Editors' Suggestion

Machine learning vortices at the Kosterlitz-Thouless transition

Matthew J. S. Beach,* Anna Golubeva, and Roger G. Melko
Department of Physics and Astronomy, University of Waterloo, Waterloo, Canada N2L 3G1
and Perimeter Institute for Theoretical Physics, Waterloo, Ontario, Canada N2L 2Y5

(Received 31 October 2017; published 25 January 2018)

Efficient and automated classification of phases from minimally processed data is one goal of machine learning in condensed-matter and statistical physics. Supervised algorithms trained on raw samples of microstates can successfully detect conventional phase transitions via learning a bulk feature such as an order parameter. In this paper, we investigate whether neural networks can learn to classify phases based on topological defects. We address this question on the two-dimensional classical XY model which exhibits a Kosterlitz-Thouless transition. We find significant feature engineering of the raw spin states is required to convincingly claim that features of the vortex configurations are responsible for learning the transition temperature. We further show a single-layer network does not correctly classify the phases of the XY model, while a convolutional network easily performs classification by learning the global magnetization. Finally, we design a deep network capable of learning vortices without feature engineering. We demonstrate the detection of vortices does not necessarily result in the best classification accuracy, especially for lattices of less than approximately 1000 spins. For larger systems, it remains a difficult task to learn vortices.

DOI: 10.1103/PhysRevB.97.045207

I. INTRODUCTION

The remarkable success of artificial neural networks in the tasks of image recognition and natural language processing has prompted interdisciplinary efforts to investigate how these new tools might benefit a broad range of sciences. One of the most intriguing areas of application is condensed-matter physics, where the exponentially large Hilbert space of a quantum many-body state provides an immense data set. In fields such as computer vision, it has been demonstrated that neural networks have the ability to extract physical features from highly complex data sets [1–4]. This gives hope that machine learning techniques may provide a tool to probe regions of the many-body Hilbert space that are currently intractable with other algorithms.

In the realm of classical statistical physics, supervised and unsupervised learning have been applied successfully to classify symmetry-broken phases [5–8]. In some cases, it is possible to deduce that the network has learned an order parameter or another thermodynamic quantity [5,6,8]. This interpretability is one major advantage of data sets derived from statistical physics and can contribute to the theoretical understanding of the behavior of neural networks in real-world applications.

Motivated by the successful application of supervised learning to conventional symmetry-breaking transitions, it is natural to ask whether neural networks are capable of distinguishing unconventional phase transitions driven by the emergence of topological defects. The prototypical example for such a system is the two-dimensional XY model, which exhibits a Kosterlitz-Thouless (KT) transition [9]. Several unsupervised learning strategies have been applied to this model previously; for example, it was found that principal component analysis (PCA) [10] performed on spin configurations captures the magnetization which is present in finite-size lattices [11–13]. Even when trained directly on vorticity, PCA is unable to resolve vortex-antivortex unbinding, which is attributed to the linearity of this method [12]. Similarly, variational autoencoders [14], a popular tool for unsupervised learning based on Bayesian inference, perform classification by learning a bulk magnetization [11,13,15].

In contrast, efforts in supervised learning have been more successful, although none have been applied directly to the XY model. In Ref. [16], a convolutional network trained on winding numbers correctly classified interacting boson phases separated by a KT transition. However, this same method failed when trained directly on raw configurations. A related problem was explored in Ref. [17], where the authors trained a convolutional network directly on Hamiltonians of one-dimensional topological band insulators labeled by their global winding number. By inspecting the trained weights the authors deduced that the network had learned to calculate the winding number correctly.

In this paper, we apply several supervised machine learning strategies to the task of identifying the KT transition in the two-dimensional XY model. We ask whether it is possible for a neural network, trained only on raw spin states labeled by their phases, to learn a representation that can be interpreted as the local vorticity of the spin variables. First, we compare supervised learning algorithms involving feed-forward and convolutional neural networks applied to both unprocessed (raw spin configurations) and processed input data (vorticity). We then use both types of input data in the semisupervised confusion scheme from Ref. [7]. Last, we explore to what degree feature engineering of the raw spin configurations

*matthew.beach@uwaterloo.ca

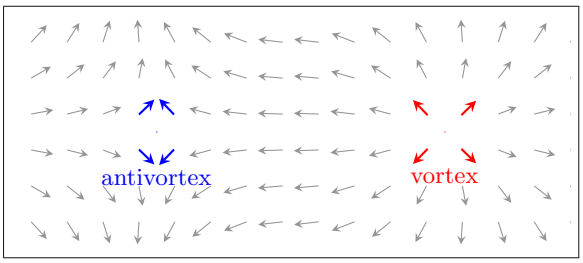

FIG. 1. Example of a vortex and antivortex in the XY model on the lattice. A vortex has winding number $k = 1$, while an antivortex has $k = -1$.

is required and whether the network can learn to process the data into something resembling vortices using additional convolutional layers.

Although it is possible to learn vortices, the network can also perform its classification task to reasonable accuracy by finding a local optimization minimum which is unrelated to topological features. We conclude with a discussion on the challenging task of seeing the vortex-antivortex unbinding transition in the two-dimensional XY model using machine learning techniques.

