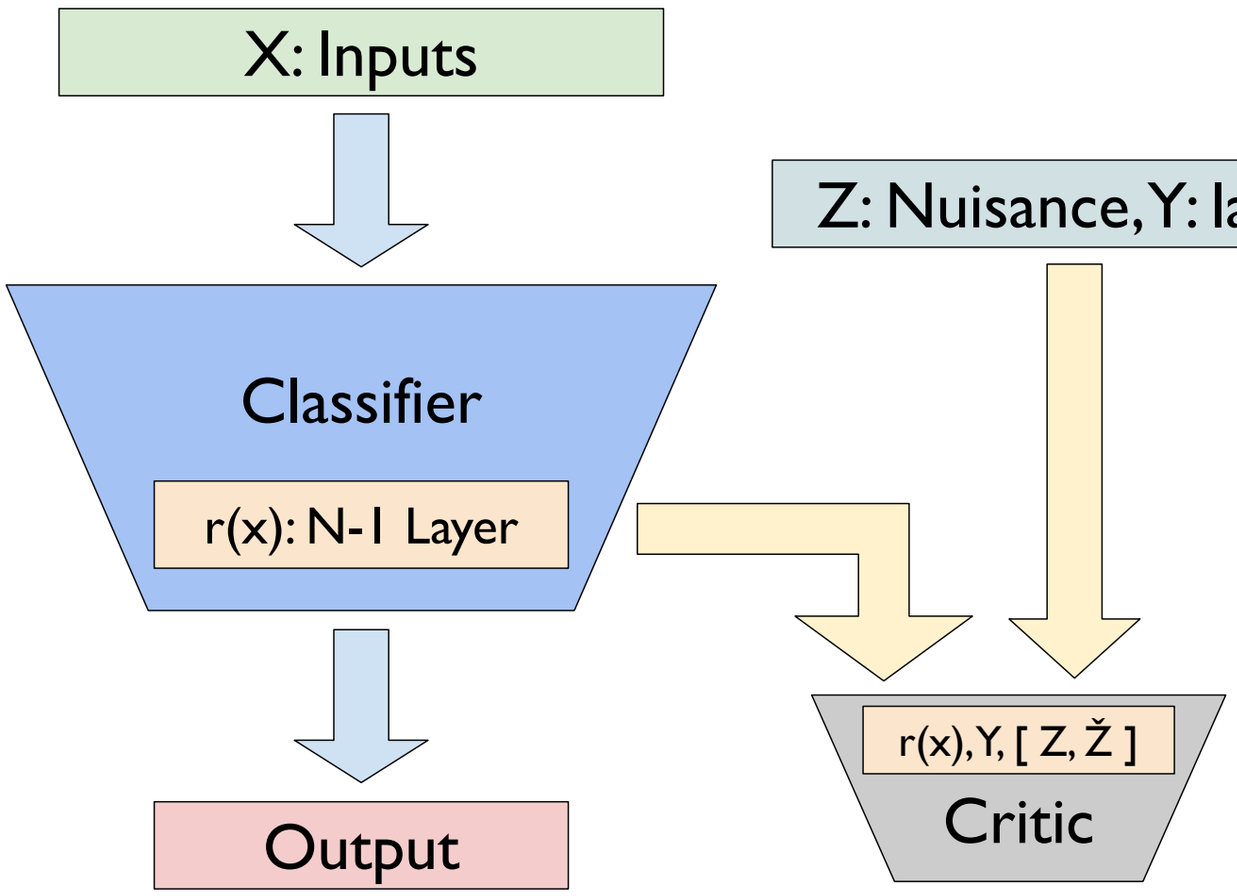




$$\mathscr{L} = w\left(CE(Y_{pred}, Y_{true}) - \lambda \log \frac{p_\phi(r_X, Y, [Z, \hat{Z}])}{1 - p_\phi} \right)$$

Nuisance Randomized Distillation

- OOD Detection:
  - Outlier Dataset: Top quarks jets
  - Use representations to build anomaly metrics



$$\mathscr{L} = w\left(CE(Y_{pred}, Y_{true}) - \lambda \log \frac{p_\phi(r_X, Y, [Z, \hat{Z}])}{1 - p_\phi} \right)$$

