Finding symmetry breaking order parameters with Euclidean neural networks

Tess E. Smidt [1,2,*] Mario Geiger,[1,3] and Benjamin Kurt Miller [1,2,4]

[1]*Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*
[2]*Center for Advanced Mathematics for Energy Research Applications (CAMERA), Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*
[3]*Department of Physics, École Polytechnique Fédérale de Lausanne, Lausanne 1015, Switzerland*
[4]*Department of Mathematics, Freie Universität Berlin, Berlin 14195, Germany*

Curie's principle states that "when effects show certain asymmetry, this asymmetry must be found in the causes that gave rise to them." We demonstrate that symmetry equivariant neural networks uphold Curie's principle and can be used to articulate many symmetry-relevant scientific questions as simple optimization problems. We prove these properties mathematically and demonstrate them numerically by training a Euclidean symmetry equivariant neural network to learn symmetry breaking input to deform a square into a rectangle and to generate octahedra tilting patterns in perovskites.

Machine learning techniques such as neural networks are data-driven methods for building models that have been successfully applied to many areas of physics, such as quantum matter, particle physics, and cosmology [1,2].

All machine learning models can be abstracted as a function f parametrized by learnable weights $W \in \mathbb{R}^N$ that maps vector space V_1 to another vector space V_2 (i.e., $f : V_1 \times W \to V_2$). Weights are updated by using a loss function which evaluates the performance of the model. In the case of neural networks which are differentiable models, the weights $w \in W$ are updated using the gradients of the loss $\mathcal{L}$ with respect to w, $w = w - \eta \frac{\partial \mathcal{L}}{\partial w}$ where η is the learning rate.

An important consideration for enhancing the performance and interpretability of these "black box" models when used for physics is how to incorporate axioms of symmetry [3–10]. Building symmetry into the model prevents the model from learning unphysical bias and can lead to new capabilities for investigating physical systems.

Symmetry invariant models only operate on invariant quantities (i.e., scalars), while symmetry equivariant models can preserve equivariant transformations (e.g., a change of coordinate system). A function is equivariant under symmetry group G if and only if the group action commutes with the function [i.e., for the group representation D_1 and D_2 acting on vector space V_1 and V_2, respectively, $f[D_1(g)x] = D_2(g)f(x)$, $\forall x \in V_1$ and $\forall g \in G$]. While equivariance is more general, invariant models are easier to build; most present-day symmetry-aware models are invariant. However, only symmetry equivariant models can fully express the richness of symmetry-related phenomena of physical systems (e.g., degeneracy and symmetry breaking).

Identifying sources of symmetry breaking is an essential technique for understanding complex physical systems. Many discoveries in physics have been made when symmetry implied something was missing (e.g., the first postulation of the neutrino by Pauli [11]); many physical phenomena are now understood to be consequences of symmetry breaking [12]: the mechanism that generates mass [13–15], superconductivity [16,17], and phase transitions leading to ferroelectricity [18].

In this Letter, we show how symmetry equivariant models can perform symmetry-related tasks without the conventional tools of symmetry analysis (e.g., character tables and related subgroup conventions). Using these networks, we can pose symmetry-related scientific questions as simple optimization problems without using explicit knowledge of the subgroup symmetry of the input or output. These networks can, for example, identify when data (input and output) are not compatible by symmetry, recover missing symmetry breaking information, find symmetry-intermediate solutions between a given input and target output, and build symmetry-compatible models from limited data.

These applications are possible due to two properties of symmetry equivariant neural networks that we prove in this Letter: (1) symmetry equivariant functions exhibit Curie's principle [19,20] and (2) gradients of an invariant loss function acting on both the network and target outputs can be used to recover the form (representation) of symmetry breaking information missing from the network input.

We organize this Letter as follows: First, we provide background on symmetry equivariant neural networks. Second, we prove the symmetry properties of the output and gradients of Euclidean symmetry equivariant neural networks and demonstrate them numerically by training a Euclidean neural

*tsmidt@lbl.gov

network to deform a square into a rectangle. Third, we use this technique on a more complex physical example, octahedral tilting in perovskites.

