A New Approach to Unsupervised Learning in Jet Physics

Contrastive Learning of Representations

Peter Sorrenson

July 6, 2021

Heidelberg Collaboratory for Image Processing &
Institute for Theoretical Physics
University of Heidelberg

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

'Contrastive learning of jet observables', hep-ph/2107.soon

Barry M. Dillon, Gregor Kasieczka, Hans Olischlager, Tilman Plehn, Peter Sorrenson, and Lorenz Vogel

Outline

1. Representation learning

2. Contrastive learning of jet observables

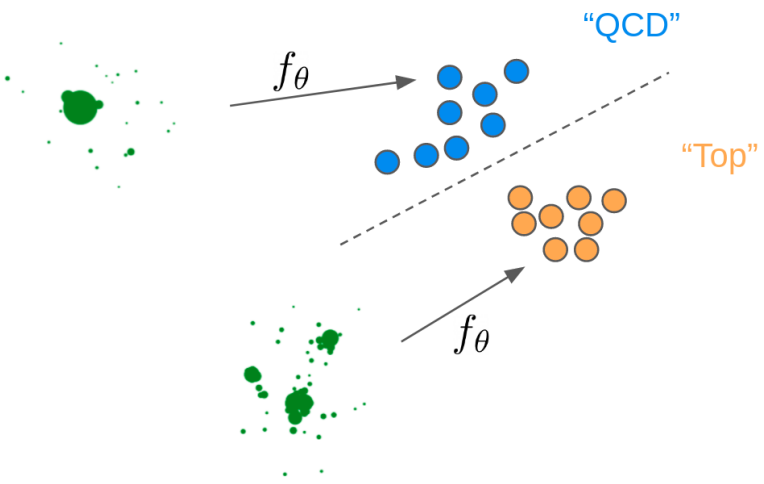
3. Summary

1. Representation learning

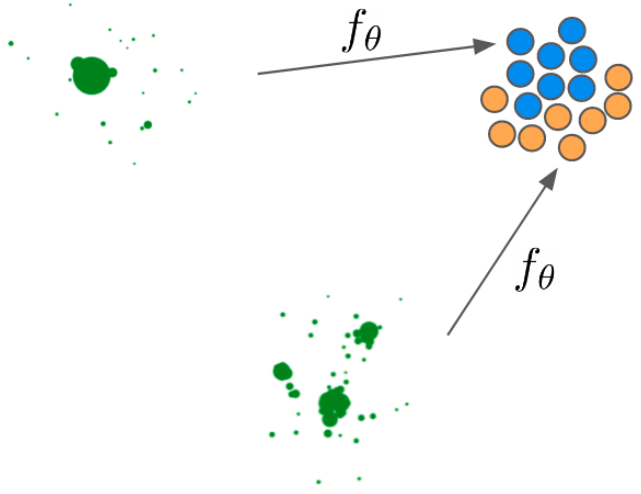
2. Contrastive learning of jet observables

3. Summary

What is a representation?



Last layer of a classifier

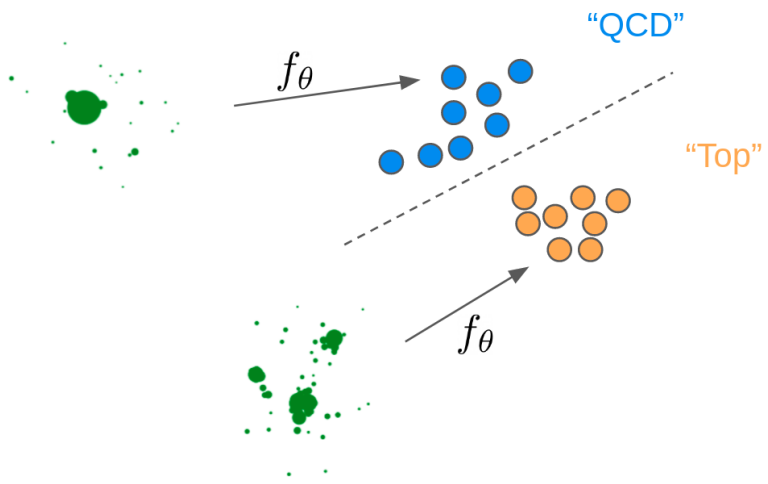


Latent space of a VAE

A representation is a vector of features which are used as input for some set of tasks.

