PELICAN: Permutation Equivariant and Lorentz Invariant or Covariant Aggregator Network for Particle Physics

Alexander Bogatskiy
Center for Computational Mathematics
Flatiron Institute, New York, NY, U.S.A.
abogatskiy@flatironinstitute.org

Timothy Hoffman
Department of Physics, University of Chicago
Chicago, IL, U.S.A.
hoffmant@uchicago.edu

David W. Miller
Department of Physics, University of Chicago
Enrico Fermi Institute
Chicago, IL, U.S.A.
David.W.Miller@uchicago.edu

Jan T. Offermann
Department of Physics, University of Chicago
Enrico Fermi Institute
Chicago, IL, U.S.A.
jano@uchicago.edu

Abstract

Many current approaches to machine learning in particle physics use generic architectures that require large numbers of parameters, often adapted from unrelated data science or industry applications, and disregard underlying physics principles, thereby limiting their applicability as scientific modeling tools. In this work, we present a machine learning architecture that uses a set of inputs maximally reduced with respect to the full 6-dimensional Lorentz symmetry, and is fully permutation-equivariant throughout. We study the application of this network architecture to the standard task of classifying the origin of jets produced by either hadronically-decaying massive top quarks or light quarks, and show that the resulting network outperforms all existing competitors despite significantly lower model complexity. In addition, we present a Lorentz-covariant variant of the same network applied to a 4-momentum regression task in which we predict the full 4-vector of the W boson from a top quark decay process.

1 Introduction

Neural networks have played a significant role in data analysis for particle physics experiments, perhaps most significantly in the study of *jets*, the collimated streams of hadronic particles produced by the decay, showering and hadronization of quarks and gluons. As demonstrated in Ref. [6], convolutional neural networks can be leveraged for identifying the type of particle that initiated a jet. However, such network architectures do not explicitly respect (or leverage) the symmetries inherent in particle physics, in particular those of the *Lorentz group*. Designing a network architecture that respects these symmetries may benefit model interpretability while reducing model complexity via physically-meaningful constraints, without sacrificing performance.

2 Equivariance and jet physics

Lorentz Invariance In what follows, "Lorentz group" refers to the *proper orthochronous Lorentz group* $\mathrm{SO}^+(1,3)$, i.e. the identity component of the full Lorentz group $\mathrm{O}(1,3)$ of linear transformations on $\mathbb{R}^4$ that preserve the Minkowski metric $\eta = \mathrm{diag}(1,-1,-1,-1)$. The classification task considered

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

in this work is Lorentz invariant, that is, the output of the network is invariant under the application of any Lorentz transformation $\Lambda \in \mathrm{SO}^{+}(1,3)$ to all of the 4-vector inputs (energy-momentum vectors in our case, for which $p = (p^0, \vec{p}) = (E, p^x, p^y, p^z)$ and $p^2 = E^2 - \vec{p}^2$). The simplest way to enforce invariance is to hand-pick a set of invariant observables (e.g. particle masses, identification labels) as inputs to a generic neural network architecture, as summarized in Ref. [2]. Another approach inspired by convolutional neural networks (CNN's) is to preserve group-equivariant latent representations in the hidden layers, see e.g. Refs. [1, 9]. As in CNN's, equivariant latent representations, as opposed to invariant ones, can regularize the network via efficient weight-sharing [25].

Here, we take a slightly different approach. Given a set of 4-vector inputs $p_1, \ldots, p_N$, we compute the *complete* set of Lorentz invariants on that set. Weyl's work [23] characterized the set of all Lorentz invariant functions of a collection of 4-vector inputs. Namely, all totally symmetric Lorentz invariants $I(p_1, \ldots, p_N)$ depend only on the invariant dot products (see related discussion in Ref. [21]):

$$I(p_1, \ldots, p_N) = I\left(\{p_i \cdot p_j\}_{i,j}\right). \tag{1}$$

The array of all $N \times N$ pairwise dot products will be the network input. We note that the idea to include dot products as inputs was recently used in Ref. [15]. As we will show, these invariant dot products alone can provide state-of-the-art performance in a significantly simpler architecture.