Euclidean neural networks are a general class of networks that has been explored by multiple groups [6–8] and build on previous work on building equivariances into convolutional neural networks [21–23].

The success of convolutional neural networks at a variety of tasks is due to them having translation equivariance (e.g., a pattern can be identified in any location). Euclidean neural networks are a subset of convolutional neural networks where the filters are constrained to be equivariant to three-dimensional (3D) rotations. To accomplish this, the filter functions are defined to be separable into a learned radial function and real spherical harmonics, $F_{c_{\text{in}}c_{\text{out}}Lm}(\vec{r}) = R^{(L)}_{c_{\text{in}}c_{\text{out}}}(|r|)Y_{Lm}(\hat{r})$, analogous to the separable nature of the hydrogenic wave functions, where L is the degree of the spherical harmonic, m is the order, and c_{in} and c_{out} are channel indices that will be described further below.

An additional consequence of Euclidean equivariance is that all "tensors" in a Euclidean neural network are geometric tensors; every component carries representation indices defining its transformation properties under rotation and parity. We express these geometric tensors in an irreducible representation (irrep) basis of $O(3)$, thus these coefficients carry representation indices (L, m) for rotation and p for parity. Irreps have parity $p = 1$ if even under spatial inversion and parity $p = -1$ if odd. Spherical harmonics which transforms as the irreps of $SO(3)$ have definite parity equal to $p = (-1)^L$.

To combine input and filter geometric tensors to produce new output, we must contract the representation indices of these two tensors and produce the representation indices of the output using the Clebsch-Gordan coefficients or Wigner $3j$ symbols (they are equivalent), forming the geometric tensor product. The point b feature tensor component, $I^{(b)}_{c_{\text{in}}L_{\text{in}}m_{\text{in}}p_{\text{in}}}$, has a channel index c_{in}, because there can be multiple instances of a given irrep, and representation indices $(L_{\text{in}}, m_{\text{in}}, p_{\text{in}})$. The filter tensor components for the relative distance vector $\vec{r}_{ab}$ of neighbors a and b, $F_{c_{\text{in}}c_{\text{out}}L_{\text{filter}}m_{\text{filter}}(p_{\text{filter}})}(\vec{r}_{ab})$, has similar indices except $p_{\text{filter}} = (-1)^{L_{\text{filter}}}$ and the component carries two channel indices; the number of filters is determined by the number of nontrivial paths between input and output representations. For simplicity, we rewrite these components with flattened representation indices as $I^{(b)}_{c_{\text{in}}L_{\text{in}}m_{\text{in}}p_{\text{in}}} = I^{(b)}_{c_{\text{in}}j}$ and $F_{c_{\text{in}}c_{\text{out}}L_{\text{filter}}m_{\text{filter}}p_{\text{filter}}}(\vec{r}_{ab}) = F_{c_{\text{in}}c_{\text{out}}k}(\vec{r}_{ab})$. Using Clebsch-Gordan coefficients contained in C, we obtain output

$$O^{(a)}_{c_{\text{out}}i} = \sum_{b\in(ab)} \sum_{c_{\text{in}}} \sum_{j} \sum_{k} C_{ijk} I^{(b)}_{c_{\text{in}}j} F_{c_{\text{in}}c_{\text{out}}k}(\vec{r}_{ab}), \qquad (1)$$

where $b \in (ab)$ denotes the sum over all neighbors b of a. Note, that the output, $O^{(a)}_{c_{\text{out}}i} = O^{(a)}_{c_{\text{out}}L_{\text{out}}m_{\text{out}}p_{\text{out}}}$, will be non-trivial for any $(L_{\text{out}}, m_{\text{out}}, p_{\text{out}})$ that satisfies the conditions $|L_{\text{in}} - L_{\text{filter}}| \leqslant L_{\text{out}} \leqslant L_{\text{in}} + L_{\text{filter}}$ and $p_{\text{out}} = p_{\text{in}} \times p_{\text{filter}}$. In practice, because the representation indices are nonrectangular, we flatten channel and representation indices into a single index and keep track of the irreps of each component using a separate list. The only conventions in these networks are how we choose to order channels from the tensor product and the

choice of the basis for the irreps of $O(3)$, which dictates the spherical harmonics and Wigner $3j$ symbols we use.

