Rapid Measurements and Phase Transition Detections Made Simple by AC-GANs

Jiewei Ding[1,2], Ho-Kin Tang[3] ⋆ and Wing Chi Yu[1,2] †

1 Department of Physics, City University of Hong Kong, Kowloon, Hong Kong
2 City University of Hong Kong Shenzhen Research Institute, Shenzhen 518057, China
3 School of Science, Harbin Institute of Technology, Shenzhen, 518055, China
⋆ denghaojian@hit.edu.cn † wingcyu@cityu.edu.hk

May 10, 2024

Abstract

In recent years, significant attention has been paid to using end-to-end neural networks for analyzing Monte Carlo data. However, the exploration of non-end-to-end generative adversarial neural networks remains limited. Here, we investigate classical many-body systems using generative adversarial neural networks. We employ the conditional generative adversarial network with an auxiliary classifier (AC-GAN) and integrate self-attention layers into the generator. This modification enables the network learn the distribution of the two-dimensional (2D) XY model's spin configurations and the physical quantities of interest. Utilizing the symmetry of the systems, we discover that AC-GAN can be trained with a very small raw dataset. This approach allows us to obtain reliable measurements for models typically demanding large samples, such as the large-sized 2D XY and the 3D constrained Heisenberg models. Moreover, we demonstrate the capability of AC-GAN to identify the phase transition points by quantifying the distribution changes in the spin configurations of the systems.

Contents

1 Introduction

In condensed matter systems, the interactions among a large number of microscopic particles can lead to fascinating collective physical phenomena [1]. Some of these phenomena offer insights for developing new materials with useful properties. However, when studying many-body systems, exact calculations are often challenging due to the vast number of coupled degrees of freedom. The stochastic nature of Monte Carlo (MC) simulation has rendered it a crucial tool for studying many-body physics [2, 3]. Nevertheless, Monte Carlo methods encounter issues such as critical slowdown around phase transitions, which may limit their applicability to complex systems. Improving the Monte Carlo algorithm or discovering alternative tools has become a vibrant research area in the field [4–10].

As a statistical tool, deep learning has attracted significant attention in physics and other scientific disciplines in recent years due to its capability to model highly complex functions [11–17]. Monte Carlo simulation is commonly used to generate a large amount of high-quality data for studying many-body systems, making the analysis of this data using deep learning an intriguing area of research. For instance, prior studies have demonstrated that unsupervised deep learning can roughly identify phase transition points in many-body systems using configuration samples from Monte Carlo simulation [18–20], while supervised learning can pinpoint these points with high precision [21,22]. Efforts have also been made to employ deep learning to identify the effective non-interacting subsystem of a many-body system and incorporate it into Monte Carlo simulation to expedite sampling of the original system [8].

Generative Adversarial Networks (GANs) are non-end-to-end deep learning models that have garnered much attention owing to its ability in generating high-quality samples from an implicit probability distribution. In computer vision, GANs are used for tasks such as image modification, generation, and enhancing image resolution [23–25]. Recently, GANs have also started to gain attention in fields such as chemistry, pharmaceuticals, and engineering [26–31]. In physics, recent research by J. Singh *et. al.* has shown that implicit-GANs can replace Monte Carlo methods to sample XY model's spin configurations and predict phase transitions without prior knowledge of symmetry breaking [32]. However, there are differences between the spin configurations sampled by implicit-GANs and those sampled by Markov Chain Monte Carlo (MCMC), leading to discrepancies in the measured physical quantities at high temperatures. In this study, we further explore the application of GANs to many-body systems, focusing on addressing these issues.

Firstly, to enhance the performance of GANs in fitting Monte Carlo data, we employ conditional GANs with an auxiliary classifier (AC-GAN) as our primary architecture [33]. To improve the Generator's ability to learn the correlation between spins, we integrate multi-head self-attention layers into the Generator. The physical quantities measured by our AC-GANs agrees almost perfectly with those measured by MCMC [34, 35]. Secondly, we discover that

an AC-GAN can be effectively trained using an augmented dataset obtained by applying rotations, translations, mirror symmetries, etc., to a very small raw dataset. This finding enables us to apply AC-GANs to many-body systems where sampling spin configurations with MCMC is challenging. Finally, we demonstrate that GANs can be utilized to locate phase transition points in many-body systems. We propose a measure of the difference between the distribution of GAN-generated spin configuration samples and MCMC samples to identify phase transition points. We observe that phase transitions are accompanied by significant changes in the configuration distribution. The method does not require any preprocessing of raw data and offers a more physically intuitive approach to locate the phase transition points compared to most other methods [18–22].

