Anomaly Detection Using Machine Learning at Belle II

David Giesegh, Nikolai Hartmann, Thomas Kuhr

LMU München

Joint Particle Physics Group Seminar
26.06.2024

Motivation

- Searches for New Physics typically motivated by specific models
- What if we are looking in the wrong places?

→ Need for generic, model agnostic search methods

Motivation

- Searches for New Physics typically motivated by specific models
- What if we are looking in the wrong places?

→ Need for generic, model agnostic search methods

- Different approaches:

Supervised / simulation driven:
Generic comparisons of measurements with theory predictions

Unsupervised / data driven:
Direct searches for anomalous/over-dense regions in the data

Motivation

- Searches for New Physics typically motivated by specific models
- What if we are looking in the wrong places?

→ Need for generic, model agnostic search methods

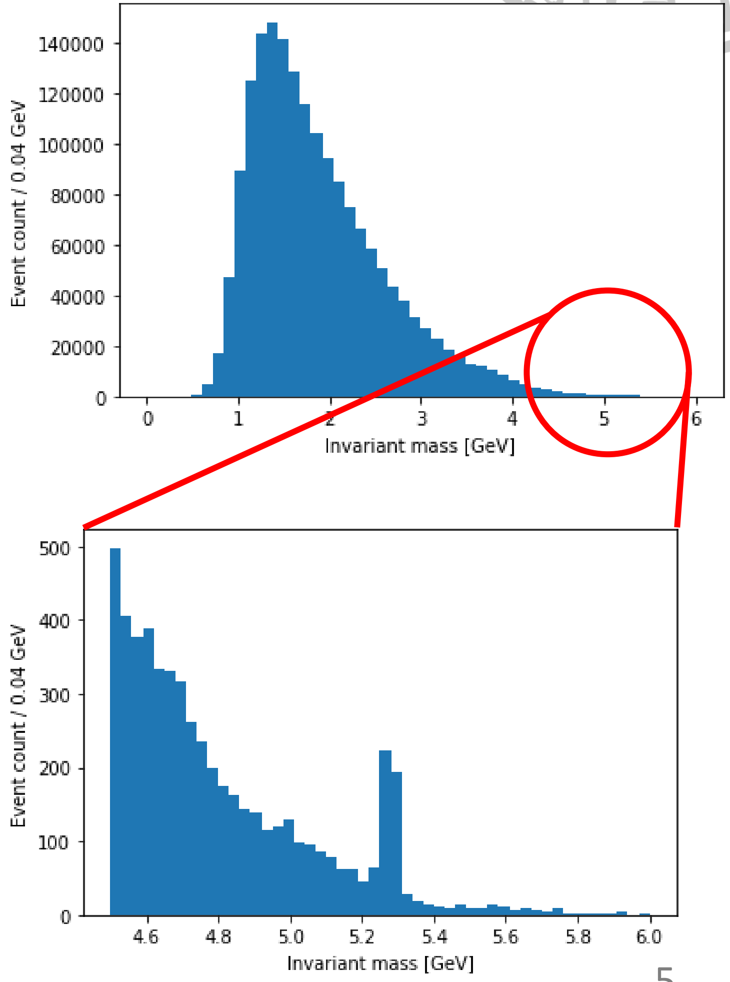
- Different approaches:

Supervised / simulation driven:
Generic comparisons of measurements with theory predictions

Unsupervised / data driven:
Direct searches for anomalous/over-dense regions in the data

Principles of Anomaly Detection

- Given a (high dimensional) dataset X, determine the datapoints that don’t seem to follow the general distribution of X
- Typical approach: assign numeric **anomaly score** to each datapoint (like classification score)
- No labels: well suited task for **unsupervised machine learning**
- Various methods:
    - Compression algorithms (**Autoencoders**)
    - Density estimation methods (**CATHODE**)



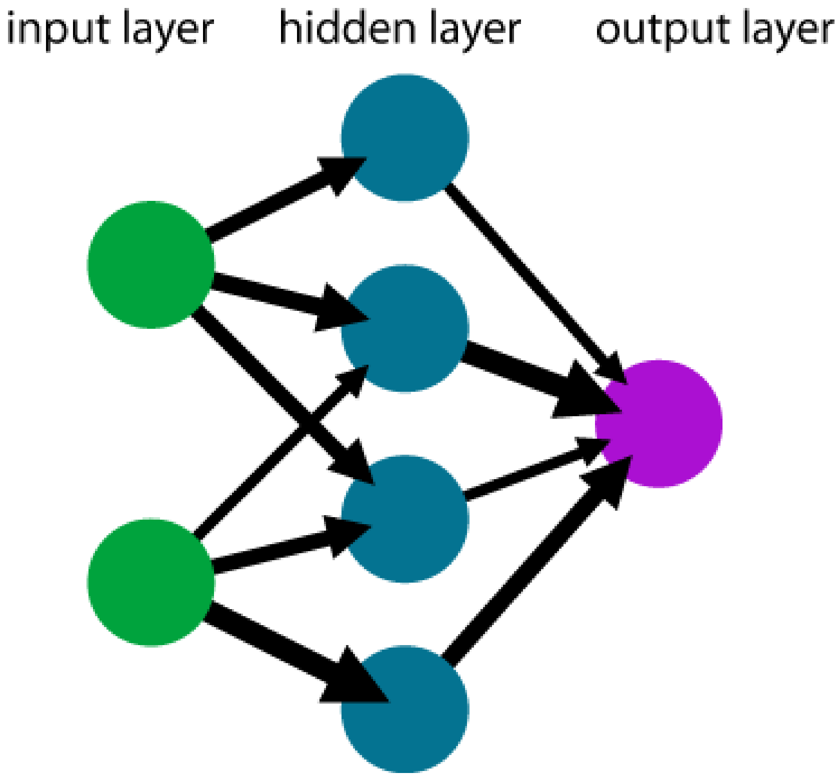
Intermezzo: Unsupervised Machine Learning

(Variational) Autoencoders and Normalizing Flows

Recap: Deep Learning

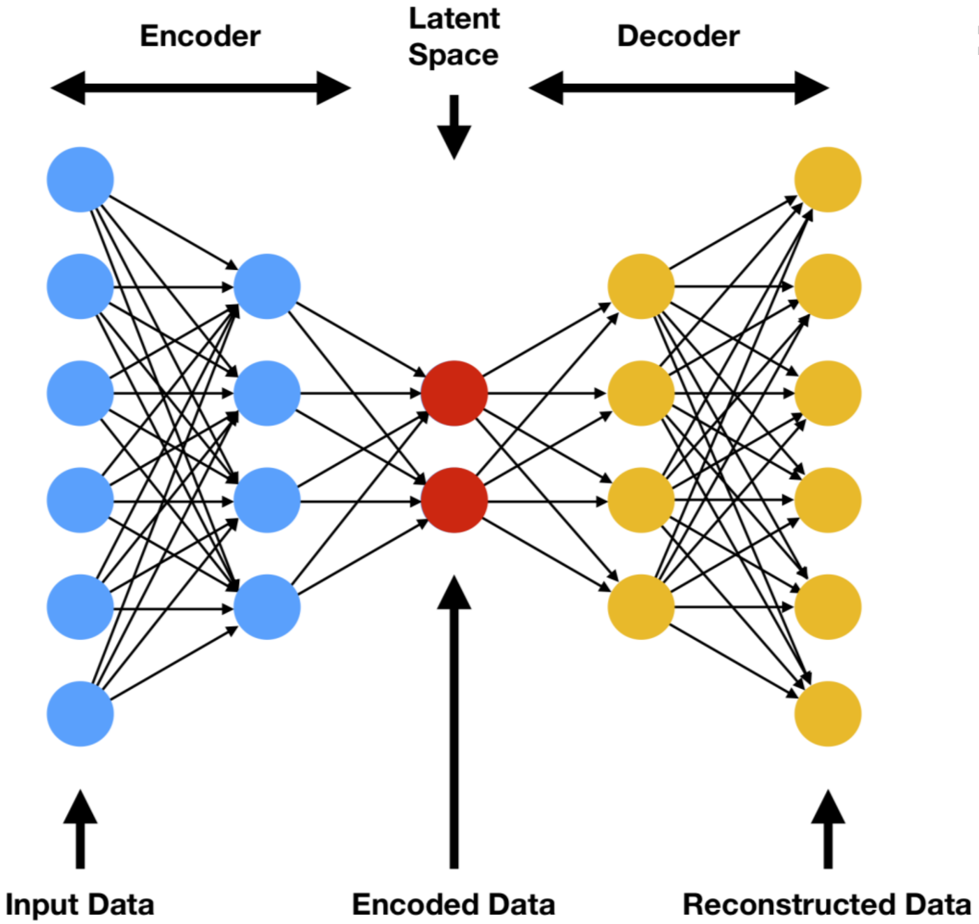
(-> See talk by Nikolai on 24.04.)