In our experiments, we use geometric tensors to express spatial functions; specifically the resulting coefficients from projecting a local point cloud onto spherical harmonics. This procedure is a common step in calculating rotation invariant descriptors of local atomic environments, such as smooth overlap of atomic positions kernels [4]. We treat a local point cloud S around a chosen origin as a set of δ functions, compute the relative distance vectors $\vec{r}$ from that origin, and evaluate the spherical harmonics at those corresponding angles (up to some maximum L, L_{max}). Then, we weight the spherical harmonic projection of each point $a \in S$ by its radial distance from the origin and finally, sum over all pointwise projections.

$$P_{Lm} = \sum_{a\in S} P_{aLm} = \sum_{a\in S} \|\vec{r}_a\| Y_{Lm}\left(\frac{\vec{r}_a}{\|\vec{r}_a\|}\right). \qquad (2)$$

The coefficients of this projection P_{Lm} form a geometric tensor with each coefficient tied to a specific spherical harmonic. The total number of coefficients is equal to the sum of all $2L + 1$ orders for all degrees L, $\sum_{L=0}^{L_{\text{max}}} 2L + 1 = (L_{\text{max}} + 1)^2$. Because spherical harmonics transform as the irreps of $SO(3)$ and Euclidean neural networks support irreps of $O(3)$, we can directly encode these coefficients as network output. From this tensor, we can then construct a signal on the sphere $f_S : S^2 \to \mathbb{R}$ by evaluating the spherical harmonics on the sphere, weighted by the corresponding coefficients, and summing over all contributions.

$$f_S(\hat{x}) = \sum_{a\in S} f_a(\hat{x}) = \sum_{a\in S} \sum_{L=0}^{L_{\text{max}}} \sum_{m=-L}^{L} P_{aLm} Y_{Lm}(\hat{x}). \qquad (3)$$

Finally, to achieve a spatial signal, we interpret the magnitude of the function f_S as the radial distance from the origin. When we construct the coefficients P_{aLm}, we rescale them to account for finite basis effects by ensuring the max of the function corresponds to the original radial distance $[f_a(\frac{\vec{r}_a}{\|\vec{r}_a\|}) = \|\vec{r}_a\|]$, hence why we have kept the a index in these expressions. The network predicts the coefficients summed over the point cloud S.

First, we prove that symmetry equivariant functions obey "Curie's principle." For a group G, vector space V, and a representation D, the symmetry group of $x \in V$ is defined as

$$\text{Sym}(x) = \{g \in G : D(g)x = x\}. \qquad (4)$$

Let $f : V_1 \to V_2$ be equivariant to group G. Curie's principle can be articulated as

