Lorentz Group Equivariant Neural Network for Particle Physics: Supplementary Material

Alexander Bogatskiy [1] **Brandon Anderson** [2 3] **Jan T. Offermann** [1] **Marwah Roussi** [1] **David W. Miller** [1 4] **Risi Kondor** [2 5]

1. Clebsch-Gordan (CG) coefficients

Here we provide further details on the Clebsch-Gordan decompositions for SU(2) and SL(2, $\mathbb{C}$) and their computer implementation. A good reference for this material is the book by Gelfand et al. (1963), however that book contains some errors that lead to an incorrect expression for the CG coefficients of the Lorentz group. Since we are not aware of a reference with the correct formulas, we re-derive them here.

We first make a note about the inverse CG mapping for SU(2). By orthogonality of the CG mapping, we have its inverse

$$B^{-1} = B^T : \quad R_{l_1} \otimes R_{l_2} \to \bigoplus_{l=|l_1-l_2|}^{l_1+l_2} \tilde{R}_l,$$

so its components, defined by the formula

$$e_{l_1,m_1} \otimes e_{l_2,m_2} = \sum (B^{-1})^{l,m}_{l_1,m_1;l_2,m_2} \tilde{e}_{l,m},$$

are given by

$$(B^{-1})^{l,m}_{l_1,m_1;l_2,m_2} = B^{l_1,m_1;l_2,m_2}_{l,m}.$$

Thus the inverse transformation of the components of vectors (which is the one we actually need in the network) reads

$$\tilde{v}^{l,m} = \sum B^{l_1,m_1;l_2,m_2}_{l,m} v^{l_1,m_1;l_2,m_2}.$$

This replaces the incorrect formula obtained in (Gelfand et al., 1963, I.§10.4 (p. 152)), which also propagated into their derivation for the Lorentz group. Now, following the derivation in (Gelfand et al., 1963, II.§6.2) with the help of the corrected formula (1), we find the formula presented in the body of this paper:

$$H^{(k_1,n_1),l_1,m_1;(k_2,n_2),l_2,m_2}_{(k,n),l,m} = \sum_{m'_1,m'_2} B^{\frac{k}{2},m'_1+m'_2;\frac{n}{2},m-m'_1-m'_2}_{l,m} B^{\frac{k_1}{2},m'_1;\frac{k_2}{2},m'_2}_{\frac{k}{2},m'_1+m'_2} B^{\frac{n_1}{2},m_1-m'_1;\frac{n_2}{2},m_2-m'_2}_{\frac{n}{2},m-m'_1-m'_2} \times \times B^{\frac{k_1}{2},m'_1;\frac{n_1}{2},m_1-m'_1}_{l_1,m_1} B^{\frac{k_2}{2},m'_2;\frac{n_2}{2},m_2-m'_2}_{l_2,m_2}.$$

For computational purposes, it is convenient to store an element $v^{(k,n)}$ of an irrep $T^{(k,n)}$ as a single column-vector with the combined index $M = (l, m)$ where $l = \frac{|k-n|}{2}, \ldots, \frac{k+n}{2}$ with indices sorted over l first and over m last. We can thus work with vectors

$$v \in T^{(k,n)} \quad \text{has components } v^M,\ M = 1, \ldots, (k+1)(n+1).$$

Similarly, the CG matrix corresponding to the (k, n) sector of the $(k_1, n_1) \otimes (k_2, n_2)$ product is a rectangular matrix of size $(k_1 + 1)(n_1 + 1)(k_2 + 1)(n_2 + 1) \times (k + 1)(n + 1)$ which can be stored as a rank 3 tensor of size $(k_1 + 1)(n_1 + 1) \times (k_2 + 1)(n_2 + 1) \times (k + 1)(n + 1)$:

$$\underset{(k,n)}{{}^{(k_1,n_1),(k_2,n_2)}}H : \quad T^{(k,n)} \to T^{(k_1,n_1)} \otimes T^{(k_2,n_2)}$$

$$\text{Components of } H : \quad \left(\underset{(k,n)}{{}^{(k_1,n_1),(k_2,n_2)}}H\right)^{M_1,M_2}_{M}.$$