The term is usually applied to the output of some network or function, but raw/preprocessed inputs can also be regarded as representations (e.g. (p_T, η, ϕ) contituents, jet images).

What is it good for?

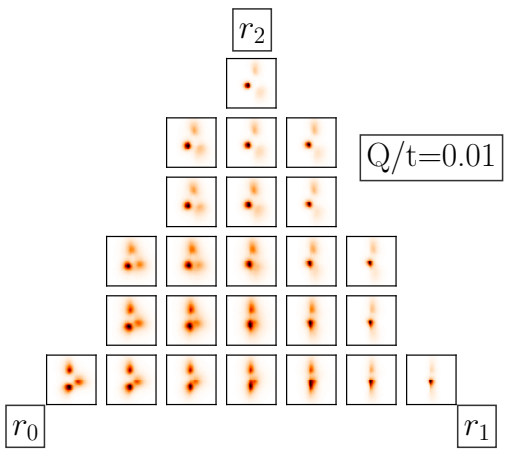


Last layer of a classifier

- Representation allows classification with very simple (linear) function
- Useful for learning another classifier with few labeled training examples: append a small head network and finetune

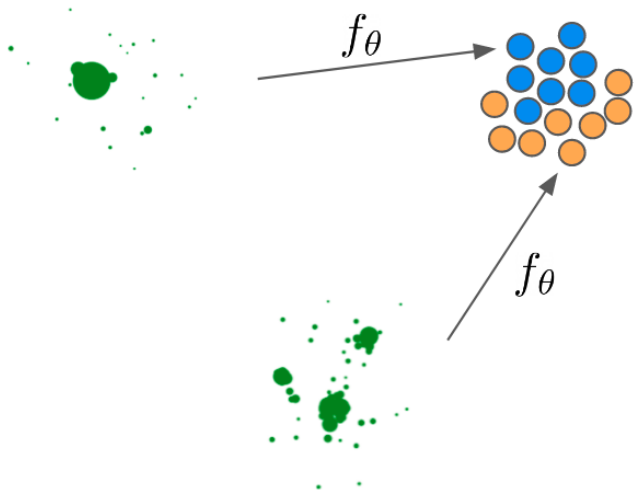
What is it good for?

- Representation may encode high-level conceptual information which can be interpreted or manipulated



Latent space of a DVAE trained on QCD and top jets

"Better Latent Spaces for Better Autoencoders", Dillon et al., 2021



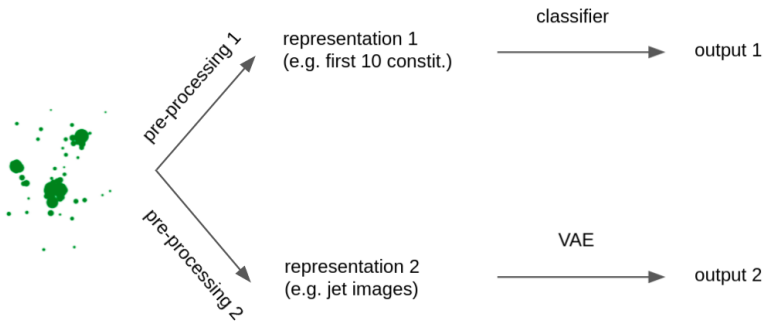
Latent space of a VAE

Why learn a representation?

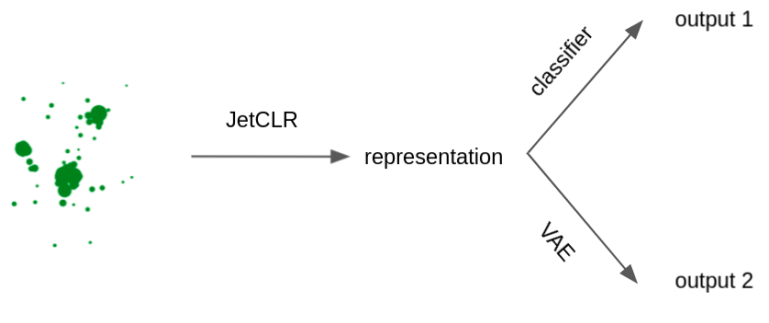
One of the big stories in ML is that learned features beat hand-engineered ones.

Recall: A representation is a vector of features which are used as input for some set of tasks.

Therefore, we should try to learn the representation.



