DEEP LEARNING INVARIANT MODELS

Radi Radev

EP-NU Meeting 17/5/2023

AGENDA

Invariance and Covariance

Toy Invariant Task

Reweighting Neutrino Models

Attention and Invariant Attention

Results

Future Plans

INVARIANCE AND COVARIANCE

A Machine Learning Perspective

Invariance and Equivariance

Invariance

- The function f is invariant to permutations of its inputs

$f($🐕🦔🐅$) =$ 🤗

$f($🐅🐕🦔$) =$ 🤗

Equivariance (Covariance)

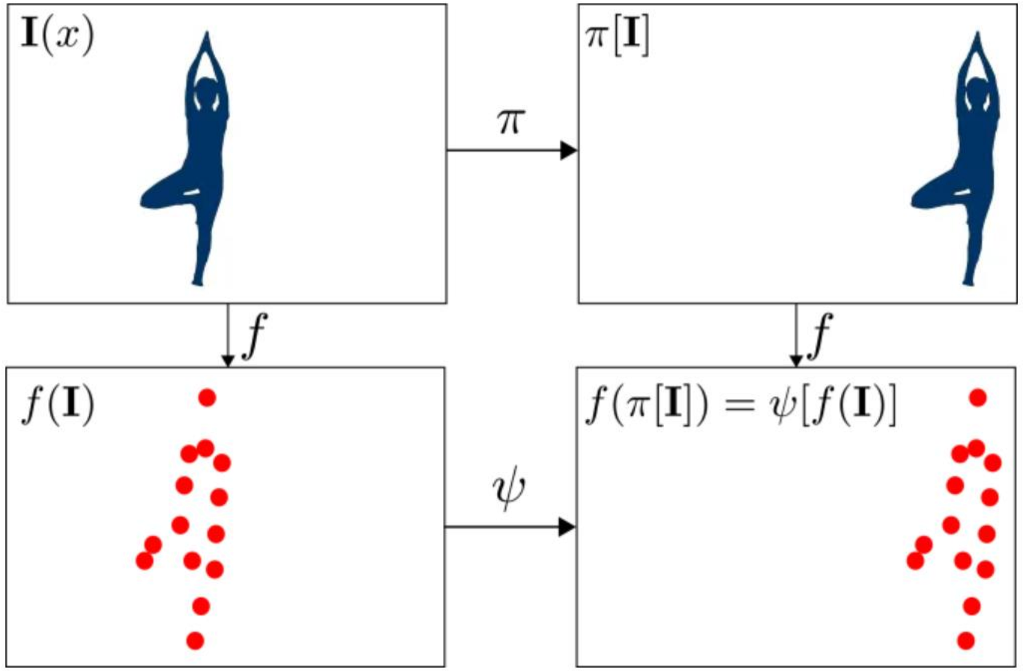
- The function g is equivariant to permutations of its inputs