Permutation Equivariance Particle data is often naturally represented by a point cloud, or a set. For such problems the ordering of the particles in the set is not physically meaningful, and thus it makes sense to use one of the permutation-equivariant architectures. One approach is that of Deep Sets [24], applied to jet tagging e.g. in Ref. [14]. It is based on the fact that any symmetric function of inputs $x_1, \ldots, x_N$ can be written in the form $\psi\left(\sum_i \varphi(x_i)\right)$, where ψ and φ can be approximated by neural networks. However, since aggregation happens only once, the network can struggle at modeling complex higher-order interactions between the particles. The sub-network representing ψ is forced to be a relatively complex (wide) fully-connected network, which makes it difficult to train [22, 25]. The alternative to permutation-invariant architectures is provided by permutation-*equivariant* ones. Equivariance is a key property of all convolutional networks – for example, in CNN's convolutions are manifestly equivariant with respect to translations (up to edge effects). Similarly, Graph Neural Networks (GNN's) use permutation equivariance, usually in the form of message passing (MP), to force architectures to respect the underlying graph structure.

Despite the benefits of MP (previously used in jet tagging [1, 9]), attempts to combine MP with Lorentz invariance run into an obstacle: the key inputs to the network are *nothing but* edge data $d_{ij} = p_i \cdot p_j$. Since traditional MP architectures use only single-label vertices, we employ the general permutation-equivariant layers proposed in Refs. [7, 19]. In the general setting, permutation equivariance is a constraint on mappings F between arrays $T_{i_1 i_2 \cdots i_r}$ of any rank r, where every index $i_k \in \{1, \ldots, N\}$ refers to a particle label, whereby permutations $\pi \in S_N$ of the particles "commute" with the map:

$$F\left(\pi \circ T_{i_1 i_2 \cdots i_r}\right) = \pi \circ F\left(T_{i_1 i_2 \cdots i_s}\right), \quad \pi \in S_N. \tag{2}$$

Here, the action of permutations is diagonal: $\pi \circ T_{i_1 i_2 \cdots i_p} = T_{\pi(i_1) \ldots \pi(i_p)}$. Thus a higher-order generalization of the MP layer can be defined as $T^{(\ell+1)} = \textsc{Agg} \circ \textsc{Msg}\left(T^{(\ell)}\right)$. Here, $\textsc{Msg}$ is a node-wise nonlinear map ("message forming") shared between all nodes, and $\textsc{Agg}$ is a general permutation-equivariant linear mapping ("aggregation") acting on the particle indices of T.

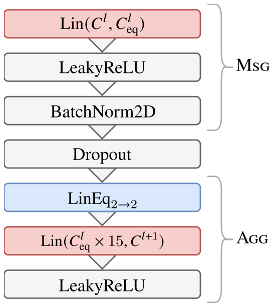
Elementary Equivariant Aggregators It remains to describe the exact structure of the equivariant aggregation layers introduced above. Since the general case is presented in [7, 19], here we will only present the layers needed for jet physics. Since the input is an array of rank 2 (of dot products), the main equivariant layer in this case is one that maps between arrays of rank 2: $T_{ij} \mapsto T'_{ij}$. The space of all linear maps of this type is 15-dimensional and its basis elements can be defined using binary arrays of rank 4. There are 15 such arrays B^a_{ijkl}, $a = 1, \ldots, 15$, (see Ref. [19] for exact expressions) and the action of the equivariant layer can be written as $T'^{a}_{ij} = \sum_{k,l=1}^{N} B^a_{ijkl} T_{kl}$. Five of the basis elements in fact contain only one non-zero component for each ij pair, which includes the identity and the transposition maps, so they can be thought of as sorts of "skip connections". The rest involve aggregations over N or N^2 elements of the input.

More generally, instead of a simple summation, aggregators can involve arbitrary (nonlinear) symmetric functions, e.g. maximum. In practice, we define aggregation as the mean of its inputs followed by an additional scaling by a factor of $\left(N/\bar{N}\right)^{\alpha}$ with learnable exponents α, where $\bar{N}$ is a constant describing the typical number of particles expected in the input.