The paper is organized as follows. In Section 2, we introduce the deep learning model AC-GANs used in this work. Then, in Section 3, we compare the AC-GANs and MCMC results on spin configurations and measured physical quantities in the two-dimensional (2D) XY model. Section 4 presents the results in the 2D XY and the three-dimensional (3D) constrained Heisenberg models with AC-GANs trained with data augmentation. In Section 5, we discuss the property of generative configurations in AC-GANs. We demonstrate how this property can be used to locate the phase transition. Finally, we provide a conclusion in Section 6.

2 Conditional GAN with auxiliary classifier

The basic structure of GANs consists of a generator (G) and a discriminator (D), which are two independent deep learning models, along with a database for storing real samples. The generator inputs a random noise matrix z and produces generated samples as output. These generated samples, along with the real samples, are then fed into the discriminator, which performs binary classification to distinguish between generated and real samples. The discriminator aims to accurately differentiate between the two types of samples, while the generator aims to produce samples that closely resemble real ones to deceive the discriminator [23]. In practice, the discriminator and generator are trained alternately in an adversarial manner, allowing both models to improve their performance iteratively through parameter updates.

In AC-GAN, along with the random noise matrix z, constraineds are included as a conditional matrix c in the input to the generator. The discriminator now has the additional task of identifying which conditional class the input sample belongs to, besides determining whether the input is a real sample or a generated one. This extra output can assist the discriminator in updating the machine's parameters during backward propagation, resulting in more stable training and faster convergence [33]. A schematic diagram of the AC-GAN architecture is depicted in Fig.1.

For the many-body systems we considered in the following sections, we used the spin configurations at each site to create input matrices. We divided the continuous temperature range into intervals of 0.2 to form N_c conditional classes. Generally, the elements in the configuration matrix are correlated due to spatial correlations present in the many-body system [36–38]. However, the kernel in the convolutional neural network (CNN) of a classical generator typically has a small receptive field (the common kernel size being 3x3, resulting in an effective receptive field of only 2x2) [39]. This limitation makes it difficult for the generator to produce configurations with long correlation lengths. To address this, we introduced self-attention layers to our generator. These layers can capture global features from the feature map and

