ML4Jets 2022 RUTGERS THE STATE UNIVERSITY OF NEW JERSEY

Neural Embedding :

Learning the Embedding of the Manifold of Physics Data

Based On https://arxiv.org/abs/2208.05484

Sang Eon Park
(MIT, IAIFI)
with Phil Harris (MIT, IAIFI), Bryan Ostdiek (Harvard, IAIFI)

Metric & Manifold Structure on Collider Physics Data

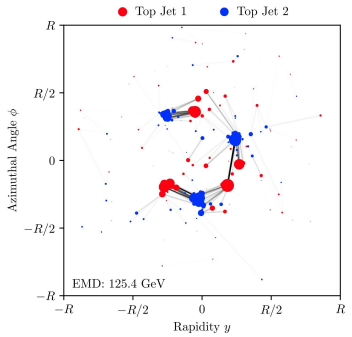
Energy Mover's Distance (Komiske, Metodiev, Thaler, 2019) , allows us compute distance between **any two collider physics events,** satisfies metric properties

An example of **optimal transport based metric** : Move one event to another by optimally moving energy around

Manifold Hypothesis : Structured data forms a lower dimensional manifold within ambient space , and complexity of data can be drastically reduced.
Physics data is highly structured (Governed by physical laws)

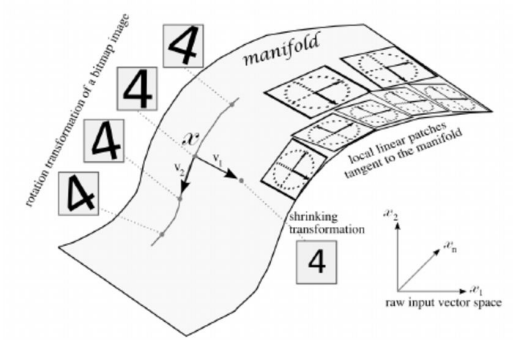
If we consider jets(events) with N particles and 3 features per particle (pT, eta, phi), the collider physics **manifold** lives in $\mathbb{R}^{3N}$

$$N \sim 50$$



1902.02346

$$\mathrm{EMD}(\mathcal{E},\mathcal{E}') = \min_{\{f_{ij}\}} \sum_{ij} f_{ij}\frac{\theta_{ij}}{R} + \left|\sum_i E_i - \sum_j E'_j\right|,$$
$$f_{ij} \geq 0,\ \ \sum_j f_{ij} \leq E_i,\ \ \sum_i f_{ij} \leq E'_j,\ \ \sum_{ij} f_{ij} = E_{\min}$$



1203.2990

Why aren't we using the manifold & metric property everywhere?

1. **Curse of dimensionality**: Intractable to use manifold property directly on the input space because of huge dimension
2. **Metric itself is complicated**: Uses complex computational tools and packages
3. **Problem with scaling:** To study relation between $O(10^6)$ events needs $O(10^{12})$ computations (Pairwise computation)

What we propose

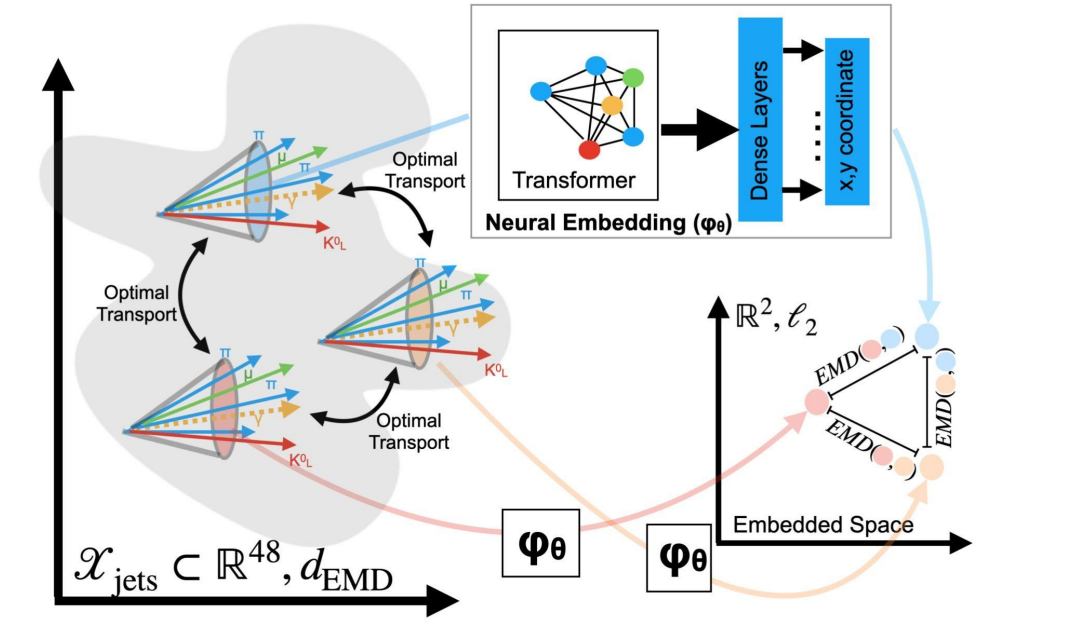
$$\phi : (\mathcal{X}, d_{\mathcal{X}}) \to (\mathcal{Y}, d_{\mathcal{Y}})$$

In high dimensional ambient space

complicated

Low dimensional

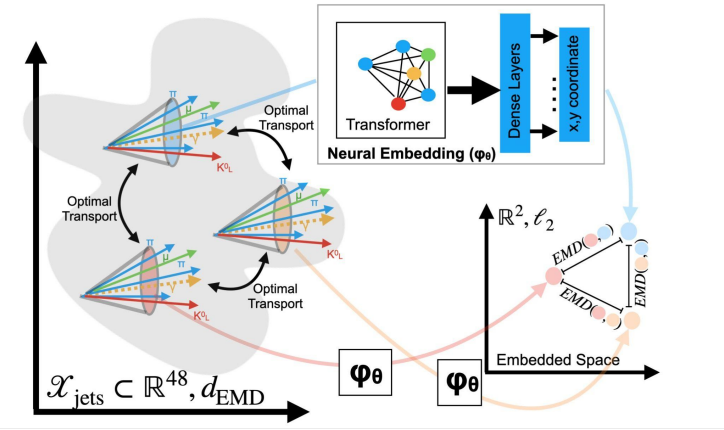
Simple



Highlight of the Method

Embedding **solves all these problems** !

1. Curse of dimensionality : Embedded space is lower-dimensional
2. Cumbersome computation : Original metric is replaced by a way simpler metric (ex. L2 norm)
3. Problem with scaling : **One forward pass** of the neural network



In addition to that, it achieves

4. Easy online application (one evaluation of the neural network)
5. Data compression, dataset organization
6. Offers alternative method of building lower dimensional spaces for searches (Compared to latent variable modeling)
7. **Quantifying Anomaly Detection search algorithms**

Learning Objective

$$\phi : (\mathcal{X}, d_{\mathcal{X}}) \to (\mathcal{Y}, d_{\mathcal{Y}})$$

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \frac{|d_{\mathcal{Y}}(\phi_\theta(u_i), \phi_\theta(v_i)) - d_{\mathcal{X}}(u_i, v_i)|}{d_{\mathcal{X}}(u_i, v_i)}, \phi_\theta \in \mathcal{F}$$

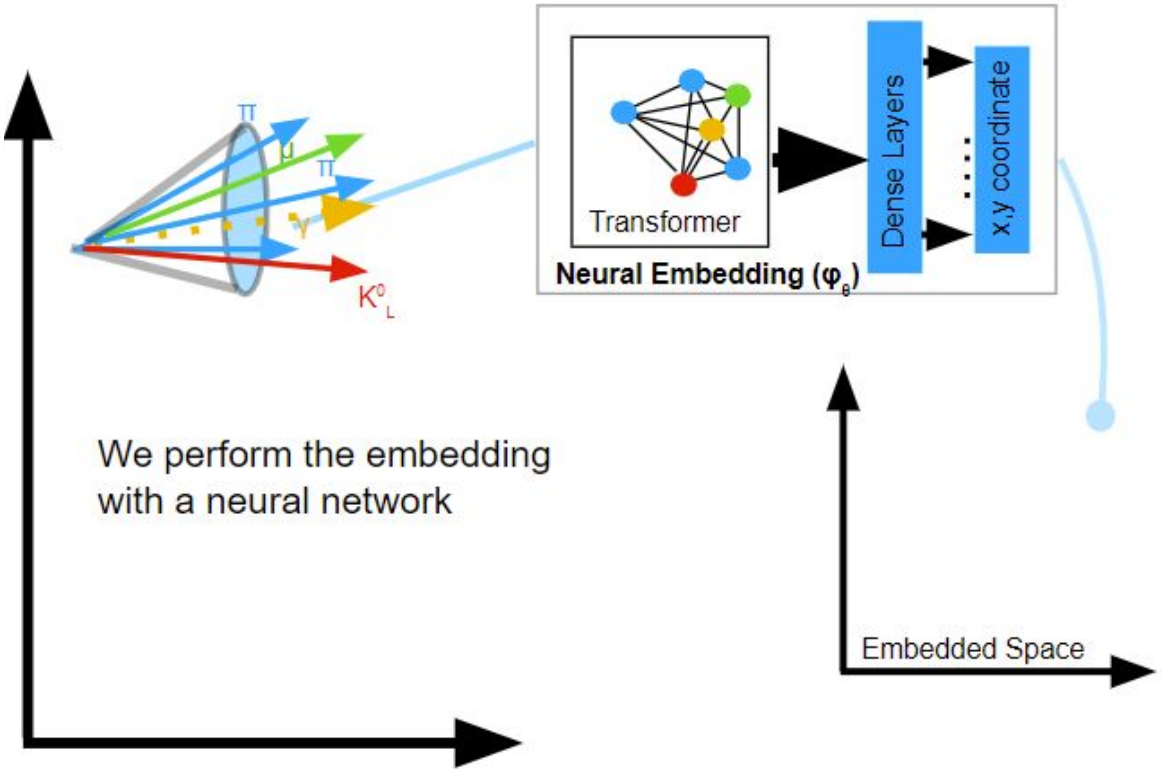
Two big choices :

1. **Which metric space** do we choose to embed into?
2. How to choose **the parametric family of functions**?