$$\text{Sym}(x) \subseteq \text{Sym}[f(x)]. \qquad (5)$$

Proof. For $g \in \text{Sym}(x)$ [i.e., $D_1(g)x = x$],

$$D_2(g)f(x) = f[D_1(g)x] = f(x) \qquad (6)$$

$$\Rightarrow g \in \text{Sym}[f(x)]. \qquad (7)$$

■

According to Eq. (5), since Euclidean neural networks are equivariant to Euclidean symmetry, the symmetry of the output can only be of equal or higher symmetry than the input.
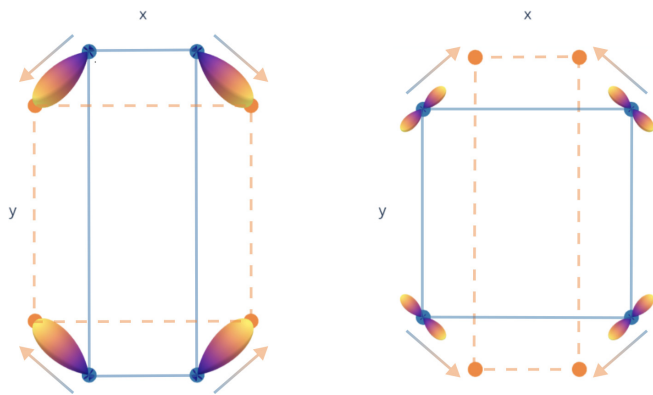
FIG. 1. Left: The predicted displacement signal for a network trained to deform the rectangle into the square. Right: The predicted displacement signal for the network trained to deform the square into the rectangle. Solid lines outline the input shape and dashed lines outline the desired shape. Arrows point from input geometry to desired geometry.

This implies that the network will also preserve any subgroup of Euclidean symmetry (e.g., point groups and space groups).

To demonstrate this, we train Euclidean neural networks to deform two arrangements of points in the xy plane into one another, one with four points at the vertices of a square, and another with four points at the vertices of a rectangle, shown as blue and orange points in Fig. 1.

To conduct our experiments, we use 3D Euclidean-symmetry-equivariant neural networks implemented in the framework E3NN [24] with PYTORCH [25]. The JUPYTER [26] notebooks used for running the experiments and creating the figures for this Letter are made available at Ref. [27]. These tasks, which each employ a single training example, achieve perfect accuracy in 100 iterations or less using a modest default network of (2×10^6) to (4×10^6) parameters (which could readily be reduced by using smaller networks for the radial functions) with three convolutional layers interspersed with gated nonlinearities (see Ref. [28]). The tasks can be trained in minutes on a modern personal computer. Further details are provided in the Supplemental Material [29].

We train each network to match the spherical harmonic projection of the desired displacement vector (i.e., final point location). As we will show, this representation is helpful for identifying degeneracies when they arise.

First, we train a Euclidean neural network to deform the rectangle into the square. This network is able to accomplish this quickly and accurately. Second, we train another Euclidean neural network to deform the square into the rectangle. No matter the amount of training, this network cannot accurately perform the desired task.

In Fig. 1, we show the output of the trained networks for both cases. On the right, we see that the model trained to deform the square into the rectangle is producing symmetric spherical harmonic signals each with two maxima. Due to being rotation equivariant, the network cannot distinguish distorting the square to form a rectangle aligned along the x axis from a rectangle along the y axis. The model automatically weighs symmetrically degenerate possibilities equally.

By Eq. (5), the output of the network has to have equal or higher symmetry than the input.

We emphasize here that the network does not "know" the symmetry of the inputs; the network predicts a degenerate answer simply because it is constrained to be equivariant. This is analogous to how physical systems operate and why physical systems exhibit Curie's prinicple.

Having a dataset where the "inputs" are higher symmetry than the "outputs" implies there is missing data—an asymmetry waiting to be discovered. In the context of phase transitions as described by Landau theory [18], symmetry breaking factors are called order parameters. To update its weights, a neural network is required to be differentiable, such that gradients of the loss can be taken with respect to every parameter in the model. This technique can be extended to the input; we use this approach to recover symmetry breaking order parameters.

To prove that this is possible, we must prove that the gradients of a G invariant scalar loss (such as the mean squared error) evaluated on the output of a G equivariant neural network $f(x)$ and ground truth data y_{true} (e.g., $\partial [f(x) - y_{\text{true}}]^2 / \partial x$) can have lower symmetry than the input.