2. Lorentz D-matrices

As was described in the paper, the irreps of SL(2, $\mathbb{C}$) can be constructed as tensor products of pairs of irreps of SU(2), that is, of pairs of Wigner-D matrices:

$$D^{k/2}(\alpha, \beta, \gamma) \otimes \overline{D^{n/2}(-\alpha, \beta, -\gamma)}.$$

However, as written these matrices act on the space $T^{(k,0)} \otimes T^{(0,n)}$ and not $T^{(k,n)}$. Since we actually want to represent these matrices in the canonical basis of the $T^{(k,n)}$ irrep, we need to conjugate the tensor product with a matrix of CG

[1]Department of Physics, University of Chicago, Chicago, IL, U.S.A. [2]Department of Computer Science, University of Chicago, Chicago, IL, U.S.A. [3]Atomwise, San Francisco, CA, U.S.A. [4]Enrico Fermi Institute, Chicago, IL, U.S.A. [5]Flatiron Institute, New York, NY, U.S.A.. Correspondence to: Alexander Bogatskiy <bogatskiy@uchicago.edu>.

Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

coefficients:

$$D_{(k,n)}(\alpha,\beta,\gamma) = \left({}^{(k,0),(0,n)}_{\quad (k,n)}H\right)^T \cdot \cdot \left(D^{k/2}(\alpha,\beta,\gamma) \otimes \overline{D^{n/2}(-\alpha,\beta,-\gamma)}\right) \cdot \cdot \left({}^{(k,0),(0,n)}_{\quad (k,n)}H\right).$$

We are not aware of a conventional name for these matrices, so for lack of a better term we call them the *Lorentz D-matrices*. On $T^{(1,1)} \cong \mathbb{R}^4$, these are the familiar 4×4 Lorentz matrices, i.e. the *standard representation* of $\mathrm{SO}^+(1,3)$.

In our network, these matrices are used only to test Lorentz equivariance, but they can also be key elements of other similar architectures.

3. Orthogonality

Wigner D-matrices are unitary, but Lorentz D-matrices are neither unitary nor orthogonal (in fact it is known that the Lorentz group doesn't have any unitary finite-dimensional irreps). Therefore it is instructive to find a Lorentz-invariant bilinear form on all irreps. Clearly on $T^{(1,1)}$ it is the Minkowski dot product, and on other integer-spin irreps it can be induced from $T^{(1,1)}$ via tensor powers. However, invariant forms actually exist on all irreps of $\mathrm{SL}(2,\mathbb{C})$. There is a prototype of a Lorentzian metric on the 2-dimensional space $R_{1/2} \cong \mathbb{C}^2$ of spinors:

$$g_{1/2} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

It is not Hermitian because we will be using it as a bilinear form and not as a sesquilinear form, that is, no complex conjugation is used in its definition:

$$\langle \psi, \psi' \rangle = \psi_+ \psi'_- - \psi_- \psi'_+.$$

Here, $\psi = (\psi_+, \psi_-) \in \mathbb{C}^2$. Thus the form can be equally viewed either as an exterior 2-form $\omega_{1/2}$ or as a pseudo-Hermitian metric $i\omega_{1/2}(\bar{\psi}, \psi')$. This form naturally induces invariant forms/metrics on all higher spin irreps of SU(2) (these forms are symmetric on integer-spin irreps). The isomorphism of representations

$$R_l \cong \left(R_{1/2}^{\otimes 2l}\right)_{\mathrm{sym}}$$

induces the forms

$$g_l = \left(g_{1/2}^{\otimes 2l}\right)_{\mathrm{sym}},$$

where the symmetrization is done separately over the two $2l$-tuples of indices. It is easy to see that in the canonical basis

$$(g_l)_{m,m'} = (-1)^{l+m}\delta_{m+m',0}.$$

For example, g_1 is exactly the negative of the standard Euclidean metric on $\mathbb{R}^3$.

