19 Parameters Is All You Need: Tiny Neural Networks 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

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

Abstract

As particle accelerators increase their collision rates, and deep learning solutions prove their viability, there is a growing need for lightweight and fast neural network architectures for low-latency tasks such as triggering. We examine the potential of one recent Lorentz- and permutation-symmetric architecture, PELICAN, and present its instances with as few as 19 trainable parameters that outperform generic architectures with tens of thousands of parameters when compared on the binary classification task of top quark jet tagging.

1 Introduction

Particle collisions at the Large Hadron Collider at CERN happen every 25 nanoseconds, producing immense amounts of data that have to be processed in real time. Much of the event filtering is done by the Level-1 trigger [1, 5], which uses algorithms implemented on FPGAs that need to operate at below-microsecond latency to avoid loss of valuable data. Low-latency tasks include charged particle track reconstruction and energy measurements. Implementing neural networks under such constraints is a significant challenge, however the most recent attempts to do so have finally surpassed their traditional non-ML counterparts. The current state-of-the-art implementations in this area are based on the JEDI-net Graph Neural Network (GNN) architecture, see [17, 23, 24]. The network input data consist of lists of jet constituents, with a certain number of geometric features describing each constituent.

GNN architectures are inherently permutation-equivariant, providing a significant boost to efficiency and model size by virtue of weight sharing, but no other physical symmetries are necessarily respected. Physics-informed architectures that are inherently equivariant with respect to rotational and Lorentz-boost symmetries have recently shown themselves to provide state-of-the-art performance at tasks such as jet tagging (see e.g. [2, 3, 4, 9, 12, 15]), and they do so despite the relatively small model size.

In this work we study the current state-of-the-art architecture for top-quark jet tagging, PELICAN [4]. It is fully Lorentz-invariant and its permutation-equivariant layers are based on the general higher-order permutation-equivariant mappings introduced in [16, 19]. The full reduction of all relevant symmetries allows small instances of PELICAN with just a few thousand parameters to perform on par with much larger models with hundreds of thousands or even millions of parameters.

Machine Learning and the Physical Sciences Workshop, NeurIPS 2023.

Moreover, the simplicity of the architecture presents a unique opportunity for explainability and even interpretability.

Our goal here is to explore the small model size limit of PELICAN and compare it against the previous state-of-the-art (and also Lorentz-equivariant) architecture, LorentzNet [9]. The benchmark task for this comparison is that of top-quark tagging due to the publicly available dataset [10] and the extensive prior exploration of architectures trained on it [12]. The input consists of a list of N 4-momenta of jet constituents, $\{p_i\}_{i=1}^N$ which PELICAN reduces to the $N \times N$ array of pairwise Lorentz-invariant dot products, $d_{ij} = p_i \cdot p_j$. Thus the reduced input is an array with one channel. We find that a stripped down version of PELICAN consisting of nothing but two linear permutation-equivariant blocks with just two channels in the hidden layer and exactly one nonlinear activation function in between outperforms generic architectures such as the fully-connected TopoDNN which has 59k parameters [12]. This model nominally has 26 parameters, but through absorption of multiplicative factors and a simplification of the output layer that number can be effectively reduced to 19. Despite the costly N^2 scaling of the memory that PELICAN requires, its symmetric architecture can provide ultra-lightweight networks that can be viably used in low-latency and high-throughput applications.

2 The original PELICAN architecture

The original PELICAN architecture consists of an input block which encodes the $N \times N$ array of pairwise dot products $\{d_{ij}\}$, followed by a sequence of so-called $\mathrm{Eq}_{2\to2}$ permutation-equivariant blocks (the index 2 indicates the rank of the input and output arrays) that produce transformed $N \times N$ arrays. Each of these blocks consists of a fully-connected "messaging" layer that mixes the channels but is shared among all components of the $N \times N$ array, and an "aggregation" layer that applies a general linear permutation-equivariant operation that exchanges information between the various components of the array. Finally, a similar $\mathrm{Eq}_{2\to0}$ block reduces the array to a permutation-invariant (rank 0) scalar, after which an output MLP layer produces the two binary classification weights $\{w_0, w_1\}$. The diagram below summarizes this architecture, see [4] for details.

$$\{d_{ij}\} \longrightarrow \mathrm{Emb} \longrightarrow [\mathrm{Eq}_{2\to2}]^L \longrightarrow \mathrm{Eq}_{2\to0} \longrightarrow \mathrm{MLP} \longrightarrow \{w_c\} \tag{1}$$