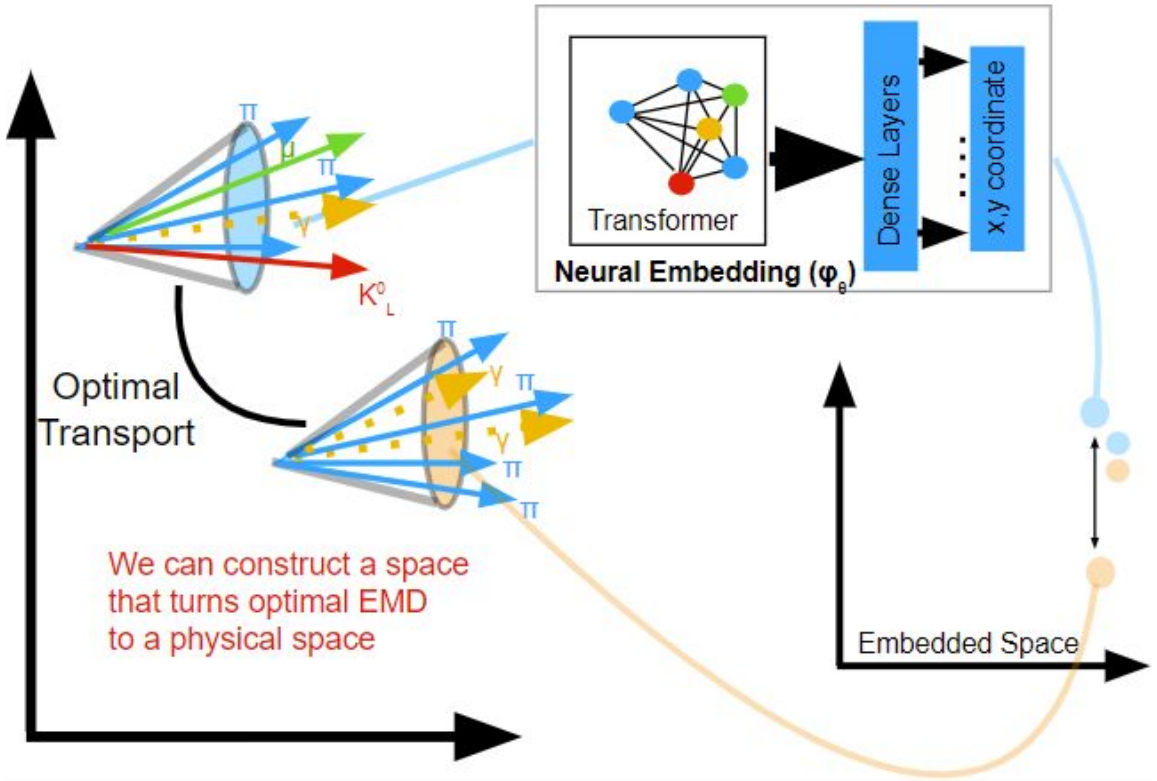
These two questions are closely tied to our modeling assumptions

Do we want to view jets as **Graph**? **Point clouds**? **Tree**?

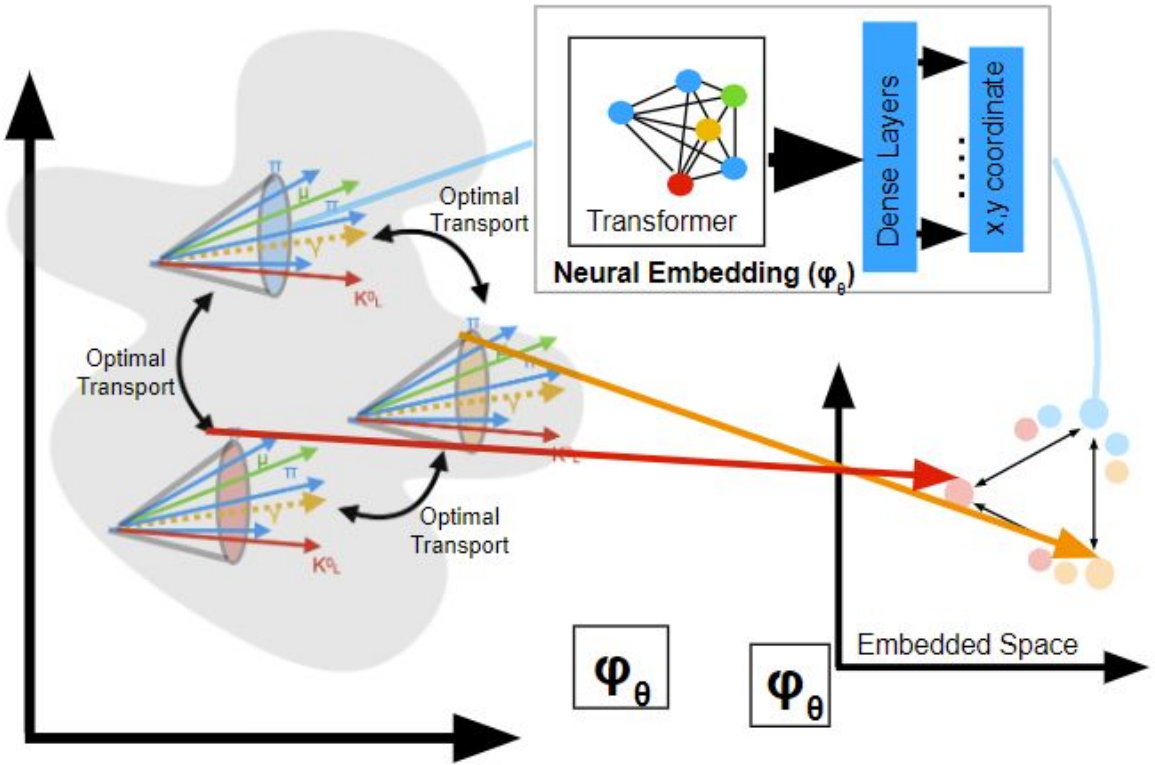
The schematic



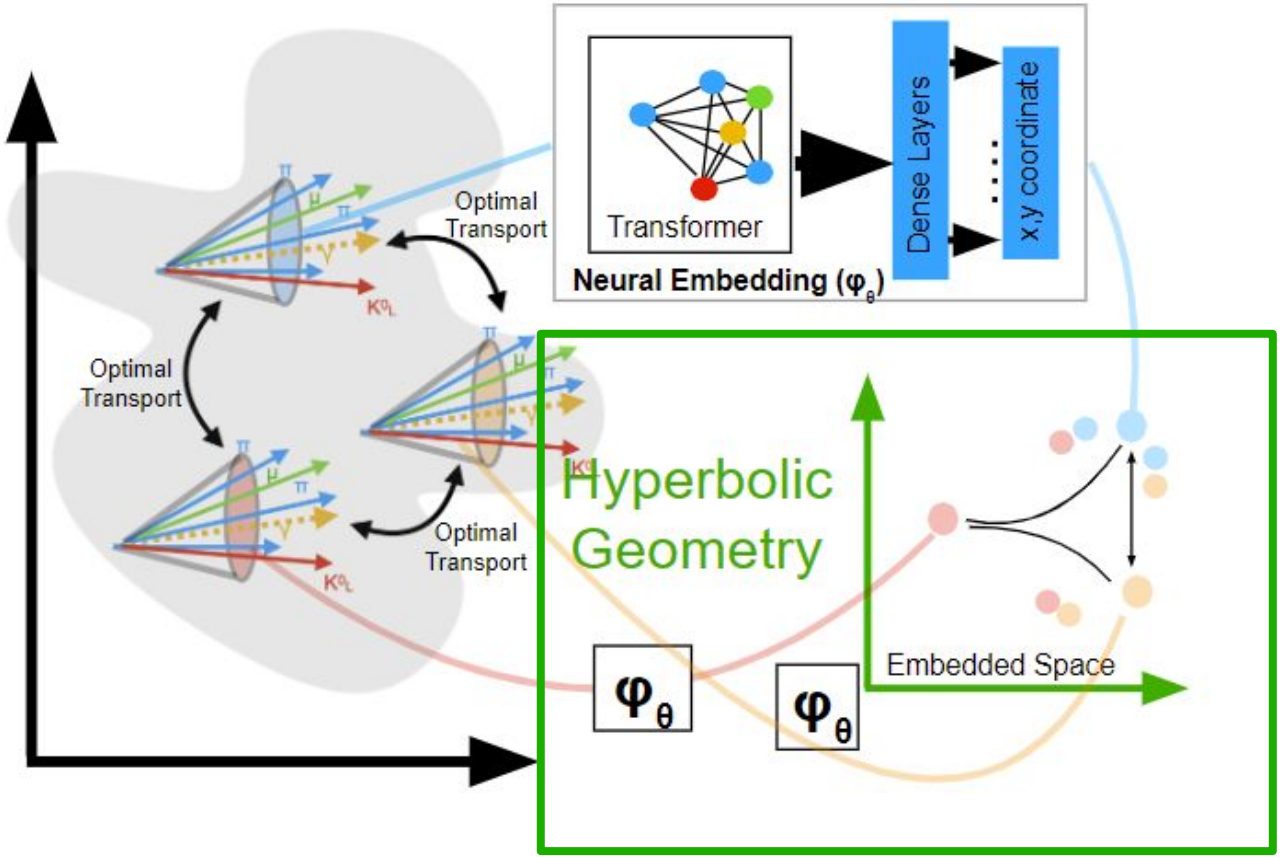
The schematic



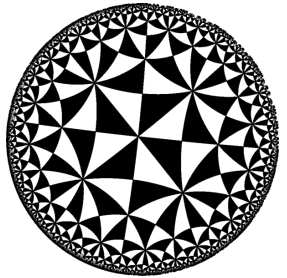
The schematic



The Schematic



Choice of Embedding Space



Preferably **lower dimensional spaces (Dim < 4)** with **a simple metric**

Euclidean spaces: Most common choice, *easy to calculate volume (more later)*

$(\mathbb{R}^2, l_2)$ $(\mathbb{R}^2, l_1)$ $(\mathbb{R}^3, l_2)$

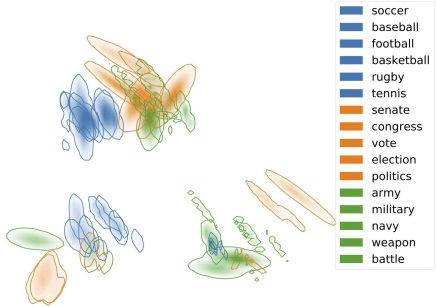
Hyperbolic spaces: Better for handling hierarchically structured data (ex. tree), biological sequences

$(\mathcal{B}^2, d_p)$

1804.03329, 2109.09740

Wasserstein Space: $\mathcal{W}_p(\mathbb{R}^2)$ Extremely big

1905.03329, 2006.09430



1905.03329

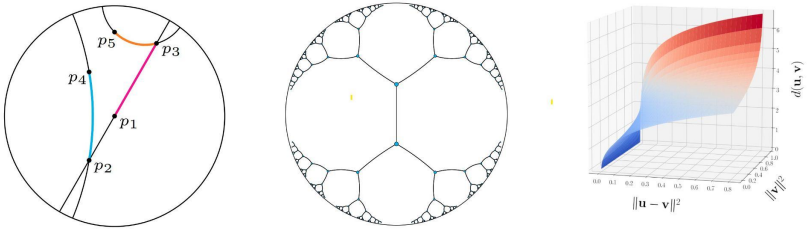
Hyperbolic Embedding

$(\mathcal{B}^n, d_p)$

$$(\mathcal{B}^n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| < 1\}$$

$$d_p(\mathbf{x}, \mathbf{y}) = \operatorname{arcosh}\left(1 + 2\frac{\|\mathbf{x}-\mathbf{y}\|^2}{(1-\|\mathbf{x}\|^2)(1-\|\mathbf{y}\|^2)}\right)$$

