Self-supervision in particle physics

Data- and symmetry-driven definition of observables using machine learning

Barry M. Dillon

May 9, 2022

Institute for Theoretical Physics
University of Heidelberg

Pheno 2022

Symmetries, Safety, and Self-Supervision, hep-ph/2108.04253

BMD, Gregor Kasieczka, Hans Olischlager, Tilman Plehn, Peter Sorrenson, and Lorenz Vogel

UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.

1. Jet physics & ML

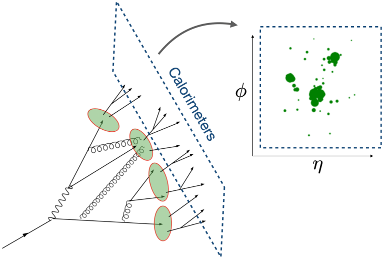
2. Self-supervision

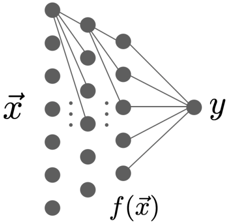
3. Results

4. Outlook

Top-tagging with machine-learning

Neural network maps kinematical data to a predicted label (supervised)





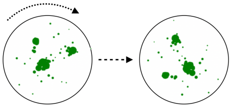
- simulations provide training data $\{\vec{x}_i\}$ and truth-labels $\{y_i'\}$
- neural network is optimised to minimise a loss function
- loss function is minimised when QCD and top jets are well-separated in y
- predicted label is a new observable used to tag top-jets

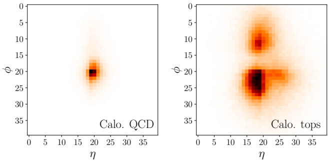
Learning physical quantities

Neural networks $\Rightarrow$ inductive bias

i.e. implicit assumptions made by the network on mapping input $\to$ output

- $\to$ neural nets are not invariant to physical symmetries in data
- $\to$ we typically try to solve this through 'pre-processing'





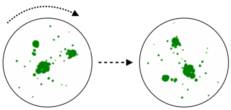
Learning physical quantities

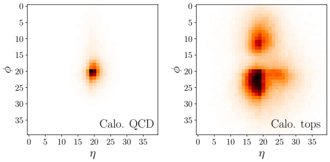
Neural networks ⇒ inductive bias

i.e. implicit assumptions made by the network on mapping input → output

→ neural nets are not invariant to physical symmetries in data

→ we typically try to solve this through 'pre-processing'





Our goal: control the training to ensure we learn physical quantities

→ rotational & translation invariant, permutation invariant, IRC safe

→ deep neural networks can never be completely interpretable

... but we can place limits on what they can learn

1. Jet physics & ML
2. Self-supervision
3. Results
4. Outlook

Self-supervision - contrastive learning

Key idea

Reframe the definition of our observables as an optimisation problem to be solved with machine-learning

What do we fundamentally want from observables?

1. invariance to certain transformations / augmentations of the jets
2. discriminative within the space of jets

Self-supervision - contrastive learning

Key idea

Reframe the definition of our observables as an optimisation problem to be solved with machine-learning

What do we fundamentally want from observables?

1. invariance to certain transformations / augmentations of the jets
2. discriminative within the space of jets

From the dataset of jets $\{x_i\}$ define:

- positive-pairs: $\{(x_i, x_i')\}$ where x_i' is an augmented version of x_i
- negative-pairs: $\{(x_i, x_j)\} \cup \{(x_i, x_j')\}$ for $i \neq j$

Augmentation: any transformation (e.g. rotation) of the original jet

Self-supervision $\Rightarrow$ training using 'pseudo-labels'

Self-supervision - contrastive learning

Train a neural net to map constituents to a high-dim representation, $f : \mathcal{J} \to \mathcal{R}$

Optimise the mapping for:

1. alignment: positive-pairs close together in $\mathcal{R} \Rightarrow$ invariance
2. uniformity: negative-pairs far apart in $\mathcal{R} \Rightarrow$ discriminative

Self-supervision - contrastive learning

Train a neural net to map constituents to a high-dim representation, $f:\mathcal{J}\to\mathcal{R}$