$f($🐕🦔🐅$) =$ 🤗😃🤪

$f($🐕🐅🦔$) =$ 🤗🤪😃

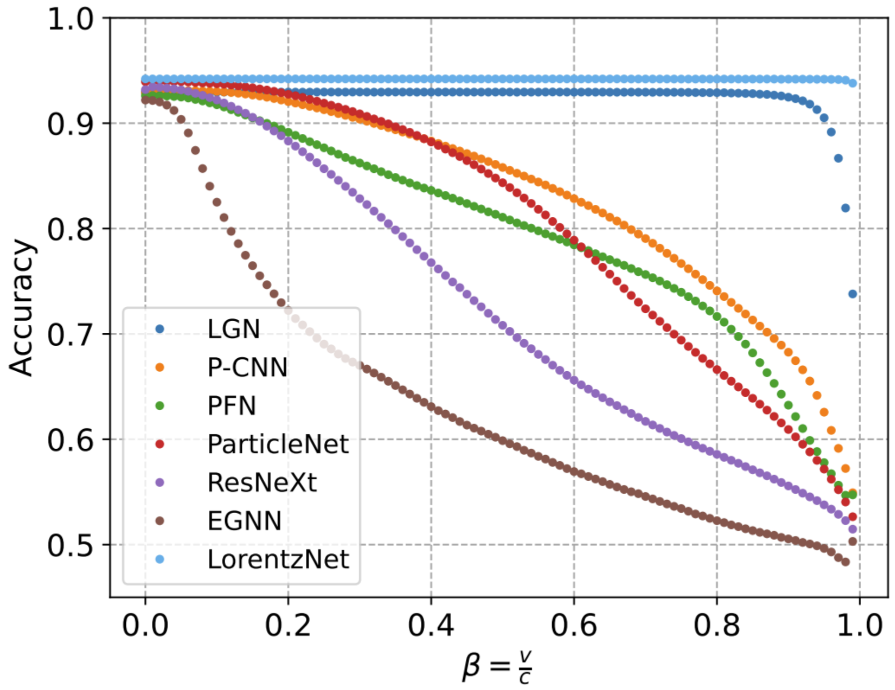
Invariant ML Models

- Convolutional models are naturally translation invariant and started the DL age
- As they learn a filter that slides along the image



Invariant ML Models in Physics

- We want to incorporate inductive biases in ML models to reduce the parameter space the model has to learn during training
- Allows the model to be more parameter and data efficient



TOY EXAMPLE

Cat vs Dog with a Spin

- Learning a classifier that tries to distinguish images of dogs and cats
- A convolutional model deals with the translational invariance in images

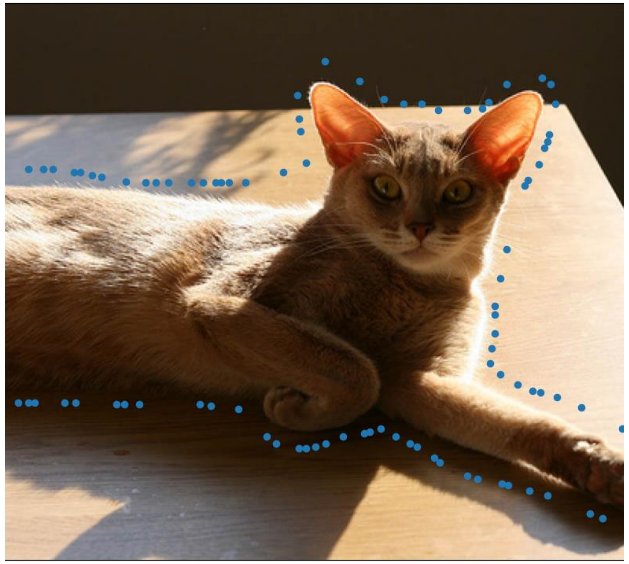
?

?

Cat vs Dog with a Spin

- To illustrate how to incorporate more general invariances, we sample points along the outline
- The model only sees the sampled points