1705.08039



(a) Geodesics of the Poincaré disk (b) Embedding of a tree in $\mathcal{B}^2$ (c) Growth of Poincaré distance

Distance gets stretched near the boundary

High encoding capability for hierarchical structures (like trees)

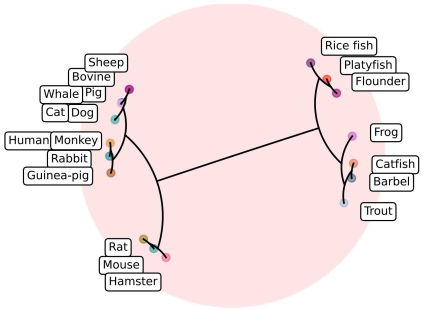
Can we view jets as hierarchical tree-like structure?(Particles shower to more particles)

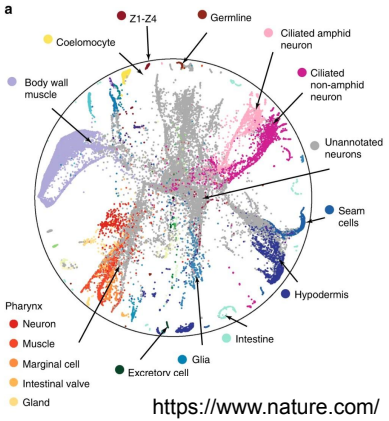
Interesting physics motivation

You can't embed non-euclidean manifold into euclidean space without big distortion

Collider Physics events probably come from non-euclidean geometry(Collisions can be viewed as expanding universe, hyperbolic geometry)

2109.09740





https://www.nature.com/articles/s41467-020-16822-4/f

Motivation: Euclidean Space is too “Narrow” for Tree-like Graphs

O.Ganea

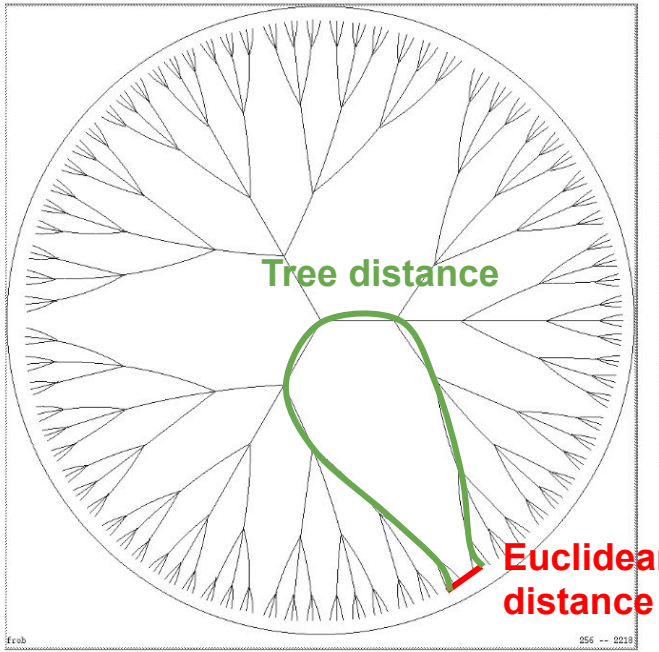
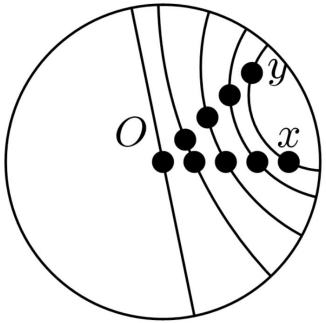
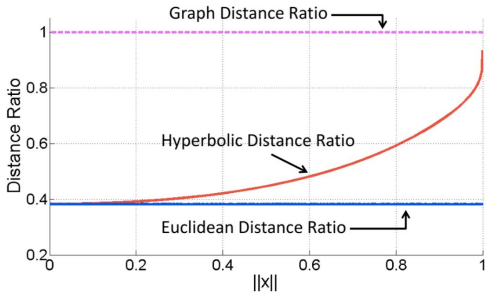


Image source: http://prior.sigchi.org

A tree: the number of nodes grows **exponentially** with the tree depth!

Distance between (x,y)/Distance between O-x





https://collignon.github.io/2020/07/notes-on-hyerpbolic-geometry/

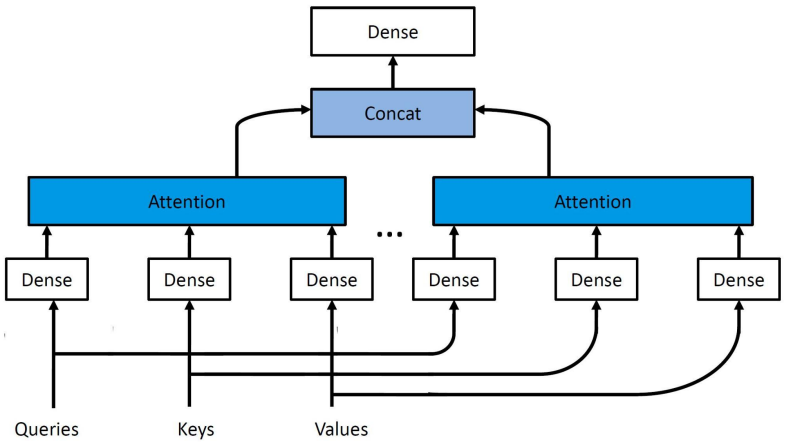
Family of Functions

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \frac{|d_{\mathcal{Y}}(\phi_\theta(u_i), \phi_\theta(v_i)) - d_{\mathcal{X}}(u_i, v_i)|}{d_{\mathcal{X}}(u_i, v_i)}, \phi_\theta \in \mathcal{F}$$

Use family of neural networks (**Neural Embedding**),

For MNIST Images : CNN **For Jets** : Transformers

2106.11342



Transformers = Fully connected graph network with attention

Treat jet data as if we have fully connected graphs

Between particles, Capture relation between every pair of particles

Datasets and Setup

We test our framework in progressively harder datasets

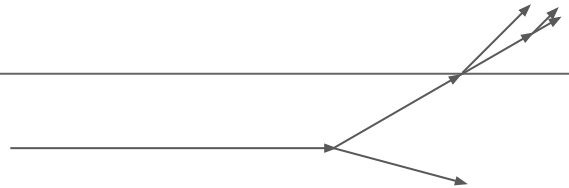
1. MNIST (Preliminary checks)
2. Toy Jets
3. Simulated QCD Jets

Tried 2 different geometry of the embedded space

Dimension of the embedded space fixed to 2

Dataset	Architecture	Geometry
MNIST [49]	CNN	Euclidean
Simple Toy Jets 3.2	Transformer	Euclidean
Realistic Toy Jets 3.3	Transformer	Euclidean
Simulated Jets 4.3	Transformer	Euclidean
Simulated Jets 4.3	Transformer	Hyperbolic

Toy Jet Generator



Build toy jets as iterative splitting (Simplified jets)

Motivation : Probe whether embedding learns the proper latent variables

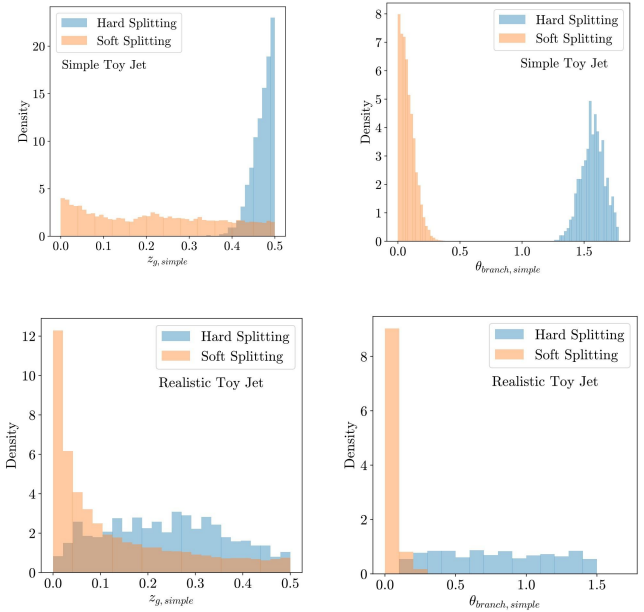
Splitting angle drawn from some fixed distribution (inspired by leading order matrix elements)

Different distributions for hard, soft splitting

Verified with the zg distribution

These generates jets with same mass, pT, number of particles with desired pronginess.

we have the data generating process, and access to variables we don't have in real jet data (ex. The first splitting angle)



$$z_g = \frac{\max p_{\mathrm{T},1}, p_{\mathrm{T},2}}{p_{\mathrm{T},1} + p_{\mathrm{T},2}}$$

Simulated QCD Jets

Events	Jet Pronginess	Jet Mass	Used in Training	Test Dataset
$X \to YY'$	2	25 GeV	✗	Extrapolation
$X \to YY'$	2	80 GeV	✓	Interpolation
$X \to YY'$	2	170 GeV	✗	Extrapolation
$X \to YY'$	2	400 GeV	✓	Interpolation
$W' \to B'T$	3	25 GeV	✗	Extrapolation
$W' \to B'T$	3	80 GeV	✓	Interpolation
$W' \to B'T$	3	170 GeV	✗	Extrapolation
$W' \to B'T$	3	400 GeV	✓	Interpolation
$V_{kk} \to (VV)V$	4	170 GeV	✗	Extrapolation
$V_{kk} \to (VV)V$	4	400 GeV	✗	Extrapolation

Events generated with MADGRAPH, showered with PYTHIA, smeared with DELPHES

Divided into Interpolation and Extrapolation sets, jet topologies in extrapolation sets aren't shown in training stage at all

How is our data represented? (Single Jet, 16 particles)