II. BACKGROUND

The classical XY model consists of unit spins with nearest-neighbor interactions given by

$$\mathcal{H}_{XY} = -J \sum_{\langle ij \rangle} \cos(\theta_i - \theta_j), \tag{1}$$

where $\langle ij \rangle$ indicates that the sum is taken over nearest neighbors and the angle $\theta_i \in [0, 2\pi)$ denotes the spin orientation on site i. Although the Mermin-Wagner theorem states that a long-range-ordered (LRO) phase cannot exist in two dimensions due to the coherence of massless spin waves [18], the formation of topological defects (i.e., vortices/antivortices) in the XY model results in a quasi-LRO phase [9,19]. The transition between the low-temperature quasi-LRO phase with an algebraically decaying correlation function and the high-temperature disordered phase with an exponentially decaying correlation function is a KT transition, and the associated temperature is denoted as T_{KT}. Transitions of this universality class can be found in a variety of systems, with one of the most famous being the superfluid transition in two-dimensional helium [20–22].

The topological defects in the XY model are quantified through the vorticity v, defined as

$$v \equiv \oint_C \nabla\theta \cdot d\vec{\ell} = 2\pi k, \quad k = \pm 1, \pm 2, \ldots, \tag{2}$$

where C denotes any closed path around the vortex core and k is the winding number of the associated spins. A vortex is defined by a positive winding number, $k = 1$, and an antivortex is defined by $k = -1$. On a lattice, the integral may be approximated by the sum of the angle differences over a plaquette. An example of a vortex and antivortex is shown in Fig. 1.

Below T_{KT}, vortex-antivortex pairs form due to thermal fluctuations, but they remain bound to minimize their total free energy. At T_{KT}, the entropy contribution to the free energy equals the binding energy of a pair, triggering vortex unbinding which drives the KT phase transition. The essential singularity of the free energy at T_{KT} means that all derivatives are finite at the transition. For example, the specific heat is observed to be smooth at the transition, with a nonuniversal peak at a $T > T_{\mathrm{KT}}$ which is associated with the entropy released when most vortex pairs unbind [23]. While the thermodynamic limit of the XY model has strictly zero magnetization for all $T > 0$, a nonzero value is found for systems of finite size [see Fig. 2(b)] [24,25].

One method to calculate T_{KT} from finite-size data is to exploit the Nelson-Kosterlitz universal jump [26,27]. This is determined from where the helicity modulus Υ crosses $\frac{2T}{\pi}$. The helicity modulus, also called spin-wave stiffness or spin rigidity, measures the response of a system to a twist in the boundary conditions (i.e., torsion). From the linearized renormalization group (RG) equations, one can derive the finite-size scaling behavior of the critical temperature $\tilde{T}_{\mathrm{KT}}$ on a $L \times L$ lattice to be

$$\tilde{T}_{\mathrm{KT}}(L) \approx T_{\mathrm{KT}} + \frac{\pi^2}{4c(\log L)^2}, \tag{3}$$

with a constant c [26]. Figure 2(a) shows the helicity modulus Υ and the scaling of T_{KT} derived from Monte Carlo simulations. From our generated samples, we find $T_{\mathrm{KT}} = 0.899 \pm 0.06$, which is consistent with the literature value of $T_{\mathrm{KT}} = 0.893$ [24,25,28]. As shown in Fig. 2(b), the magnetization evaluated at the critical point $M|_{T_{\mathrm{KT}}}$ is of significant magnitude and scales with $L^{-1/8}$ as expected [24], to within a 4% error.

In the next section, we explore which neural networks can accurately distinguish the phases above and below the thermodynamic temperature T_{KT}. We employ two standard network architectures motivated by canonical problems in machine learning (such as classification of the MNIST data set) using XY spin configurations for finite-size systems as input data. Based on previous observations that conventional phase boundaries estimated by supervised learning follow established finite-size scaling [5], we compare the scaling of $\tilde{T}_{\mathrm{KT}}$ predicted by our neural networks with the $(\log L)^{-2}$ form above.

III. METHODS AND RESULTS

We study the binary classification of the two phases of the XY model, labeling configurations as belonging to either the low $T < T_{\mathrm{KT}}$ or high $T > T_{\mathrm{KT}}$ temperature phases. Our goal is to confirm whether simple supervised learning with neural networks is capable of correctly classifying spin configurations according to these labels. In particular, we wish to interpret whether the network relies on the (finite-size) magnetization, or on topological defects. Further, we inquire as to what specific network architecture is required to achieve this goal and what features different architectures may utilize.

We employ standard Monte Carlo simulation methods to generate spin configuration of the XY model [29,30]. For the training set, we generate 1000 configurations per temperature, with 64 temperatures ranging from 0.1 to 3.0, for lattice sizes $L = 8, \ldots, 64$ in increments of 8. The test set is generated