Similarly, on the 2-dimensional irreps $(1,0)$ and $(0,1)$ of $\mathrm{SL}(2,\mathbb{C})$, we choose the same form $g_{(1,0)} = g_{(0,1)} \coloneqq g_{1/2}$. Now the tensor product decomposition $T^{(k,n)} \cong \left(T^{(1,0)}\right)^{\otimes k} \otimes \left(T^{(0,1)}\right)^{\otimes n}$ induces the form

$$g_{(k,n)} = \left(g_{1/2}\right)_{\mathrm{sym}}^{\otimes k} \otimes \left(g_{1/2}\right)_{\mathrm{sym}}^{\otimes n}.$$

Another CG map can be applied to represent this product in the canonical basis, and the result is exactly the same as for SU(2) on each fixed-l subspace:

$$\left(g_{(k,n)}\right)_{(l,m),(l',m')} = (-1)^{l+m}\delta_{l,l'}\delta_{m+m',0}.$$

For instance, $g_{(1,1)}$ is precisely the standard Lorentzian metric on $\mathbb{R}^4$.

CG products and D-matrices respect these forms in the sense that tensor products of two such forms generate the same forms, and D-matrices are orthogonal with respect to them (here we write this out for $\mathrm{SL}(2,\mathbb{C})$ since SU(2) can be considered a special case by setting $n = 0$):

$$g_{(k_1,n_1)} \otimes g_{(k_2,n_2)} = \bigoplus_{(k,n)} g_{(k,n)},$$
$$D^T_{(k,n)} g_{(k,n)} D_{(k,n)} = g_{(k,n)}.$$

Note that we use transposition instead of Hermitian conjugation because we treat the metric as $\mathbb{C}$-bilinear.

4. Equivariant Universal Approximation

This section provides more details on the derivation of the equivariant universal approximation theorem stated in the body of the paper.

Recall that a polynomial $f : V \to \mathbb{R}$ is called a polynomial G-invariant if $f(g \cdot v) = f(v)$ for all $g \in G, v \in V$. Similarly, a map $\widetilde{f} : V \to U$ between two representations is called a polynomial equivariant if it is equivariant and $l \circ \widetilde{f}$ is a polynomial for any linear functional $l : U \to \mathbb{R}$. Hilbert's finiteness theorem (Hilbert, 1890; 1893) states that for completely reducible representations V and U, the ring of polynomial invariants $f : V \to \mathbb{R}$ is finitely generated by a set $\{f_1, \ldots, f_{N_{\mathrm{inv}}}\}$. Similarly, all polynomial equivariants $\widetilde{f} : V \to U$ constitute a finitely generated module over the ring of invariants by a basis set $\{\widetilde{f}_1, \ldots, \widetilde{f}_{N_{\mathrm{eq}}}\}$ (Worfolk, 1994). By an extension of a standard universal approximation theorem, it was shown in (Yarotsky, 2018) that for completely reducible representations, any continuous equivariant map can be approximated by a single-layer perceptron with a non-polynomial activation function σ, with the invariant generators as inputs and the equivariant generators

as coefficients of the outputs. That is, there is a complete system consisting of the functions

$$\widetilde{f}_i(v) \cdot \sigma\left(\sum_{j=1}^{N_{\text{inv}}} w_{ij} f_j(v) + b_i\right), \quad i = 1, \ldots, N_{\text{eq}},$$

where each of the weights w_{ij}, b_i spans the real line.

Therefore our network, aside from including traditional nonlinear layers acting on polynomial invariants, has to generate the basis of polynomial invariants $\{f_i\}$ and equivariants $\{\widetilde{f}_j\}$.

To talk about neural networks, we adopt the definition of feed-forward neural networks from (Kondor & Trivedi, 2018):

Definition 4.1. Let $J_0, \ldots, J_L$ be a sequence of index sets, $V_0, \ldots, V_L$ vector spaces, $\phi_0, \ldots, \phi_L$ linear maps $\phi_k : V_{k-1}^{J_{k-1}} \to V_k^{J_k}$, and $\sigma_k : V_k \to V_k$ appropriate potentially nonlinear functions (acting pointwise in the sense that they are independent of the index in J_k). The corresponding *multilayer feed-forward neural network* is then a sequence of maps $f_0, f_1, \ldots, f_L$, where $f_k = \sigma_k \circ \phi_k \circ f_{k-1}$.

Now we define an equivariant analog of a feed-forward neural network.