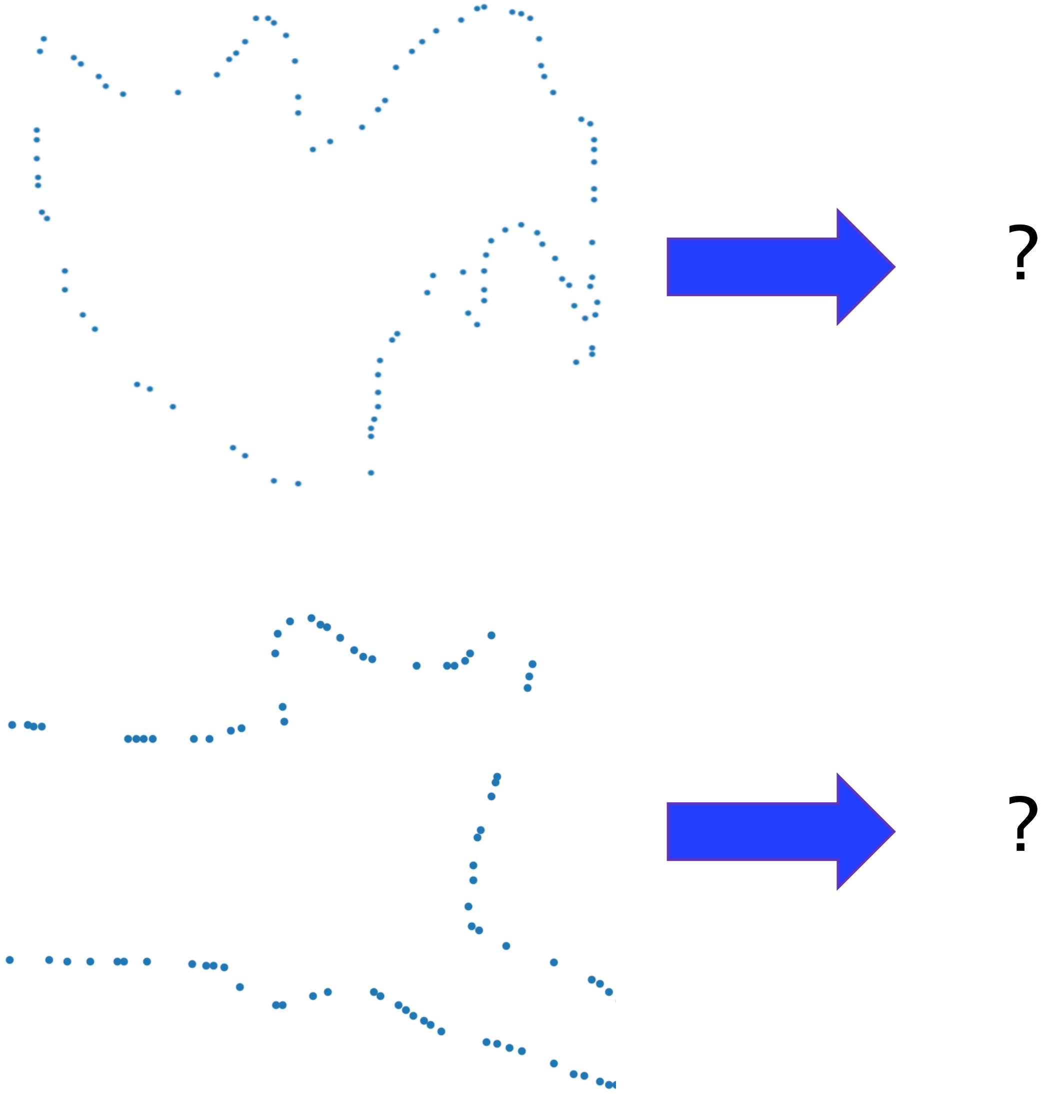
?



?