Equivariance and Jet Physics There are several reasons for enforcing the full Lorentz symmetry in our ML models. First and foremost, it is a fundamental symmetry of the space to which the inputs belong. If an analyzer in the lab frame establishes that a given collection of particles resulted from a top quark decay, then the same is true for all other reference frames. The breaking of the Lorentz symmetry implicit in the running of the QCD couplings notwithstanding, there is no question that both the original protons and the final (asymptotic) decay products are accurately represented by a collection of 4-vectors subject to the global Lorentz symmetry. Another reason for symmetry-restricted modeling is that, from the geometric perspective, only some mathematical operations are permissible when working with objects that transform in a certain way under a symmetry group. A non-equivariant neural network effectively neglects the vector nature of the inputs by treating individual components of the input vectors as scalars. Despite *a priori* improving network expressivity, non-equivariance fails to deliver physically interpretable models. Ultimately, a statement about equivariance is a statement about what quantities are not truly self-contained *features* of the inputs – e.g. a single x-component of a 2D vector is not a feature of that vector unless the input also contains the vector $(1, 0)$.

3 PELICAN architecture

Permutation Equivariant Blocks The main equivariant block, $\texttt{Eq}_{2\to 2}$, consists of a simple dense layer Msg and an aggregation block Agg. The aggregation block applies 15 linear aggregation functions ($\texttt{LinEq}_{2\to 2}$) as outlined in Section 2. Note that this is a non-parametric transformation performed on each channel separately. Each of the $C^l_{\text{eq}} \times 15$ resulting aggregation values is then independently multiplied by $N^{\alpha}/\bar{N}^{\alpha}$ with a trainable exponent α (initialized as a random float in $[0, 1]$, allowed to become negative), where N is the number of particles in the corresponding event. This allows for some flexibility in the aggregation process, for example $\alpha = 1$ returns the sum aggregation function, and combining multiple aggregators is known to boost accuracy [7]. PELICAN stacks five such blocks for optimal results according to a loss-minimizing hyperparameter search.



$\text{Lin}(C^l, C^l_{\text{eq}})$
LeakyReLU
BatchNorm2D
Dropout
$\text{LinEq}_{2\to 2}$
$\text{Lin}(C^l_{\text{eq}} \times 15, C^{l+1})$
LeakyReLU
Msg
Agg

PELICAN Classifier To build a classifier, aside from the $\texttt{Eq}_{2\to 2}$ equivariant layer one needs a $\texttt{Eq}_{2\to 0}$ layer that reduces the rank 2 array to permutation-invariant scalars. This layer involves just 2 aggregation functions instead of 15 – the trace and the total sum of the input square matrix, but is otherwise identical to the equivariant block described above. The inputs d_{ij} are positive with a very steeply decaying distribution at large values, therefore after forming the matrix of pairwise dot products the input layer applies a set of encoding functions of the form $((1+x)^{\delta} - 1)/\delta$, with $\delta = \beta^2$ and learnable β's. From the input block, the tensor is passed through several equivariant $\texttt{Eq}_{2\to 2}$ blocks, and a $\texttt{Eq}_{2\to 0}$ block, all with dropout. One final dense layer mixes the channels down to 2 classification weights per event. A cross-entropy loss function is then used for optimization.

PELICAN Regressor The same architecture can also be easily adapted for 4-momentum regression tasks, such as momentum reconstruction. Any Lorentz-equivariant map from a collection of 4-momenta $p_1, \dots, p_N$ to one (or several) 4-momentum has the form

$$F(p_1, \dots, p_N) = \sum_{i=1}^{N} f_i(p_1, \dots, p_N) \cdot p_i, \tag{3}$$

where f_i's are Lorentz-invariant functions [21]. Combining this with permutation-invariance, we conclude that the multi-valued map $(p_1, \dots, p_N) \mapsto (f_1, \dots, f_N)$ must also be equivariant with respect to the permutations of the inputs. The only change required to the architecture we've introduced for classification is that $\texttt{Eq}_{2\to 0}$ must be replaced with $\texttt{Eq}_{2\to 1}$ and the final output layer must have only one output channel. The $\texttt{Eq}_{2\to 1}$ layer is again identical to $\texttt{Eq}_{2\to 2}$ except that it uses only 4 linear aggregation functions. For the loss function we use $5\left|m_p - m_t\right| + \left|\vec{p}_p - \vec{p}_t\right|$ where subscripts p, t stand for predicted and true values, respectively. We avoid squares due to their sensitivity to outliers, and the coefficients of the two terms are chosen to roughly balance their magnitudes on our dataset, forcing the network to simultaneously predict the mass and the spatial momentum.