Example traditional learning pipeline to achieve several tasks on the same data

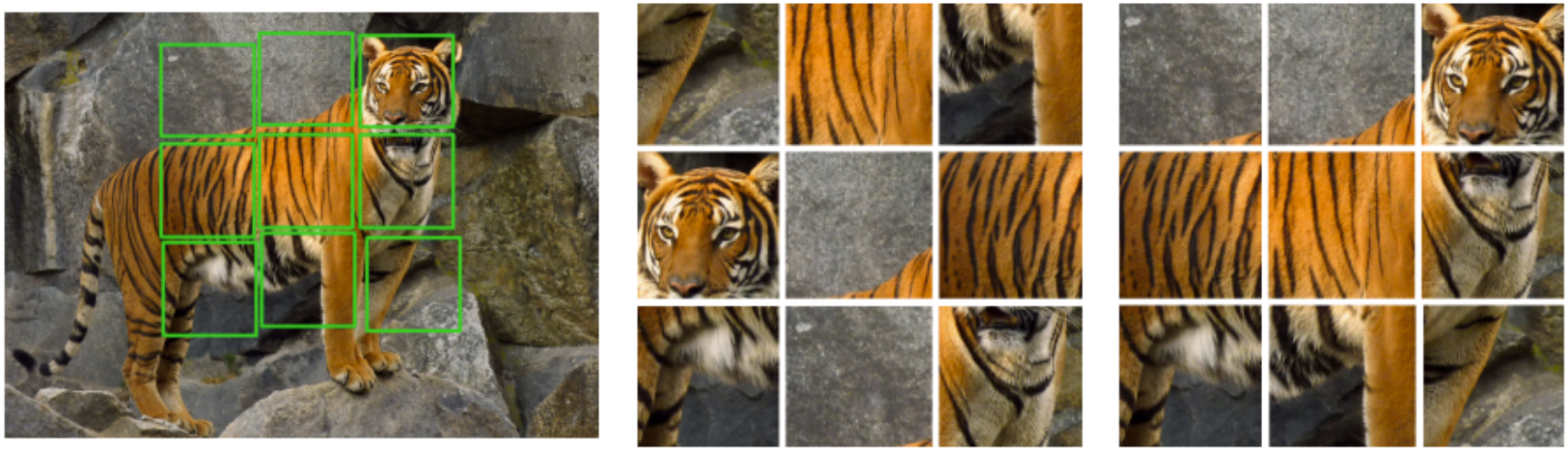


Example representation learning approach: once the representation is learned, the additional learned functions (e.g. classifier, VAE) can be much simpler

How do we learn a representation?

Self-supervised learning: Set up a non-trivial task that can be constructed from unlabeled data. Often a classification task with constructed labels.