The symmetry of the combined inputs to the invariant loss function is equal to or higher than the intersection of the symmetries of the predicted and ground truth outputs

$$\text{Sym}(x) \cap \text{Sym}(y) \subseteq \text{Sym}(\alpha x + \beta y) \qquad (8)$$
$$\forall x, y \in V, \alpha, \beta \in \mathbb{R}.$$

Proof. For $g \in \text{Sym}(x) \cap \text{Sym}(y)$,

$$D(g)(\alpha x + \beta y) = \alpha D(g)x + \beta D(g)y = \alpha x + \beta y. \qquad \blacksquare \quad (9)$$

Furthermore, if $\mathcal{L}$ is a differentiable and invariant function $\mathcal{L} : V \to \mathbb{R}$, then $\nabla \mathcal{L}$ is equivariant to G by the equivariance of differentiation

$$\nabla f[D(g)x] = [D(g)^{-T}] \nabla \mathcal{L}(x). \qquad (10)$$

Thus, if the symmetry of the ground truth output is lower than the input to the network, the gradients can have symmetry lower than the input, allowing for the use of gradients to update the input to the network to make the network input and output symmetrically compatible. This procedure can be used to find symmetry breaking order parameters missing in the original data but implied by symmetry.

Now, we demonstrate the symmetry properties of Euclidean neural networks according to Eqs. (8) and (10) can be used to learn symmetry breaking order parameters to deform the square into the rectangle.

In this task, we allow for additional inputs for each point of irreps $1_e \oplus 1_o \oplus 2_e \oplus 2_o \oplus 3_e \oplus 3_o \oplus 4_e \oplus 4_o$, where the number denotes the irrep degree L and the subscript denotes even and odd parity, respectively (e.g., 1_o transforms as a vector and 1_e transforms as a pseudovector). These irreps are initialized to be zero and we modify the training procedure. We require the input to be the same on each point, such that we learn a "global" order parameter. We also add an identical componentwise mean absolute error loss on each $L > 0$ component of the input feature to encourage sparsity. We train the network in the coordinate frame that matches the conventions of point group tables.
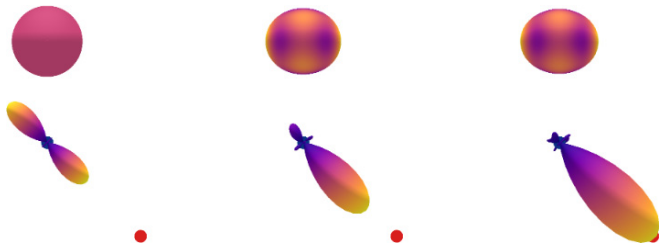
FIG. 2. Input parameters (top row) and output signal (bottom row) for one of the square vertices at (from left to right) the start, middle, and end of the model and order parameter optimization. For simplicity, we only plot components with the same parity as the spherical harmonics (2_e^2 and 4_e^2) as they require less familiarity with how parity behaves to interpret. The starting input parameter (on the left) is only a scalar of value 1 ($0_e^0 = 1$), hence it being a spherically symmetric signal. As the optimization procedure continues, the symmetry breaking parameters become larger, gaining contribution from components other than 0_e^0 and the model starts to be able to fit to the target output. When the loss converges, the input parameters have nonzero 2_e^2, 3_e^{-2}, 4_e^2, and 5_e^{-2} components with other nonscalar components close to zero and the model is able to fit to the target output.

We first train the model normally until the loss no longer improves. Then we alternate between updating the parameters of the model and updating the input using gradients of the loss. As the loss converges, we find that the input for $L > 0$ consists of nonzero order parameters comprising only 2_e^2, 3_e^{-2}, 4_e^2, and 5_e^{-2}, where the superscript denotes the order m of the irrep where $-L \leqslant m \leqslant L$. See Fig. 2 for images of the evolution of the input and output signals during the model and order parameter optimization process. The order parameters distinguish the x direction from the y direction while maintaining the full symmetry of the rectangle.

Our optimization returns four order parameters 2_e^2, 3_e^{-2}, 4_e^2, and 5_e^{-2} because the gradients cannot break the degeneracy between these equally valid order parameters. To recover only, for example, the 2_e^2 order parameter using Euclidean neural networks, we can do one of two things to break this degeneracy: limit the possible input order parameters (e.g., $1_e \oplus 1_o \oplus 2_e \oplus 2_o$) or add a loss that penalizes higher degree L order parameters. Thus, Euclidean neural networks can recover both the most general order parameters (including degeneracies) and more constrained order parameters (e.g., by using a custom loss function).

