Self-supervision in particle physics

Barry M. Dillon

March 30, 2022

Institute for Theoretical Physics
University of Heidelberg

IRN-Terascale @ Bonn

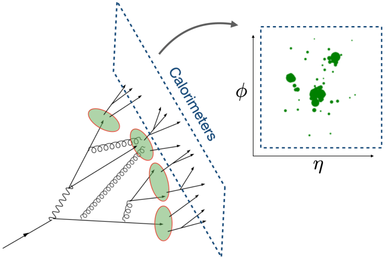
Symmetries, Safety, and Self-Supervision, hep-ph/2108.04253

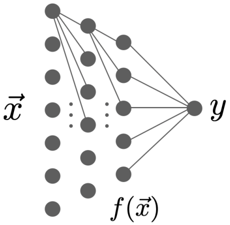
BMD, Gregor Kasieczka, Hans Olischlager, Tilman Plehn, Peter Sorrenson, and Lorenz Vogel

UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.

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: $\mathcal{L}_i = y'_i \log(y_i) + (1 - y'_i) \log(1 - y_i)$
- 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

Top-tagging with machine-learning

Neural networks don't explicitly learn the invariances associated with jets

⋆ we can't know exactly what features the network learns (..simulation artefacts?..)

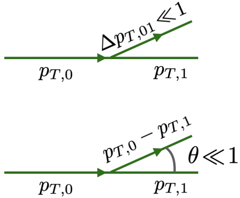
Top-tagging with machine-learning

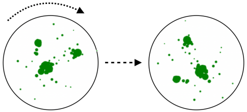
Neural networks don't explicitly learn the invariances associated with jets

⋆ we can't know exactly what features the network learns (..simulation artefacts?..)

What do we want the network to learn?

- rotational invariance
- translational invariance
- permutation invariance
- IR safety
- collinear safety





$$f(R\vec{x}) = f(\vec{x}) = y$$

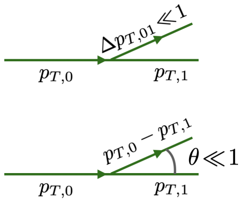
Top-tagging with machine-learning

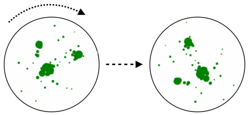
Neural networks don't explicitly learn the invariances associated with jets

⋆ we can't know exactly what features the network learns (..simulation artefacts?..)

What do we want the network to learn?

- rotational invariance
- translational invariance
- permutation invariance
- IR safety
- collinear safety





$f(R\vec{x}) = f(\vec{x}) = y$

→ How can we control what a neural network learns?

Can we force it to learn invariances from the raw data?

Optimising observables / representations

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

Optimising observables / representations

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**

neural networks are optimised using pseudo-labels, not truth labels

$\rightarrow$ independent of signal-types + can run directly on expt. data

Optimising observables / representations

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**

neural networks are optimised using pseudo-labels, not truth labels

$\rightarrow$ independent of signal-types + can run directly on expt. data

⋆ Contrastive-learning (SimCLR, Google Brain, Hinton et al)

map raw jet data to a new representation / observables

Contrastive learning of jet representations

arxiv:2107.soon, 'Contrastive learning of jet observables'
BMD, G. Kasieczka, H. Olischlager, T. Plehn, P. Sorrenson, and L. Vogel

Dataset: mixture of top-jets and QCD-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
  related by augmentation
- negative-pairs: $\{(x_i, x_j)\} \cup \{(x_i, x_j')\}$ for $i \neq j$
  not related by augmentation

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

positive and negative pairs = pseudo-labels

Contrastive learning of jet representations

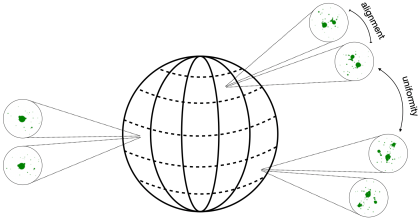
Train a network to map raw data to a new representation space, $f : \mathcal{J} \rightarrow \mathcal{R}$, minimising the contrastive loss:

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

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



This optimises for:

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

Contrastive learning of jet representations

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

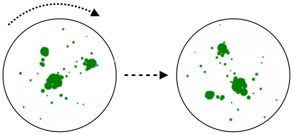
Contrastive learning of jet representations

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

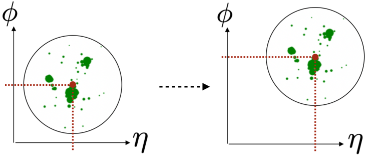
rotations

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



translations

Translation distance sampled randomly



Contrastive learning of jet representations

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:

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

Contrastive learning of jet representations

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

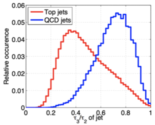
- ⋆ based on 'self-attention' mechanism
- ⋆ output invariant to constituent ordering

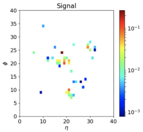
more info. in additional slides

Quality measure of observables

Many representations used in practice:

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





Quality measure of observables

Many representations used in practice:

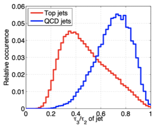
- raw constituent data
- jet images
- Energy Flow Polynomials
  (Thaler et al: arXiv:1712.07124)

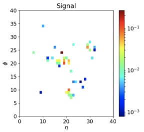
Compare these using a Linear Classifier Test (LCT)

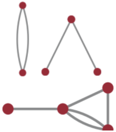
- ⋆ use top-tagging as a test
- ⋆ **linear cut** in the observable space
- ⋆ supervised - uses simulations
- ⋆ measures:

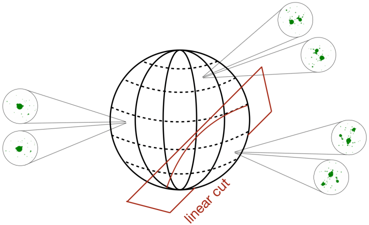
  ϵ_s - true positive rate

  ϵ_b - false positive rate



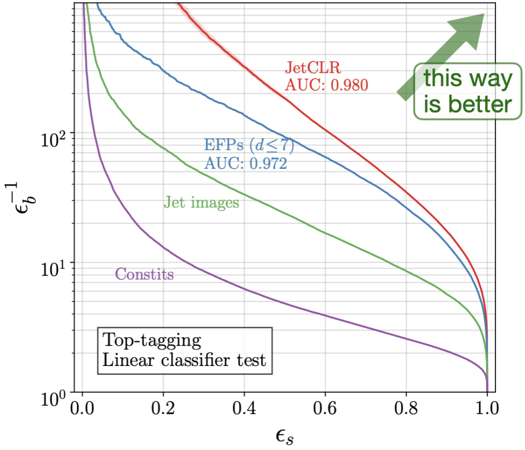






1. ML and jet physics
2. Self-supervision
3. Results
4. Outlook

Linear classifier test results



Linear classifier test results

Where does the performance come from?

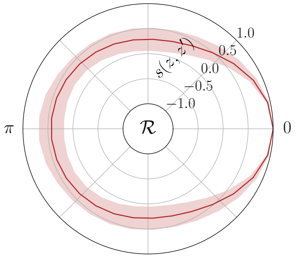
Augmentation	$\epsilon_b^{-1}(\epsilon_S = 0.5)$	AUC
none	15	0.905
translations	19	0.916
rotations	21	0.930
soft+collinear	89	0.970
all combined (default)	181	0.980

- ⋆ soft + collinear has the biggest effect

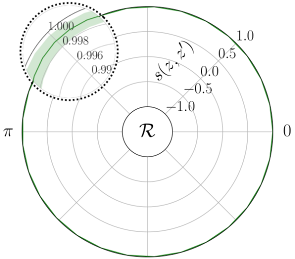
  translations + rotations also significant in final combination
- ⋆ also not very sensitive to S/B