Example: Jigsaw puzzle

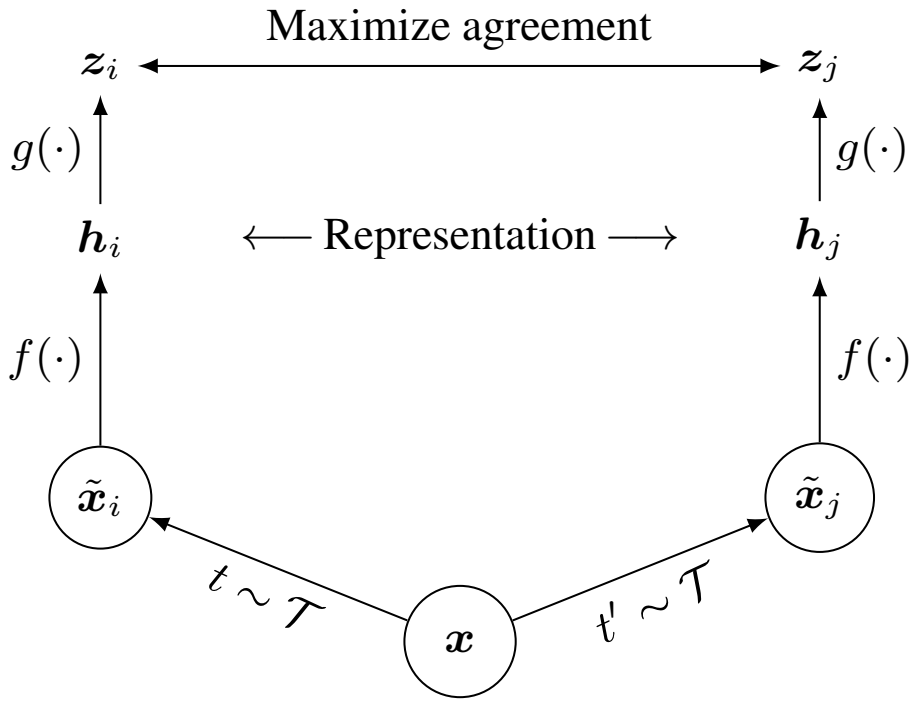


A picture is divided into 'jigsaw pieces'. The network has to learn how to correctly arrange the pieces. In doing so, it learns to extract relevant features from the data.

"Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles", Noroozi and Favaro, 2016

How do we learn a representation?

Contrastive learning: A type of self-supervised learning where the network has to learn to match similar examples and push apart dissimilar examples. Pairs of similar examples are created by different transformations (augmentations) of the same data.



SimCLR: "A Simple Framework for Contrastive Learning of Visual Representations", Chen et al., 2020

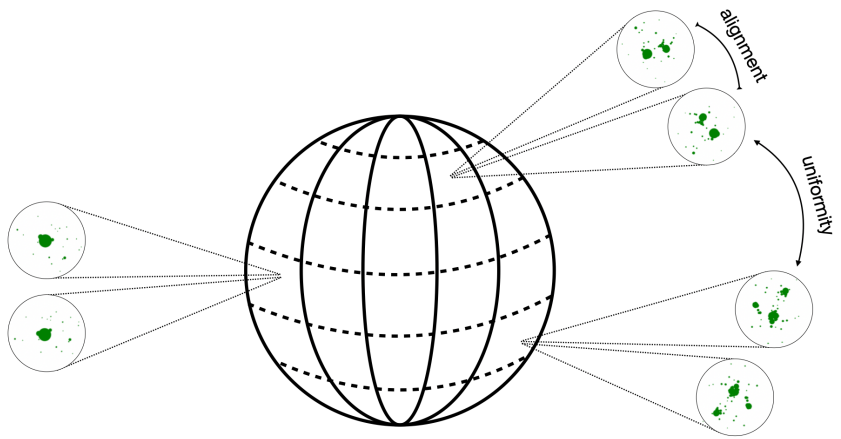
How do we learn a representation?

Define: $z = f_\theta(x)$ is the output of the network, τ is a temperature parameter and sim is the cosine similarity: $\text{sim}(z_i, z_i') = \frac{z_i \cdot z_i'}{\|z_i\| \|z_i'\|}$

Loss function: normalized temperature-scaled cross-entropy (NT-Xent)

$$\mathcal{L}_i = -\log \frac{\exp(\text{sim}(z_i, z_i')/\tau)}{\sum_{j \in \text{batch}} \mathbb{I}_{i \neq j} \left[\exp(\text{sim}(z_i, z_j)/\tau) + \exp(\text{sim}(z_i, z_j')/\tau)\right]}$$

- alignment: $\text{sim}(z_i, z_i') \to 1$, numerator is maximised
- denominator minimised when z_i distributed uniformly on the hyper-sphere



1. Representation learning

2. Contrastive learning of jet observables

3. Summary

Which augmentations are relevant to jet physics?

The network will learn to be invariant to the augmentations applied (under perfect convergence). This means we can impose physically motivated symmetries:

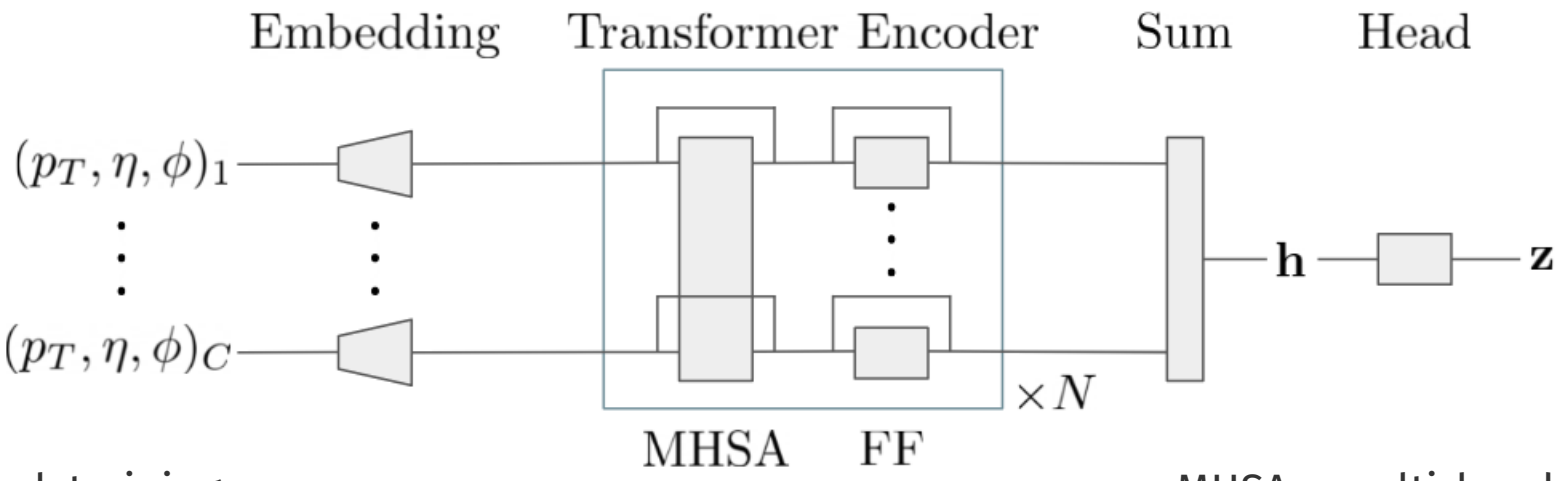
- Rotation around the jet axis
- Invariance to permutations of constituents

Two ways to impose (approximate) symmetries: through augmentations and through model constraints. We choose to use a permutation invariant network and impose rotational symmetry through augmentations. Note that we always need to impose some invariances through augmentations in order to do contrastive learning at all.

Permutation Invariance: transformer network

We use a transformer network whose output is **invariant to the constituent ordering**.

Similar to Deep-Sets/Energy-Flow-Networks: arXiv:1810.05165, P. T. Komiske, E. M. Metodiev, J. Thaler



Network training:

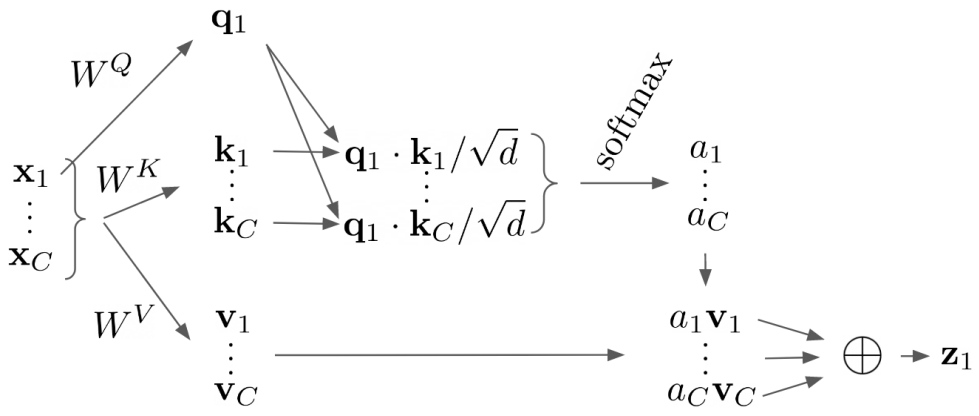
1. sample batch of jets $\{x_i\}$
2. create augmented batch of jets $\{x_i'\}$
3. forward-pass both batches through the network
4. compute $\mathcal{L}$ and derivatives and back-propagate
5. representation is taken before the head network