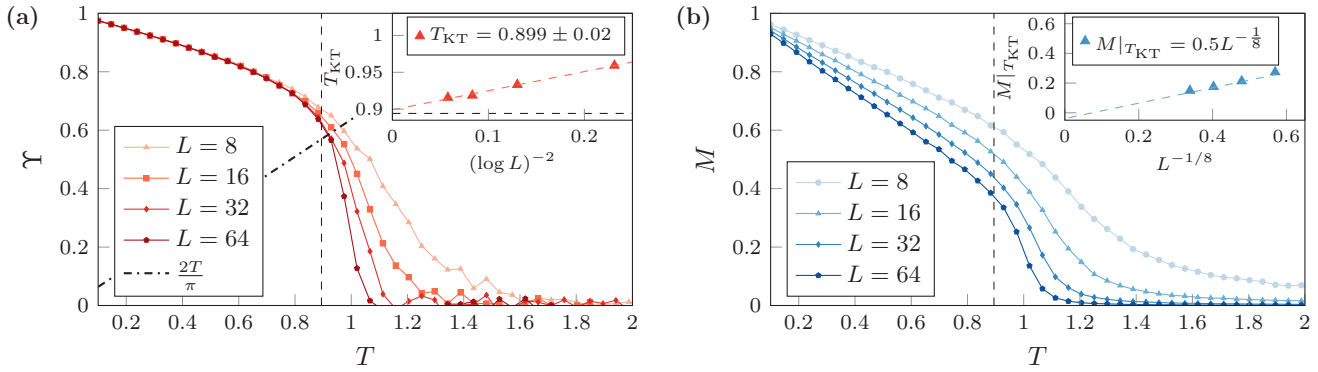

FIG. 2. Estimators of the XY model on a $L \times L$ lattice with periodic boundary conditions computed via Monte Carlo sampling. (a) The helicity modulus for various lattice sizes L. The estimated critical point $\tilde{T}_{\mathrm{KT}}$ is determined by the Nelson-Kosterlitz universal jump where the helicity modulus Υ intersects the line $\frac{2T}{\pi}$. The inset shows how $\tilde{T}_{\mathrm{KT}}$ scales with $(\log L)^{-2}$ towards the thermodynamic T_{KT} shown by the black dashed line. (b) The nonzero magnetization present in the finite-size XY model. The magnetization vanishes as $L^{-\frac{1}{8}}$ in the thermodynamic limit with the scaling shown in the inset.

separately, with 100 configurations per temperature. From the training data, we randomly select 10% for cross validation in order to decrease the chance of overfitting and to identify a definitive stopping point for training using early stopping [31–33].

The network is trained to minimize the loss function $L(y^{\mathrm{pred}}, y^{\mathrm{true}})$, where y^{true} represents the true binary labels and y^{pred} represents the predicted ones. We take the loss function to be the standard cross entropy

$$L(y^{\mathrm{pred}}, y^{\mathrm{true}}) = -\sum_i y_i^{\mathrm{true}} \log y_i^{\mathrm{pred}}. \tag{4}$$

The parameters of the network (weights and biases) are then optimized through backpropagation to minimize the loss function on the training data [2]. Each network is trained until the loss function *evaluated on the validation set* fails to decrease after 50 training epochs. Early stopping with cross validation is commonly used to choose the network parameters with minimal generalization error [32]. We implement the networks with the KERAS library using the TENSORFLOW back end [34,35].

We employ two different standard network architectures: a one-layer feed-forward network (FFNN) and a deep convolutional network (CNN). The FFNN consists of one hidden layer of 1024 sigmoid activation units and one sigmoid output unit. The CNN starts with a two-dimensional convolutional layer consisting of eight filters of size 3×3 with rectified linear (ReLu) activation functions. The output from this layer is passed to another identical convolution layer with 16 filters before applying 2×2 max pooling. The network is then reshaped and fed into a fully connected layer with 32 ReLu units and passed to a single sigmoid output unit. Because there is a total of $1024L^2 + 2049$ trainable parameters in the FFNN, it can be difficult to train in comparison to the $128L^2 - 1024\,L + 3361$ parameters in the CNN. This is because the CNN explicitly takes advantage of the two-dimensional structure of the input to vastly improve performance. This architecture is one of the simplest that can attain over 99% accuracy on the standard MNIST data set. In our experience, changing the hyperparameters has a negligible effect on the accuracy.

A. Finite-size scaling of supervised learning

One goal in modern machine learning is to minimize the amount of feature engineering required. In our case, this corresponds to treating the raw spin configurations as direct inputs to the neural networks. For the XY model, these data are formatted as angle values, $\theta_i \in [0, 2\pi)$, on an $L \times L$ lattice with periodic boundary conditions.

For a given spin configuration, the output value of the final neuron in the network gives the probability of the configuration belonging to the low- (or high-) temperature phase. Due to thermal fluctuations, it is difficult to accurately classify states near the critical point. In accordance with intuition about phase transitions, we take the point where the probability is exactly 0.5 to be the inferred critical temperature $\tilde{T}_{\mathrm{KT}}$. This is further established in Ref. [5], where the authors show that this point scales with the correct correlation length critical exponent and predicts the thermodynamic critical temperature accurately for the Ising model. In that case, training a FFNN with a single hidden layer of 100 sigmoid units was sufficient ($100L^2 + 202$ total parameters) to achieve high classification accuracy and correctly predict the critical temperature.

Similarly, we study the performance of both a FFNN and a CNN in predicting T_{KT} for the XY model. To get an estimate for the statistical variance, the training process is repeated ten times with different validation sets.

As illustrated in the inset of Fig. 3(a), the FFNN has low classification accuracy (i.e., percentage of correctly classified configurations) for $L > 48$. This results in the very poorly predicted critical temperature $\tilde{T}_{\mathrm{KT}}$ in the main plot. In contrast, the accuracy of the CNN continually improves as L increases. However, as evident from Fig. 3(a), there is no clear finite-size scaling trend in the predicted T_{KT}. To interpret this we note that for each system size, the network is supervised on the thermodynamic value of T_{KT}. Thus, we speculate that each network could simply be learning to discriminate phases based on a robust, global feature which takes a unique value above and below T_{KT} for any L.