- Metrics:
  - Calculate the distance from samples in representation space

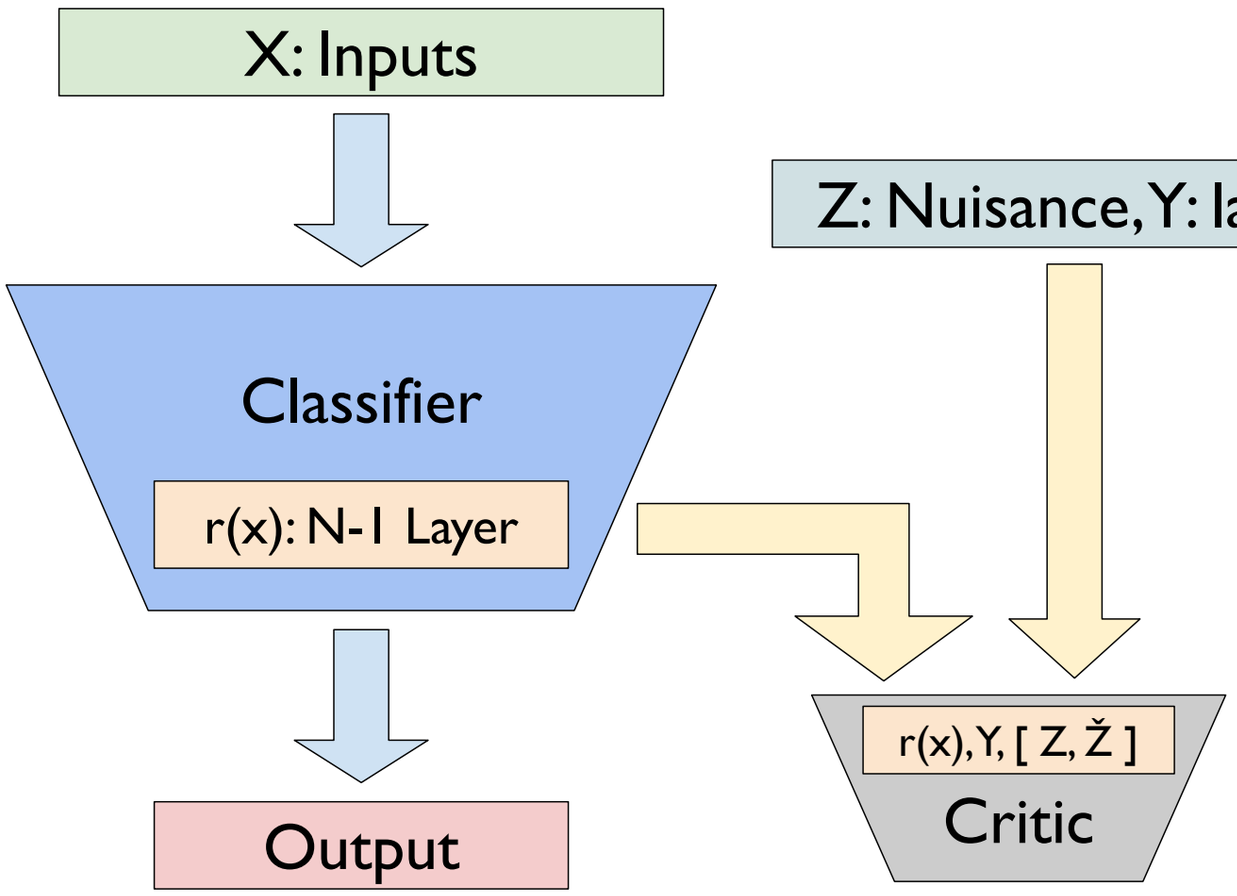
$$d_A = (r_X - \mu_A)\ \Sigma_A^{-1}\ (r_X - \mu_A)^T$$

(dist. from BKG A)

- Obtain distance from all BKG samples
  - Here: $[d_{QCD}, d_{WZ}]$
- Use this to find anomalies

Nuisance Randomized Distillation

- OOD Detection:
  - Outlier Dataset: Top quarks jets
  - Use representations to build anomaly metrics