Definition 4.2. Let G be a group. Let $V_0, \ldots, V_{2L}$ be finite-dimensional vector spaces that are also linear representations of G, $\sigma_k : V_k \to V_{k+1}$, $k = 0, 2, \ldots, 2(L-1)$, – potentially nonlinear G-equivariant maps, and $\phi_k : V_k \to V_{k+1}$, $k = 1, 3, \ldots, 2L-1$, – G-equivariant linear maps. Then the corresponding *G-equivariant multilayer feed-forward neural network* is the sequence of maps $f_0, \ldots, f_L$, where $f_k = \phi_{2k+1} \circ \sigma_{2k} \circ f_{k-1}$.

Definition 4.3. A *polynomial G-equivariant feed-forward neural network* is a G-equivariant one in the sense of Def. 4.2 in which all nonlinearities σ_k are polynomial. Specifically, all such σ_k can be expressed using tensor products and G-equivariant linear maps. A minimal example with a quadratic nonlinearity is $\sigma_k(v) = v \oplus (v \otimes v)$.

Lemma 4.1. *If $\sigma : V \to U$ is a polynomial G-equivariant map of degree d between two completely reducible finite-dimensional representations V, U of G, then there exist G-equivariant maps $\alpha_p : V^{\otimes p} \to U$, $p = 0, \ldots, d$, such that*

$$\sigma = \sum_{p=0}^{d} \alpha_p \left(v^{\otimes p}\right). \tag{1}$$

Proof. Decompose σ into homogeneous components $\sigma = \sum_{i=0}^{d} p_i$. Since the action of G is linear, each p_i separately is G-equivariant: $p_i(\rho_V(g) \cdot v) = \rho_U(g) \cdot p_i(v)$. Thus, without loss of generality, we can assume that σ is homogeneous.

If σ is homogeneous of degree d, it can be written as

$$\sigma(v) = p(\underbrace{v, \ldots, v}_{d})$$

for some symmetric d-multilinear map $p : V^d \to U$. Such a multilinear map is identified with an element of the tensor product space

$$t \in S^d(V^*) \otimes U, \tag{2}$$

where $S^d(V^*) = (V^*)^{\otimes d}_{\text{Sym}}$ is the symmetric tensor power of V^*. Therefore all polynomial equivariants on V are indeed tensor polynomials, i.e. p can be viewed as a *linear* equivariant map $p : V^{\otimes d} \to U$. Since this tensor is symmetric, this proves the existence of a linear equivariant α_d such that $\sigma(v) = \alpha_d\left(v^{\otimes d}\right)$. □

Lemma 4.2. *Given two completely reducible finite-dimensional representations of a group G, the space of polynomial G-equivariant maps from V to U is isomorphic to the subspace of invariants in the tensor product $S(V^*) \otimes U$, where $S(V^*)$ is the symmetric tensor algebra over V^*:*

$$\text{Pol}_G(V, U) \cong \left((S(V^*) \otimes U\right)^G.$$

Proof. As shown in the proof of Lemma 4.1, there is an isomorphism with the space of G-equivariant linear maps mapping $S(V^*) \to U$:

$$\text{Pol}_G(V, U) \cong \text{Hom}_G(S(V), U).$$

Since the hom-functor is the adjoint of the tensor product functor, we have

$$\text{Hom}_G(S(V), U) \cong \text{Hom}_G(S(V) \otimes U^*, \mathbb{R}) = (S(V^*) \otimes U)^G.$$

See also (Miller, 1971). □

Remark 4.1. The computation of this space clearly comes down to finding an isotypic decomposition of the tensor algebra over V (we expand on this in Remark 4.2). The isotypic decomposition of the symmetric tensor algebra $S(V^*)$ thus provides a complete system of polynomial equivariants. Namely, assuming without loss of generality that U is irreducible, any $\sigma \in \text{Pol}_G(V, U)$ can be written as in (1), where each α_p is a composition $\alpha_p = \beta_p \circ P_U^p$ of the projector $P_U^p : V^{\otimes p} \to U^\tau$ onto the U-type isotypic component of $V^{\otimes p}$ and a G-equivariant linear map $\beta_p : U^\tau \to U$.

These lemmas imply that the seemingly nonlinear problem of constructing all polynomial equivariants on V can be reduced to the *linear* problem of computing the isotypic decompositions of tensor powers of V. We now state more precisely our equivariant approximation theorem.