Optimise the mapping for:

1. alignment: positive-pairs close together in $\mathcal{R} \Rightarrow$ invariance
2. uniformity: negative-pairs far apart in $\mathcal{R} \Rightarrow$ discriminative

Contrastive loss:

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

Self-supervision - contrastive learning

Train a neural net to map constituents to a high-dim representation, $f : \mathcal{J} \to \mathcal{R}$

Optimise the mapping for:

1. alignment: positive-pairs close together in $\mathcal{R} \Rightarrow$ invariance
2. uniformity: negative-pairs far apart in $\mathcal{R} \Rightarrow$ discriminative

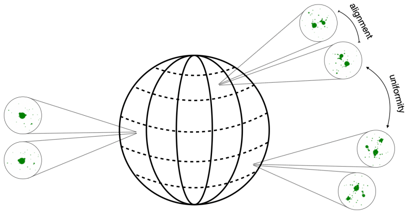
Contrastive loss:

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

Similarity measure in $\mathcal{R}$:

$s(z_i, z_j) = \frac{z_i \cdot z_j}{|z_i||z_j|}$

$\Rightarrow$ defined on unit-hypersphere

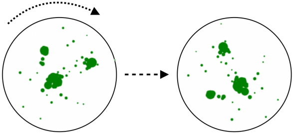


JetCLR $\to$ code at https://github.com/bmdillon/JetCLR

Augmenting jets

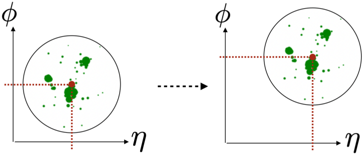
rotations

Angles sampled from $[0, 2\pi]$



translations

Translation distance sampled randomly



collinear splittings

some constituents randomly split,

$$p_{T,a} + p_{T,b} = p_T, \quad \eta_a = \eta_b = \eta$$
$$\phi_a = \phi_b = \phi$$

low p_T smearing

(η, ϕ) co-ordinates are re-sampled:

$$\eta' \sim \mathcal{N}\left(\eta, \frac{\Lambda_{\text{soft}}}{p_T} r\right)$$
$$\phi' \sim \mathcal{N}\left(\phi, \frac{\Lambda_{\text{soft}}}{p_T} r\right).$$

Network and training

Transformer-encoder network

- ⋆ based on 'self-attention' mechanism
- ⋆ output invariant to constituent ordering

  $\Rightarrow$ permutation invariance by construction

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

more info. in additional slides

Network and training

Transformer-encoder network

- ⋆ based on 'self-attention' mechanism
- ⋆ output invariant to constituent ordering

  $\Rightarrow$ permutation invariance by construction

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

more info. in additional slides

The training loop:

1. sample batch of jets, x_i
2. create an augmented batch of jets, x_i'
3. forward-pass both through the network
4. compute the contrastive loss & update weights