4 Experiments

Top tagging We perform top-tagging on the reference dataset [13] explored in Ref. [2]. This dataset consists of 2M entries, each entry corresponding to a single hadronic top jet or the leading jet from a QCD dijet event. The events were generated with `PYTHIA` 8.2 [20], with `DELPHES` [8] used for detector interactions. For each jet, the four-momenta of up to 200 constituents are listed. The model was trained in batches of 100 events on A100 80GB GPU's using the AdamW optimizer [17] with a linear warmup of the learning rate up to $2.5 \cdot 10^{-3}$ for the first 4 epochs, followed by 28 epochs of `CosineAnnealingWarmRestarts` with $T_0 = 4$, $T_{\text{mult}} = 2$, and 3 more epochs of exponential schedule with $\gamma = 0.5$. A dropout rate of 1% was used. The Msg blocks output 35 channels and the Agg blocks output 60. Hyperparameters were tuned by manual optimization.

The metrics of several top taggers are compared in Table 1, quoted from Refs. [2, 9]. $1/\epsilon_B$ stands for the background rejection rate at efficiency rate of 0.3. Among these PFN, LGN and LorentzNet are physics-motivated architectures, with LorentzNet using a Lorentz-invariant version of Message Passing.

Table 1: Comparison of top-taggers.

Architecture	Accuracy	AUC	$1/\epsilon_B$	# Params
LGN	0.929(1)	0.964(14)	424 ± 82	4.5k
PFN	0.932	0.982	891 ± 18	82k
ResNeXt	0.936	0.984	1122 ± 47	1.46M
ParticleNet	0.938	0.985	1298 ± 46	498k
LorentzNet	0.942	0.9868	2195 ± 173	220k
PELICAN	0.9425(1)	0.9869(1)	2289 ± 204	45k

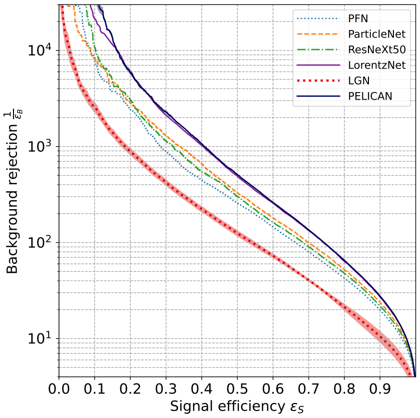


Fig. 1: Top-tagger ROC curves.

Our results are averaged over 5 random initialization seeds and the uncertainties are given by the standard deviation. As can be seen from these results, equivariant architectures provide good models for top-tagging despite much lower model complexity as measured by the number of trainable parameters. PELICAN in particular provides state-of-the-art accuracy with almost 5 times fewer parameters than the next best model. Model complexity is of great importance for many real-world applications, such as online detector triggering, which requires extremely low microsecond latencies [4, 5, 10, 11, 16]. Normally reduction in model size comes at the cost of accuracy, but equivariant architectures can avoid such compromises. Aside from accuracy, physics applications also demand high background rejection rates, where PELICAN also provides state-of-the-art performance.

4-Momentum reconstruction For regression, we use a dataset [18] containing 1M entries, each corresponding to a single hadronic top jet from an event generated with `PYTHIA` 8, with settings as in Ref. [13]. We prepare two versions of this dataset: One with jets clustered from truth-level, final-state particles, the other clustered from the output of detector simulation with `DELPHES`, each using the same events. Each entry contains the 200 leading jet constituents, and the parton-level 4-momenta of the top quark, and of the b-quark and W-boson to which it decays.