Theorem 4.1. *Let G be a classical Lie group and V, U two completely reducible finite-dimensional representations of G. Then any continuous equivariant map $F : V \to U$ can be uniformly approximated by equivariant feed-forward neural networks in the sense of Def. 4.2, in which all nonlinearities are based on tensor products, except perhaps when acting on G-invariants. For example, given a non-polynomial function $\tilde{\sigma}_k : \mathbb{R} \to \mathbb{R}$, we can have*

$$\sigma_k(v) = \tilde{\sigma}_k\left(P_{\text{inv}}(v)\right) \oplus v \oplus (v \otimes v), \tag{3}$$

where P_{inv} is the projector onto invariants and the action of $\tilde{\sigma}_k$ on the vector of invariants is component-wise.

Proof. This theorem follows immediately from Remark 4.1. Indeed, Yarotsky (2018) showed that, given a basis of polynomial invariants and equivariants, a conventional neural network can uniformly approximate an equivariant function. We have further demonstrated that such a basis can be generated up to an arbitrary polynomial degree by an equivariant feed-forward neural network which can construct all possible tensors of the inputs and compute the isotypic components of these tensors. A nonlinearity such as (3) iterated sufficiently many times constructs a basis for all tensors of v and applies scalar nonlinearities to all G-invariants. □

Remark 4.2. Here we further specify the form of the equivariant tensors constructed above. Since V admits a decomposition into a direct sum $V \cong \bigoplus_i V_{\alpha_i}$ of irreps R_{α_i} labeled by their highest weight α_i, then an equivariant $\tilde{f} : V \to U$ viewed as a function of several vectors $f(v_1, v_2, \ldots)$ with $v_i \in V_{\alpha_i}$, has to be a homogeneous polynomial of some degree k_i in each v_i. As shown in the Lemmas above, this allows one to view $\tilde{f}$ as a *multilinear* U-valued function t of $\sum_i k_i$ vectors, where each v_i is repeated k_i times:

$$\tilde{f}(v_1, v_2, \ldots) = t(\underbrace{v_1, \ldots, v_1}_{k_1 \text{ times}} \underbrace{v_2, \ldots, v_2}_{k_2 \text{ times}}, \ldots).$$

Just like in the proof of Lemma 4.1, this multilinear function can be interpreted as an element of the symmetric tensor product

$$t \in \left(\bigotimes_i \left(R^*_{\alpha_i}\right)^{\otimes k_i}_{\text{Sym}} \right) \otimes U. \tag{4}$$

Assuming without loss of generality that $U = R_\alpha$ is an irrep, the problem of constructing all equivariants $V \to R_\alpha$ is reduced to computing the R_α-isotypic component of this tensor algebra.

More information on these constructions in classical invariant theory can be found in e.g. (Goodman & Wallach, 2009) and (Weyl, 1946). As a side note, we restate the following classical theorem (Goodman & Wallach, 2009, Thm. 5.5.21):

Theorem. *If G is a classical Lie group, say,* SU(2), SL(2, $\mathbb{C}$), SO(3), *or* $\text{SO}^+(1,3)$, *and V is its fundamental representation (of dimension 2, 2, 3, and 4, respectively), then any finite-dimensional irrep of G occurs as a G-invariant subspace of the tensor power $V^{\otimes k}$ for a sufficiently high k.*

Considering the case of the Lorentz group, taking all possible tensor products of input 4-vectors and decomposing into irreducibles we will generate tensors that transform under arbitrary irreps of the group. Therefore there are no restrictions on the type of equivariant outputs that our architecture can produce. In fact, the dimensions of the spaces of equivariants mapping a set of 4-vectors to an irrep $U = R_\alpha$ of the Lorentz group are known (Miller, 1971).

5. Equivariance Tests

We have conducted experiments to verify Lorentz invariance of our neural network. The network itself had exactly the same values of hyper-parameters as in the main application, but the inputs were replaced by random 4-momenta, whose components are drawn uniformly from $[-1, 1]$, with 20 particles in each event and 20 events in a batch. The outputs of the network are then arrays w of shape 2×20. We compute the outputs for the same 4-momentum inputs with and without a Lorentz matrix applied to them at the start. Calling these two outputs w and $\tilde{w}$, we define the relative deviation as $\text{mean}(w - \tilde{w})/\text{mean}(w)$. We computed these quantities for a number of Lorentz boosts with varying Lorentz factor γ and averaged the results over 10 sets of random inputs and random initializations of the model (60 events with 20 particles each in total). The computations here are done using double precision and the relative error remains within 0.1% up to gamma factors of about 5000, which well covers the physically relevant domain of about $[10, 200]$. When using 32 bit precision, the error remains this low only up to $\gamma \sim 70$ and grows to over 10% after $\gamma \sim 200$.