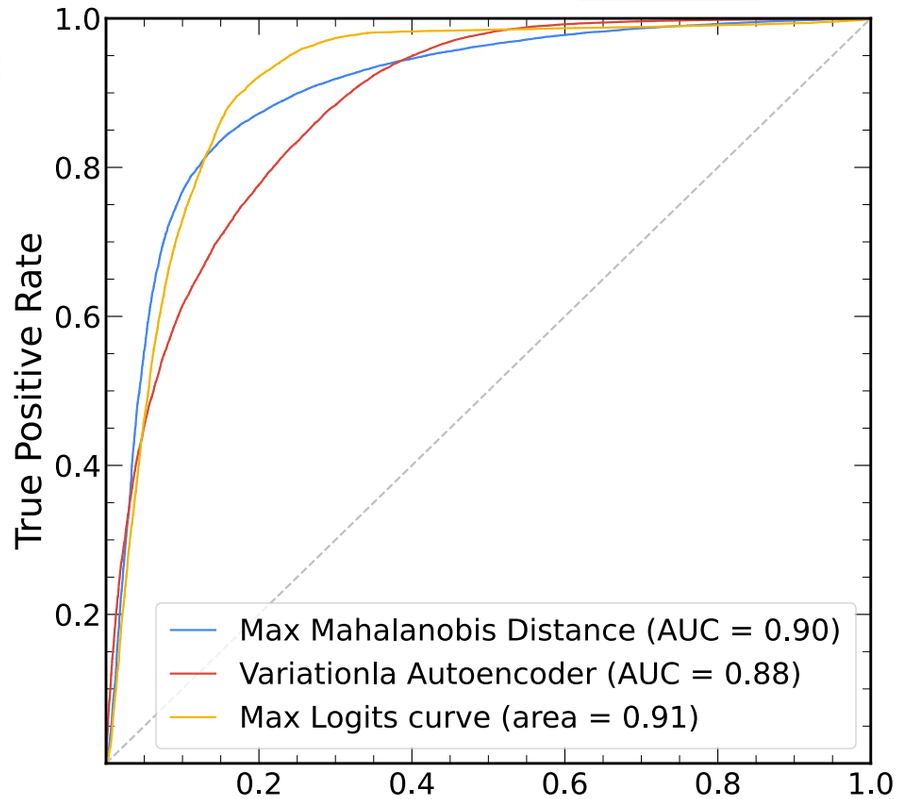


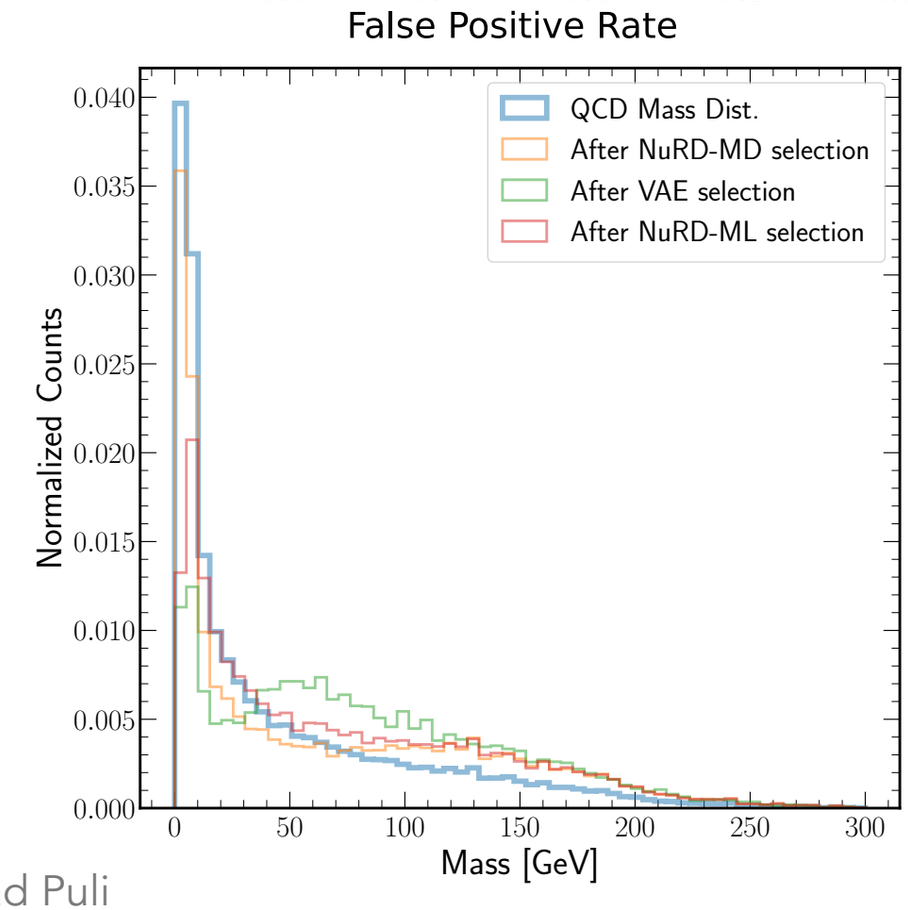
$$\mathscr{L} = w\left(CE(Y_{pred}, Y_{true}) - \lambda \log \frac{p_\phi(r_X, Y, [Z, \hat{Z}])}{1 - p_\phi} \right)$$