Invariances in representation space

without rotational invariance



with rotational invariance



⋆ $s(z,z') = \frac{z \cdot z'}{|z||z'|}$, $z = f(\vec{x})$, $z' = f\big(R(\theta)\vec{x}\big)$

$\Rightarrow$ The network $f(\vec{x})$ is approx rotationally invariant

1. ML and jet physics

2. Self-supervision

3. Results

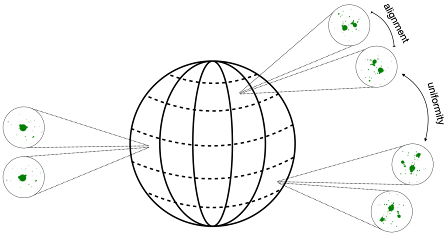
4. Outlook

Outlook

Self-supervision allows for:

1. data-driven definition of observables
2. invariance to pre-defined symmetries/augmentations
3. 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
3. high discriminative power

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

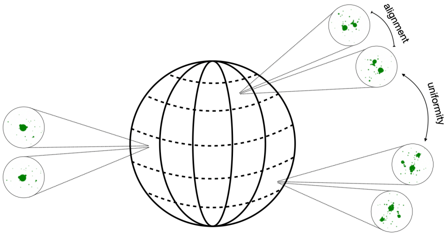
On-going work:

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

  better representations

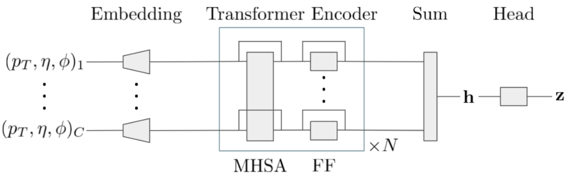
  ⇒ better results!

  (coming soon...)



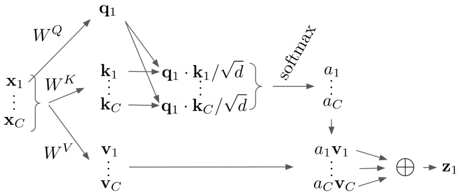
The network

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



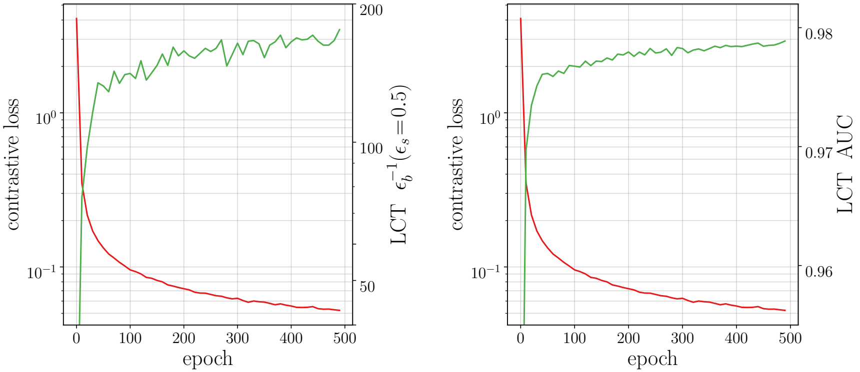
Equivariance $\longrightarrow$ invariance is similar to Deep-Sets/Energy-Flow-Networks: arXiv:1810.05165, P. T. Komiske, E. M. Metodiev, J. Thaler

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



Linear classifier test results

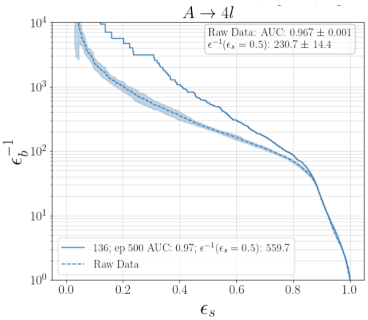
Performance as a function of training time / epochs



Self-supervised anomaly-detection (PRELIMINARY)

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)



Other anomaly metrics taken directly from the self-supervised latent space also show promise

$\to$ work in progress...