- Neural Network: Series of **nodes** arranged in **layers**
- Node = linear transformation plus
  non-linear activation
- Training: updating **weights** by minimizing a **loss function** through **backpropagation** (i.e. chain rule)
- Different architectures and loss functions for different tasks



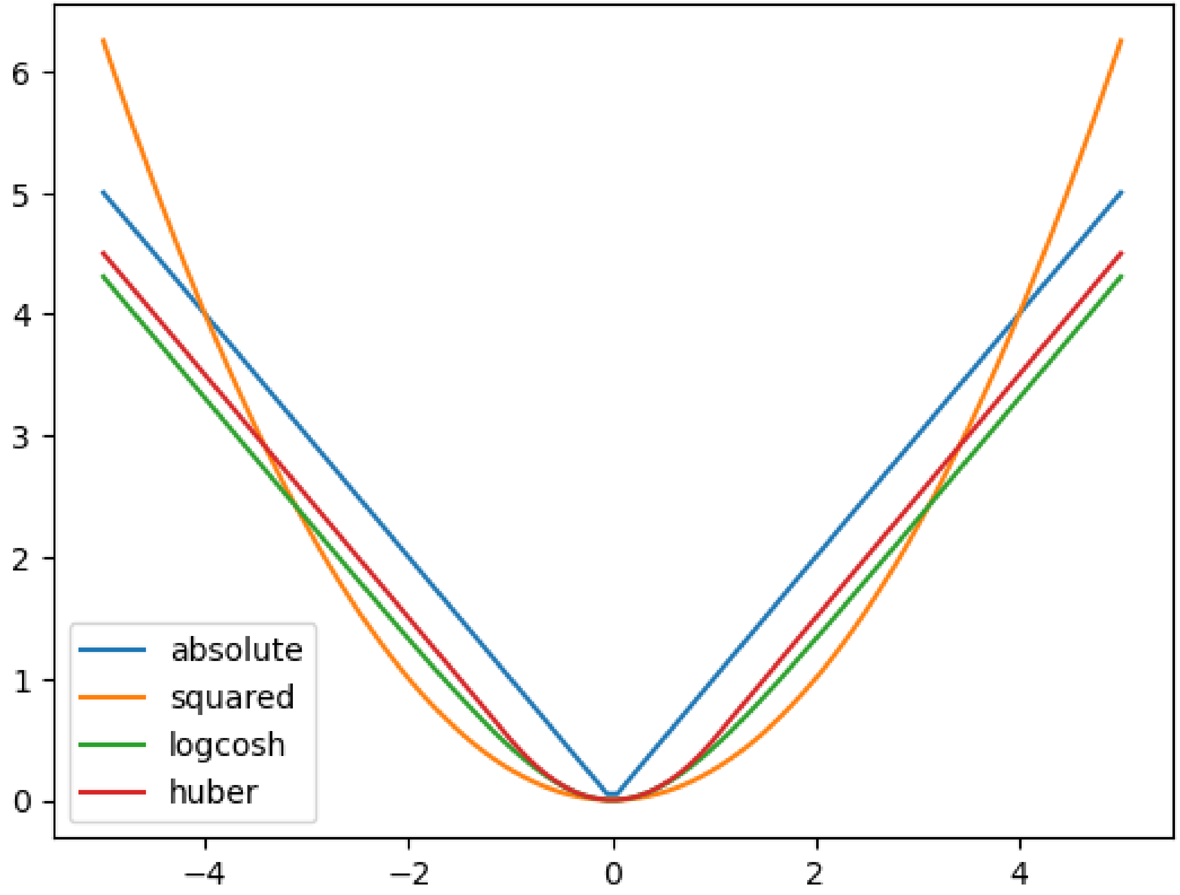
Autoencoders

- Same-size input and output layer
- Twist: make middle layer smaller than input layer (**latent space**)
- Typical loss: mean squared error between input and output (also mae, huber, logcosh, …)



Autoencoders

- Same-size input and output layer
- Twist: make middle layer smaller than input layer (**latent space**)
- Typical loss: mean squared error between input and output (also mae, huber, logcosh, …)

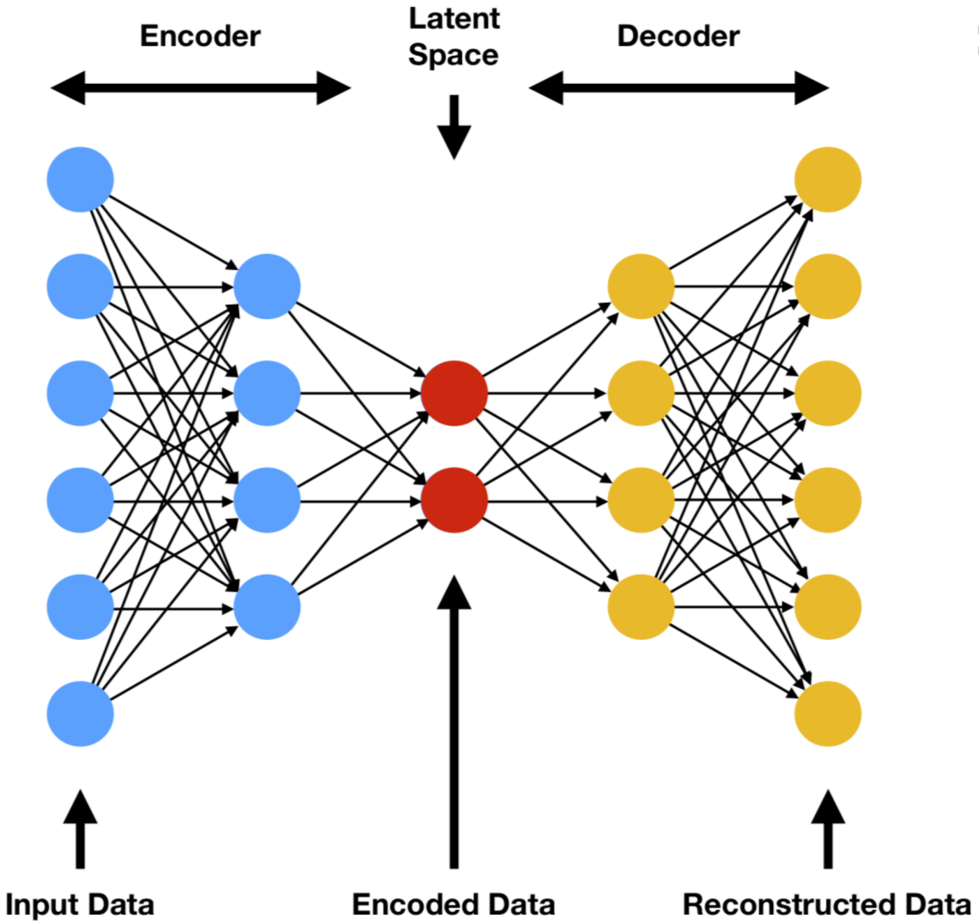


Autoencoders

- Same-size input and output layer
- Twist: make middle layer smaller than input layer (**latent space**)
- Typical loss: mean squared error between input and output (also mae, huber, logcosh, …)

→ Model learns essential features of dataset

→ However: reconstruction never perfect

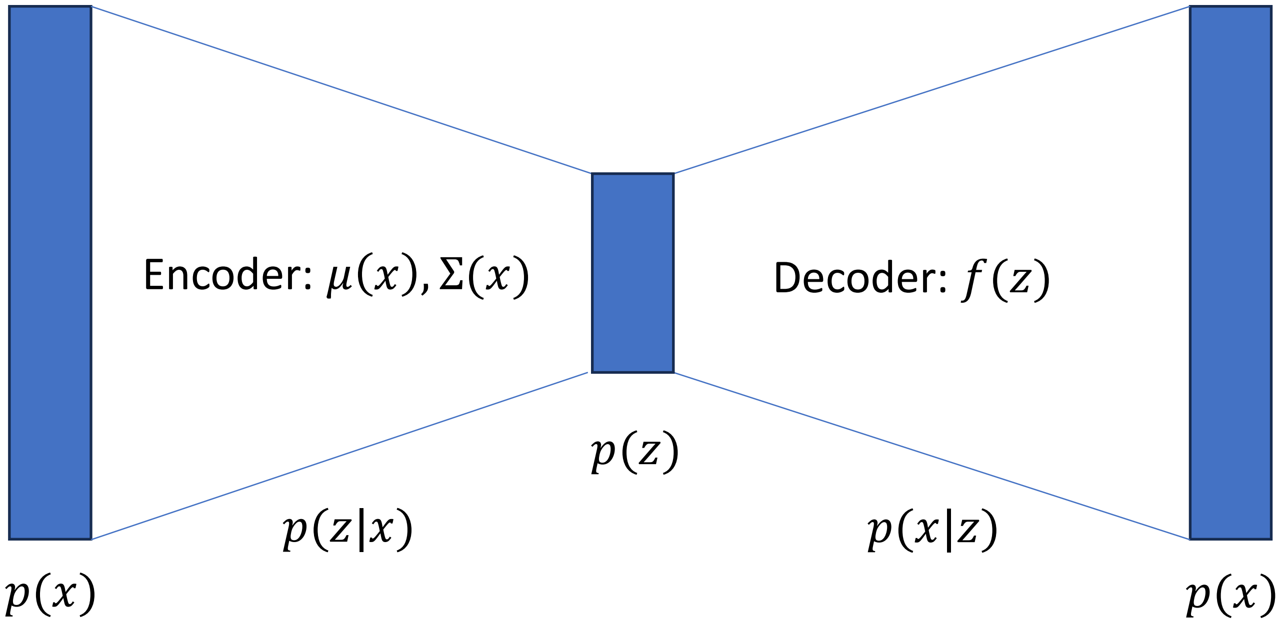


Sidenote: Variational Autoencoders

- What if we want to generate new data (sample from latent space)?

→Problematic since distribution in latent space not known

→Idea: control this distribution (i.e. set prior on latent space)



$p(z) = \mathcal{N}(z, 1)$

$p(x|z) = \mathcal{N}(x - f(z), \alpha)$

Approximation:
$p(z|x) = \mathcal{N}(z - \mu(x), \Sigma(x))$

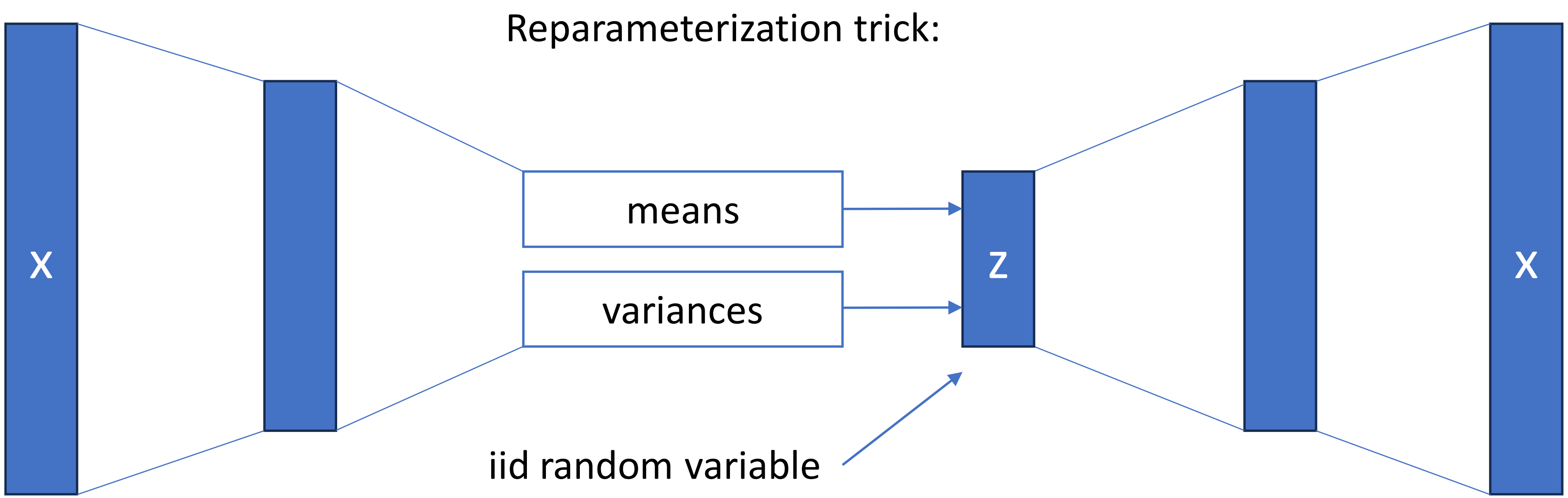
→ Loss: Reconstruction loss + KL-divergence between $p(z|x)$ and $p(z)$

Sidenote: Variational Autoencoders

- What if we want to generate new data (sample from latent space)?