Similarly we have tested rotational invariance, however the error is remains strictly of the order 10^{-16} when using double precision (the Euler angle of the rotation ranged from 0 to 10), so we are not showing a separate plot for it. It is clear that the source of the error is just the rounding errors in float arithmetic, so larger inputs produce larger relative errors. That is why applying large boosts increases the error, but rotations do not have the same effect.

Finally, the internal equivariance of the network was tested as well by applying Lorentz matrices to the inputs and comparing the values of the resulting activations of the network to an application of corresponding Lorentz D-matrices to them. The errors are similarly small, so we do not show separate statistics for them.

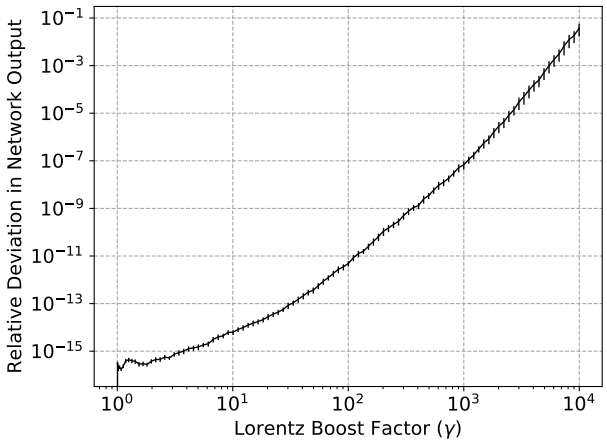

Figure 1. Relative deviation of the outputs of the network as a function of the boost factor γ applied to its inputs.

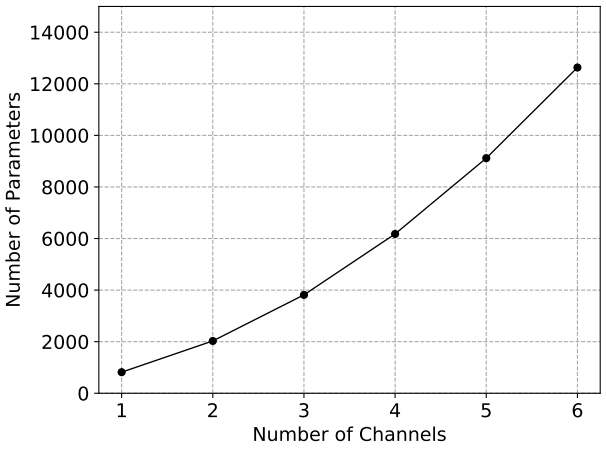

Figure 3. The number of network parameters as a function of the number of channels per layer, with 3 layers.

6. Computational Cost

Here we present the plots of the GPU memory (Fig. 2) and the number of parameters (Fig. 3) as functions of the number of channels (which here is uniform across all layers). These numbers correspond to the same model as the one trained for our main experiment, except for the modified number of channels. We note that the usage of GPU memory is much more efficient when the sizes of all tensors are multiples of 32. The size of most tensors is $2 \times B \times N_{\text{obj}}^{s} \times N_{\text{ch}} \times d$ with B being the batch size, N_{obj} the number of particles (202 for the top-tagging dataset), the power $s = 1$ or 2, and d the dimension of an irrep. The number of model parameters grows roughly quadratically with the number of channels.