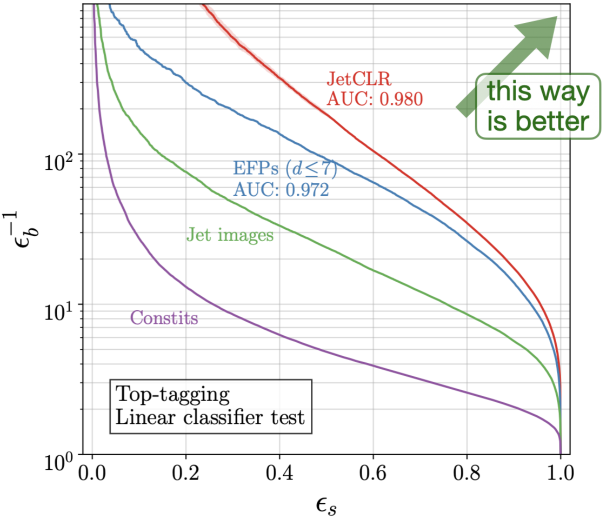
1. Jet physics & ML

2. Self-supervision

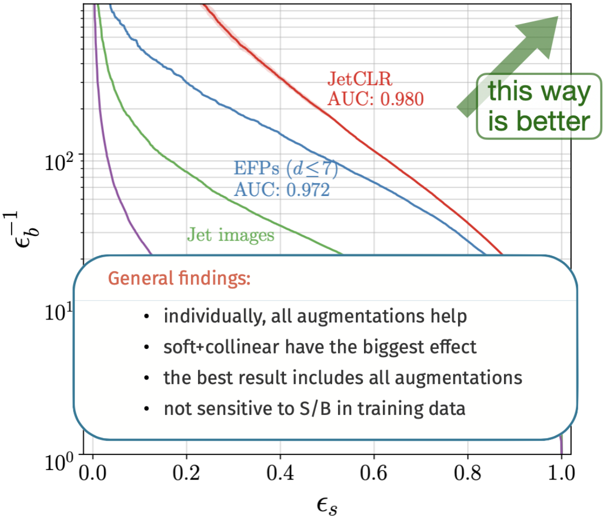
3. Results

4. Outlook

Linear classifier test results



Linear classifier test results



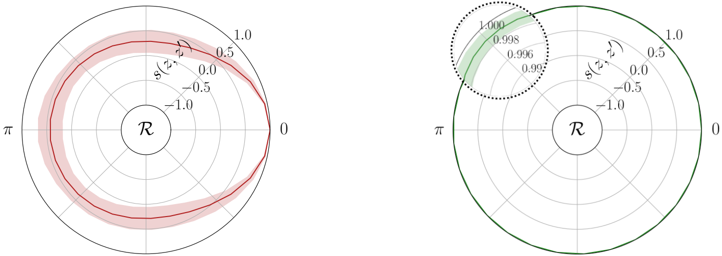
General findings:

- individually, all augmentations help
- soft+collinear have the biggest effect
- the best result includes all augmentations
- not sensitive to S/B in training data

Invariances in representation space

without rotational invariance

with rotational invariance



- ⋆ $s(z,z') = \frac{z \cdot z'}{|z|\,|z'|}$, $z = f(\vec{x})$, $z' = f\left(R(\theta)\vec{x}\right)$
- $\Rightarrow$ The network $f(\vec{x})$ is approx rotationally invariant

1. Jet physics & ML
2. Self-supervision
3. Results
4. Outlook

Outlook

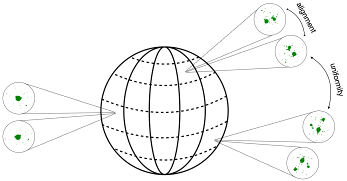
Self-supervision allows for:

1. data-driven definition of observables
2. invariance to pre-defined symmetries/augmentations

$\rightarrow$ definition of observables based on data and symmetries only

$\rightarrow$ high discriminative power

An example: **JetCLR** (contrastive learning of jet observables)



Outlook

Self-supervision allows for:

1. data-driven definition of observables
2. invariance to pre-defined symmetries/augmentations

$\rightarrow$ definition of observables based on data and symmetries only

$\rightarrow$ high discriminative power

An example: **JetCLR** (contrastive learning of jet observables)

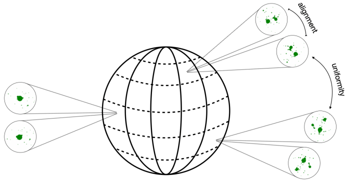
On-going work:

- Robust jet representations
- **anomaly-detection**

  better representations

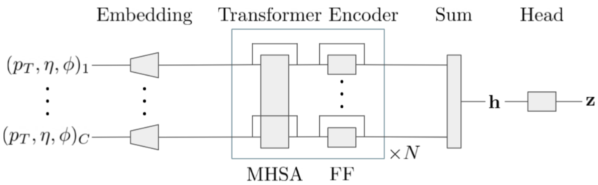
  $\Rightarrow$ better results!

  (coming soon...)