Notably, the aggregation step inside $\mathrm{Eq}_{2\to2}$, called $\mathrm{LinEq}_{2\to2}$, applies 15 different operations that provide a basis for the space of all linear permutation-equivariant transformations of rank 2 arrays, which temporarily increases the size of the activation by a factor of 15, marking the peak of PELICAN's memory utilization. This is followed by a trainable linear layer that applies $(C_{\mathrm{in}} \times 15) \times C_{\mathrm{out}}$ weights and adds two biases per channel (one bias added to the entire $N \times N$ array and one only to the diagonal), where C_{in} and C_{out} are the number of input and output channels. Similarly, $\mathrm{Eq}_{2\to0}$ involves only 2 aggregators (total sum and trace), a linear layer of shape $(C_{\mathrm{in}} \times 2) \times C_{\mathrm{out}}$, and one bias per channel.

3 nanoPELICAN architecture

In this section we simplify the PELICAN architecture to a single hidden layer and reduce parameters further based on symmetry arguments, which we refer to as nanoPELICAN (nPELICAN). The only two linear symmetric observables that can be constructed from the input dot products (assuming sum-based aggregation that does not explicitly depend on the multiplicity N) are N, the jet mass $m_J^2 = \sum_{i,j} d_{ij}$, and the total mass $\sum_i d_{ii}$. The top-tagging dataset has only massless constituents, so the latter observable is irrelevant. A non-parametric top-tagger based on a simple jet mass cut achieves an AUC of only 90.6%. A linear PELICAN, which outputs $p(N)m_J^2 + q(N)$ with some learned polynomials p and q, cannot far exceed this.

To this end, we set out to find the smallest and most interpretable modification of PELICAN that is still nonlinear and performs competitively on the top-tagging task. We thus omit the input embedding layer, all messaging layers, and the output MLP, and are left with just two linear equivariant blocks, $\mathrm{LinEq}_{2\to2}$ and $\mathrm{LinEq}_{2\to0}$, separated by a single activation function, which we choose to be ReLU. The architecture is summarized in the following diagram:

$$\{d_{ij}\} \longrightarrow \mathrm{LinEq}_{2\to2}^{\mathrm{nano}} \longrightarrow \mathrm{ReLU} \longrightarrow \mathrm{LinEq}_{2\to0} \longrightarrow \{w_c\}. \tag{2}$$

Here, we also notice that since the array of dot products, $\{d_{ij}\}$, is symmetric, and since the constituents in the top-tagging dataset are massless ($d_{ii} = 0$), many of the 15 basis aggregators in $\mathrm{LinEq}_{2\to2}$ are redundant. We remove 5 aggregators that depend only on the diagonal of the input, and one from each of 4 pairs of aggregators that attain the same value on symmetric inputs. We are left with just 6 aggregators which constitute $\mathrm{LinEq}^{\mathrm{nano}}_{2\to2}$. To help with training, each equivariant layer is still preceded by a `Dropout` layer. Moreover, keeping `BatchNorm` layers just before the dropout can also help the model converge, meanwhile the extra parameters from these layers can be almost completely absorbed into the linear layers for inference. Namely, the multiplicative weights of `BatchNorm` can be absorbed into the following $\mathrm{LinEq}_{2\to2}$, whereas the biases can be either left or absorbed into the biases of $\mathrm{LinEq}_{2\to2}$ at the cost of turning them into quadratic polynomials of N, adding 2 parameters per output channel. Since there are two distinct bias parameters per channel, such a `BatchNorm` effectively adds $\min\{C_{\mathrm{in}}, 4C_{\mathrm{out}}\}$ parameters. In the case of $\mathrm{LinEq}_{2\to0}$ there is only one bias parameter per channel, thus the number of added parameters is $\min\{C_{\mathrm{in}}, 2C_{\mathrm{out}}\}$.