Based on previous experience, a global magnetization is a feature very easily detected in a supervised learning scheme [5,6,8]. Since the finite-size configurations of the XY model

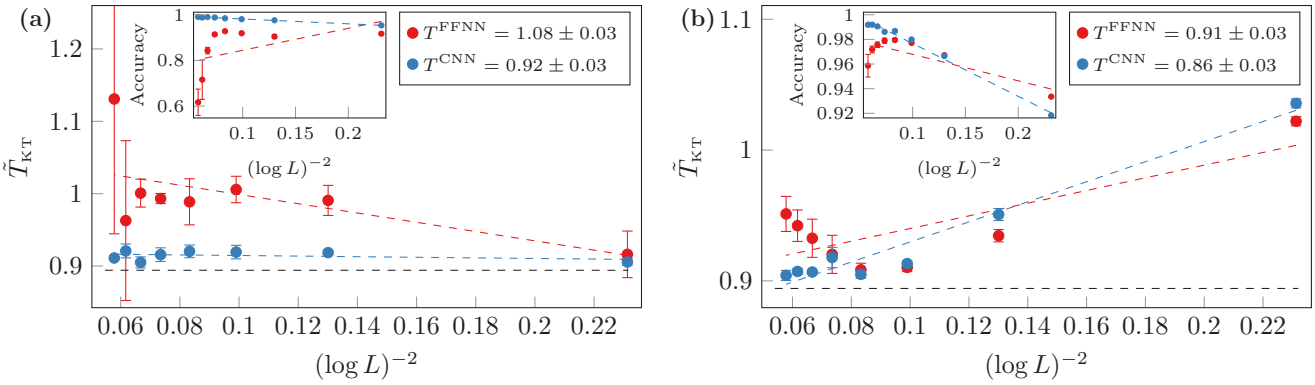

FIG. 3. Finite-size scaling of the predicted T_{KT} for FFNN and CNN trained on either (a) raw spin configurations or (b) the vorticity. In either case the FFNN performs worse than the CNN according to the test classification accuracy (insets). The critical temperature is determined by the point where the sigmoid output, as a function of temperature, crosses 0.5. Each data point and variance is obtained by training ten networks with stochastic gradient descent until the validation loss function fails to improve after 50 epochs (early stopping).

themselves contain a nonzero magnetization at $T > 0$ [see Fig. 2(b)], it is reasonable to hypothesize that the CNN simply learns this threshold value of the magnetization for each system size separately. Because of the Mermin-Wagner theorem, however, it is known that a global magnetization is not a relevant feature for T_{KT} in the thermodynamic limit. Thus, in this case, some amount of feature engineering is crucial to achieve our goal of detecting a phase transition mediated by topological defects.

In the next step, we preprocess the spin configurations into the associated vorticity and train the networks on these configurations. To calculate the vorticity, one computes the angle differences $\Delta\theta_{ij} \in [-2\pi, 2\pi]$ between each pair of neighboring spins i and j on a plaquette and converts these to the range $(-\pi, \pi]$. This can be done by applying the sawtooth function,

$$\mathrm{saw}(x) = \begin{cases} x + 2\pi, & x \leqslant -\pi, \\ x, & -\pi \leqslant x \leqslant \pi, \\ x - 2\pi, & \pi \leqslant x, \end{cases} \tag{5}$$

to each $\Delta\theta_{ij}$. The sum of the rescaled angle differences gives the vorticity from Eq. (2).

Trained on the vortex configurations, Fig. 3(b) shows that both the FFNN and CNN achieve high accuracy and scale with L towards the correct value of T_{KT}. However, once again, we observe that the FFNN begins to perform poorly for $L > 32$, whereas the CNN continually improves. We note that the scaling seems consistent with Eq. (3), particularly for the CNN. However, from this scaling alone, we cannot determine precisely what the CNN learns. For example, it could potentially classify the phases based on the sum of the squared vorticity (which is approximately zero below T_{KT}), or it might represent a more complicated function such as the average distance between vortex-antivortex pairs. Regardless, the scaling behavior may serve as a useful diagnostic to determine whether a given network is learning bulk features or topological effects.

B. Learning by confusion

We further investigate the difference between training on spin configurations and vortex configurations by employing a confusion scheme [7,36]. Learning by confusion offers a semisupervised approach to finding the critical temperature separating two phases by training many supervised networks on data that is deliberately mislabeled. The binary label "0" is assigned to a configuration if its temperature is less than a proposed T^*, and the label "1" is assigned otherwise. A new network is trained on each new labeling of the data, (i.e., for each T^*). It is expected that the highest accuracy is achieved when the labeling is close to the true value and, trivially, at the end points. This results in a $\vee\vee$ shape when plotting the test accuracy as a function of T^* [7]. The peak on either end point can be attributed to the network being trained and tested exclusively on one class, in which case it will always place test data into that class. The key assumption in the confusion scheme is the existence of a true physical labeling of the data which the network is capable of learning more accurately than false labelings.

Since we have shown that the CNN is more successful at classification than the FFNN, we consider only the CNN for the present confusion scheme. The results of training on raw spin and vortex configurations are shown in Fig. 4. Learning on the raw spins results in a $\overline{\vee}$ shape rather than the expected $\vee\vee$. As mentioned above, the finite-size XY model has a nonzero magnetization for $T < T_{\mathrm{KT}}$, and this algorithm can easily classify any division $T^* < T_{\mathrm{KT}}$ by a threshold magnetization. This supports our hypothesis from Sec. III A that trained on raw spins, a CNN learns the magnetization.