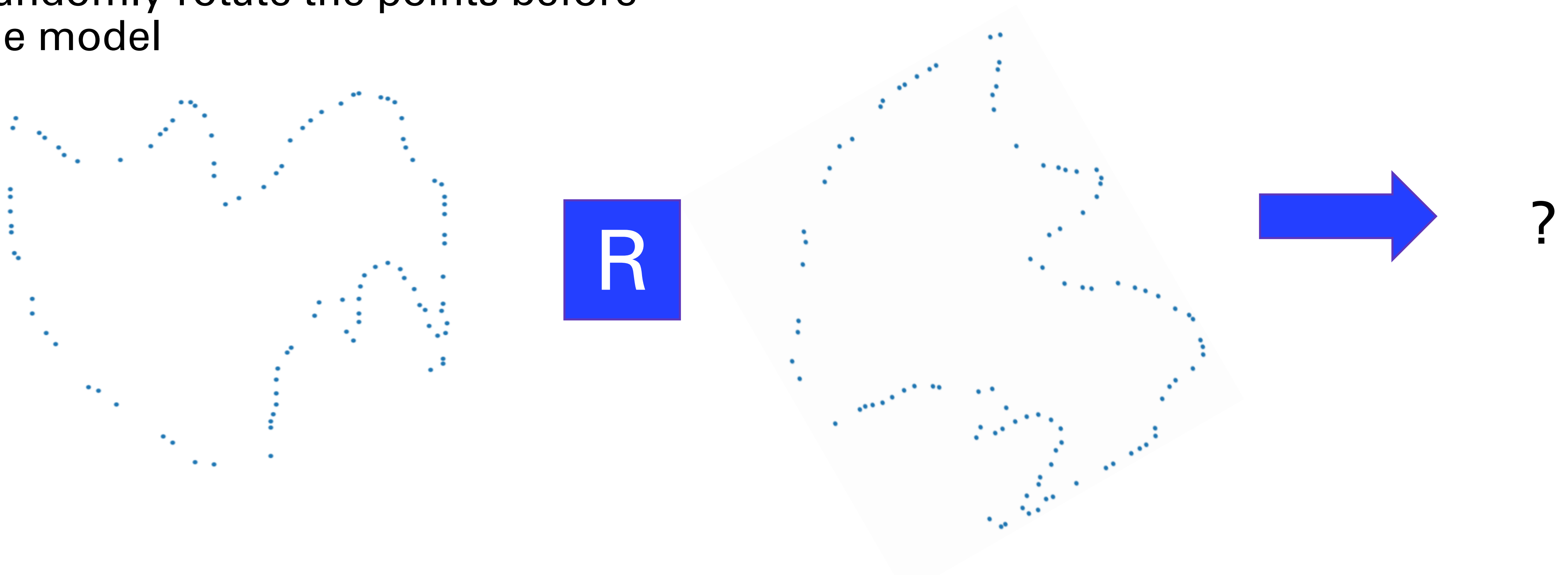
Cat vs Dog with a Spin

- To illustrate how to incorporate more general invariances, we sample points along the outline
- The model only sees the sampled points



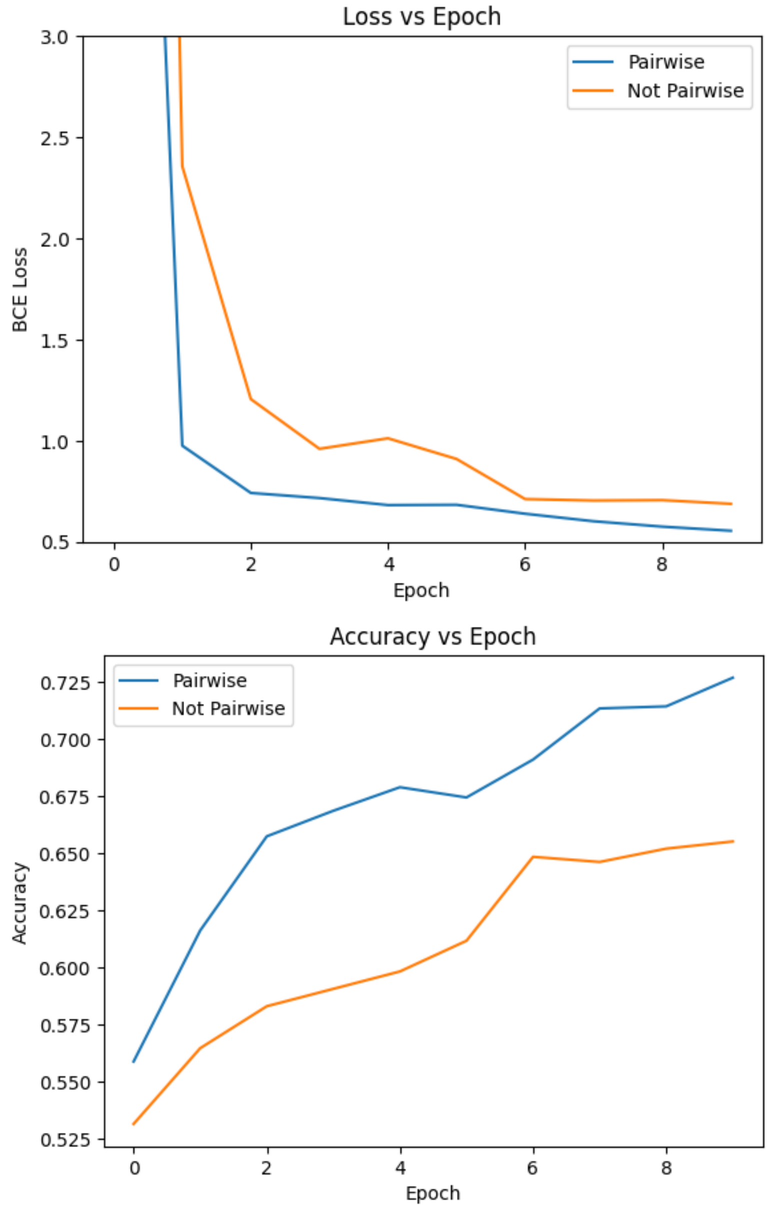
Cat vs Dog with a Spin

- To illustrate how to incorporate more general invariances, we sample points along the outline
- The model only sees the sampled points
- Randomly rotate the points before the model



Cat vs Dog with a Spin

- Train two models one with directly with the coordinates the other with the pairwise distances
- The model invariant to Euclidean transformations performs better and learns faster!