The only remaining hyperparameter is C_{hidden}, the number of channels in the hidden layer (between $\mathrm{LinEq}_{2\to2}$ and $\mathrm{LinEq}_{2\to0}$). The total number of parameters is then $1 \times 6 \times C_{\mathrm{hidden}} + 2 \cdot C_{\mathrm{hidden}} + C_{\mathrm{hidden}} \times 2 \times 2 + 2 = 12C_{\mathrm{hidden}} + 2$ (ignoring `BatchNorm`). In addition, since for binary classification only the difference in weights $w_1 - w_0$ matters, it is possible to have only 1 output channel, in which case we have $10C_{\mathrm{hidden}} + 1$ parameters. The models presented below produce only one output weight called w. Leaving in the two `BatchNorm` layers effectively adds only 3 new parameters if $C_{\mathrm{hidden}} > 1$ and 2 otherwise. Finally, since we're using the ReLU activation, which is a homogenous function, one more multiplicative factor can be absorbed in each channel. The final number of parameters then is $9C_{\mathrm{hidden}} + 4$ for $C_{\mathrm{hidden}} > 1$ and 12 otherwise.

4 Top tagging performance

The top tagging dataset [10] consists of anti-k_T jets [6] corresponding with top quarks (signal) and light quarks or gluons (background). It includes up to 200 jet constituents per entry, each represented by a 4-momentum in Cartesian coordinates. A converted version of the dataset that can be directly used with PELICAN can be found at [11]. For our models we only use the 80 constituents with the highest transverse momentum $p_T = \sqrt{p_x^2 + p_y^2}$, which is typically enough to saturate our network's performance. We follow a training regime almost identical to that in [4], using an Nvidia H100 GPU. The only changes are that we disable weight decay, extend the training to 140 epochs (4 epochs of linear warm-up, 124 epochs of `CosineAnnealingLR` with $T_0 = 4$ and $T_{\mathrm{mult}} = 2$, and 12 epochs of exponential decay with $\gamma = 0.5$), and increase the batch size to 512. Training took about 30 ms per batch, and the evaluation took about 23 ms per batch (including overhead).

Table 1: Comparison of tiny top-taggers. Averaged over the top 5 (lowest loss) out of 25 random seeds. Uncertainty given by the standard deviation. $1/\epsilon_B$ is the background rejection at 30% signal efficiency.

Architecture	Accuracy	AUC	$1/\epsilon_B$	# Params
$\mathrm{LorentzNet}_{n_{\mathrm{hidden}}=3}$	0.907(2)	0.966(3)	174±44	120
$\mathrm{nPELICAN}_{C_{\mathrm{hidden}}=10}$	0.921(1)	0.9748(1)	327±20	101
$\mathrm{nPELICAN}_{C_{\mathrm{hidden}}=3}$	0.919(1)	0.9730(4)	256±12	31
$\mathrm{nPELICAN}_{C_{\mathrm{hidden}}=2}$	0.918(1)	0.9718(6)	243±18	21
$\mathrm{nPELICAN}_{C_{\mathrm{hidden}}=1}$	0.895(1)	0.950(2)	81±12	11

We train several models with C_{hidden} ranging from 1 to 10. For comparison, we also train instances of LorentzNet with only one message passing block and the number of channels in the hidden layers set to 3 and the batch size of 512 (no other changes to hyperparameters and training were made). The results are reported in Table 2. We report the accuracy, the area under the ROC, and the background rejection (inverse false positive rate) at 30% signal efficiency (true positive rate). For each architecture, we pick the model with the lowest cross-entropy loss out of 10 trained instances initialized with different random seeds.

We observe that nPELICAN achieves competitive AUC and background rejection with as few as 2 channels in the hidden layer. In fact, its AUC surpasses that of the fully-connected TopoDNN with 59k parameters [12] (its average accuracy was 0.929(1), AUC 0.964(14), and background rejection 424 ± 82). Moreover, the AUC of nPELICAN with 10 channels (101 parameters) is only about 1% behind that of ParticleNet (498k parameters) [21] and even ParT (2.1M parameters) [22]. Meanwhile,