When trained on vortices, the expected $\vee\vee$ shape emerges, although it is skewed because we choose our training data from a nonsymmetric region around T_{KT}. Despite having a powerful deep network, it is unable to learn any arbitrary partition and performs best near T_{KT}. This may be attributed to the fact that for low T, the vortex configurations are fundamentally similar; there are few vortices, and they are logarithmically bound. This is in contrast to the raw spin configurations, which may possess distinguishing features like the magnetization. Near T_{KT}, the network can distinguish the phases with high accuracy because of the true physical partition due to vortex unbinding. At high T^* the vortex configurations look sufficiently random that the network again misclassifies for an arbitrary partition.

We also observe significant finite-size effects in the $\vee\vee$ and $\overline{\vee}$ shapes, both broadening and shallowing with increasing

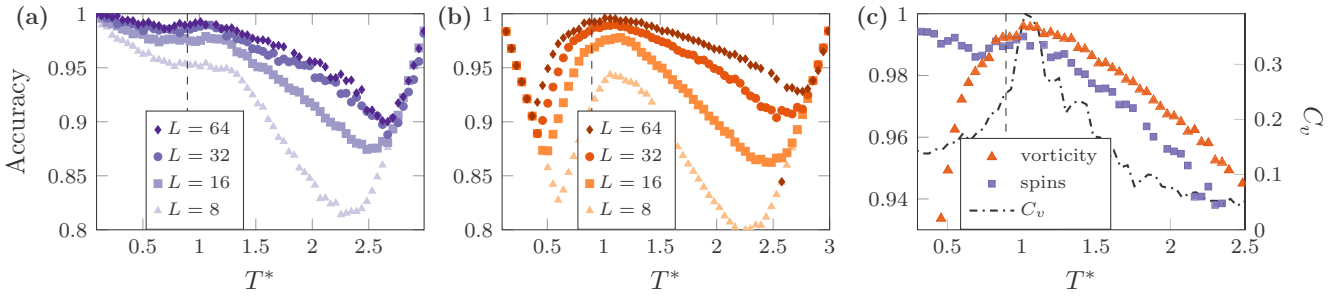

FIG. 4. The learning by confusion scheme for a CNN applied to (a) raw spin configurations and (b) vorticity configurations. The test accuracy is expected to form a $\vee\vee$ shape with the center peak at $T^* = T_{\text{KT}}$. (c) The peak in specific heat C_v compared to the peak of the test accuracy for a system of size $L = 64$. The dashed vertical line shows the thermodynamic T_{KT}.

L. The finite-size scaling behavior of the peak does not trend towards T_{KT} in the vortex case, but rather always stays above it, similar to the specific-heat peak [see Fig. 4(c)]. Surprisingly, in Fig. 4(c), we see the confusion scheme achieves higher accuracy at $T^* \approx 1$ than $T_{\text{KT}} = 0.89$, which indicates that the false $T^* \approx 1$ phase boundary is easier for the network to learn than the temperature $\tilde{T}_{\text{KT}}$ predicted by the universal jump. While this effect might disappear in the thermodynamic limit, it is still troubling. Matters are even worse for training on raw spins since all $T^* < T_{\text{KT}}$ have accuracy greater than 98.5% for $L = 64$, so it is even unclear where $\tilde{T}_{\text{KT}}$ is.

For finite-size systems, the test accuracy curve will never go flat. The reason is that a single spin configuration does not uniquely belong to a particular temperature, but rather, it occurs probabilistically for all temperatures (although perhaps infinitesimally). This will always results in some classification error. In particular, for isotropic or highly thermal regions, it is impossible to accurately correctly classify states, and therefore, a $\vee$ shape occurs. For other regions, the curve will go flat in the thermodynamic limit. Interestingly, the confusion scheme inadvertently tells us information about the variances in possible temperatures of a state.

The confusion scheme for the XY model offers insight into what our CNN prefers to learn. In the case of the raw spin configurations, we infer that it learns the finite magnetization of the spin configurations instead of topological features. Near T_{KT}, the network trained on vortices achieves slightly higher accuracy [see Fig. 4(c)]; therefore, in this case, the network would benefit from learning vortices. Despite this argument, we stress that we have no strong evidence that our CNN is even capable of finding vortices. To address this, in the next section we propose a custom network designed for vortex detection and test if it works in practice.

C. Custom architecture for learning vortices

In the previous sections, we compared networks trained on the raw spin configurations to those trained on vortex configurations which were constructed manually (i.e., feature engineered). We now explore the possibility of a custom network architecture designed specifically for learning vortices as an intermediate representation before performing classification. It is one of the remarkable features of deep neural networks that each layer may represent a new level of abstraction [3,4,37]. For example, in facial image recognition, the first convolution layer may extract edges, while the final layer encodes complex features such as facial expressions [1]. We aim to design a network which may similarly be interpreted as representing vortices in an intermediate layer.

Below, we derive the appropriate weights for a three-layer network which computes the vorticity from input spin configurations. The entire network is visualized in Fig. 5.