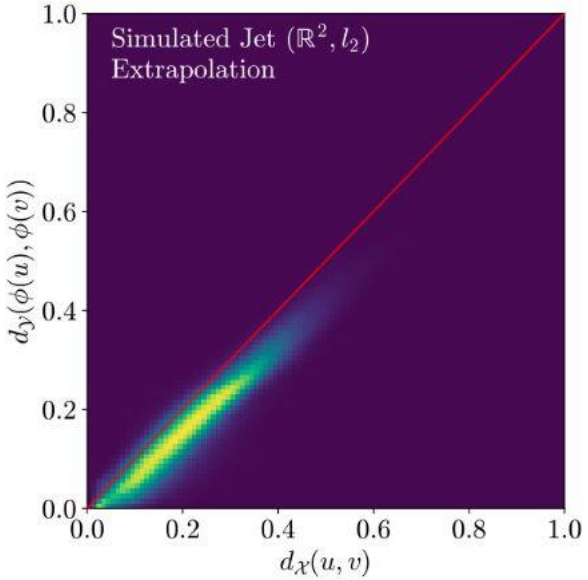
QCD

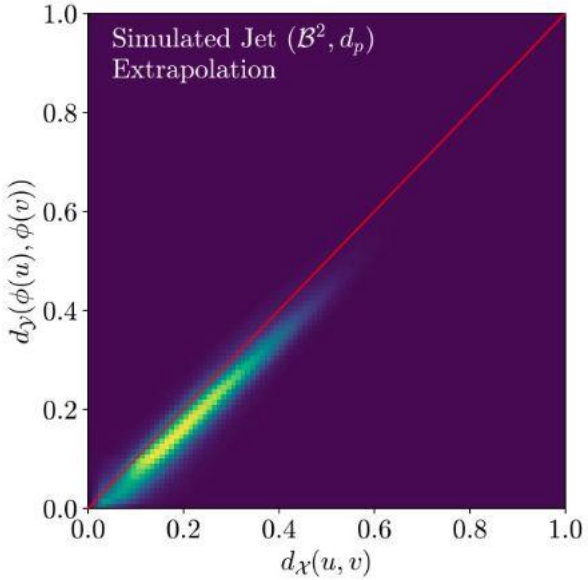
$X \to YY'$
2 prong

$W' \to B'T$
3 prong

$V_{kk} \to (VV)V$
4 prong

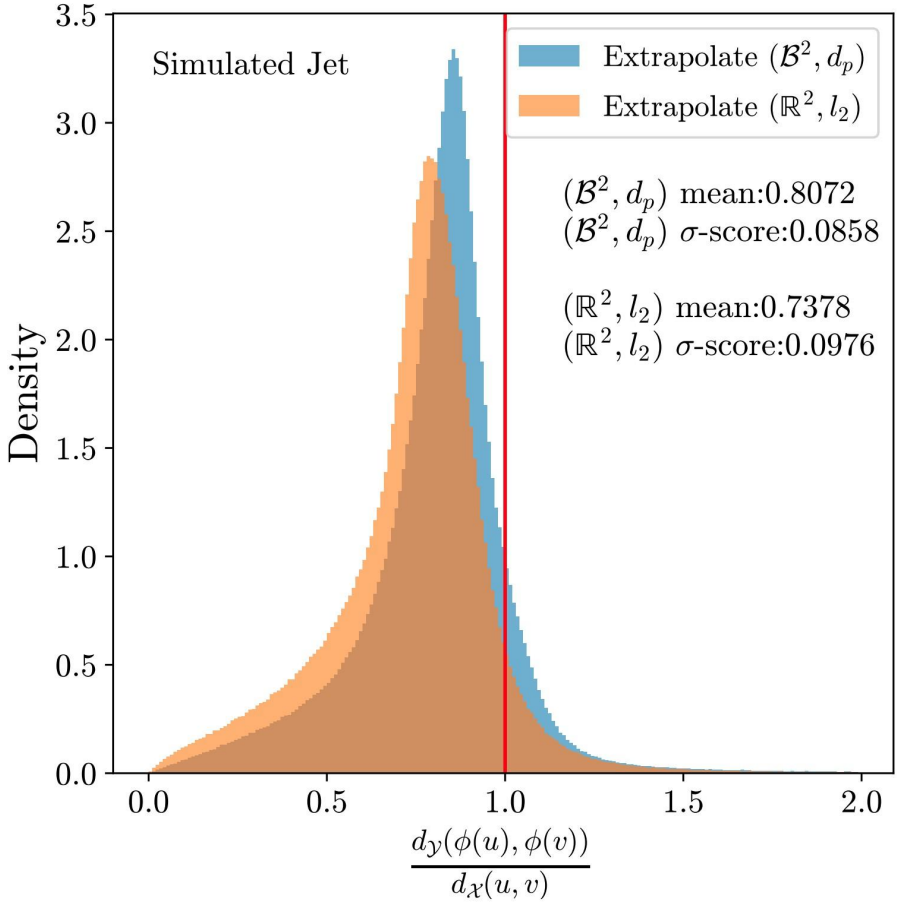
Does It Work ?





Almost falls on y = x line
Result on simulated QCD Jets
Neural network is extrapolating, and also just in 2 dimensions

Quantitative measurement of distortion



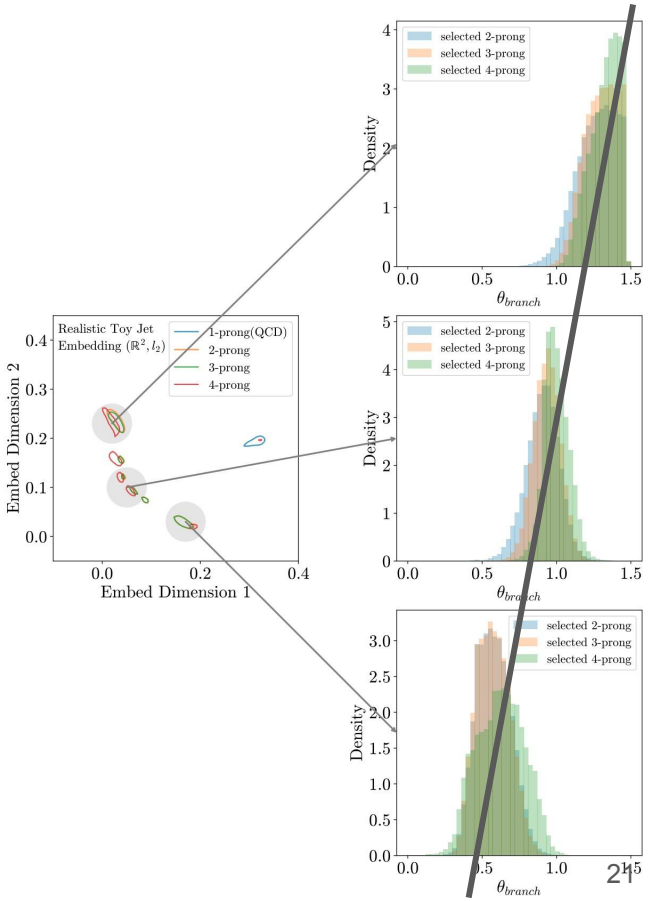
Does it learn the latent structure?

$$\phi_{\theta,\text{Transformer}} : (\mathcal{X}_{\text{jets}} \subset \mathbb{R}^{48}, d_{\text{EMD}}) \to (\mathbb{R}^2, l_2)$$

For different regions of the embedded space, plot the first parton splitting angle

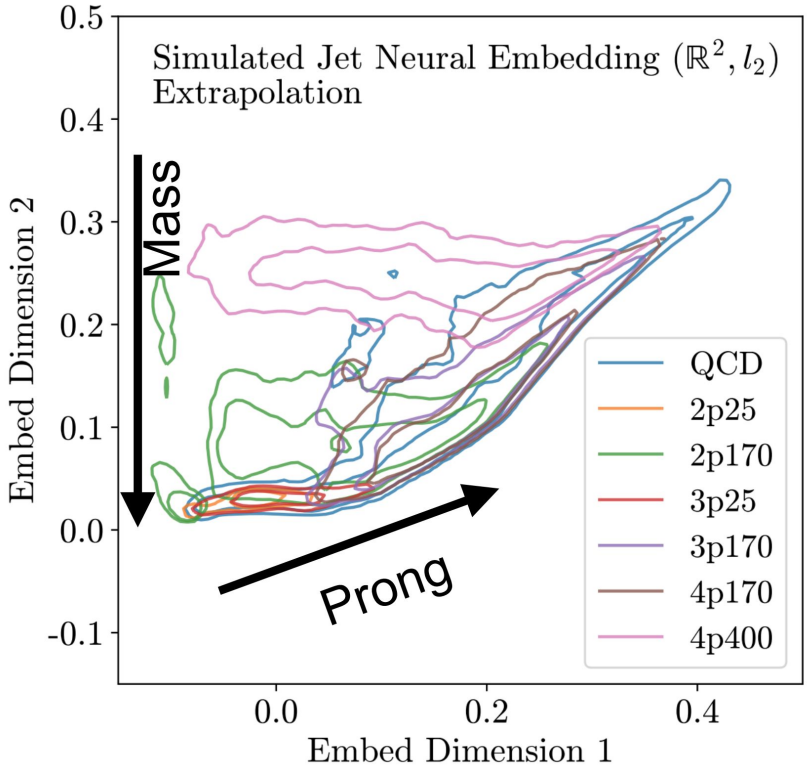
Embedding does learn the first splitting angle very well

Passes the closure test (learns the latent structure very well)

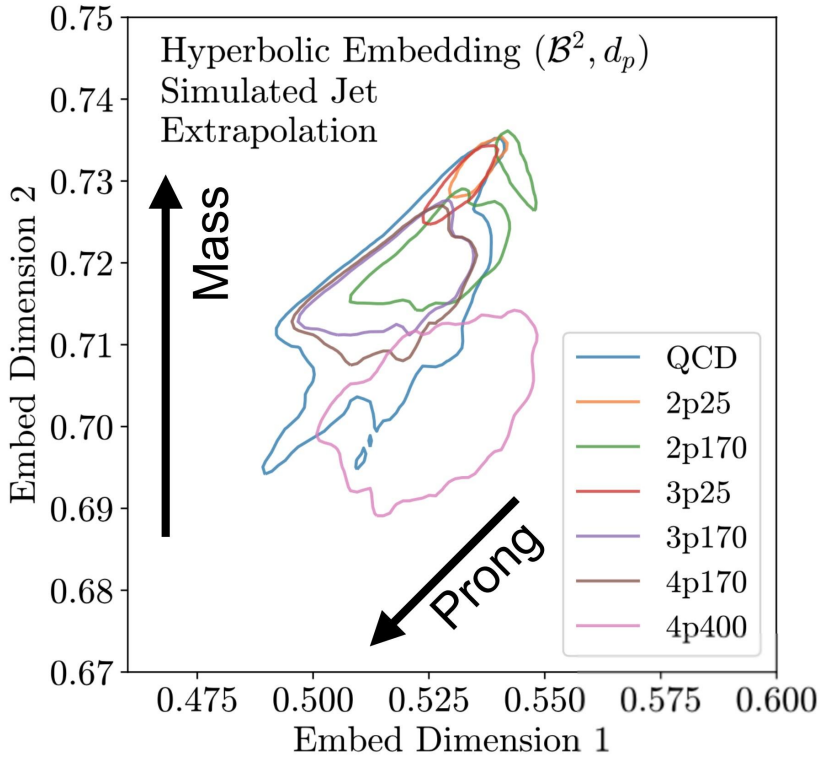


Embedding with simulated jets

$\phi_{\theta,\text{Transformer}} : (\mathcal{X}_{\text{jets}} \subset \mathbb{R}^{48}, d_{\text{EMD}}) \to (\mathbb{R}^2, l_2)$