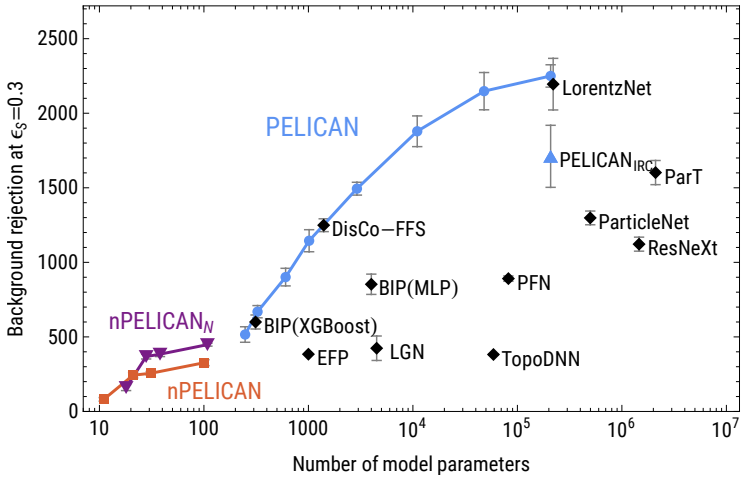

Fig. 1: Comparison of top-tagger background rejection performance at signal efficiency $\epsilon_S = 0.3$ as a function of the number of parameters in each model considered. Results other than nPELICAN are taken from refs. [2, 4, 7, 9, 12, 13, 14, 18, 20, 21, 22]. Note that the curve for the original PELICAN was obtained by varying only the network width, so only the rightmost point is fully optimized.

LorentzNet with one message-passing block lags far behind nPELICAN with a similar number of parameters. A visual comparison with many existing models is presented in Figure 1.

Table 2: Performance of nPELICAN$_N$ – nPELICAN with N^α-scaled aggregators. Metrics defined as in Table 2.

nPELICAN$_N$ width	Accuracy	AUC	$1/\epsilon_B$	# Params
$C_{\text{hidden}} = 10$	0.923(1)	0.9764(1)	448±10	108
$C_{\text{hidden}} = 3$	0.9214(3)	0.9752(2)	384±16	38
$C_{\text{hidden}} = 2$	0.9200(3)	0.9745(1)	368±17	28
$C_{\text{hidden}} = 1$	0.902(2)	0.960(2)	150±16	18

Interestingly, at such low depth the removal of fully connected messaging layers from PELICAN actually improved the performance. The element of the original network that can boost nPELICAN's performance the most with only a few new parameters is the N-dependent scaling of aggregators. In our tests, replacing sum aggregation with means led to very low performance of nPELICAN, however enabling PELICAN's original flexible scaling of the means by an extra factor of $N^\alpha/\bar{N}^\alpha$ turns out to be very beneficial, see Table 2.

5 Interpreting nanoPELICAN

Considering the extremely low complexity and relatively high performance of nPELICAN, there is high potential for a full interpretation of the model. Before attempting to interpret the weights, it is crucial to minimize any redundancies. In particular, since `ReLU` is a homogenous function, one multiplicative factor from the weights in LinEq$_{2\to0}$ can be absorbed into the weights and biases of LinEq$_{2\to2}$ in each channel of the hidden layer. For the model with $C_{\text{hidden}} = 2$ this means that the number of parameters can effectively be reduced to 19. Explicitly, the model can be written analytically as

$$w = b^{2\to0} + \sum_{h=1}^{C_{\text{hidden}}} c_{0h}^{2\to0} \frac{1}{\bar{N}^2} \sum_{i,j} \text{ReLU}\left(\sum_{b=1}^{6} c_{bh}^{2\to2} \text{Agg}_b(d)_{ij} + b_h^{2\to2} + b_{\text{diag},h}^{2\to2} \delta_{ij} \right) + \\ + \sum_{h=1}^{C_{\text{hidden}}} c_{1h}^{2\to0} \frac{1}{\bar{N}} \sum_{i=j} \text{ReLU}\left(\sum_{b=1}^{6} c_{bh}^{2\to2} \text{Agg}_b(d)_{ij} + b_h^{2\to2} + b_{\text{diag},h}^{2\to2} \delta_{ij} \right). \quad (3)$$

Here, w is the output score (the jet is tagged as a top quark if $w > 0$); $c^{2\to2}$, $b^{2\to2}$, and $b_{\text{diag}}^{2\to2}$ are the weights and biases of LinEq$_{2\to2}$; $c^{2\to0}$ and $b^{2\to0}$ are the weights and the bias of LinEq$_{2\to0}$; index b enumerates the 6 aggregators of LinEq$_{2\to2}$; index h enumerates the channels in the hidden layer. $\bar{N}$ is a hyperparameter that is used to control the magnitude of the sums over constituents, here set to be 49 (it is similarly used inside the aggregators Agg$_b$).