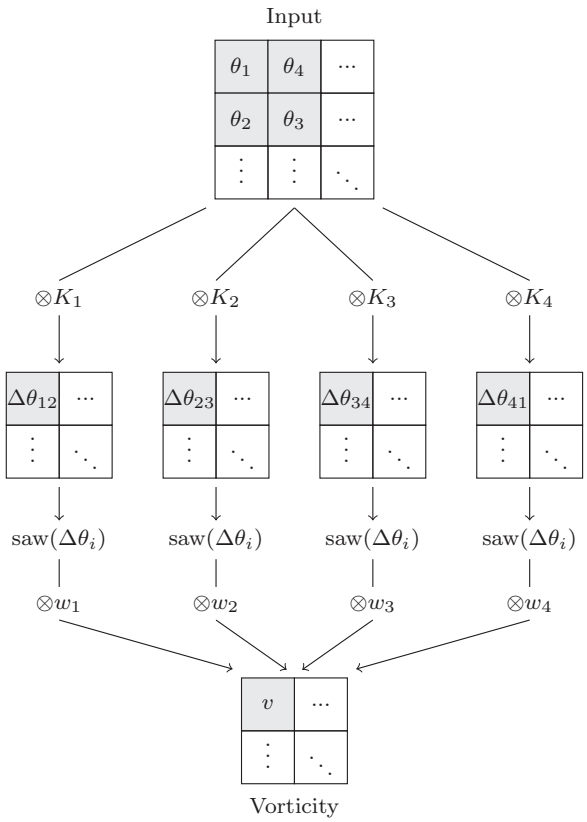

FIG. 5. Visual representation of how the custom network architecture can compute the vorticity. We denote the convolution operation with $\otimes$ and ignore biases for the purpose of the diagram. Applying the four 2×2 filters K_i partitions the data into four $L \times L$ arrays where each element is an angle difference in one lattice direction $\Delta\theta_{ij}$. The angle differences are then converted into the range $\Delta\theta_{ij} \in [-\pi,\pi)$ by applying the sawtooth function from Eq. (5). A single 1×1 convolution filter with weights $w = [1,1,1,1]$ and zero biases then sums the four shifted angle differences into the vorticity.

The first layer, which acts on the input angle values θ_i, is a convolution layer with four 2×2 convolution filters given by

$$K_1 = \begin{bmatrix} -1 & 1 \\ 0 & 0 \end{bmatrix}, \quad K_2 = \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}, \quad K_3 = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix}, \quad K_4 = \begin{bmatrix} 0 & 0 \\ 1 & -1 \end{bmatrix}. \tag{6}$$

The effect of these filters is to compute the nearest-neighbor angle differences $\Delta\theta_{ij}$ within each plaquette. The next layer we apply is hard coded to map the angle differences, $\Delta\theta_{ij} \in [-2\pi, 2\pi]$, into the range $[-\pi, \pi)$. This is done by applying the sawtooth function from Eq. (5) to each element in the $(L, L, 4)$-dimensional array. The final processing layer computes a weighted sum of the four angle differences by applying a single 1×1 convolution filter. Uniform weights with zero biases would compute the vorticity exactly up to a multiplicative constant.

While the network described above is capable of representing vortices within an internal layer (vorticity layer in Fig. 5), it might fail to do so in practice. To explore this we consider three possible variations of the initializations of the network parameters.

The first variation consists of fixing the weights (and biases) in the first three layers such that the network computes the vorticity exactly. This is, of course, engineering the relevant features by hand; however, it provides a useful benchmark. The second variation is performed by initializing the weights exactly to those of the fixed network, then relaxing the constraints as training is continued. This step shows whether the original (vortex) minimum is stable. The third variation is simply the naive choice where the network parameters are initialized randomly.

For all three variations, we train for binary classification by minimizing the cross-entropy loss from Eq. (4). Each network is trained ten times with different validation sets. As per Sec. III A, we implement early stopping to terminate training once the loss function on the validation set fails to improve after 50 epochs. We train on lattice sizes of $L = 8, \ldots, 72$ in increments of 8.

We can understand the three variations by looking at the loss function evaluated on the test set as in Fig. 6. For small L, the loss function of the fixed network is much larger than the others, indicating that it is *not* beneficial to represent the vortices for $L < 16$. In this small-lattice region, learning vortices hinders classification. However, near $L \sim 32$ the fixed network outperforms the other two. Hence, we conclude that only for the large-lattice region, $L > 32$, is it beneficial for a network to learn an intermediate representation of the vorticity. This also agrees with the findings in Ref. [12], in which the topologically invariant winding number could be learned for systems of size $L > 32$.

We can check what each network learns by looking at the histogram distribution of the outputs of the vorticity layer in Fig. 5. For the fixed network, we would see exactly integral quantities corresponding to the quantized vorticity. For the vortex-initialized network, Fig. 7 shows that for small L, it does not learn the true vorticity distribution, but for $L \geqslant 32$ it does. This is consistent with the hypothesis that learning vortices is beneficial only for $L > 32$. The randomly initialized network does not produce a histogram consistent with the learning of vortices for any system size studied.

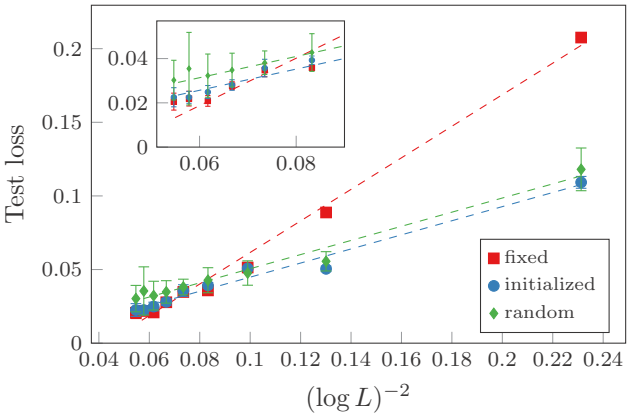

FIG. 6. The loss function from Eq. (4) evaluated on the test set for three variations of the custom architecture for various lattice sizes L. For small $L < 16$, the fixed network with hard-coded weights performs poorly compared to the others. For large $L > 32$, the fixed network performs best, possibly due to a reduced number of trainable parameters. The inset shows a magnified region for $32 \leqslant L \leqslant 72$.