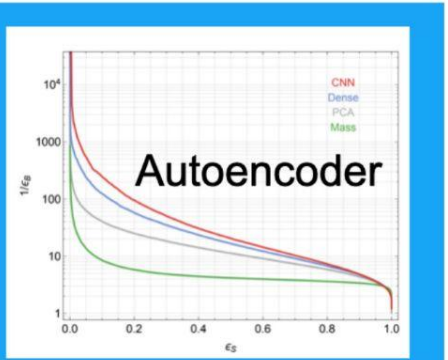
$\phi_{\theta,\text{Transformer}} : (\mathcal{X}_{\text{jets}} \subset \mathbb{R}^{48}, d_{\text{EMD}}) \to (\mathcal{B}^2, d_p)$

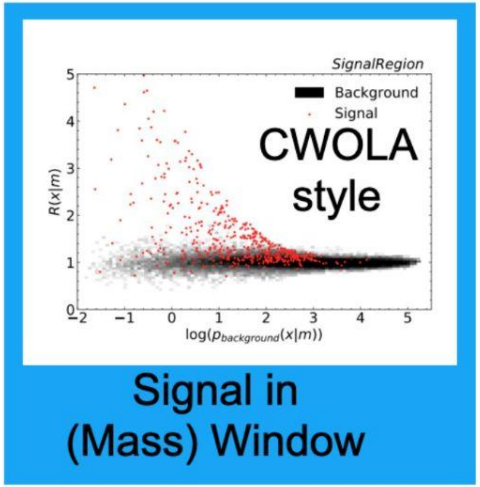


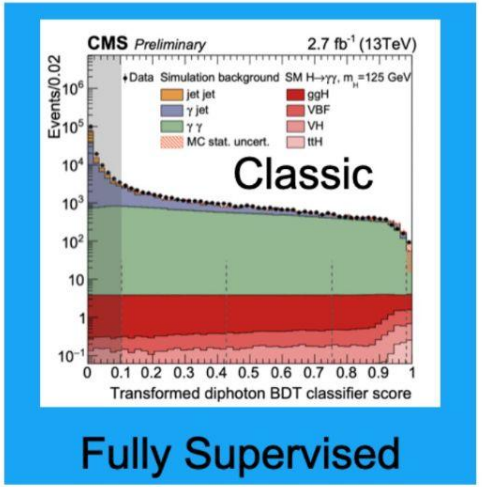
Model Agnostic Searches @ the LHC

100s of algorithms, a big paradigm shift

Prior Free → Fully Supervised







Only use background
ex) AE trained on background

Use both background, specific signal
ex) BDT, Supervised ML algorithms S vs B

Application to anomaly detection algorithms

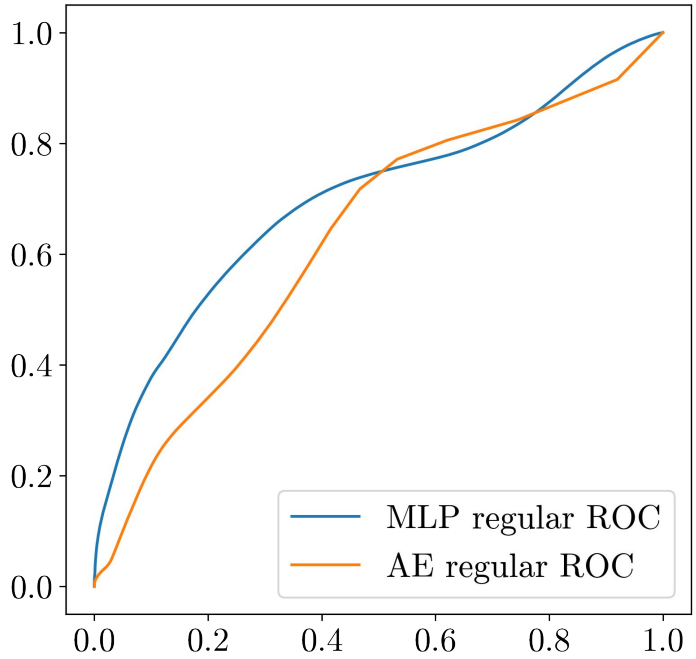
Quantifying different anomaly detection algorithms is hard, and is missing from the studies

Typically: Compare different algorithms by making ROC curves with a **specific evaluation dataset**

It doesn't tell the full story of how wide the algorithm searches the phase space & is highly dependent on the selection of the evaluation dataset

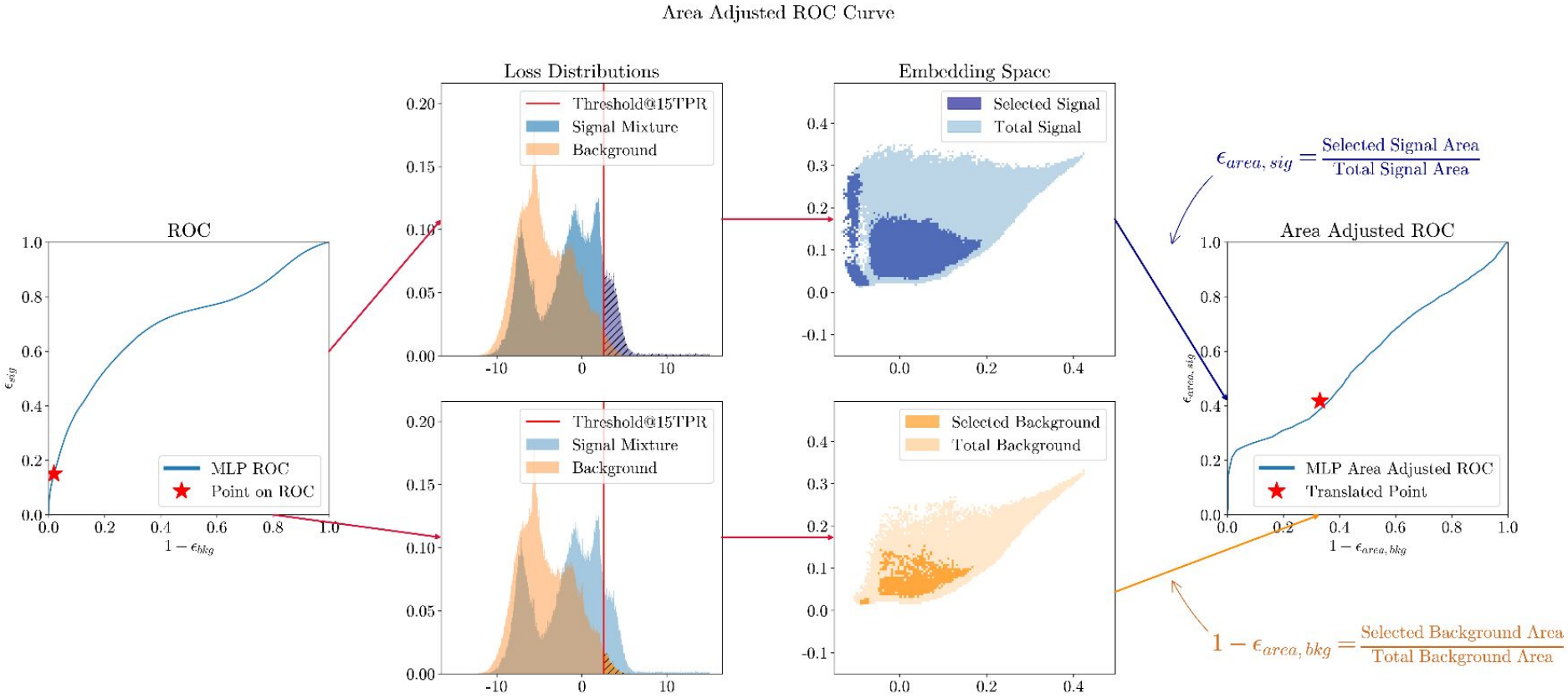
Use the **notion of n-volume** in **n-dimensional embedded euclidean** space!