→Problematic since distribution in latent space not known

→Idea: control this distribution (i.e. set prior on latent space)

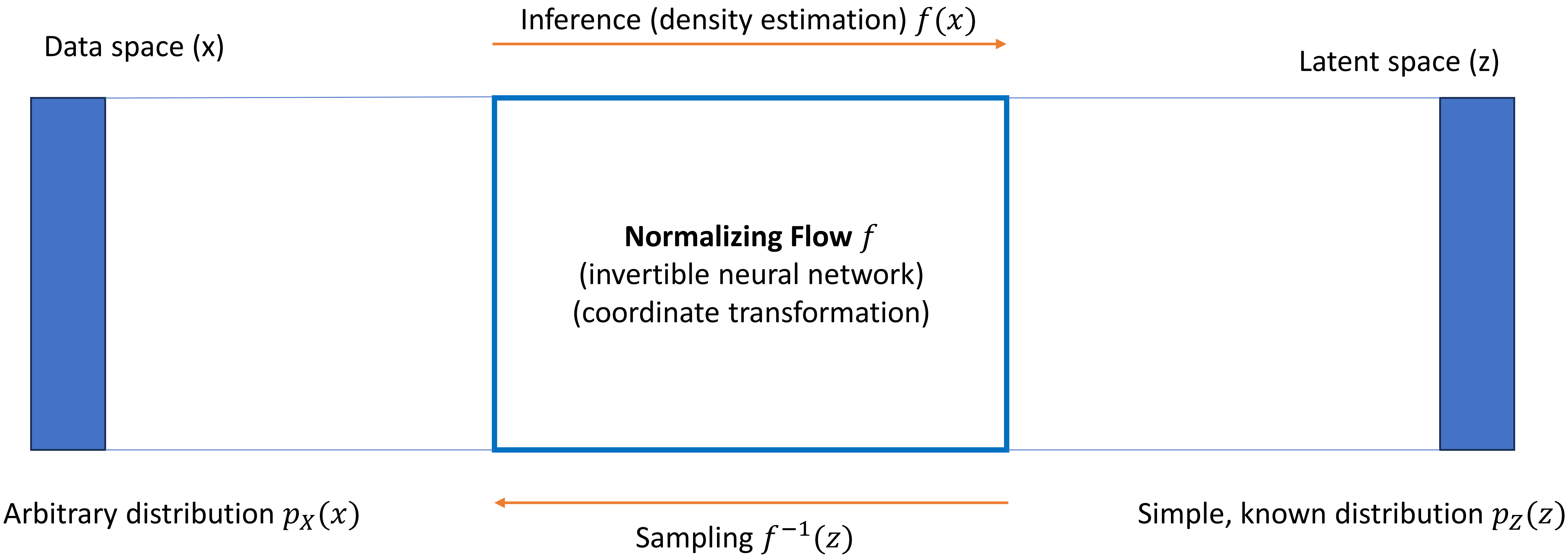


Density estimation: Normalizing Flows

- Suppose we want $p(x)$ → need to make network invertible

Density estimation: Normalizing Flows

- Suppose we want $p(x)$ → need to make network invertible



Density estimation: Normalizing Flows

How do you train this?

- Infer density for training data → compare to prior on latent space (again KL-divergence)
- In this case: minimizing KL divergence equivalent to maximizing likelihood of data under latent space distribution

Problem:

$$P(z \in V) = \int_V p_Z(z)\, dz = \int_{f^{-1}(V)} p_Z\big(f(x)\big)\, |\det(J_f)| dx = \int_{f^{-1}(V)} p_X(x)\, dx$$

→ Inference requires Jacobian of our network!

Density estimation: Normalizing Flows

Simplest idea: $\vec{z} = f(\vec{x}) = A\vec{x} + \vec{t}$, A diagonal and positive

→ $A = e^{S}, S = \mathrm{diag}(\vec{s})$ → $\vec{z} = e^{\vec{s}} \circ \vec{x} + \vec{t}$

Jacobian:

Inverse: $\vec{x} = f^{-1}(\vec{z}) = e^{-\vec{s}}(\vec{z} - \vec{t})$

$$J = A = \mathrm{diag}\left(e^{\vec{s}}\right)$$

$$\det J = \prod_i e^{s_i}$$

This fulfils our requirements but is obviously too simple!

Density estimation: Normalizing Flows

Solution: **Coupling Flows**

Split $\vec{x}, \vec{z}$ into $\vec{x}_1, \vec{x}_2$ and $\vec{z}_1, \vec{z}_2$

$$\vec{x}_1 \rightarrow \vec{z}_1 = \vec{x}_1$$

$$\vec{x}_2 \rightarrow \vec{z}_2 = \vec{x}_2 \circ e^{\vec{s}(\vec{x}_1)} + \vec{t}(\vec{x}_1)$$

Inverse:

$$\vec{z}_1 \rightarrow \vec{x}_1 = \vec{z}_1$$

$$\vec{z}_2 \rightarrow \vec{x}_2 = \left(\vec{z}_2 - \vec{t}(\vec{z}_1)\right) \circ e^{-\vec{s}(\vec{z}_1)}$$

Jacobian:

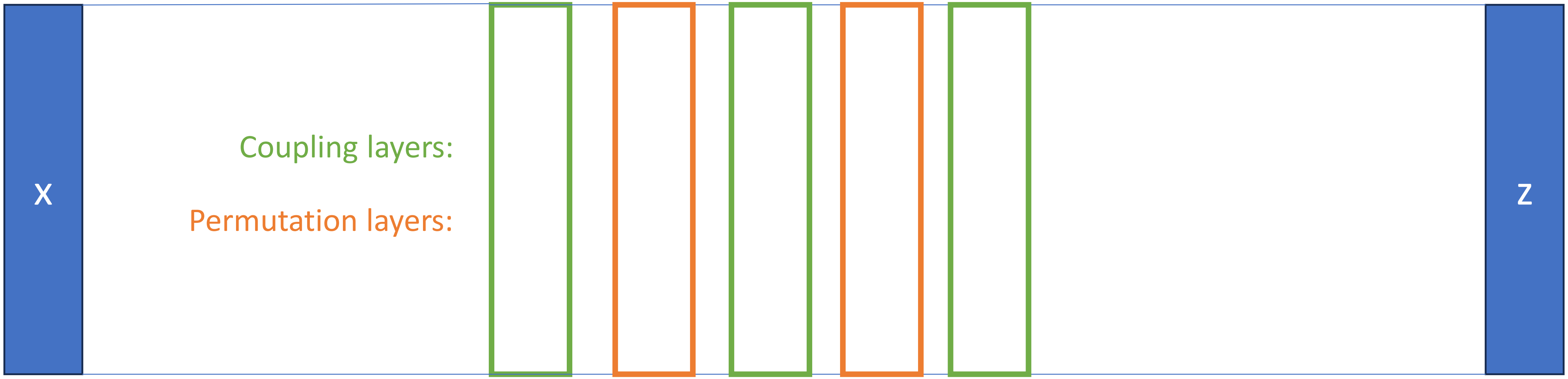
$$\begin{array}{c|c|c} & \vec{x}_1 & \vec{x}_2 \\ \vec{z}_1 & I & 0 \\ \hline \vec{z}_2 & \text{potentially complicated stuff} & \mathrm{diag}\left(e^{\vec{s}(\vec{x}_1)}\right) \end{array}$$

$$\det J = \prod_i e^{s_i(\vec{x}_1)}$$

Density estimation: Normalizing Flows

Obvious problem: this only transforms half of the dimensions

→ Stack multiple layers with permutation layers in between



$$\det(J) = \det(J_1) \cdot 1 \cdot \det(J_2) \cdot 1 \cdot \det(J_3)$$

Density estimation: Normalizing Flows

Coupling Flows

Split $\vec{x}, \vec{z}$ into $\vec{x}_1, \vec{x}_2$ and $\vec{z}_1, \vec{z}_2$

$$\vec{x}_1 \to \vec{z}_1 = \vec{x}_1$$

$$\vec{x}_2 \to \vec{z}_2 = \vec{x}_2 \circ e^{\vec{s}(\vec{x}_1)} + \vec{t}(\vec{x}_1)$$

Inverse:

$$\vec{z}_1 \to \vec{x}_1 = \vec{z}_1$$

$$\vec{z}_2 \to \vec{x}_2 = \left(\vec{z}_2 - \vec{t}(\vec{z}_1)\right) \circ e^{-\vec{s}(\vec{z}_1)}$$

Jacobian:

$$\begin{array}{c|c|c} & \vec{x}_1 & \vec{x}_2 \\ \hline \vec{z}_1 & I & 0 \\ \hline \vec{z}_2 & \text{potentially complicated stuff} & \mathrm{diag}\left(e^{\vec{s}(\vec{x}_1)}\right) \end{array}$$

$$\det J = \prod_i e^{s_i(\vec{x}_1)}$$

Density estimation: Normalizing Flows

Solution: **Coupling Flows** → Autoregressive Flows, …

Split $\vec{x}, \vec{z}$ into $\vec{x}_1, \vec{x}_2$ and $\vec{z}_1, \vec{z}_2$

$$\vec{x}_1 \to \vec{z}_1 = \vec{x}_1$$

$$\vec{x}_2 \to \vec{z}_2 = \boxed{\vec{x}_2 \circ e^{\vec{s}(\vec{x}_1)} + \vec{t}(\vec{x}_1)}$$