Interpreting the behavior of the neural network for large L is not straightforward. As Fig. 6 shows, the model with fixed features and fewer trainable parameters performs better for large L. This can likely be attributed to a lower-dimensional optimization landscape. We cannot conclude whether the vortex representation is a global minimum for the fully adjustable (randomly initialized) network variation. While it certainly performs best in fixed computational time, the higher dimensionality of the adjustable network may have another global minimum not present in the lower-dimensional case. We can claim, however, that the vortex minimum is at least a stable local minimum since a network initialized to it never escapes, as demonstrated by the initialized variation for large L in Fig. 6.

Adding a custom regularization term could potentially alter the optimization landscape to aid the network in detecting vortices. One method would be to enforce integral quantities for an intermediate output of the network, but in our

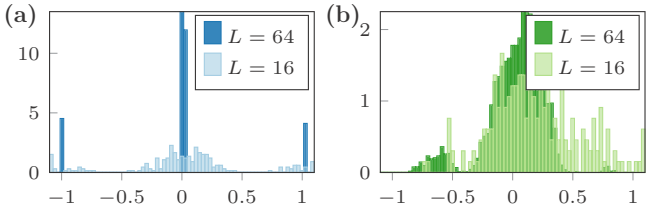

FIG. 7. Histogram of the values of the vortex layer from Fig. 5 which (ideally) computes the vorticity for (a) the network initialized to compute vorticity and (b) the randomly initialized network. In (a), we see for small L that the vorticity is not quantized, indicating that the network did not learn to compute the local vorticity; however, for large L, the histogram looks as expected for vortex detection. Conversely, the distribution in (b) appears to be unrelated to vortices for any L.

attempts, this results in the intermediate quantity being peaked sharply around zero. There is also the possibility of adding a regularization to the initial kernels to learn only nearest-neighbors interactions, but this is overly restrictive and defeats the purpose of automated machine learning.

IV. CONCLUSION

In this paper, we asked whether it is possible for a neural network to learn the vortex unbinding at the KT transition in the two-dimensional classical XY model. We demonstrated the significant effects that feature engineering and finite lattice sizes have on the performance of supervised learning algorithms.

Treating spin configurations as raw images and training on the thermodynamic value for T_{KT}, we found that naive supervised learning with a feed-forward network failed to converge to an accurate estimate for the KT transition temperature for moderate lattice sizes ($L \approx 32$). Conversely, a convolutional network performed consistently well with increasing L. Since the prediction of T_{KT} from the convolutional network was insensitive to L, we inferred that the network extracted features related to the magnetization, which are present in any finite-size lattice. This conclusion was further supported by the observation that in the confusion method any false phase boundary T^* below T_{KT} could easily be learned by a network when trained on the raw spin configurations.

By preprocessing the spin configurations into vorticity, both network architectures displayed finite-size scaling behavior consistent with the thermodynamic value of T_{KT}. In particular, the performance of the convolutional network continually improved as the system size increased, whereas the one-layer network's performance plateaued around $L = 32$. When the confusion scheme was trained on vortices, it did not predict the correct critical temperature; instead, the test accuracy reached a maximum near $T^* \approx 1$. This demonstrates the need for further study of the confusion scheme for the semisupervised learning of phase transitions.

We further explored if such extreme feature engineering could be relaxed while retaining acceptable accuracy. We devised a deep-layered structure of weights that could be constrained to extract vortices from the raw spin configurations or left free to explore other minima in the learning process. We found that it is beneficial for the network to discover vortices only for lattices with of over 1000 spins. Yet, even for large system sizes, a randomly initialized network settled into a local minimum not related to vortices. It is likely that the optimization landscape of our designed network is sufficiently rough that stochastic gradient descent would take exponentially long to find a minimum where the learned features correspond to vortices.

The difficulty that these standard supervised learning techniques have in discriminating the phases of the XY model underscores the challenge that unsupervised learning techniques could face in learning the KT transition from unlabeled data. Our work emphasizes the need for further study into how much feature engineering is required before topological features can be used reliably for the machine learning of unconventional phases and phase transitions.

ACKNOWLEDGMENTS

The authors would like to thank J. Carrasquilla, C. X. Cerkauskas, L. E. Hayward Sierens, B. Kulchytskyy, R. Lewitzky, A. Morningstar, P. Ponte, J. Rau, S. Sachdev, R. Scalettar, R. Singh, G. Torlai, F. Verstraete, and C. Wang for many useful discussions. This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canada Research Chair program, the Perimeter Institute for Theoretical Physics, and the National Science Foundation under Grant No. NSF PHY-1125915. This work was made possible by the facilities of the Shared Hierarchical Academic Research Computing Network (SHARCNET) and Compute/Calcul Canada. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. Research at the Perimeter Institute is supported by the government of Canada through Industry Canada and by the province of Ontario througdh the Ministry of Research & Innovation.