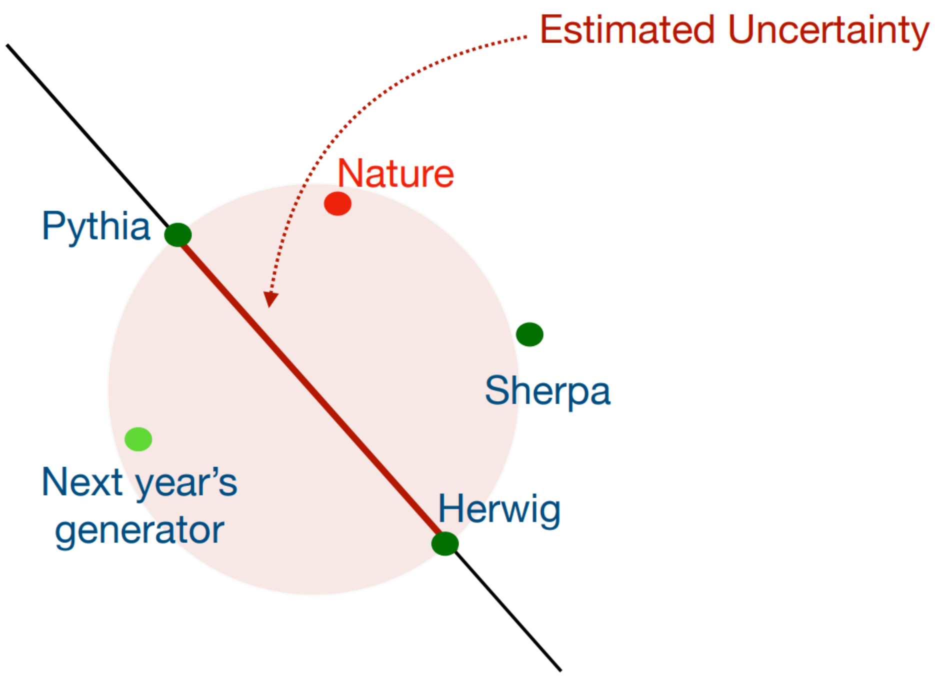


HIGH-DIMENSIONAL REWEIGHTING FUNCTIONS

And Where to Find Them

Neutrino Interaction Models

- Evaluating analyses with different neutrino interaction models allows us to evaluate uncertainties even if we don't have specific dials
- Detector simulation is computationally expensive 😟
- Reweighting allows us to compare analyses with different neutrino generators without having to rerun the detector simulation 😊