To arrive at this conclusion from the perspective of a conventional symmetry analysis, first, the symmetry of the square and rectangle must be identified as point group D_{4h} and point group D_{2h}, respectively. Second, the lost symmetries need to be enumerated; going from the square to the rectangle, eight symmetry operations are lost—two fourfold axes, two twofold axes, two improper fourfold axes, and two mirror planes. Then, the character table for the point group D_{4h} is used to find which direct sum of irreps break these symmetries. In this case, there is 1 one-dimensional irrep of D_{4h} that breaks all these symmetries—B_{1g}. The character table additionally lists that irrep B_{1g} has a basis function of $x^2 - y^2$ (i.e., 2_e^2) in the coordinate system with z being along the highest symmetry axis and x and y aligned with two of the mirror planes.
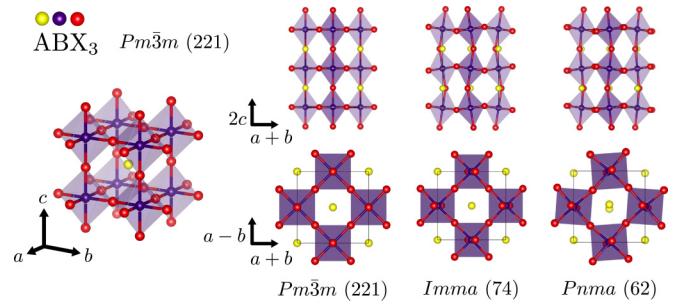

FIG. 3. Perovskite crystal structure with chemical formula of the form ABX_3 and parent symmetry of $Pm\bar{3}m$ (221). Octahedra can tilt in alternating patterns. This increases the size of the unit cell needed to describe the crystal structure. The larger unit cell directions are given in terms of the parent unit cell directions. The tilting of rotation axes for the $Pnma$ (62) structure is made of a 3D "checkerboard" of alternating rotations in the plane perpendicular to $a + b$ and a 2D checkerboard of alternating rotations in the ab plane along c. The tilting of rotation axes for the $Imma$ (74) structure does not possess any alternating tilting in the ab plane direction around the c direction. The structure in space group $Pnma$ (62) corresponds to Glazer notation $a^+b^-b^-$ and Ref. [31] notation ($a000bb$). The structure in space group $Imma$ (74) corresponds to Glazer notation $a^0b^-b^-$ and Ref. [31] notation ($0000bb$).

Character tables only typically report basis functions up to $L \leqslant 3$, so the higher order irreps 3_e^{-2}, 4_e^2, and 5_e^{-2} are not listed, but one can confirm with simple calculations that they transform as B_{1g}. This conventional approach becomes more involved for objects with more complicated symmetry. In such cases, it is standard practice to employ computer algorithms to find, for example, relevant isotropy subgroups. However, many databases and tools for performing conventional symmetry analyses are not open source, making them difficult to incorporate into specific efforts.

Now, we demonstrate this method on a more complicated example and use it to find symmetrically intermediate structures. Perovskite crystal structures are composed of octahedrally coordinated transition metal sites on a cubic lattice where the octahedra share corner vertices (Fig. 3). Perovskites display a wealth of exotic electronic and magnetic phenomena, the onset of which is often accompanied by a structural distortion of the parent structure in space group $Pm\bar{3}m$ (221) caused by the softening of phonon modes or the onset of magnetic orders [30].

The octahedra in perovskites can distort in a variety of ways, one of which is by developing tilting patterns, commonly classified using Glazer notation introduced in Ref. [32]. Using the same procedure as in the previous section, we recover the order parameters for two perovskite structures with different octahedral tilting. We use periodic boundary conditions to treat the crystal as an infinite periodic solid.