Table 2: Momentum reconstruction results for the Johns Hopkins (JH) tagger and PELICAN. We report the relative p_T and mass resolutions, and the interquantile range for the angle $\psi \in (0, \pi)$ between predicted and true momenta. PELICAN uncertainties are within the last significant digit.

	Method	σ_{p_T} (%)	σ_m (%)	σ_ψ (centirad)
Without DELPHES	JH	0.70%	1.29%	0.162
	PELICAN	0.83%	1.21%	0.388
	PELICAN|JH	0.28%	0.60%	0.089
	PELICAN|FC	0.32%	0.76%	0.111
With DELPHES	JH	10.8 %	8.3 %	8.9
	PELICAN	5.6 %	3.2 %	4.2
	PELICAN|JH	3.8 %	2.9 %	2.7
	PELICAN|FC	4.4 %	3.1 %	3.0

Our regression task consists of predicting the 4-momentum of the parton-level W-boson in the lab frame. The results are summarized in Table 2 by the resulting p_T and mass resolutions – given by half of the central 68th interquantile range of $(x_{\text{predict}} - x_{\text{true}})/x_{\text{true}}$, where x is m or p_T – and the lower 68th interquantile range for ψ, the angle between predicted and true momenta. To serve as a baseline regression method, we use the W-boson identification of the Johns Hopkins top tagger [12] implemented in FASTJET [3]. The tagger has a 36% efficiency on the dataset and can only identify W-boson candidates for jets it tags, so we report PELICAN results both on the tagged jets (PELICAN|JH) and on the full dataset. In addition, we report PELICAN results on the subset of jets that are *fully-contained* (PELICAN|FC), which we define as jets where both quarks from W-boson decay are within the jet radius. This is in fact a strict subset of the tagged jets, and we highlight this subpopulation of jets as we do not expect accurate W-boson momentum reconstruction in the case of jets that fail to capture a significant fraction of the W-boson decay products. Notably, fully-contained events comprise about 75% of the dataset, which is still significantly higher than JH tagger's efficiency. The regression network uses 36k parameters and was trained in the same way as the classifier. The code for PELICAN can be found at github.com/abogatskiy/PELICAN.

Conclusion We have introduced a new neural network architecture designed to respect some basic symmetry constraints in particle physics. PELICAN delivers state-of-the-art results in a top-tagging benchmark despite its relatively low complexity. It also shows potential for more complex tasks such as momentum reconstruction, significantly outperforming an established non-ML approach from Ref. [12].

5 Acknowledgements

We would like to acknowledge Brian Nord and the Deep Skies Lab as a community of multi-domain experts and collaborators who've facilitated an environment of open discussion, idea-generation, and collaboration.

6 Broader Impacts Statement

This work will potentially have a positive impact on basic physics research: Specifically through the use of PELICAN as a method in particle physics measurements, and perhaps more broadly in providing further inspiration for the use of symmetry-respecting/physically-constrained neural network architectures in scientific research. PELICAN's ability to not only tag particles, but also accurately reconstruct their 4-momenta, opens up possibilities of improving precision measurements of the Standard Model and searches for new physics. This basic research, in turn, will likely continue to yield positive impacts through the innovation and development of new technologies that it drives. Beyond its potential influence in furthering basic high-energy physics research, this work is unlikely to have other societal impacts (good or bad), as PELICAN does not have clear direct applications outside of physics research, and in fact the use of symmetry-preserving architectures is already present to some extent in industry and government applications – specifically, the use of convolutional neural networks for image recognition.

References

[1] A. Bogatskiy, B. Anderson, J. T. Offermann, M. Roussi, D. W. Miller, and R. Kondor. "Lorentz Group Equivariant Neural Network for Particle Physics". In: *ICML 2020*. ICML, June 2020.

[2] A. Butter et al. "The Machine Learning Landscape of Top Taggers". *SciPost Phys.*, **7**, 014, 2019.

[3] M. Cacciari, G. P. Salam, and G. Soyez. "FastJet User Manual". *Eur. Phys. J. C*, **72**, 1896, 2012.

[4] T. Chen et al. "Only Train Once: A One-Shot Neural Network Training And Pruning Framework". In: *Adv Neural Inf Process Syst 34*. 19637–19651. 2021.