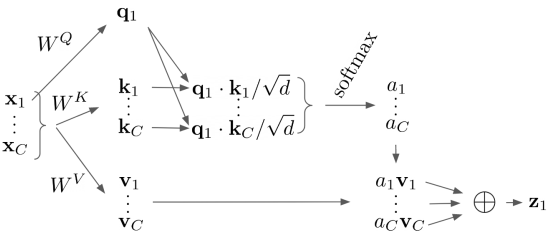
The network

We use a transformer-encoder network $\longrightarrow$ permutation invariance



Equivariance $\longrightarrow$ invariance is similar to Deep-Sets/Energy-Flow-Networks: arXiv:1810.05165, P. T. Komiske, E. M. Metodiev, J. Thaler

The attention mechanism captures correlations between constituents by allowing each constituent to assign attention weights to every other constituent.



Training loop in more detail

The training procedure:

1. sample batch of jets, x_i
2. create an augmented batch of jets, x_i'
3. forward-pass both through the network
4. compute the loss & update weights

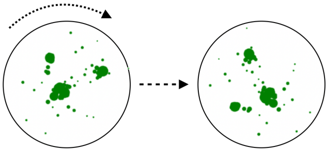
Training loop in more detail

The training procedure:

1. sample batch of jets, x_i
2. create an augmented batch of jets, x_i'
3. forward-pass both through the network
4. compute the loss & update weights

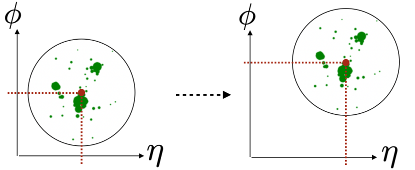
rotations

Angles sampled from $[0, 2\pi]$



translations

Translation distance sampled randomly



Training loop in more detail

The training procedure:

1. sample batch of jets, x_i
2. create an augmented batch of jets, x_i'
3. forward-pass both through the network
4. compute the loss & update weights

collinear splittings

some constituents randomly split,

$$p_{T,a} + p_{T,b} = p_T, \quad \eta_a = \eta_b = \eta$$
$$\phi_a = \phi_b = \phi$$

low p_T smearing

(η, ϕ) co-ordinates are re-sampled:

$$\eta' \sim \mathcal{N}\left(\eta, \frac{\Lambda_{\text{soft}}}{p_T} r\right)$$
$$\phi' \sim \mathcal{N}\left(\phi, \frac{\Lambda_{\text{soft}}}{p_T} r\right).$$

Training loop in more detail

The training procedure:

1. sample batch of jets, x_i
2. create an augmented batch of jets, x'_i
3. forward-pass both through the network
4. compute the loss & update weights

permutation invariance

Transformer-encoder network

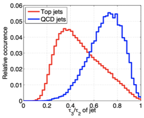
- ⋆ based on 'self-attention' mechanism
- ⋆ output invariant to constituent ordering

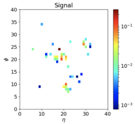
more info. in additional slides

Quality measure of observables

Many representations used in practice:

- raw constituent data
- jet images
- Energy Flow Polynomials
  (Thaler et al: arXiv:1712.07124)



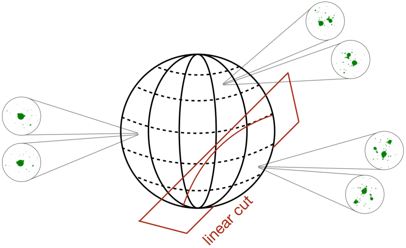


Compare these using a Linear Classifier Test (LCT)

- ⋆ use top-tagging as a test
- ⋆ **linear cut** in the observable space
- ⋆ supervised - uses simulations
- ⋆ measures:

  ϵ_s - true positive rate

  ϵ_b - false positive rate

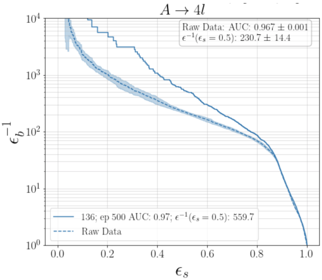


Self-supervised anomaly-detection (PRELIMINARY)

1 - Self-supervised representations + autoencoders (w. Friedrich Feiden)

- CMS anomaly-detection challenge
- Events:

  MET, 10 jets, 4 electrons, 4 muons
- Signal $A \to 4l$
- Self-supervision increases background rejection by O(5)



2 - Representation-norm as an anomaly measure

3 - Anomaly augmentations