For this demonstration we compare the parent structure in $Pm\bar{3}m$ (221) to the structure in the subgroup $Pnma$ (62). We use the same training procedure as above except for the following: we only apply order parameters to the B sites and we allow each B site to have its own order parameter. We also add a penalty that increases with the L of the candidate order parameter.

From training, the model is able to recover that each B site has a nontrivial pseudovector order parameter with equal magnitude; this can be intuitively interpreted as different rotation axes with equal rotation angle for each B site. The pattern of rotation axes and corresponding octahedral tilting is described and shown in Fig. 3. If we look at the character table for $Pm\bar{3}m$ we can confirm that this pattern of pseudovectors matches the irreps $M^{3+} \oplus R^{4+}$, the irreps recovered in Ref. [31]. In contrast to conventional symmetry analysis, our method provides a more clear geometric interpretation of these order parameters as rotation axes. Additionally, the same model can be used to determine the form of the order parameter and build a model that can predict the amplitude of this distortion (e.g., based on composition and the parent structure).

We can also learn to produce output that is symmetrically intermediate between input and ground truth output by restricting learnable order parameters. If we train an identical model, but constrain the pseudovector order parameter to be zero along the c direction and nonadjacent B sites to have identical order parameters, we recover an intermediate structure in the space group $Imma$ (74) described and shown in Fig. 3.

In contrast to conventional symmetry analysis which requires classifying the symmetry of given systems, we perform symmetry analyses with Euclidean neural networks by learning equivariant mappings. This allows us to gain symmetry insights using standard neural network training procedures. Our methods do not rely on any tabulated information and can be directly applied to tensor fields of arbitrary complexity.

Symmetry equivariant neural networks act as "symmetry compilers": they can only fit data that is symmetrically compatible and can be used to help find symmetry breaking order parameters necessary for compatibility. The properties proven in this Letter generalize to any symmetry-equivariant network and are relevant to any branch of physics using symmetry-aware machine learning models to create surrogate or generative models of physical systems. The same procedures demonstrated in this Letter can be used to find order parameters of other physical systems, for example, missing environmental parameters of an experimental setup (such as anisotropy in the magnetic field of an accelerator magnet) or identifying other symmetry-implied information unbeknownst to the researcher.

[1] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, Rev. Mod. Phys. **91**, 045002 (2019).

[2] *Machine Learning Meets Quantum Physics*, edited by K. T. Schütt, S. Chmiela, O. A. von Lilienfeld, A. Tkatchenko, K. Tsuda, and K.-R. Müller (Springer International Publishing, New York, 2020).

[3] J. Behler and M. Parrinello, Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces, Phys. Rev. Lett. **98**, 146401 (2007).

[4] A. P. Bartók, R. Kondor, and G. Csányi, On representing chemical environments, Phys. Rev. B **87**, 184115 (2013).

[5] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, SchNet—A deep learning architecture for molecules and materials, J. Chem. Phys. **148**, 241722 (2018).

[6] N. Thomas, T. E. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds, arXiv:1802.08219.

[7] R. Kondor, Z. Lin, and S. Trivedi, Clebsch–Gordan nets: A fully Fourier space spherical convolutional neural network, in *Advances in Neural Information Processing Systems 32* (Curran Associates, 2018), pp. 10117–10126.

[8] M. Weiler, M. Geiger, M. Welling, W. Boomsma, and T. Cohen, 3D steerable CNNs: Learning rotationally equivariant features in volumetric data, in *Advances in Neural Information Processing Systems 32* (Curran Associates, 2018), pp. 10402–10413.

[9] B. Anderson, T. S. Hy, and R. Kondor, Cormorant: Covariant molecular neural networks, in *Advances in Neural Information Processing Systems* (Curran Associates, 2019), pp. 14537–14546.

[10] A. Grisafi, D. M. Wilkins, G. Csányi, and M. Ceriotti, Symmetry-Adapted Machine Learning for Tensorial Properties of Atomistic Systems, Phys. Rev. Lett. **120**, 036002 (2018).