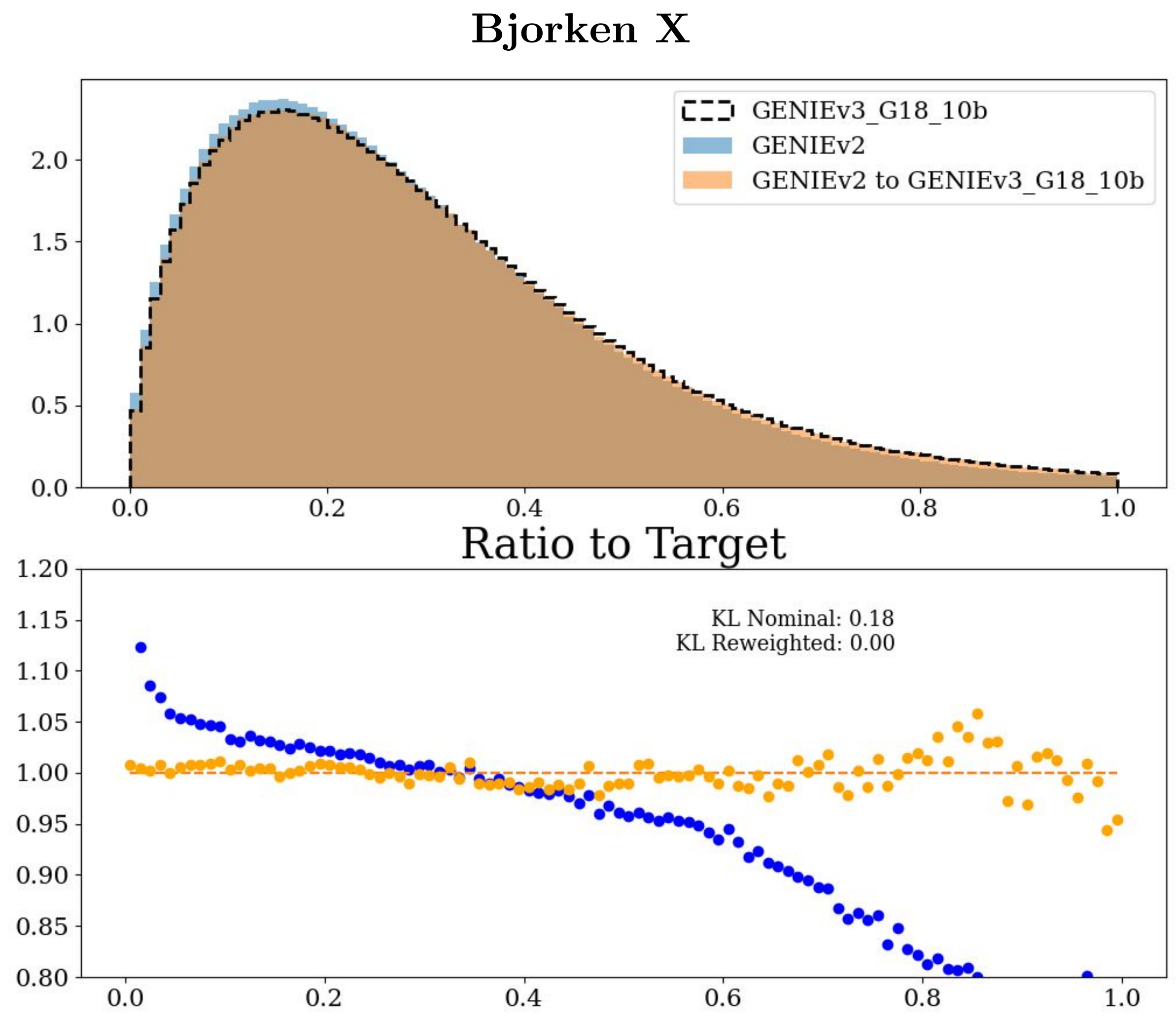
Reweighting Neutrino Models

- An event can be represented as a set of n four-momenta $p_1, p_2, \dots, p_n$ with each $p_i = (p_x, p_y, p_z, E)$
- Using those we can calculate extra variables at the particle and event level.
- Our goal is to calculate weights which shift our nominal model to look like the target model
- For 1D and 2D (event level) we can simply take ratios of histograms

ML Reweighting

- Machine learning models allow us to perform high-dimensional unbinned reweighting
- A classifier $f(x)$ trained with the binary cross-entropy loss would learn the likelihood-ratio:

$$w_x = \frac{f(x)}{1 - f(x)}$$



ATTENTION MECHANISM

Attention

Similarity

- For two vectors q and k we can compute their similarity as:

$$q \cdot k$$

- Instead, if we want to compute this for n vectors, then we can represent them as matrices:

$$QK^T$$

Complete Definition

- We can convert this to probabilities using the softmax function and use it to update our values:

$$f(Q, K, V) = \mathrm{softmax}(QK^T)V$$

- Where the softmax is normalized exponentiation:

$$\mathrm{softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

Invariant Attention

- Compute invariant inter-particle quantities (i.e minkowski norm and inner product

$$w = \left(\left|p_i \; - p_j \;\right|\right), (\langle p_i \;, p_j \;\rangle)$$