Signal: 2prong 170GeV Jet, Background: QCD

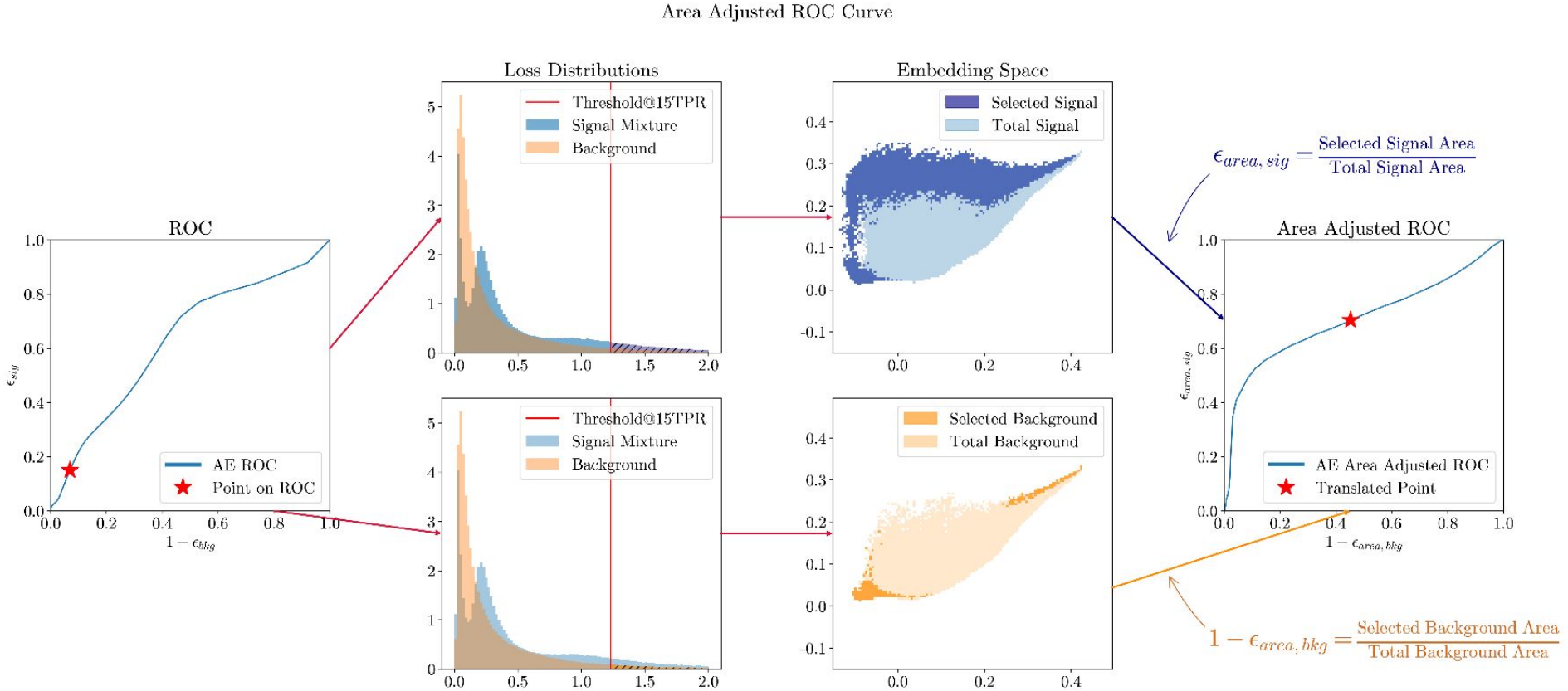


Fully supervised algo vs Unsupervised

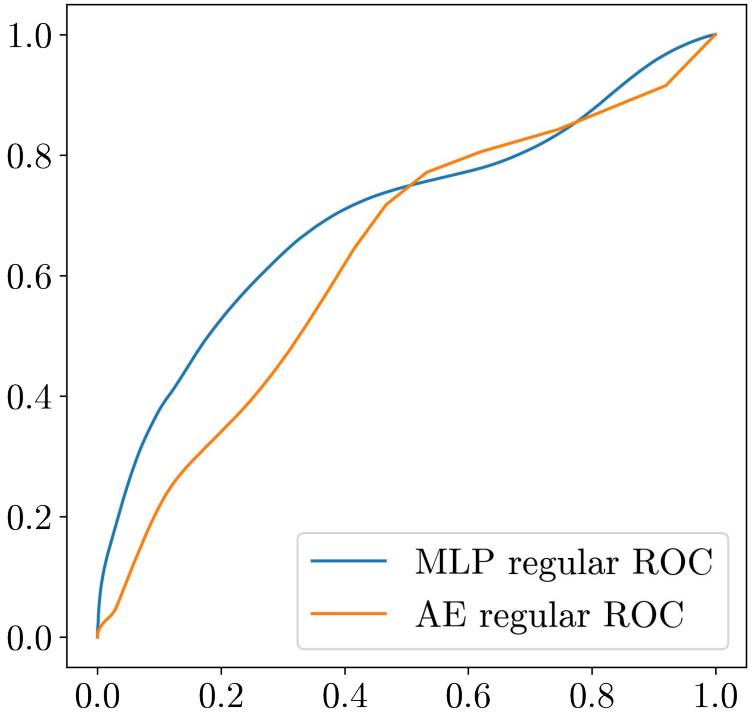
Volume Adjusted ROC Curve - MLP (Fully Supervised)

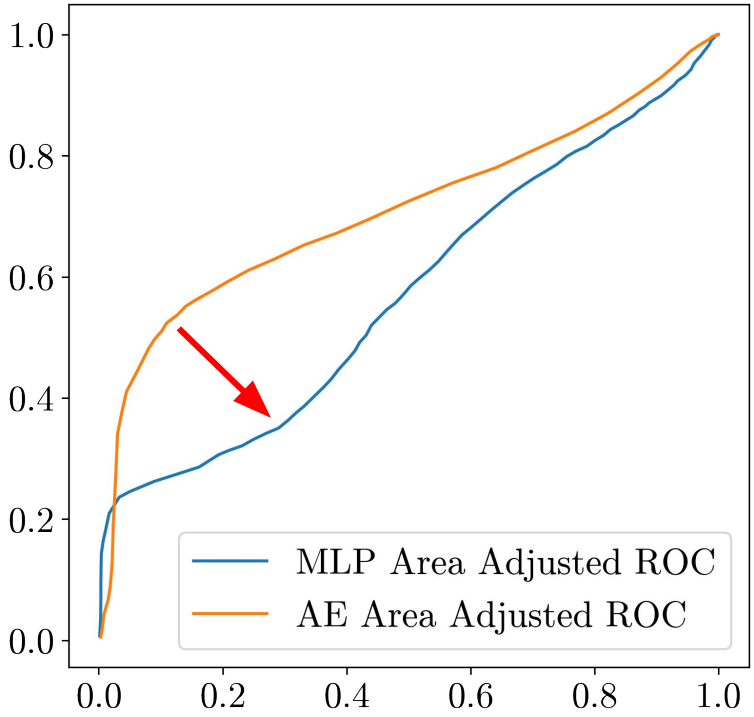


Volume Adjusted ROC Curve (Autoencoder - Unsupervised)

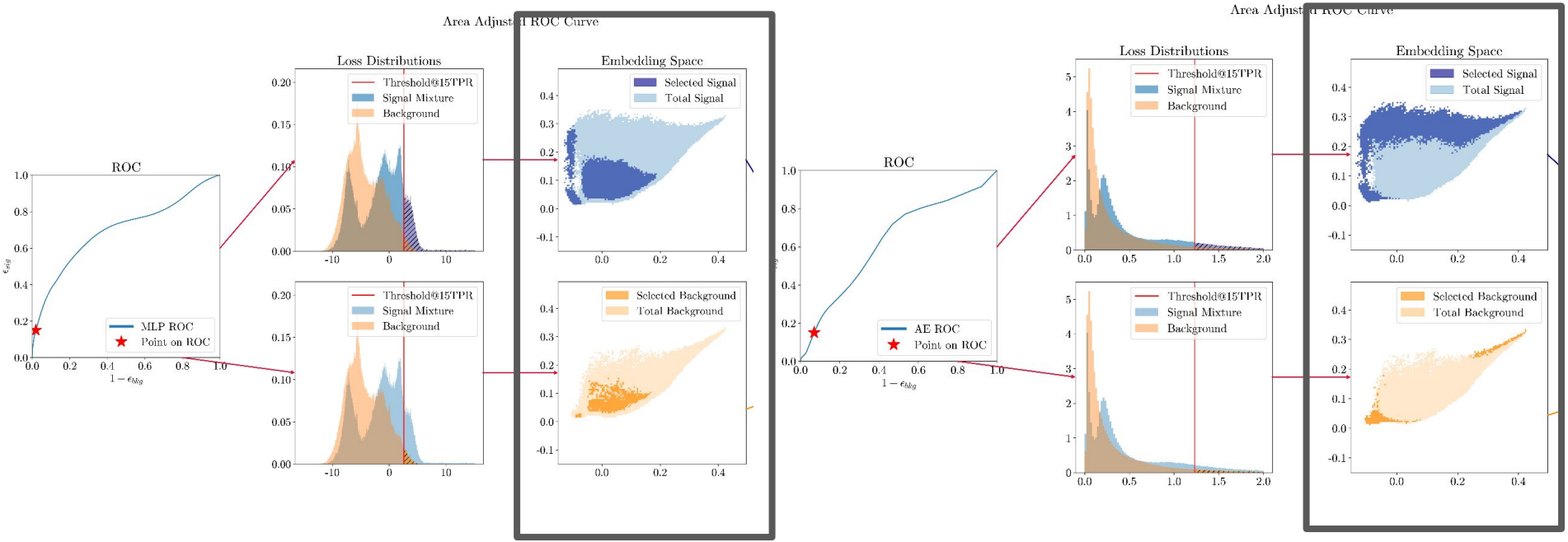


Regular ROC vs Area-adjusted ROC Curve





Visualizing the space that each method probes



Reducing the evaluation dataset dependence

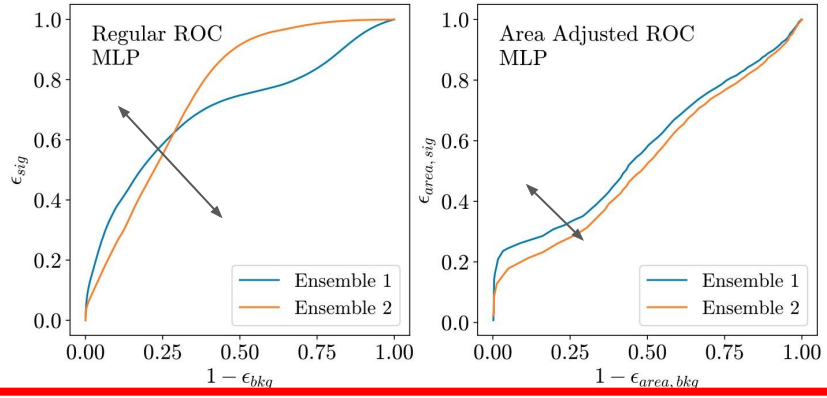
Ensemble 1 vs ensemble 2 Comparison

For normal ROC curves, they move around a lot

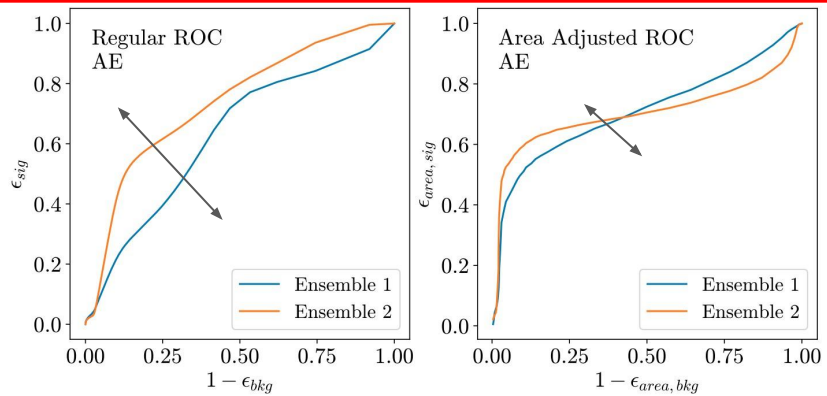
Area adjusted ROC is more stable

Gives more freedom to choose evaluation dataset without worrying about robustness

MLP, Supervised



AE, Unsupervised



Conclusion

Introduced general framework : Extended in lots of exciting ways

Interpretability of the space allows us to solve many big problems -

First attempt at quantifying anomaly detection algorithms

Embedding **unlocks full potential of manifold properties** of the collider events

Please check out https://arxiv.org/abs/2208.05484

Backup

Distance preserving embedding

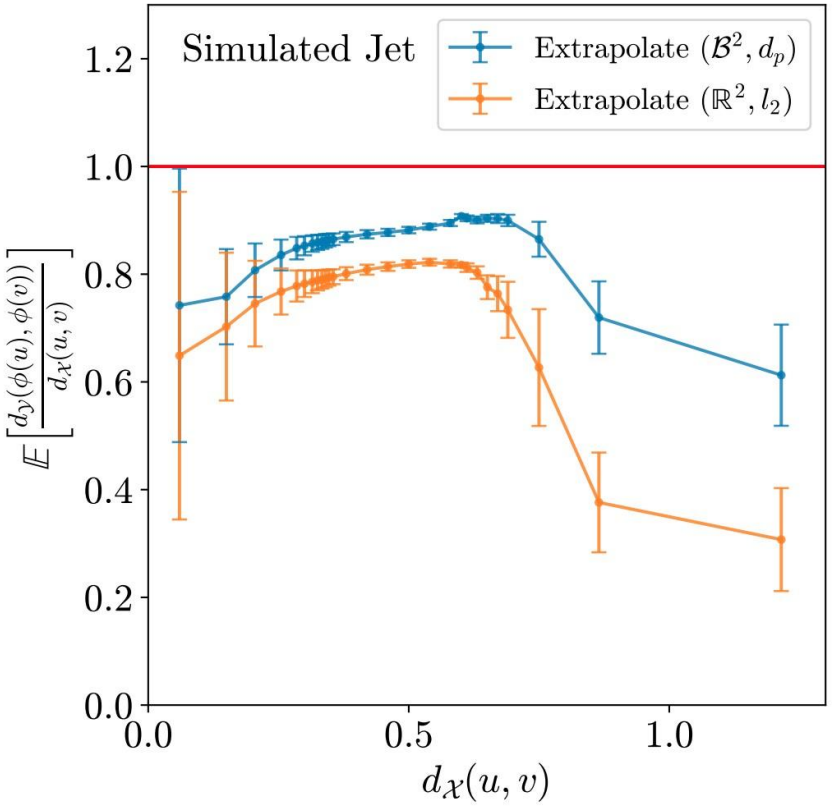
Want to embed original space to some other space $\phi : (\mathcal{X}, d_{\mathcal{X}}) \rightarrow (\mathcal{Y}, d_{\mathcal{Y}})$