Inverse:

Could be a general invertible function!

$$\vec{z}_1 \to \vec{x}_1 = \vec{z}_1$$

$$\vec{z}_2 \to \vec{x}_2 = \left(\vec{z}_2 - \vec{t}(\vec{z}_1)\right) \circ e^{-\vec{s}(\vec{z}_1)}$$

Jacobian:

$$\begin{array}{c|c|c} & \vec{x}_1 & \vec{x}_2 \\ \hline \vec{z}_1 & I & 0 \\ \hline \vec{z}_2 & \text{potentially complicated stuff} & \mathrm{diag}\left(e^{\vec{s}(\vec{x}_1)}\right) \end{array}$$

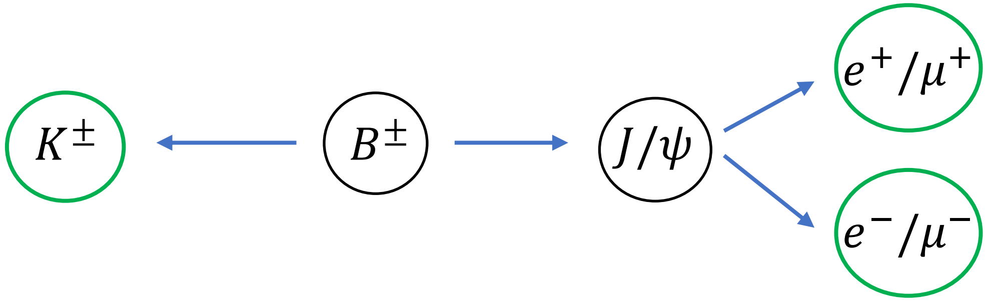
Useless zeros!

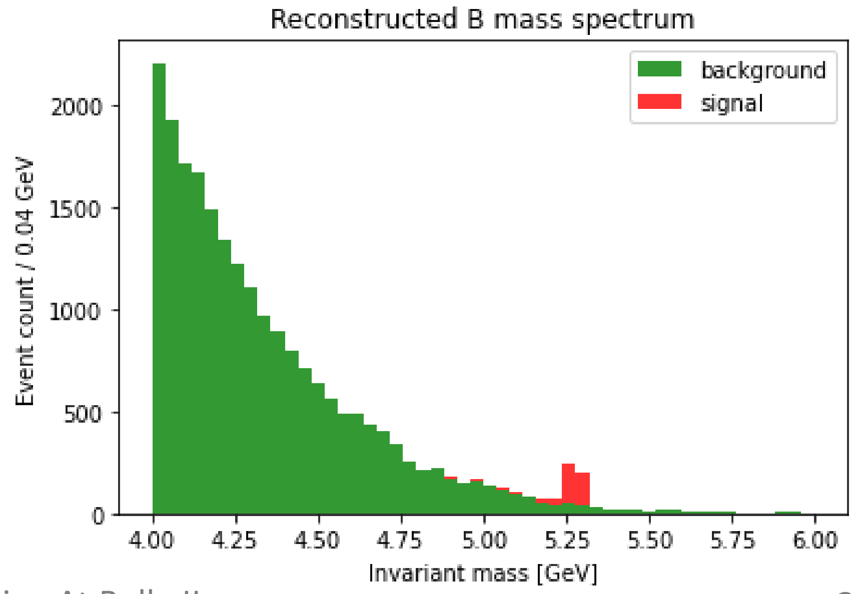
$$\det J = \prod_i e^{s_i(\vec{x}_1)}$$

Back to Anomaly Detection

Simple Performance Test

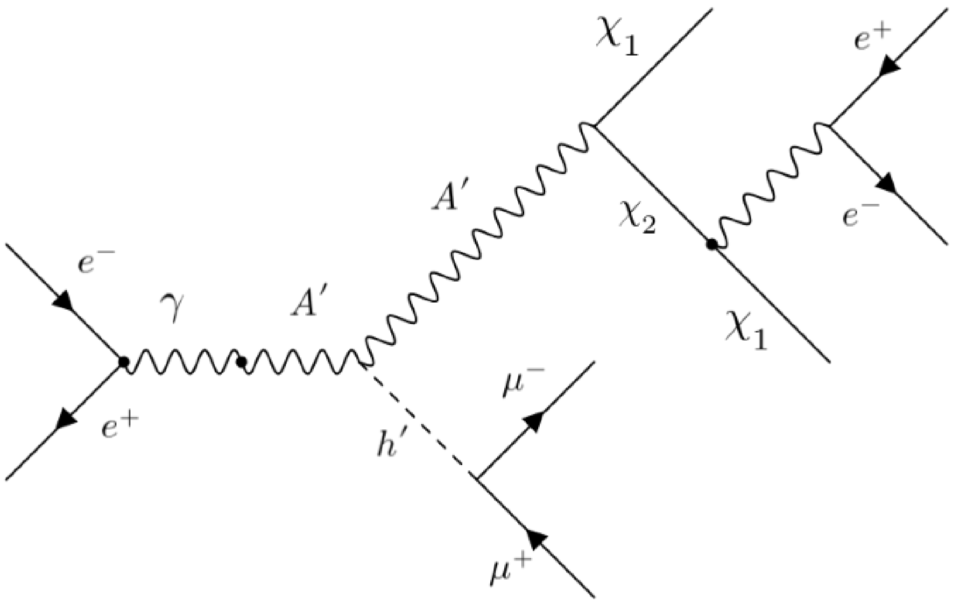
- Idea for performance test:
  - Choose an easily reconstructable B decay with small branching fraction
  - Reconstruct B without cuts (and define signal region in B mass spectrum)
  - Calculate significance improvement after cuts on anomaly score
- Simple choice: $B^{\pm} \to J/\psi\ K^{\pm}$
  - Hadronic: nice bump in mass spectrum
  - J/ψ from dileptonic decays





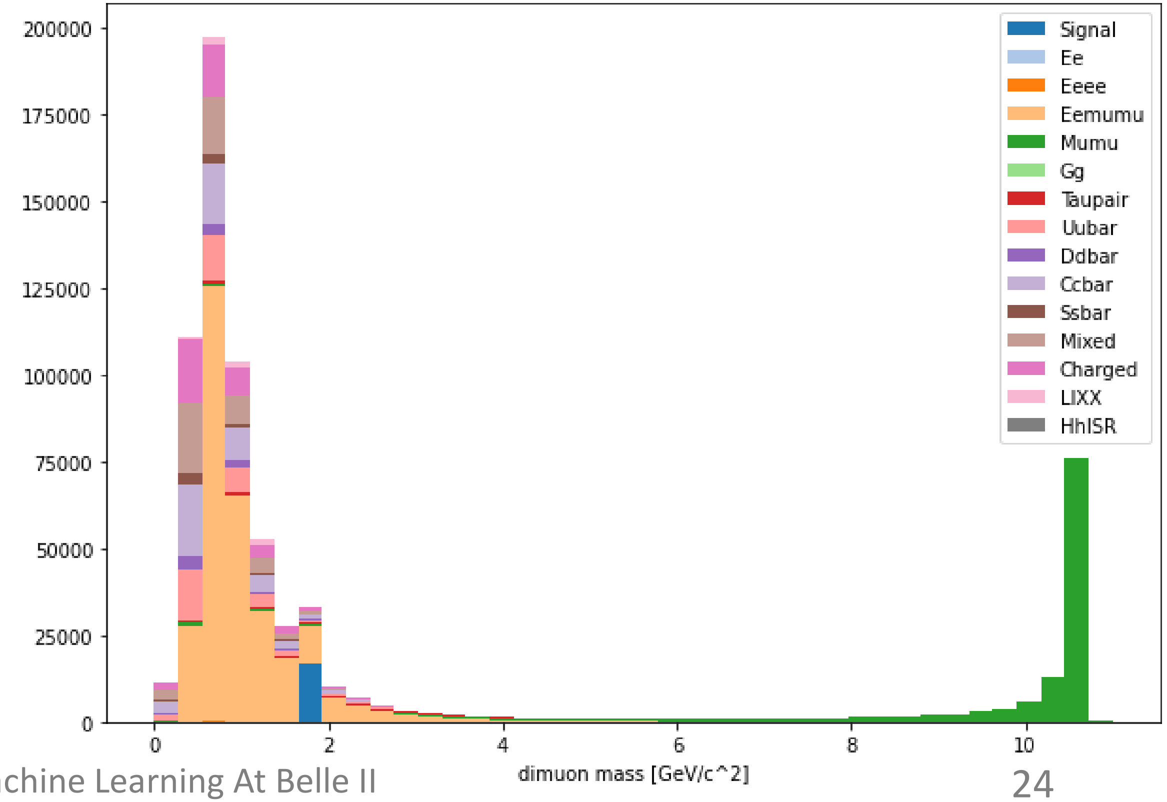
Better performance test

- New Physics Sample: dark matter model with dark Higgs and Photon (kindly received from Jonas Eppelt at KIT)
- Again: calculate significance improvement after anomaly cuts
- Resonant variable: dimuon mass