[1] M. D. Zeiler and R. Fergus, in *Computer Vision ECCV 2014*, Lecture Notes in Computer Science (Springer, Cham, 2014), pp. 818–833.

[2] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature (London) **521**, 436 (2015).

[3] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, Deep learning for visual understanding: A review, Neurocomputing **187**, 27 (2016).

[4] A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM **60**, 84 (2017).

[5] J. Carrasquilla and R. G. Melko, Machine learning phases of matter, Nat. Phys. **13**, 431 (2017).

[6] S. J. Wetzel and M. Scherzer, Machine learning of explicit order parameters: From the Ising model to SU(2) lattice gauge theory, Phys. Rev. B **96**, 184410 (2017).

[7] E. P. L. van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Learning phase transitions by confusion, Nat. Phys. **13**, 435 (2017).

[8] P. Ponte and R. G. Melko, Kernel methods for interpretable machine learning of order parameters, Phys. Rev. B **96**, 205146 (2017).

[9] J. M. Kosterlitz and D. J. Thouless, Ordering, metastability and phase transitions in two-dimensional systems, J. Phys. C **6**, 1181 (1973).

[10] I. T. Jolliffe, *Principal Component Analysis*, Springer Series in Statistics (Springer, New York, 2002).

[11] S. J. Wetzel, Unsupervised learning of phase transitions: From principal component analysis to variational autoencoders, Phys. Rev. E **96**, 022140 (2017).

[12] W. Hu, R. R. P. Singh, and R. T. Scalettar, Discovering phases, phase transitions, and crossovers through unsupervised machine learning: A critical examination, Phys. Rev. E **95**, 062122 (2017).

[13] C. Wang and H. Zhai, Machine learning of frustrated classical spin models. I. Principal component analysis, Phys. Rev. B **96**, 144432 (2017).
[14] D. P. Kingma and M. Welling, Auto-encoding variational bayes, Proceedings of the 2nd International Conference on Learning Representations (ICLR), 2014, arXiv:1312.6114 (2013).
[15] M. Cristoforetti, G. Jurman, A. I. Nardelli, and C. Furlanello, Towards meaningful physics from generative models, arXiv:1705.09524 (2017).
[16] P. Broecker, F. F. Assaad, and S. Trebst, Quantum phase recognition via unsupervised machine learning, arXiv:1707.00663 (2017).
[17] P. Zhang, H. Shen, and H. Zhai, Machine learning topological invariants with neural networks, arXiv:1708.09401 (2017).
[18] N. D. Mermin and H. Wagner, Absence of Ferromagnetism or Antiferromagnetism in One- or Two-Dimensional Isotropic Heisenberg Models, Phys. Rev. Lett. **17**, 1133 (1966).
[19] J. M. Kosterlitz, The critical properties of the two-dimensional xy model, J. Phys. C **7**, 1046 (1974).
[20] P. Kapitza, Viscosity of liquid helium below the γ-Point, Nature (London) **141**, 74 (1938).
[21] J. F. Allen and A. D. Misener, Flow of liquid helium II, Nature (London) **141**, 75 (1938).
[22] D. Bishop and J. Reppy, A precision measurement of the superfluid density near the transition of a 2D superfluid, J. Phys. Colloques **39**, C6-339 (1978).
[23] P. M. Chaikin and T. C. Lubensky, *Principles of Condensed Matter Physics* (Cambridge University Press, Cambridge, 2000).
[24] S. G. Chung, Essential finite-size effect in the two-dimensional XY model, Phys. Rev. B **60**, 11761 (1999).
[25] P. Olsson, Monte Carlo analysis of the two-dimensional XY model. II. Comparison with the Kosterlitz renormalization-group equations, Phys. Rev. B **52**, 4526 (1995).
[26] D. R. Nelson and J. M. Kosterlitz, Universal Jump in the Superfluid Density of Two-Dimensional Superfluids, Phys. Rev. Lett. **39**, 1201 (1977).
[27] P. Minnhagen and G. G. Warren, Superfluid density of a two-dimensional fluid, Phys. Rev. B **24**, 2526 (1981).
[28] Y. Komura and Y. Okabe, Large-scale Monte Carlo simulation of two-dimensional classical XY model using multiple GPUs, J. Phys. Soc. Jpn. **81**, 113001 (2012).
[29] U. Wolff, Collective Monte Carlo Updating for Spin Systems, Phys. Rev. Lett. **62**, 361 (1989).
[30] J. C. Walter and G. T. Barkema, An introduction to Monte Carlo methods, Phys. A (Amsterdam, Neth.) **418**, 78 (2015).
[31] C. Wang, S. S. Venkatesh, and J. S. Judd, in *Proceedings of 1995 IEEE International Symposium on Information Theory* (IEEE Press, Piscataway, NJ, 1995), pp. 303–310.
[32] L. Prechelt, in *Neural Networks: Tricks of the Trade,* Lecture Notes in Computer Science Vol. 1524 (Springer, Berlin, 1997), pp. 55–69.
[33] L. Prechelt, Automatic early stopping using cross validation: quantifying the criteria, Neural Networks **11**, 761 (1998).
[34] F. Chollet *et al.*, KERAS (2015), https://github.com/fchollet/keras.
[35] M. Abadi *et al.*, TENSORFLOW (2015), https://www.tensorflow.org/.
[36] Y.-H. Liu and E. P. L. van Nieuwenburg, Self-learning phase boundaries by active contours, arXiv:1706.08111 (2017).
[37] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks **61**, 85 (2015).