You generally want to embed into lower dimensional space

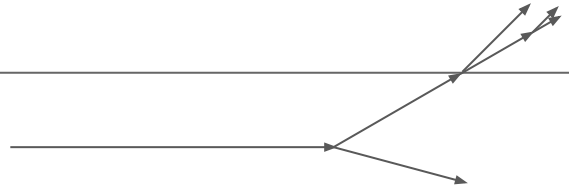
$$\forall u, v \in \mathcal{X},\ L \cdot d_{\mathcal{Y}}(\phi(u), \phi(v)) < d_{\mathcal{X}}(u, v) < C \cdot d_{\mathcal{Y}}(\phi(u), \phi(v))$$

With reasonable L, C values

Hyperbolic Embedding



Toy Jet Generator



View jets as iterative splitting of the original parton(Simplified picture)

Motivation : Probe whether embedding learns the proper latent variables

Splitting angle drawn from some prior distribution

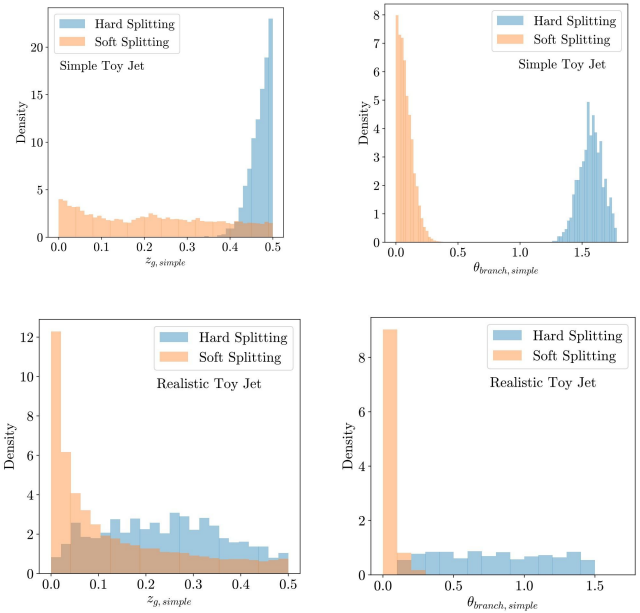
Different distributions for hard, soft splitting

Iteratively split the energetic parton, in order to make N prong jets we first perform N-1 hard splittings, and then do the soft splittings till we reach the desired number of particles

In the end = collection of 16 particles, N prong jets go through N-1 hard splittings , then the rest soft splittings till we have 16 particles

Generated with same mass and pT.

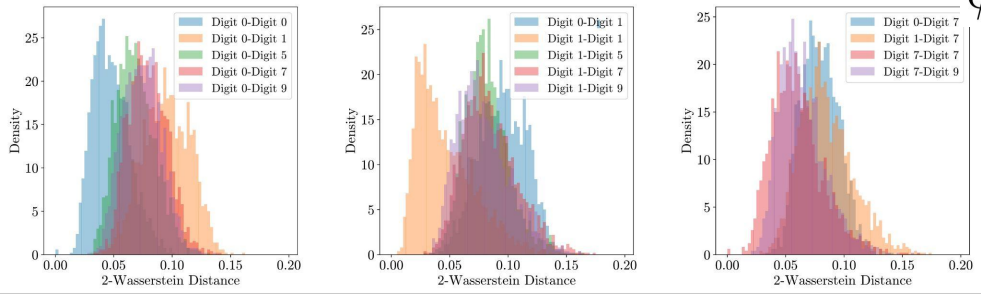
We have the data generating process, and access to variables we don't have in real jet data (ex. first splitting angle)



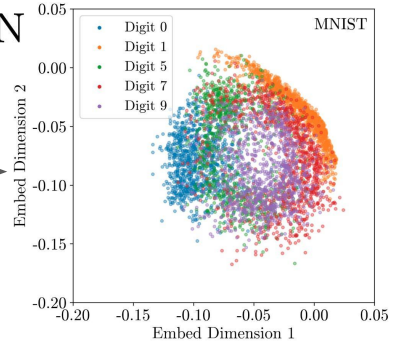
$$z_g = \frac{\max p_{\mathrm{T},1}, p_{\mathrm{T},2}}{p_{\mathrm{T},1} + p_{\mathrm{T},2}}$$

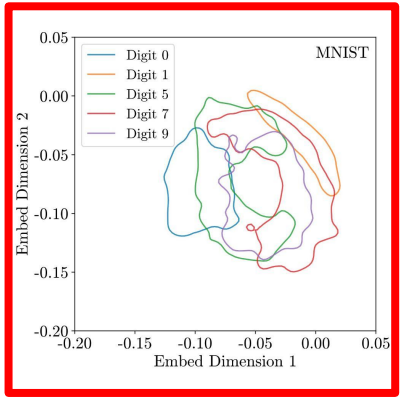
Presentation of Results

$$\phi_{\theta,\mathrm{CNN}} : (\mathcal{X}_{\mathrm{digits}} \subset \mathbb{R}^{784}, \mathcal{W}_2) \to (\mathbb{R}^2, l_2)$$

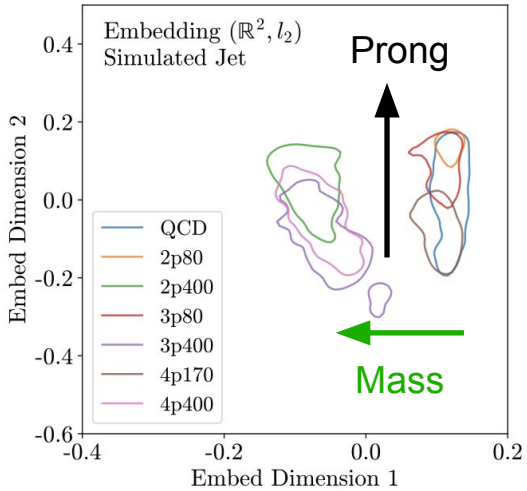


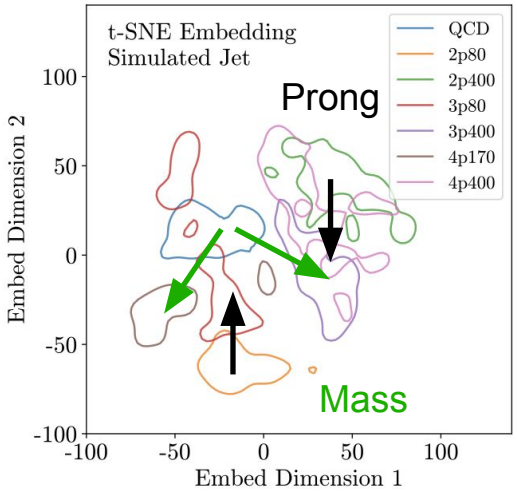
$\phi_{\theta,\mathrm{CNN}}$

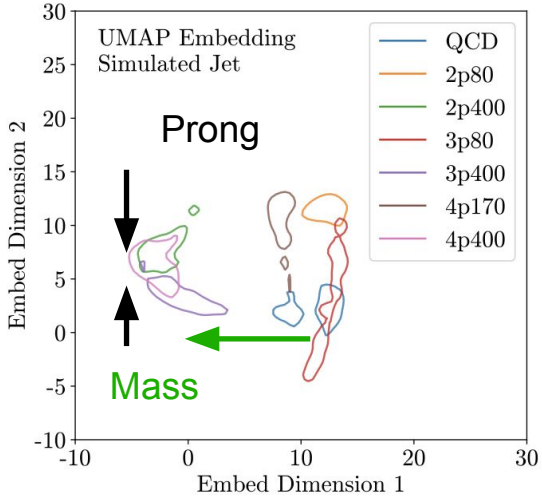




Other Manifold Learning methods

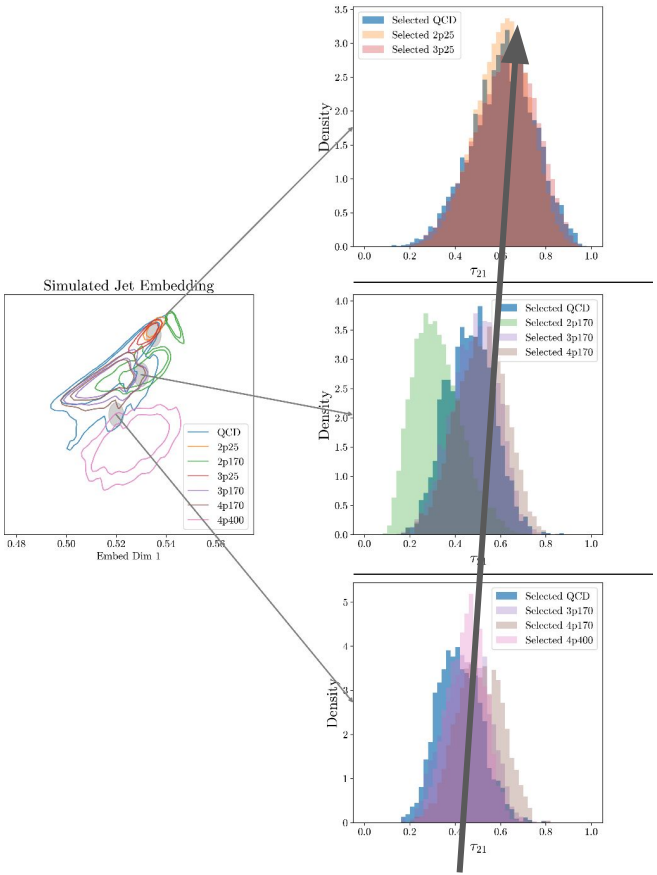






1. Distance in UMAP and tSNE space is heavily morphed : uninterpretable
2. They do not have the nice scaling property that neural embedding has (parallel evaluation and linear scaling)
3. tSNE and UMAP has to fit every time dataset changes, whereas embedding you train once and evaluate on any arbitrary dataset