Better performance test

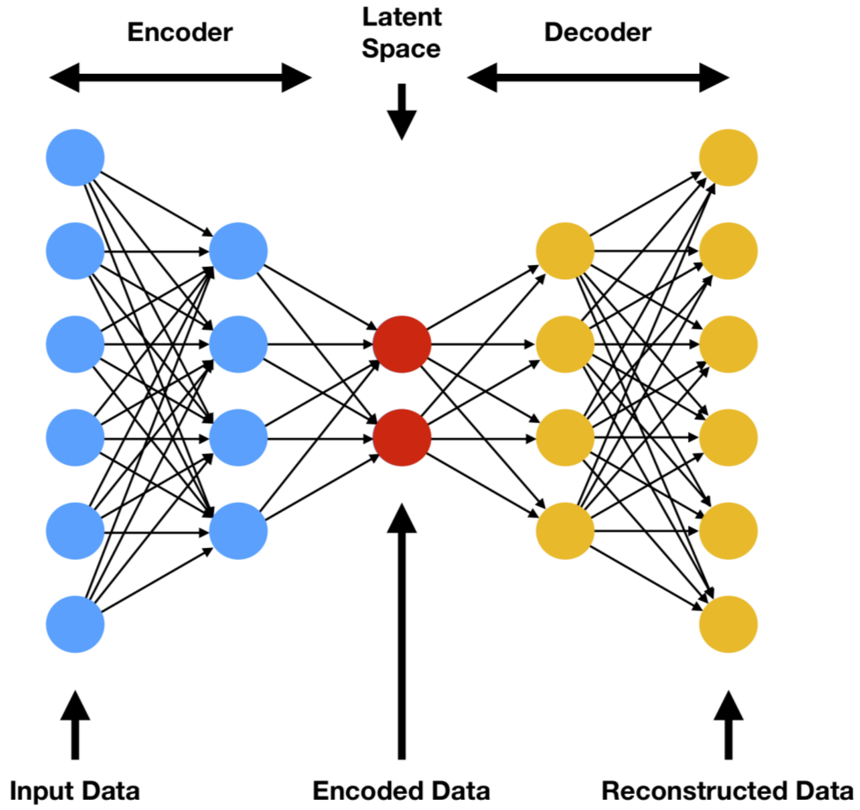
- New Physics Sample: dark matter model with dark Higgs and Photon (kindly received from Jonas Eppelt at KIT)
- Again: calculate significance improvement after anomaly cuts
- Resonant variable: dimuon mass



Anomaly Detection With Autoencoders

- Reminder: Autoencoder learns a **lower dimensional representation** of input data
- Imperfect reconstruction → reconstruction loss

- **Data-driven approach:**
  1. Train AE on subset of data (assumption: anomalies are rare)
  2. Applied to the total dataset, anomalies are expected to have a higher reconstruction loss → Use as anomaly score



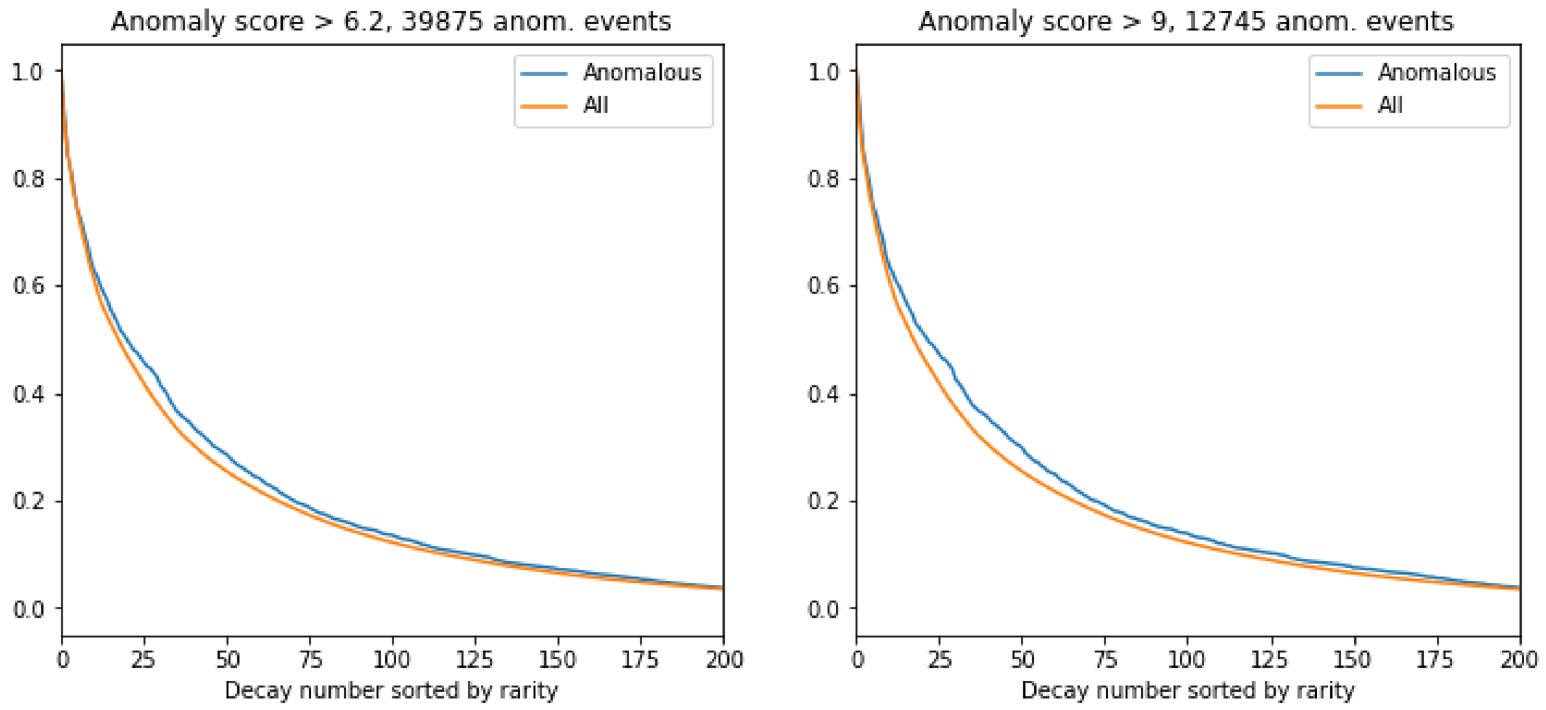
Specific Architecture (Previous Approach)

- **Inputs**: Tried different approaches using either
  - directly the reconstructed four-momenta of particles or
  - derived quantities such as n-body inv. masses, angles between particles, …

  → No difference in performance (also cross-checked with a supervised classifier)
- Variation of **depth** and **latent space size** had little to no effect
  - Settled arbitrarily on 8 latent dims and 5 hidden layers in total
- Currently redoing these studies with a combination of the above inputs, improved encoding, and on a larger unskimmed dataset

Simple preliminary test

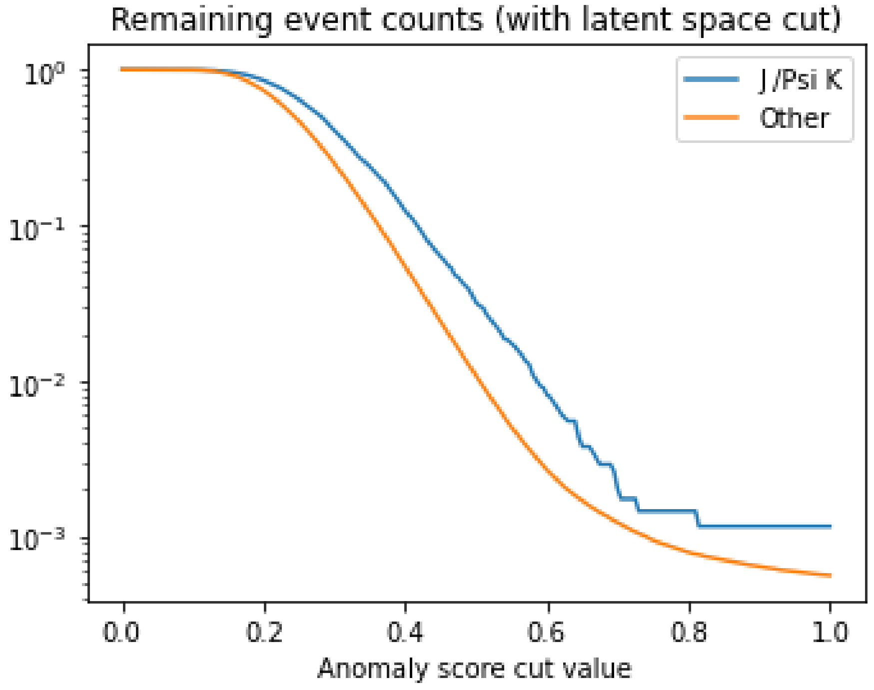
- AE trained on 250k simulated generic B decays