[11] L. M. Brown, The idea of the neutrino, Phys. Today **31**(9), 23 (1978).

[12] P. W. Anderson, More is different, Science **177**, 393 (1972).

[13] F. Englert and R. Brout, Broken Symmetry and the Mass of Gauge Vector Mesons, Phys. Rev. Lett. **13**, 321 (1964).

[14] P. W. Higgs, Broken Symmetries and the Masses of Gauge Bosons, Phys. Rev. Lett. **13**, 508 (1964).

[15] G. S. Guralnik, C. R. Hagen, and T. W. B. Kibble, Global Conservation Laws and Massless Particles, Phys. Rev. Lett. **13**, 585 (1964).

[16] J. Bardeen, L. N. Cooper, and J. R. Schrieffer, Theory of Superconductivity, Phys. Rev. **108**, 1175 (1957).

[17] Y. Nambu, Quasi-particles and gauge invariance in the theory of superconductivity, Phys. Rev. **117**, 648 (1960).

[18] L. Landau, On the theory of phase transitions, in *Collected Papers*, Vol. 1 (Nauka, Moscow, 1969), pp. 234–252, originally published in Zh. Eksp. Teor. Fiz. **7**, 19 (1937).

[19] P. Curie, Sur la symétrie dans les phénomènes physiques, symétrie d'un champ électrique et d'un champ magnétique, J. Phys. Théor. Appl. (in French) EDP Sciences. **3**, 393 (1894).

[20] A. F. Chalmers, Curie's principle, Br. J. Philos. Sci. **21**, 133 (1970).

[21] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, Harmonic networks: Deep translation and rotation equivariance, in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2017).

[22] R. Kondor and S. Trivedi, On the generalization of equivariance and convolution in neural networks to the action of compact groups, in *Proceedings of the 35th International Conference*

on Machine Learning (ICML 2018), Proceedings of Machine Learning Research Vol. 80, edited by J. G. Dy and A. Krause (PMLR, 2018), pp. 2752–2760.

[23] T. S. Cohen, M. Geiger, and M. Weiler, A general theory of equivariant CNNs on homogeneous spaces, in *Advances in Neural Information Processing Systems 32*, edited by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett (Curran Associates, 2019), pp. 9142–9153.

[24] M. Geiger, T. E. Smidt, B. K. Miller, W. Boomsma, K. Lapchevskyi, M. Weiler, M. Tyszkiewicz, and J. Frellsen, e3nn: A modular PyTorch framework for Euclidean neural networks, 2020, doi: 10.5281/zenodo.4071988, https://github.com/e3nn/e3nn.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, Pytorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., Red Hook, NY, 2019), pp. 8024–8035.

[26] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay *et al.*, Jupyter notebooks—A publishing format

for reproducible computational workflows, in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, edited by F. Loizides and B. Schmidt (IOS, Amsterdam, 2016), pp. 87–90.

[27] T. E. Smidt, Code repository for "Finding symmetry breaking order parameters with Euclidean neural networks," 2020, doi: 10.5281/zenodo.4087189, https://github.com/blondegeek/e3nn_symm_breaking.

[28] T. Cohen and M. Welling, Steerable CNNs, in *International Conference on Learning Representations (ICLR)* (OpenReview.net, 2017).

[29] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevResearch.3.L012002 for network training details and additional proofs.

[30] N. A. Benedek and C. J. Fennie, Why are there so few perovskite ferroelectrics? J. Phys. Chem. C **117**, 13339 (2013).

[31] C. J. Howard and H. T. Stokes, Group-theoretical analysis of octahedral tilting in perovskites, Acta Crystallogr., Sect. B: Struct. Sci. **54**, 782 (1998).

[32] A. M. Glazer, The classification of tilted octahedra in perovskites, Acta Crystallogr., Sect. B: Struct. Crystallogr. Cryst. Chem. **28**, 3384 (1972).