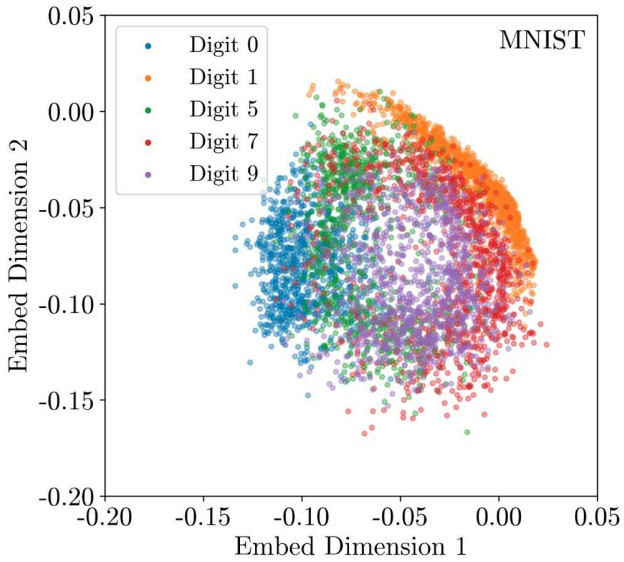
Hyperbolic Embedding

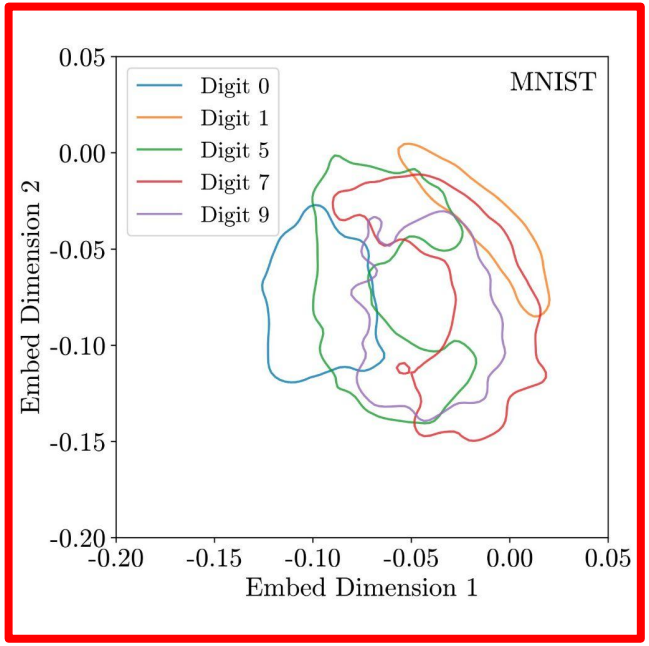


We can examine the observable such as n-subjettiness for different regions of the embedded space

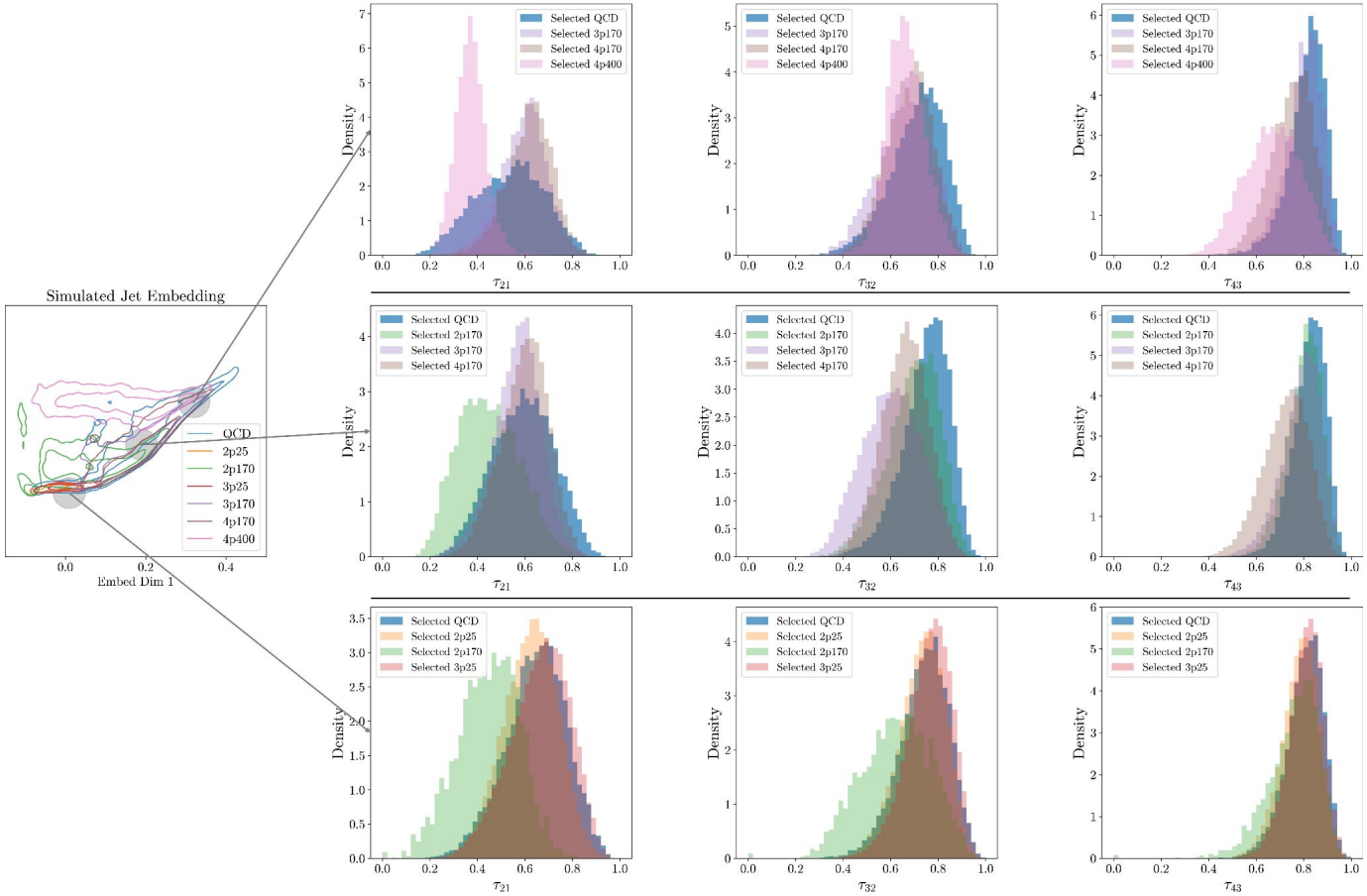
Can observe a strong correlation for τ_{21}

How results are presented





Does it learn the latent structure?



Transformer Networks

Originally developed for seq2seq modeling in NLP

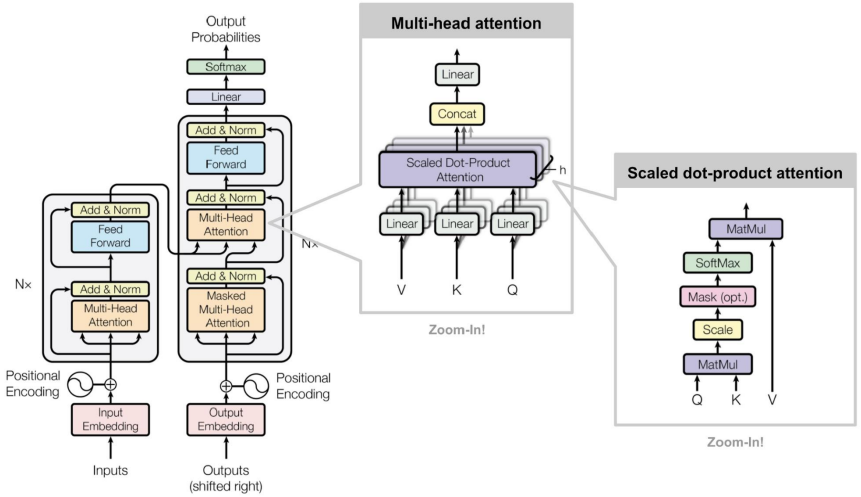
(Attention is all you need, 2017)

Quickly realized they are useful for other tasks (ViT, Video, Biological Sequencing)

Leading SOTA development & benchmarks in many ML tasks (BERT, GPT3, ROBERTa)

Best performing collider physics tagger also uses transformers (2M parameters)

- ParticleTransformer(2202.03772)



https://lilianweng.github.io/posts/2020-04-07-the-transformer-family/

Transformer Networks

Relies on **attention mechanism** : Basically a dictionary lookup made differentiable

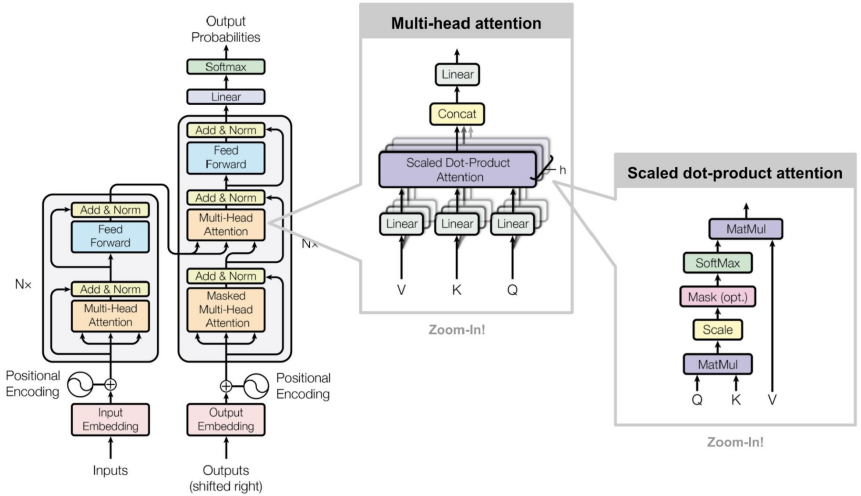
You have key, query, value weight matrices, with multiplication input tensor you get K, Q, V tensors

Compare query to each key, then retrieve convex combination of the values f((Q,K), V)

How you combine them gives a flavor of attention (like choosing kernel for kernel based ML methods)

Multiple attention heads capture different notions of similarity between inputs (input particles)

Attention can capture relationship between any pair of input (particles) : Equivalence between fully connected graph NN



https://lilianweng.github.io/posts/2020-04-07-the-transformer-family/

Details on the Transformer Architecture

16(particles) by 3 (pt, eta, phi) input tensors go through dense embedding layer, Maps 3 features to 32 features

Treat both **number of attention heads** per encoder layer and **number of encoder layers** as hyperparameters and do the scan

In each attention head, use scaled dot-product attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Then we stack N heads together, then this tensor goes through a series of dense nets to the desired dimension (of the embedded space)