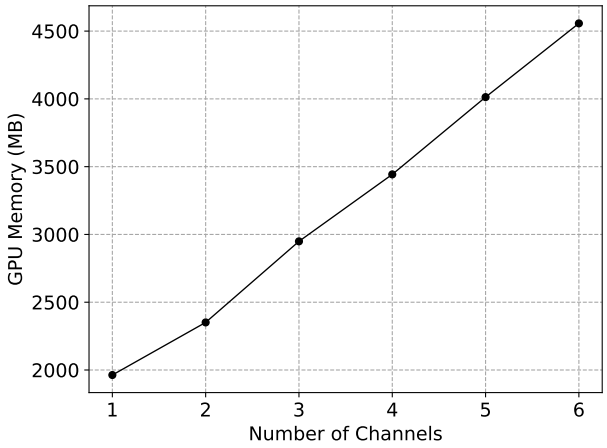

Figure 2. GPU memory usage as a function of the number of channels per layer, with 3 layers.

Since the sizes of some of the tensors involved grow quadratically with the number of particles N_{obj}, and we take tensor products of them, the evaluations of this model take a much longer time than simpler models. This can be mitigated by optimizing the tensor product operation. Namely, since Clebsch-Gordan coefficients satisfy several symmetry relations and "conservation laws", one may replace the tensor product followed by the CG operation with a single operation performed efficiently on the GPU. A custom CUDA kernel for this purpose is under development.

7. Network Metrics

Lastly, we display the evolution of some of the metrics of the network with the number of epochs – these were measured from the ensemble of networks from our main experiment. The accuracy (Fig 4) and AUC (Fig 5) score appear to reach a rough ceiling partway through training, whereas the background rejection (Fig 6) and loss (Fig 7) continue to improve throughout.

8. Source Code

The source code is available at `https://github.com/fizisist/LorentzGroupNetwork`. It requires PyTorch and CUDA for training on a GPU (not yet parallelized across multiple GPU's). It also uses NumPy and Scikit-Learn for some diagnostics, and H5py for reading data from HDF datasets.

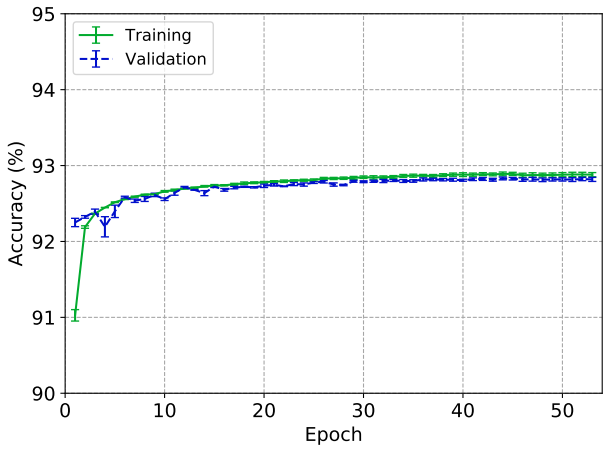

Figure 4. The average network accuracy as a function of epoch number, sampled over 4 independent trained instances. The two data series correspond with results from the training and validation subsets of the dataset (Kasieczka et al., 2019). The error bar width is given by the standard deviation.

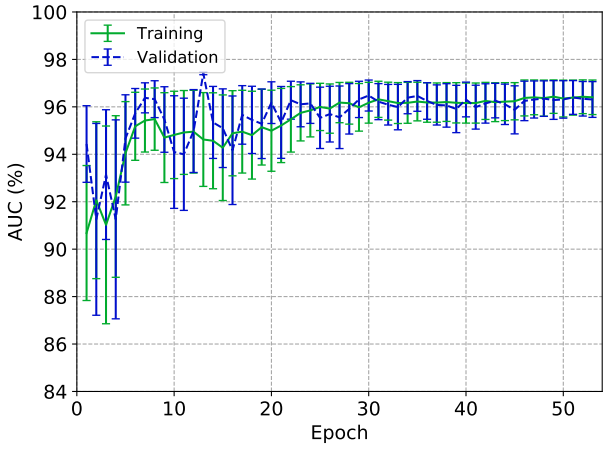

Figure 5. The average area under the ROC curve (AUC), as a function of epoch number. The error bar width is given by the standard deviation.