- Add the invariant quantities as a bias to the queries and keys

$$Q = W_V(h) + \mathrm{MLP}(w)$$
$$V = W_Q(h) + \mathrm{MLP}(w)$$
$$K = W_K(h)$$

- Update the particle features

$$\hat{h} = \mathrm{softmax}(QK)V$$

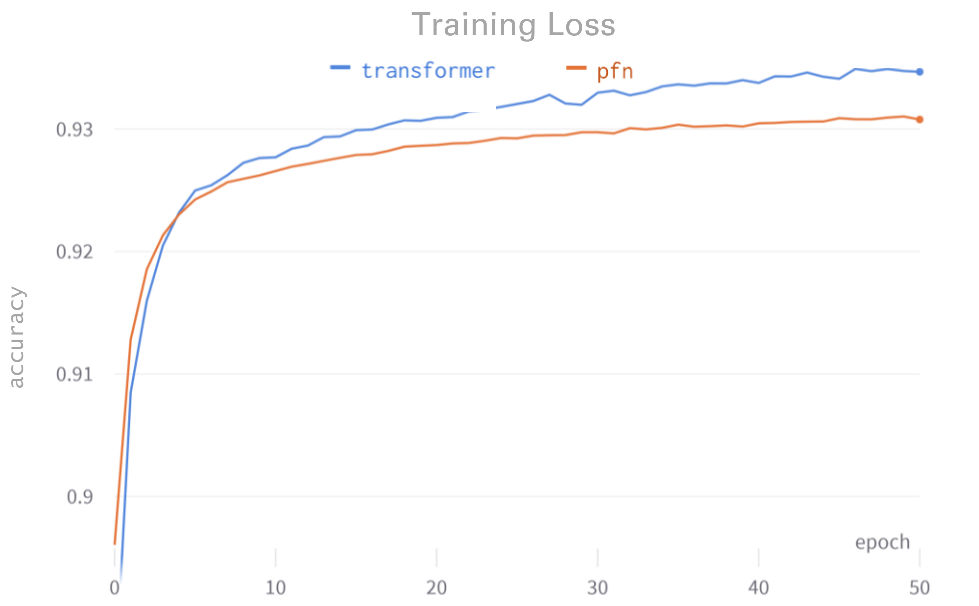
- Update the coordinates

$$\widehat{p_i} = \mathrm{softmax}\big(\mathrm{MLP}(QK)\big)p_i$$

RESULTS

Results

- Training the Lorentz transformer on Top Quark Tagging data.
- The network outperforms the baseline by a good margin and very close to state of the art
- Can be improved by further tuning



Result of models on the Top Tagging dataset

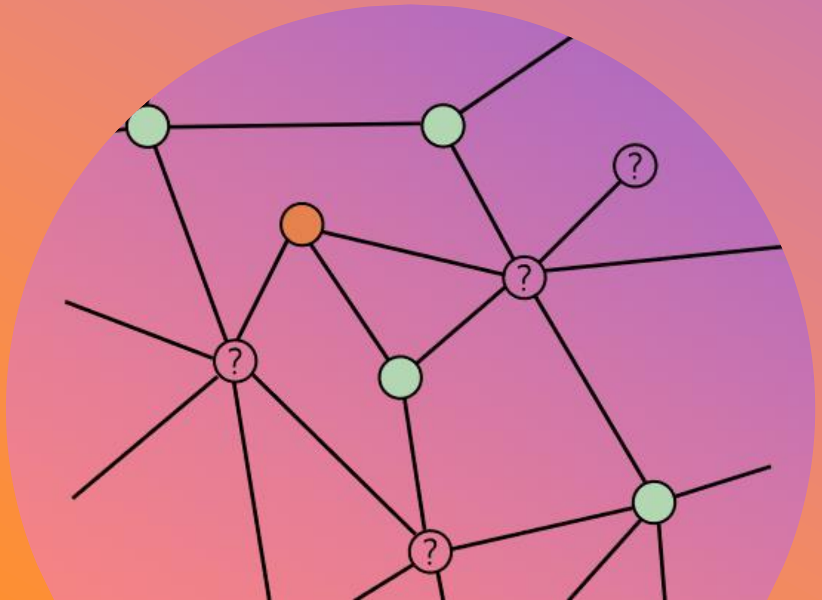
Model	Accuracy	AUC	$1/\varepsilon_B$ $(\varepsilon_S = 0.5)$	$1/\varepsilon_B$ $(\varepsilon_S = 0.3)$
ResNeXt	0.821	0.8960	30.9	80.8
P-CNN	0.827	0.9002	34.7	91.0
PFN	–	0.9005	34.7 ± 0.4	–
ParticleNet	0.840	0.9116	39.8 ± 0.2	98.6 ± 1.3
EGNN	0.803	0.8806	26.3 ± 0.3	76.6 ± 0.5
LGN	0.803	0.8324	16.0	44.3
LorentzNet	**0.844**	**0.9156**	$\mathbf{42.4 \pm 0.4}$	$\mathbf{110.2 \pm 1.3}$