- Metrics:
  - Obtain distance d_A from all BKG samples
    - Here: $[d_{QCD}, d_{WZ}]$
- Alternative Metrics:
  - Max(Logits) also serves as a OOD Score
  - Max $_{\text{Logits}}$ (OOD) < Max $_{\text{Logits}}$ (BKG)

Experiments and Results

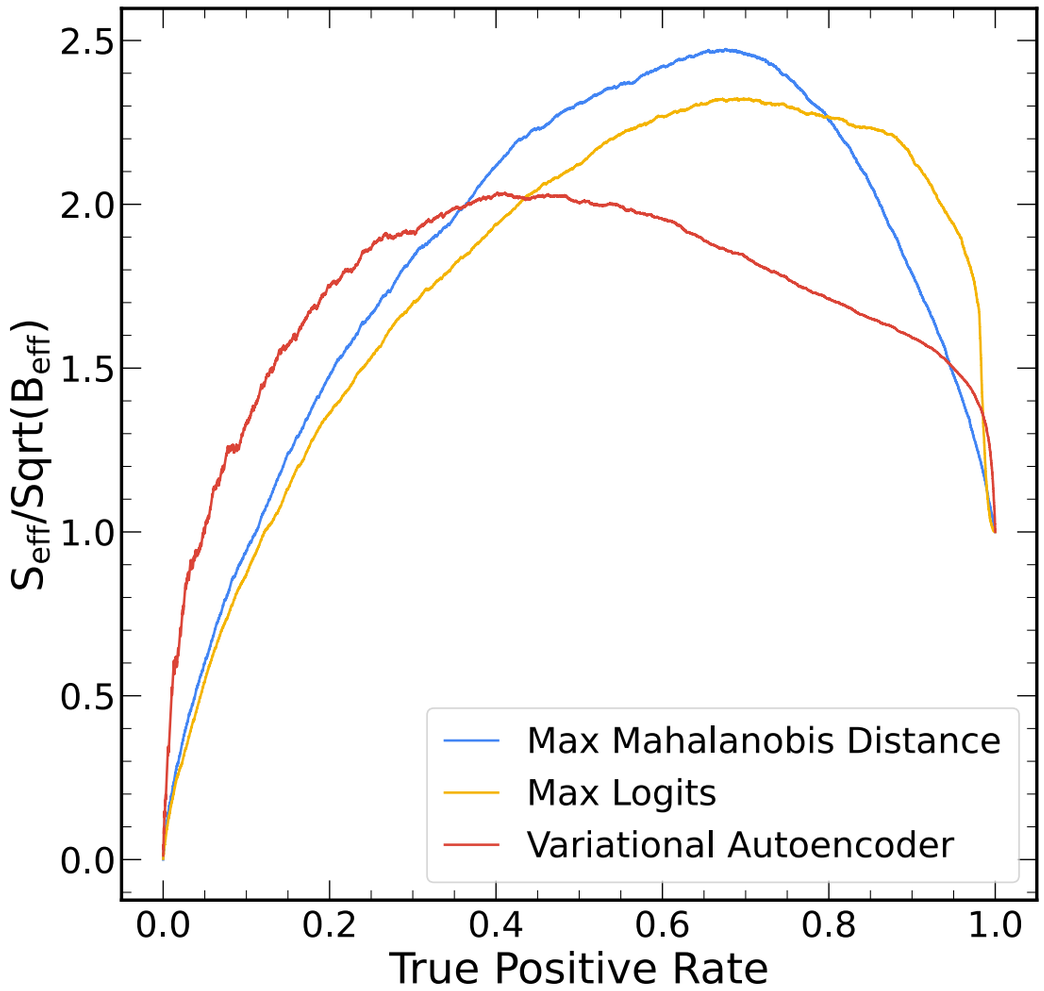
- Trained on QCD and W/Z labeled data to build out the representation space
  - Representation space is has a dimension of 20
  - The critic model :3 layers w/ 256, 128, 68 neurons
- Examined OOD performance w/ two metrics
  - AUC w/ Mahalnobis distance: 0.90
  - AUC w/ Max(Logits) score: 0.91
    - (Baseline: AUC w/ plain AE : 0.88)
- Representation w/ Joint independence gives us robustness:
  - Performance guarantees across different BKG-distributions





Results

- Obtained representations denotes the diversity of what is *typical*
  - While keeping relevant info for anomaly detection
  - Achieves this while staying decorrelated with kinematics of the jet
- Classifier in NuRD is built on the "Encoder" block on baseline VAE
  - Resulting AD model is lighter and faster
  - Technique can be adapted to any network architecture



Method	AUC ↑	JSD ↓	Sig. Imp. ↑
VAE	0.88	0.065	2.03
NuRD-MD	0.90	**0.013**	**2.47**
NuRD-ML	**0.91**	0.027	2.32

Summary

- In HEP (often many other fields) we have multiple backgrounds. We should use information contained in all of them.