MHSA = multi-headed self-attention

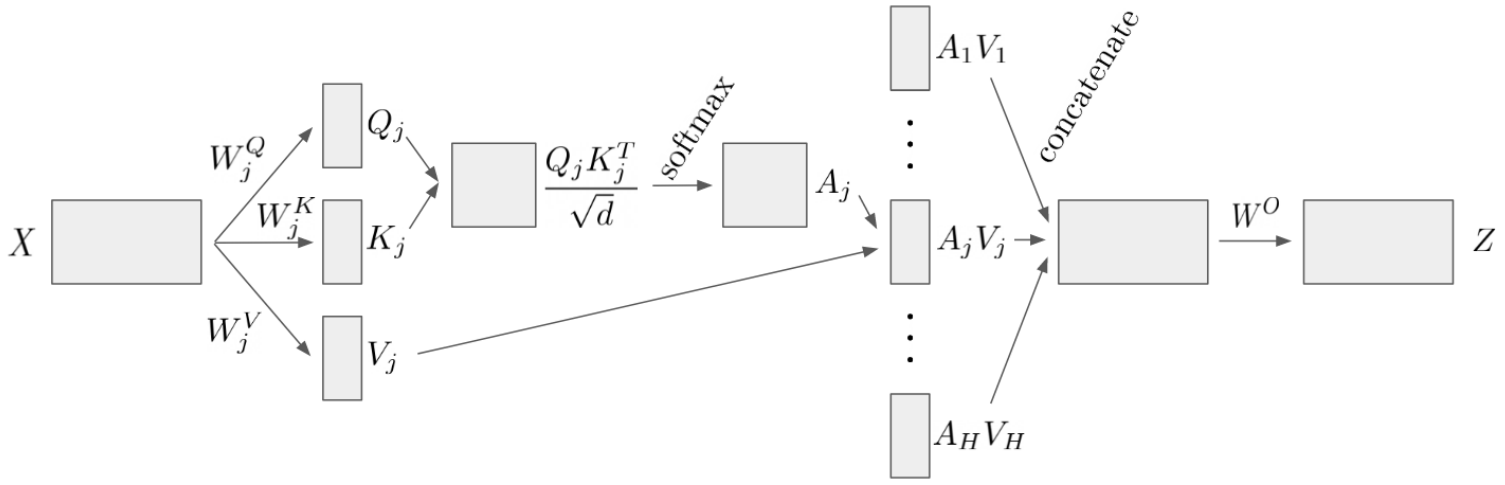
FF = feed forward (fully connected)

The transformer network can accept variable-length inputs and supports masking

Network detail: multi-headed self-attention



Self-attention applied to the single sequence element $\mathbf{x}_1$



Multi-headed self-attention. Each box represents a tensor

Evaluating performance: linear classifier test

In order to evaluate the performance of our unsupervised representation learning method, we test how well a supervised linear classifier performs on the representation.

It reflects the idea that a good representation should encode relevant information about the inputs roughly linearly.

This test is an imperfect proxy for the quality of the representation, but is commonly used in the representation learning literature.