Future Plans

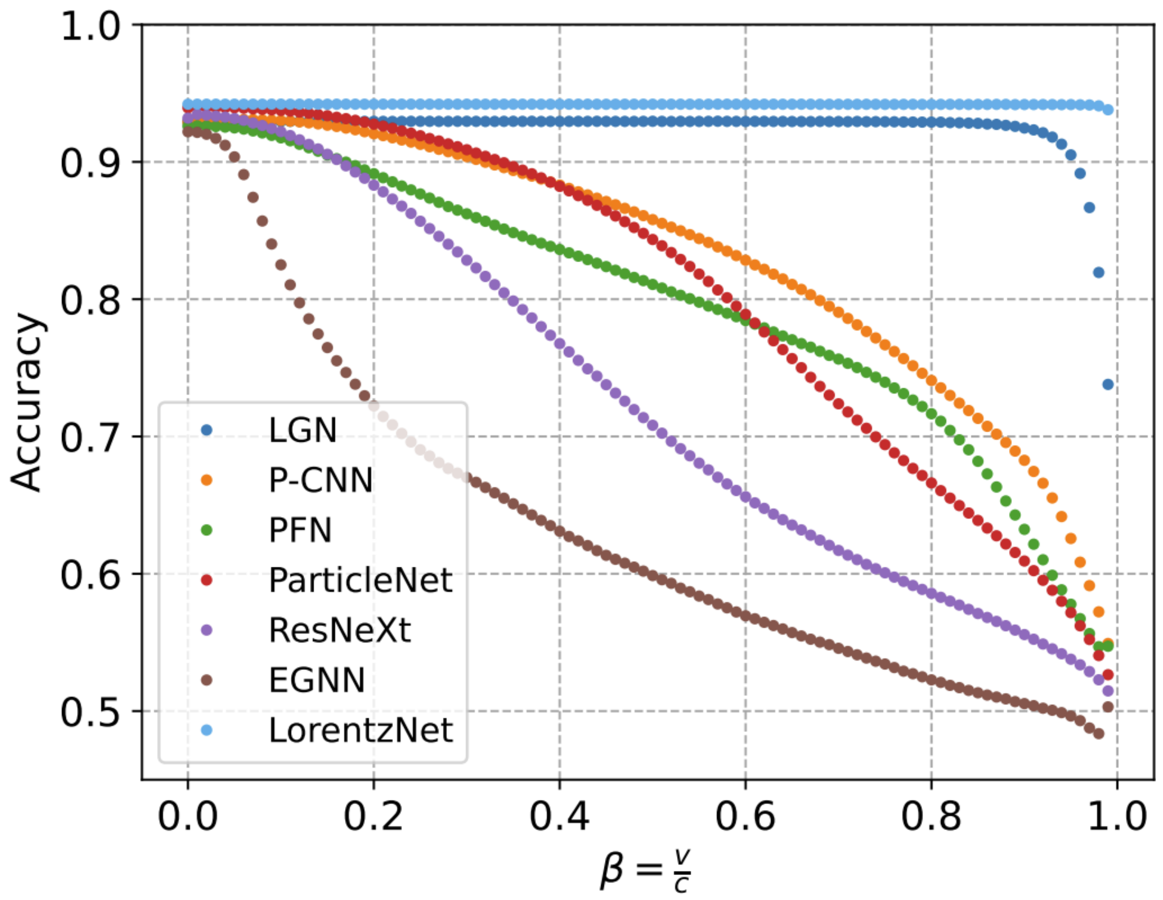
- Long(er) training (multi-GPU)
- Optimize architecture and training setup (e.g., bucket batching)
- Apply the Lorentz Transformer to the reweighting task
- Consider other use cases of this approach and other physics-aware methods (e.g., incorporating systematic uncertainties)

THANK YOU

Radi Radev



Backup



Accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.0
0.2
0.4
0.6
0.8
1.0
$\beta = \frac{v}{c}$
LGN
P-CNN
PFN
ParticleNet
ResNeXt
EGNN
LorentzNet