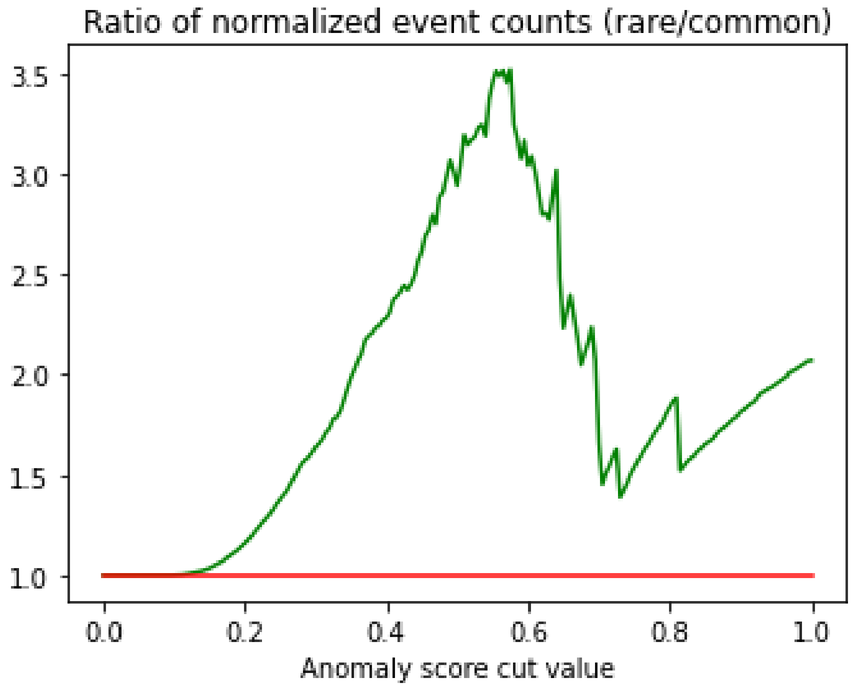


→ Increase in rare events after anomaly cut

Some performance graphs

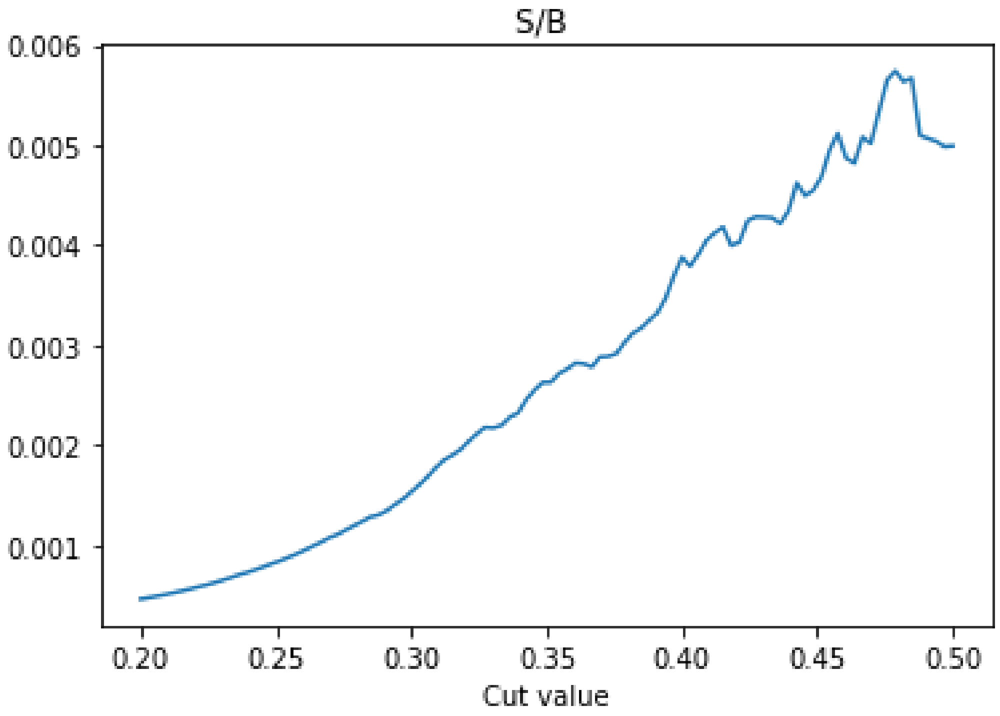
- AE trained on 250k simulated generic B decays
- Normalized $B^{\pm} \to J/\psi\, K^{\pm}$ event counts after anomaly cuts:

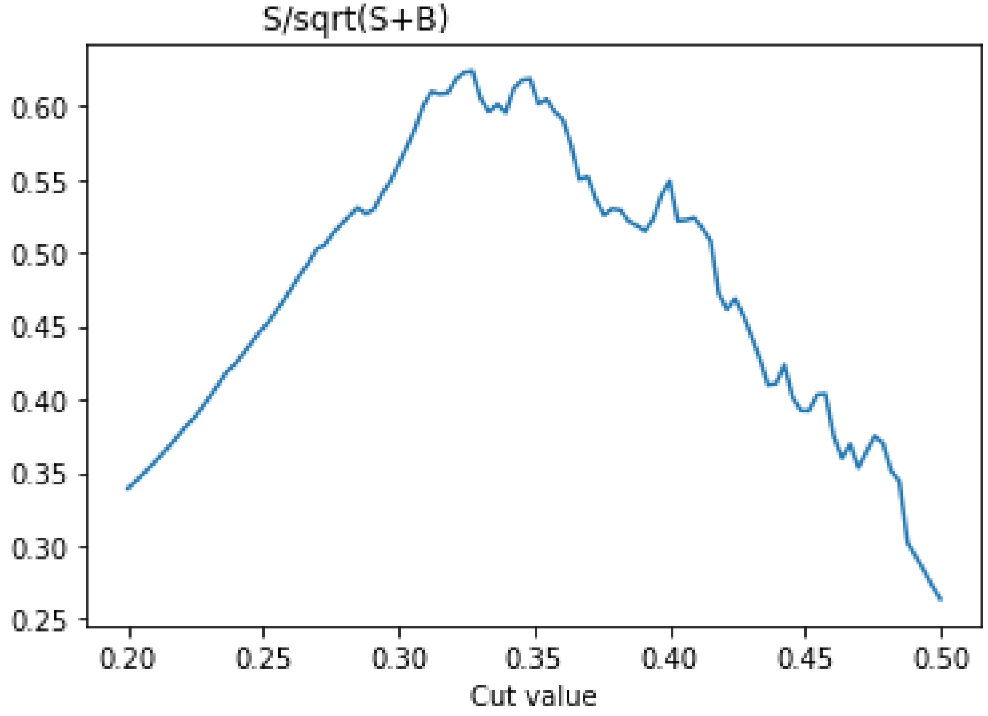




Some performance graphs

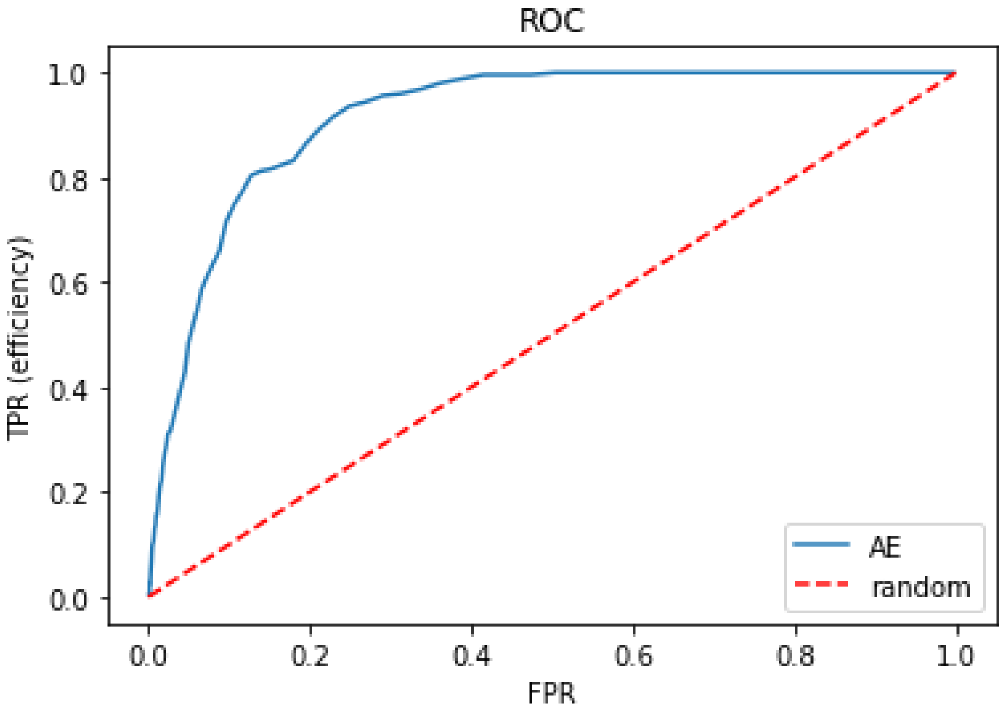
- Simple reconstruction of $B^{\pm} \rightarrow J/\psi\, K^{\pm}$
- Significance estimate: $S/\sqrt{S+B}$

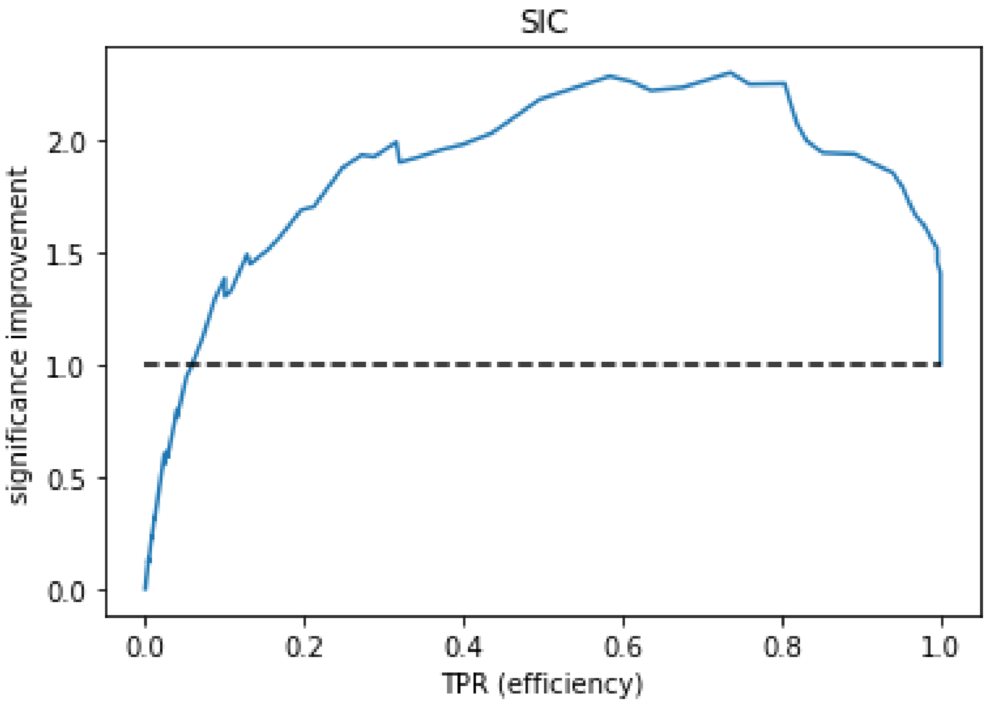




Some performance graphs

- Simple reconstruction of $B^{\pm} \rightarrow J/\psi\, K^{\pm}$
- Significance estimate: $S/\sqrt{S+B}$