[5] C. J. N. Coelho Jr. et al. "Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors". *Nat. Mach. Intell.*, **3**:8, 675–686, 2021.

[6] J. Cogan, M. Kagan, E. Strauss, and A. Schwartzman. "Jet-Images: Computer Vision Inspired Techniques for Jet Tagging". *JHEP*, **02**, 118, 2015.

[7] G. Corso et al. "Principal Neighbourhood Aggregation for Graph Nets". In: *NeurIPS*. Vol. 33, 13260–13271. Curran, 2020.

[8] J. de Favereau et al. "DELPHES 3, A modular framework for fast simulation of a generic collider experiment". *JHEP*, **02**, 057, 2014.

[9] S. Gong et al. "An efficient Lorentz equivariant graph neural network for jet tagging". *JHEP*, **2022**:7, 30, 2022.

[10] T. M. Hong et al. "Nanosecond machine learning event classification with boosted decision trees in FPGA for high energy physics". *JINST*, **16**:08, P08016, 2021.

[11] Y. Iiyama et al. "Distance-Weighted Graph Neural Networks on FPGAs for Real-Time Particle Reconstruction in High Energy Physics". *Frontiers in Big Data*, **3**, 2021.

[12] D. E. Kaplan, K. Rehermann, M. D. Schwartz, and B. Tweedie. "Top Tagging: A Method for Identifying Boosted Hadronically Decaying Top Quarks". *Phys. Rev. Lett.*, **101**, 142001, 2008.

[13] G. Kasieczka, T. Plehn, J. Thompson, and M. Russel. "Top Quark Tagging Reference Dataset". 2019.

[14] P. T. Komiske, E. M. Metodiev, and J. Thaler. "Energy flow networks: deep sets for particle jets". *JHEP*, **2019**:1, 121, 2019.

[15] C. Li et al. "Does Lorentz-symmetric design boost network performance in jet physics?", 2022.

[16] D. Linthorne and D. Stolarski. "Triggering on emerging jets". *Phys. Rev. D*, **104**, 035019, 2021.

[17] I. Loshchilov and F. Hutter. Decoupled Weight Decay Regularization. 2017.

[18] J. T. Offermann, A. Bogatskiy, and T. Hoffman. "Top Quark Momentum Reconstruction Dataset". 2022.

[19] H. Pan and R. Kondor. "Permutation Equivariant Layers for Higher Order Interactions". In: *AISTATS*. 5987–6001. PMLR, 28–30 Mar 2022.

[20] T. Sjöstrand et al. "An introduction to PYTHIA 8.2". *Comput. Phys. Commun.*, **191**, 159–177, 2015.

[21] S. Villar, D. W. Hogg, K. Storey-Fisher, W. Yao, and B. Blum-Smith. "Scalars are universal: Equivariant machine learning, structured like classical physics". 2021.

[22] E. Wagstaff, F. Fuchs, M. Engelcke, I. Posner, and M. A. Osborne. "On the Limitations of Representing Functions on Sets". In: *ICML 2019*. Vol. 97. Proceedings of Machine Learning Research, 6487–6494. PMLR, 2019.

[23] H. Weyl. The Classical Groups. Their Invariants and Representations. 2nd ed. Princeton University Press, Princeton, N.J., 1946.

[24] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. "Deep Sets". In: *Adv Neural Inf Process Syst 30*, 3391–3401. Curran Associates, Inc., 2017.

[25] A. Zweig and J. Bruna. "Exponential Separations in Symmetric Neural Networks". 2022.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No] In this context, it's not exactly clear what the "limitations" would be – the network design is quite general for particle physics measurements, and we do not make any claims of using it outside of that domain.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] This work does not have any clear direct applications that would result in negative societal impact.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] A link to a GitHub repository with the network code is provided, and the dataset is cited with a Zenodo link – this Zenodo listing, in turn, provides a link to the GitHub repository for the code used to generate that dataset.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] For the classification benchmark, we provide explicit errors. For the (new) momentum reconstruction benchmark we do not provide these explicitly, but comment on their magnitude.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite the reference dataset for the top-tagging benchmark.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]