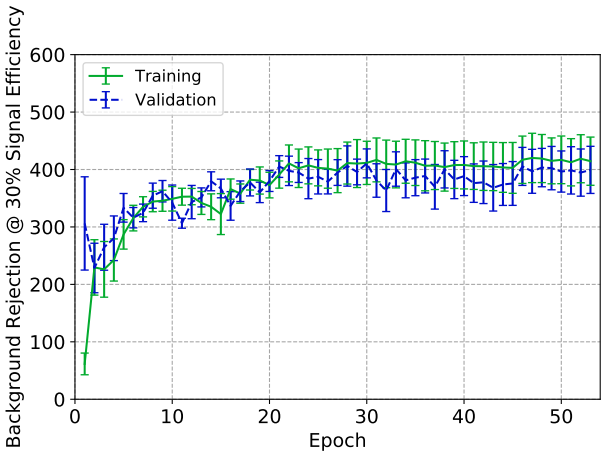

Figure 6. The average background rejection at 30% signal efficiency, as a function of epoch number. The error bar width is given by the standard deviation.

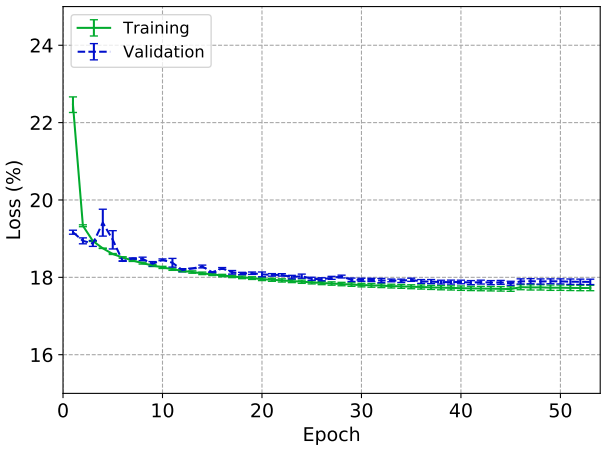

Figure 7. The average loss, as a function of epoch number. The error bar width is given by the standard deviation.

References

Gelfand, I. M., Minlos, R. A., and Shapiro, Z. Y. *Representations of the Rotation and Lorentz Groups and Their Applications*. Graduate Texts in Mathematics. Pergamon Press, 1963. ISBN 9780080100692.

Goodman, R. and Wallach, N. R. *Symmetry, representations, and invariants*, volume 255 of *Graduate Texts in Mathematics*. Springer, Dordrecht, 2009. ISBN 978-0-387-79851-6. doi: 10.1007/978-0-387-79852-3. URL https://doi.org/10.1007/978-0-387-79852-3.

Hilbert, D. Ueber die Theorie der algebraischen Formen. *Math. Ann.*, 36(4):473–534, 1890. ISSN 0025-5831. doi: 10.1007/BF01208503. URL https://doi.org/10.1007/BF01208503.

Hilbert, D. Ueber die vollen Invariantensysteme. *Math. Ann.*, 42(3):313–373, 1893. ISSN 0025-5831. doi: 10.1007/BF01444162. URL https://doi.org/10.1007/BF01444162.

Kasieczka, G., Plehn, T., Thompson, J., and Russel, M. Top quark tagging reference dataset, 2019. URL https://zenodo.org/record/2603256.

Kondor, R. and Trivedi, S. On the Generalization of Equivariance and Convolution in Neural Networks to the Action of Compact Groups. In Dy, J. and

Krause, A. (eds.), *Proceedings of the 35th ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2747–2755, Stockholm, Sweden, 7 2018. PMLR. URL http://proceedings.mlr.press/v80/kondor18a.html.

Miller, Jr., W. Invariant tensor fields in physics and the classical groups. *SIAM J. Appl. Math.*, 20:503–519, 1971. ISSN 0036-1399. doi: 10.1137/0120052. URL https://doi.org/10.1137/0120052.

Weyl, H. *The Classical Groups. Their Invariants and Representations*. Princeton University Press, Princeton, N.J., 2 edition, 1946. ISBN 0-691-07923-4.

Worfolk, P. A. Zeros of equivariant vector fields: algorithms for an invariant approach. *J. Symbolic Comput.*, 17(6): 487–511, 1994. ISSN 0747-7171. doi: 10.1006/jsco.1994.1031. URL https://doi.org/10.1006/jsco.1994.1031.

Yarotsky, D. Universal approximations of invariant maps by neural networks. *CoRR*, 2018. URL http://arxiv.org/abs/1804.10306.