Some sidenotes

Also tried

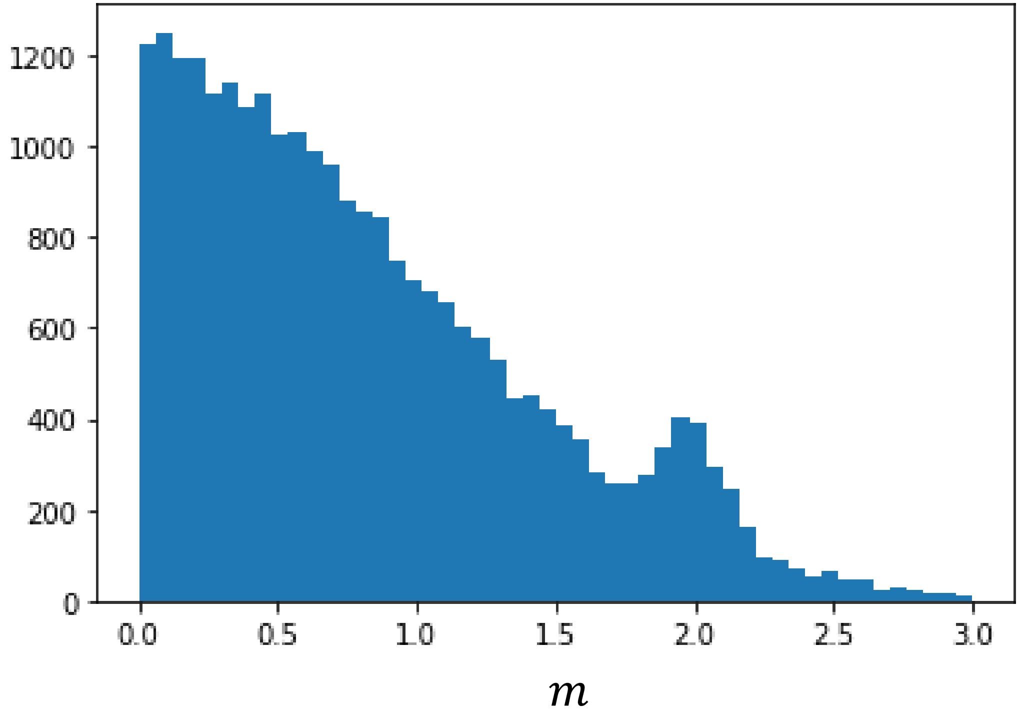
- Variational Autoencoders
  → Worse performance with more difficult training
- **Encoding** of MC information (rarity) in **latent space**
  → Slight improvement in performance but breaks data-driven approach

Anomaly Detection With Density Estimation

- Different methods ((R-)ANODE, CATHODE)
- Basic principle always the same:

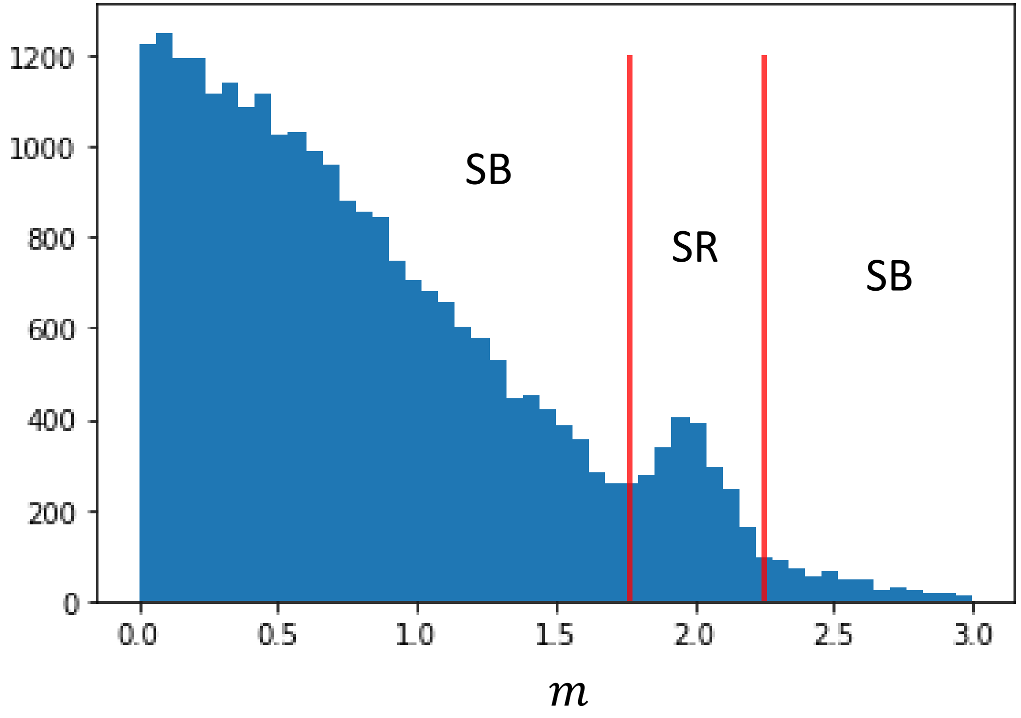
Anomaly Detection With Density Estimation

- Different methods ((R-)ANODE, CATHODE)
- Basic principle always the same:
  - Choose a variable in which to look for a localized signal



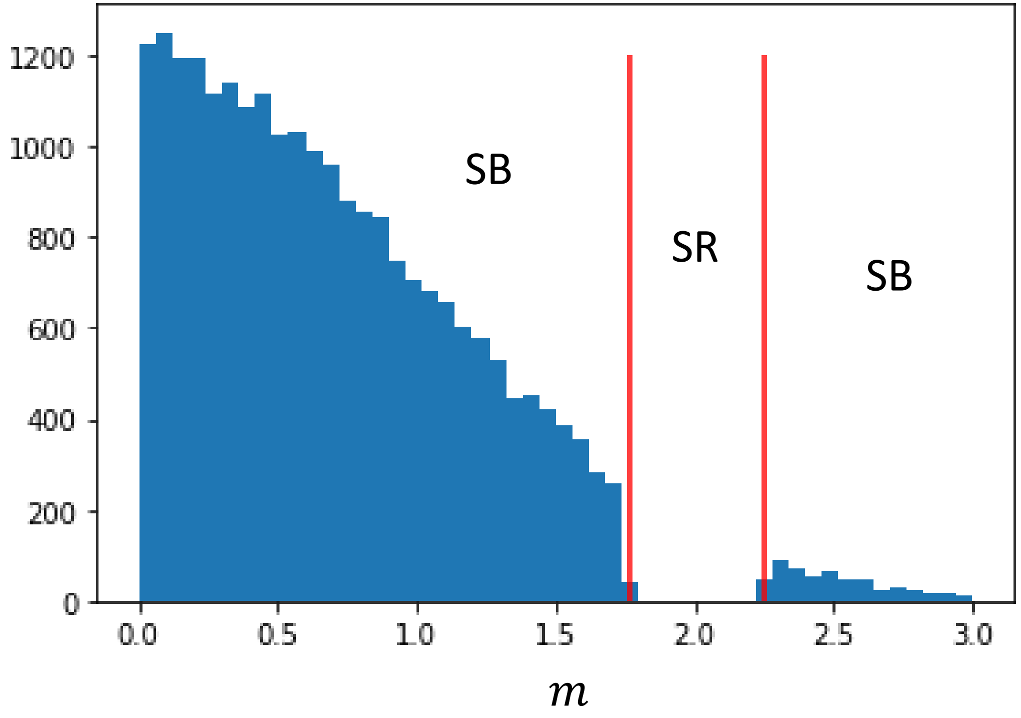
Anomaly Detection With Density Estimation

- Different methods ((R-)ANODE, CATHODE)
- Basic principle always the same:
  - Choose a variable in which to look for a localized signal
  - Define a signal region (SR)



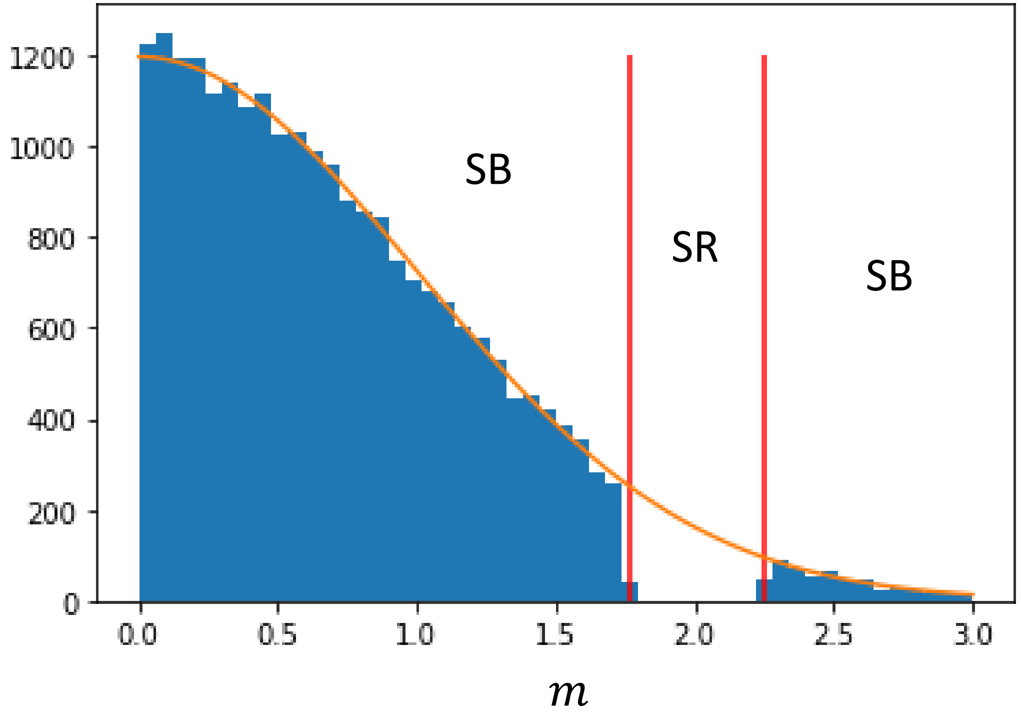
Anomaly Detection With Density Estimation

- Different methods ((R-)ANODE, CATHODE)
- Basic principle always the same:
  - Choose a variable in which to look for a localized signal
  - Define a signal region (SR)
  - Train a density estimator on the sidebands (everything except SR)