Therefore the ReLU effectively sets a linear constraint on the dot products, and the output takes a sum over only those pairs (i, j) that satisfy the constraint. More explicitly, denoting the jet momentum by $J = \sum_i p_i$, the argument of the ReLU is a linear combination of relative masses ($m_{ij}^2 = -(p_i - p_j)^2 = 2d_{ij}$), jet-frame masses $p_i \cdot J$, $p_j \cdot J$, the jet mass $m_J^2 = \sum_{ij} d_{ij}$, and a constant term. In addition, we found that these parameters are stable across multiple random initializations, indicating that they can be directly interpreted as unique physical constraints that encode Lorentz-invariant quantities such as the top quark mass, which we intend to elucidate in future work.

6 Conclusions

We have presented nPELICAN, a miniaturized a version of the PELICAN architecture, which is both surprisingly performant relative to much larger networks and can be rewritten simply as a constraint on Lorentz invariant quantities with a single ReLU activation function. This represents a novel development in interpretability for neural networks in particle physics, and gives hope for the interpretability of much larger networks. Future studies will exploit the stability of the nPELICAN parameters to determine their dependencies on features of the training data such as jet energies and particle masses, as well as relating these parameters to traditional, discriminating kinematic variables for jet-tagging, such as jet constituent multiplicity, subjet multiplicity [25] and jet shapes [8]. The code can be found at https://github.com/abogatskiy/PELICAN-nano.

References

[1] G. Aad et al. "Performance of the ATLAS Level-1 topological trigger in Run 2". *Eur. Phys. J. C*, **82**:1, 7, 2022.

[2] 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.

[3] A. Bogatskiy, T. Hoffman, D. W. Miller, and J. T. Offermann. "PELICAN: Permutation Equivariant and Lorentz Invariant or Covariant Aggregator Network for Particle Physics". 2022.

[4] A. Bogatskiy, T. Hoffman, D. W. Miller, J. T. Offermann, and X. Liu. "Explainable Equivariant Neural Networks for Particle Physics: PELICAN". 2023.

[5] W. Buttinger. "The ATLAS Level-1 Trigger System". On behalf of the ATLAS collaboration, 2012.

[6] M. Cacciari, G. P. Salam, and G. Soyez. "The anti-k_t jet clustering algorithm". *JHEP*, **04**, 063, 2008.

[7] R. Das, G. Kasieczka, and D. Shih. Feature Selection with Distance Correlation. 2022.

[8] S. D. Ellis, Z. Kunszt, and D. E. Soper. "Jets at hadron colliders at order $\alpha - s^{3:}$ A Look inside". *Phys. Rev. Lett.*, **69**, 3615–3618, 1992.

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

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

[11] G. Kasieczka, T. Plehn, J. Thompson, and M. Russel. "Converted Top Tagging Dataset". 2020.

[12] G. Kasieczka et al. "The Machine Learning landscape of top taggers". *SciPost Phys.*, **7**, 014, 2019.

[13] P. T. Komiske, E. M. Metodiev, and J. Thaler. "Energy flow polynomials: A complete linear basis for jet substructure". *JHEP*, **04**, 013, 2018.

[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] H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman. "Invariant and Equivariant Graph Networks". 2018.

[17] E. A. Moreno et al. "JEDI-net: a jet identification algorithm based on interaction networks". *Eur. Phys. J. C*, **80**:1, 58, 2020.

[18] J. M. Munoz, I. Batatia, and C. Ortner. "Boost invariant polynomials for efficient jet tagging". *Machine Learning: Science and Technology*, **3**:4, 04LT05, 2022.

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

[20] J. Pearkes, W. Fedorko, A. Lister, and C. Gay. "Jet Constituents for Deep Neural Network Based Top Quark Tagging". 2017.

[21] H. Qu and L. Gouskos. "Jet tagging via particle clouds". *Phys. Rev. D*, **101**:5, 2020.

[22] H. Qu, C. Li, and S. Qian. "Particle Transformer for Jet Tagging". 2022.

[23] Z. Que, M. Loo, H. Fan, M. Pierini, A. Tapper, and W. Luk. "Optimizing Graph Neural Networks for Jet Tagging in Particle Physics on FPGAs". In: *2022 32nd International Conference on Field-Programmable Logic and Applications (FPL)*. 327–333. 2022.

[24] Z. Que et al. "LL-GNN: Low Latency Graph Neural Networks on FPGAs for High Energy Physics". 2023.

[25] J. Thaler and K. Van Tilburg. "Identifying Boosted Objects with N-subjettiness". *JHEP*, **03**, 015, 2011.