- This is a new take on building a representation space to detect anomalies:

    - Training w/ background labels gives us good performance.

    - NuRD, via joint independence, helps

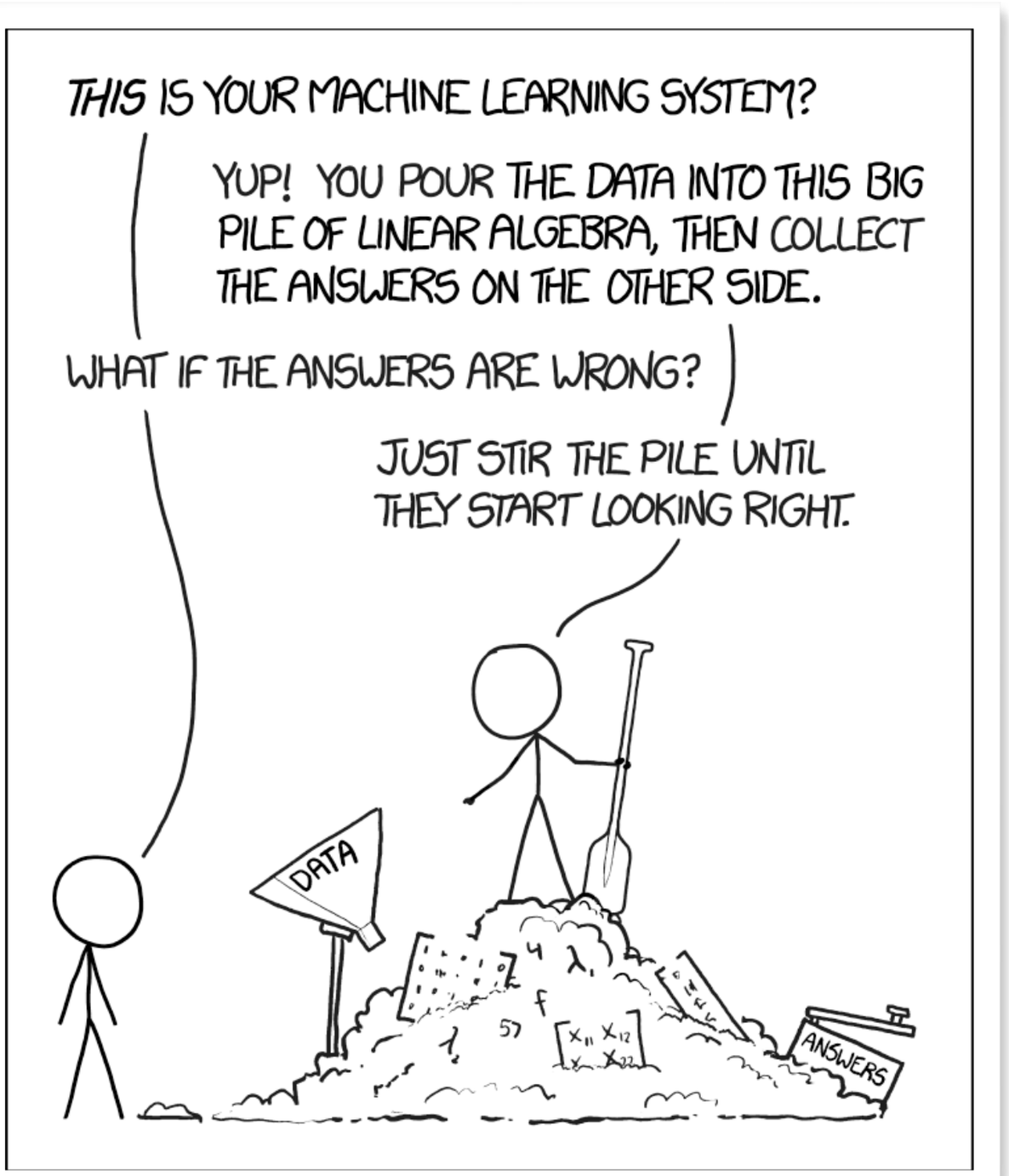
        - Maximize physics learnt while decorrelating nuisances

- This technique although takes longer to train, results in smaller models

    - A primary benefit of increased robustness.

- Paper will be out on Arxiv soon (w/ code)

Thank you



https://xkcd.com/2451/