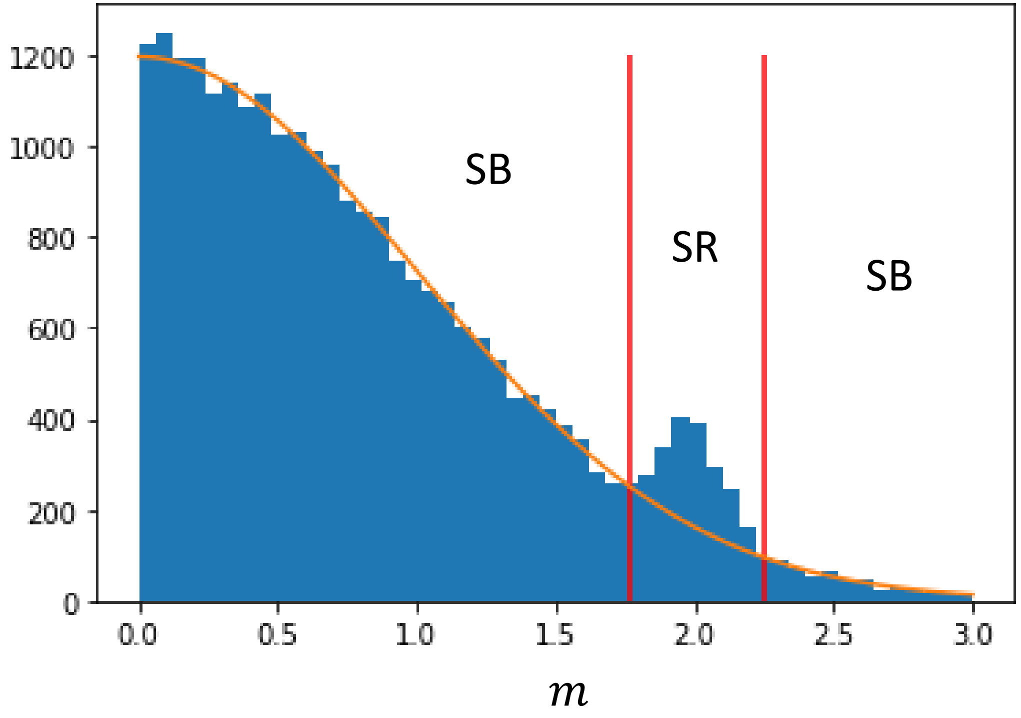
Anomaly Detection With Density Estimation

- Different methods ((R-)ANODE, CATHODE)
- Basic principle always the same:
  - Choose a variable in which to look for a localized signal
  - Define a signal region (SR)
  - Train a density estimator on the sidebands (everything except SR)
  - Extrapolate this learned density into the SR



Anomaly Detection With Density Estimation

- Different methods ((R-)ANODE, CATHODE)
- Basic principle always the same:
  - Choose a variable in which to look for a localized signal
  - Define a signal region (SR)
  - Train a density estimator on the sidebands (everything except SR)
  - Extrapolate this learned density into the SR
  - Compare to actual distribution
    → This is where the models differ



CATHODE[1]

- **Sample** from the extrapolated distribution
- Train a **binary classifier** to distinguish sample from actual data in SR
- Expectation for classification score:
  - For background no distinction possible → peak at 0.5
  - For signal tail to higher values

→ Use classification score as anomaly score

[1]Based on arXiv:2109.00546

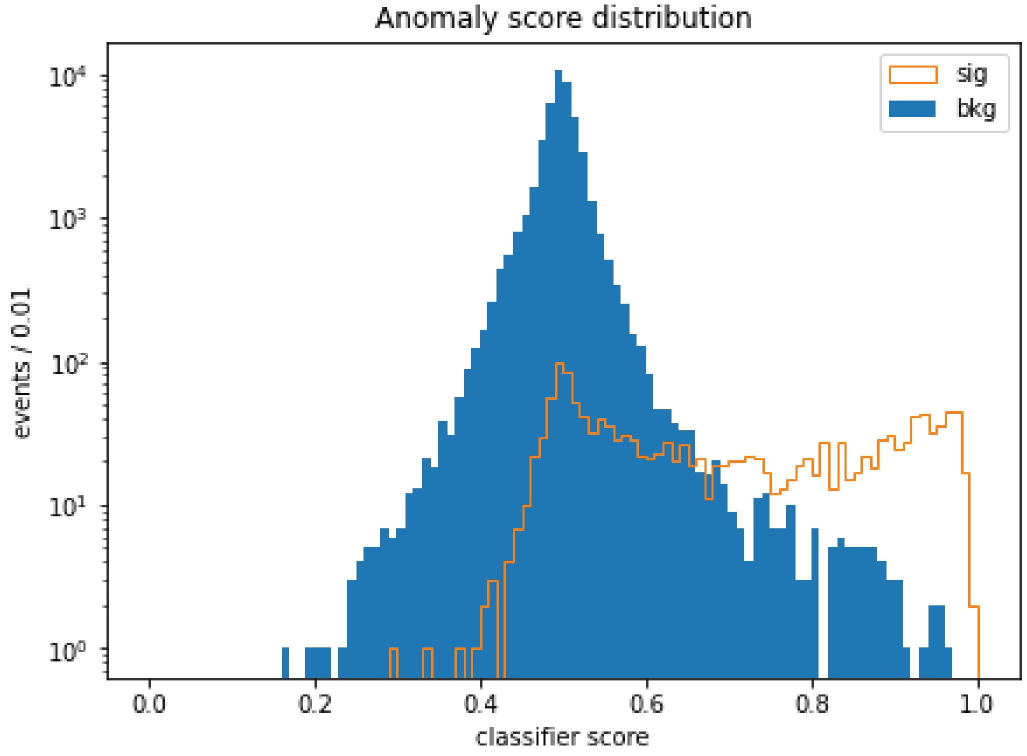
Simple Demonstration

- Implementation tested on public dataset from the LHC Olympics AD Challenge[1] (anomaly in dijet mass distribution)

[1]Publicly available under https://zenodo.org/records/4536377

Simple Demonstration

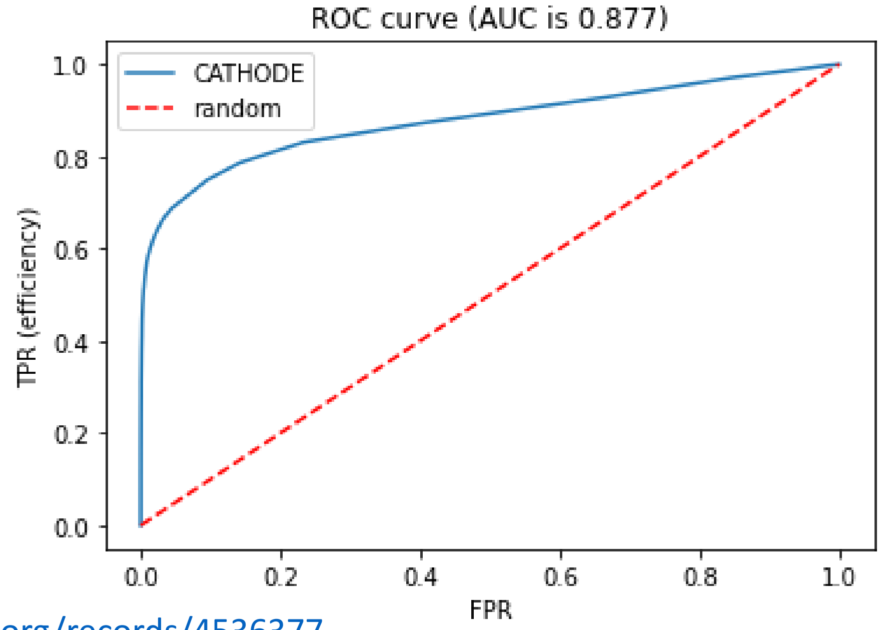
- Implementation tested on public dataset from the LHC Olympics AD Challenge[1] (anomaly in dijet mass distribution)
- Anomaly (classification) score distribution:

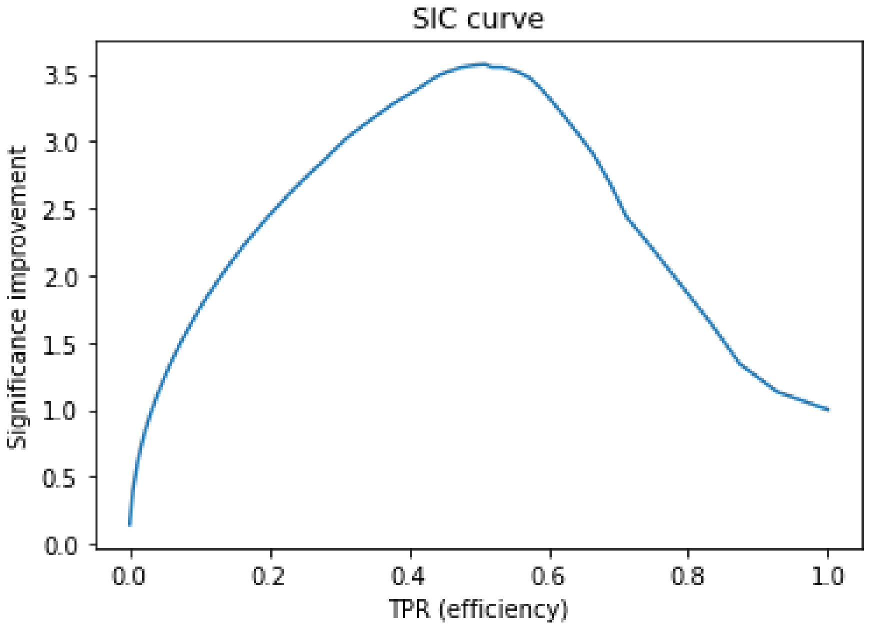


[1]Publicly available under https://zenodo.org/records/4536377

Simple Demonstration

- Implementation tested on public dataset from the LHC Olympics AD Challenge[1] (anomaly in dijet mass distribution)
- Performance:





[1]Publicly available under https://zenodo.org/records/4536377

Outlook

Density Estimation:

- Presented scenario is a bit artificial (known signal region) → needs a scanning procedure (probably not in the scope of my thesis)
- Currently working on Belle II implementation on New Physics sample

Autoencoders:

- Current studies on unskimmed dataset don't show promise for the J/Psi K analysis → investigating modifications
- Very preliminary results on New Physics sample show more promise

Thank you!