Linear classifier test results

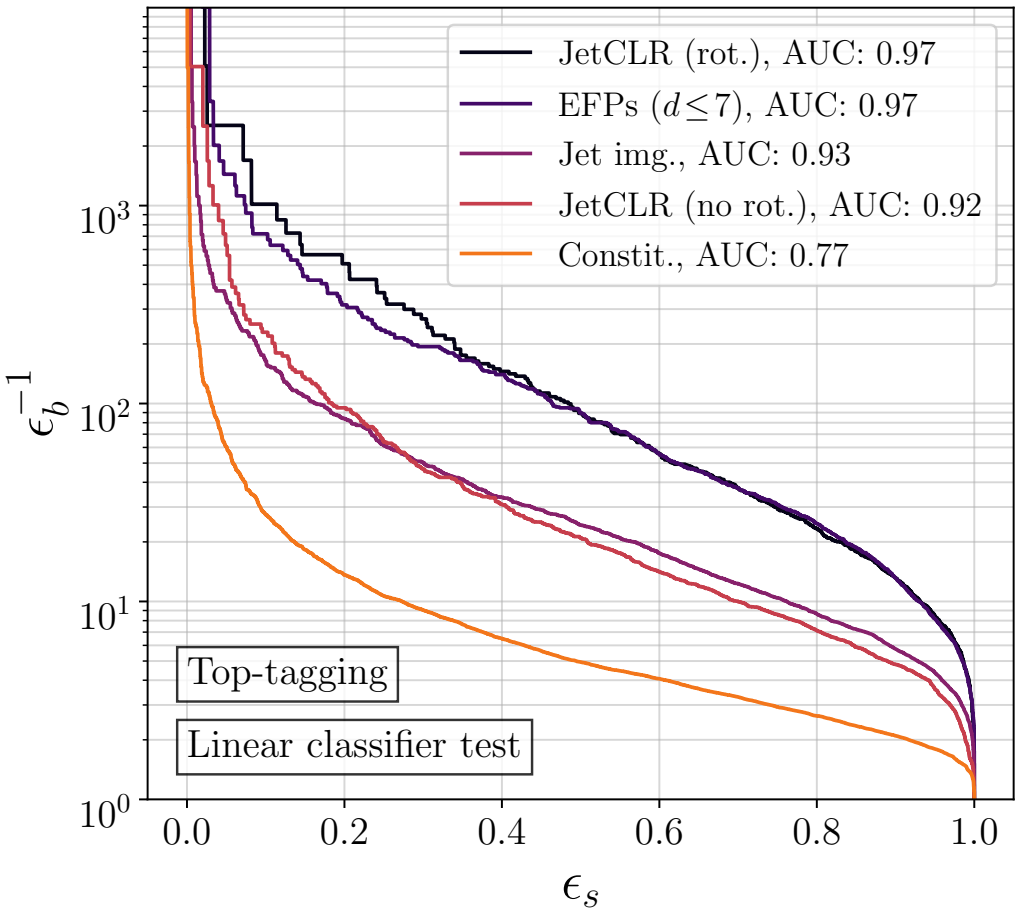
We call this method JetCLR

code to come with paper

The representation is 1000 dimensional

Data: 100k top and 100k QCD. Train/test split = 90/10

The unsupervised JetCLR repr. performs just as well as energy flow polynomials (EFPs), using only the contrastive learning concept, permutation inv and rotational inv



1. Representation learning

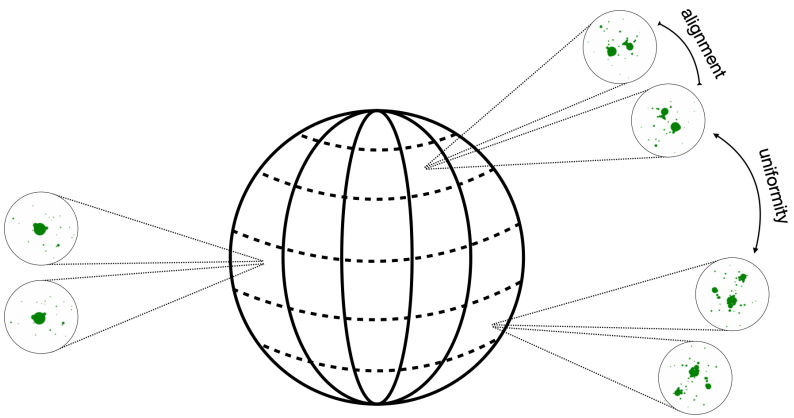
2. Contrastive learning of jet observables

3. Summary

Summary

Contrastive Learning

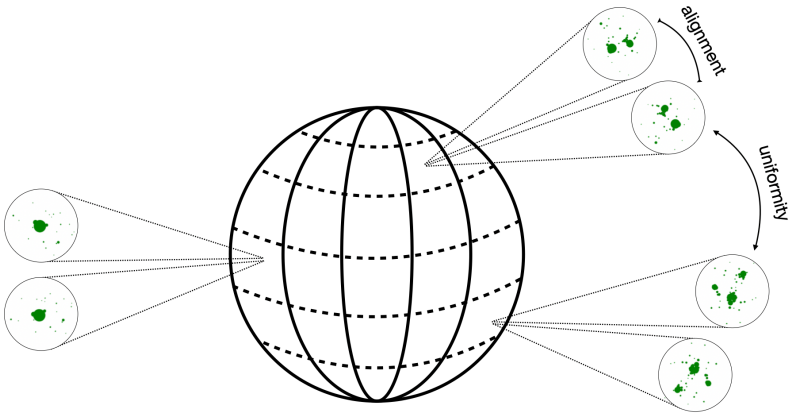
Unsupervised representation learning using data augmentations (symmetries/invariances).
Learns to pull together similar inputs and push apart dissimilar ones.



Summary

Contrastive Learning

Unsupervised representation learning using data augmentations (symmetries/invariances).
Learns to pull together similar inputs and push apart dissimilar ones.



JetCLR

Unsupervised learning of jet observables with contrastive learning and symmetries.

paper/code $\longrightarrow$ soon

JetCLR can already incorporate other data: tracking info, particle ID, etc, all while retaining the rotational and permutation invariance of the constituents.