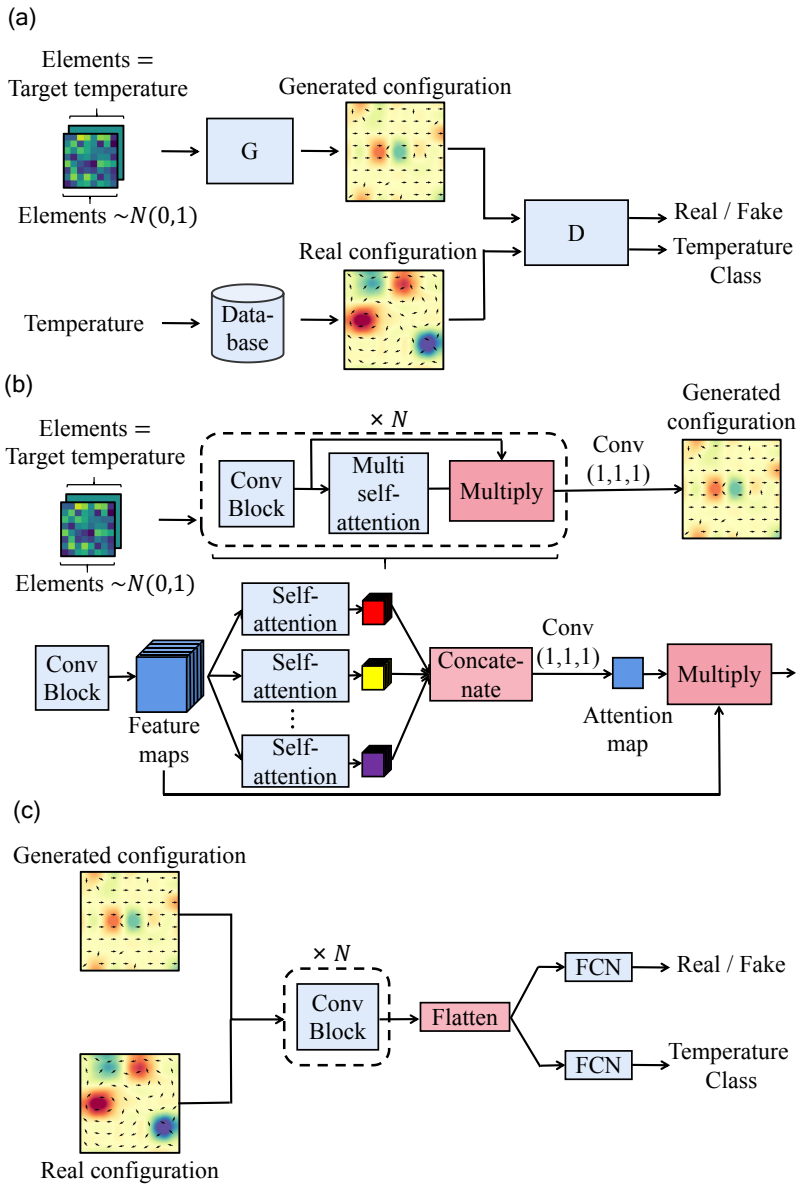

Figure 1: (a) depicts the main architecture of the AC-GAN. The block labelled "G" and "D" represents the generator and the discriminator, respectively, and their detailed architecture is shown in Appendix A. The generator takes as input a random noise matrix with elements generated from a Gaussian distribution and a condition matrix. (b) and (c) illustrate the architectures of the generator and the discriminator, respectively. The generator comprises a series of CNN blocks and self-attention blocks, while the discriminator consists solely of a series of CNN blocks and outputs the real/fake classification and condition class classification.

update the configuration matrix elements based on the extracted long-range information during the forward propagation of the neural network (see Appendix A for an introduction to self-attention layers and the detailed architecture). This modification allows our AC-GAN to generate configurations in larger size systems [34, 35].

The elements of the random noise matrix are generated from a Gaussian distribution with mean 0 and variance 1. Along with the conditional matrix, the random noise matrix is input

into the generator, which then passes through several CNN blocks and multi-head self-attention layers (Fig. 1(b)). These neural networks transform the input data into spin configurations that meet the temperature conditions and generate diverse spin configurations from the random noise matrix. The discriminator is a classical CNN neural network comprising multiple CNN blocks (Fig. 1(c)).

The loss functions of the generator and the discriminator are

$$\begin{aligned} GLoss = &\left\langle -\log D_1\left(X_{\text{real}}\right) + \log\left(1 - D_1\left(X_{\text{gen}}\right)\right)\right\rangle \\ &- \left\langle Y\left(X_{\text{real}}\right)\log D_2\left(X_{\text{real}}\right)\right\rangle - \left\langle Y\left(X_{\text{gen}}\right)\log\left(D_2\left(X_{\text{gen}}\right)\right)\right\rangle, \end{aligned} \tag{1}$$

and

$$\begin{aligned} DLoss = &\left\langle -\log D_1\left(X_{\text{real}}\right) - \log\left(1 - D_1\left(X_{\text{gen}}\right)\right)\right\rangle \\ &- \left\langle Y\left(X_{\text{real}}\right)\log D_2\left(X_{\text{real}}\right)\right\rangle - \left\langle Y\left(X_{\text{gen}}\right)\log D_2\left(X_{\text{gen}}\right)\right\rangle, \end{aligned} \tag{2}$$

respectively [33]. In the above equations, D_1 is one output branch of the discriminator that classifies the input configuration as either real data or generated data, and D_2 is another branch that determines which conditional class the input configuration belongs to. X_{real} and X_{gen} represent the real data and the generated data, respectively, while Y denotes the true condition class corresponding to the X fed into the discriminator, and $\langle\cdots\rangle$ represents the expectation value. It can be observed that $GLoss$ expects the discriminator to misclassify real and generated data, while $DLoss$ expects the discriminator to correctly classify the data. Furthermore, both $GLoss$ and $DLoss$ expect the discriminator to correctly classify the conditional classes of the input samples.

3 Generating spin configurations of the XY model

3.1 The 2D XY Model

The Hamiltonian of the 2D XY model is given by

$$H = -J\sum_{\langle i,j\rangle}\cos(\theta_i - \theta_j), \tag{3}$$

where J represents the strength of the spin-spin interaction, $\theta_i \in (0, 2\pi]$ denotes the spin angle at the i-th site, and the sum is taken over all nearest neighbors. The XY model on a square lattice is a classical model that displays a Kosterlitz-Thouless (KT) transition, where the spin-spin correlations decay algebraically and exponentially in the low and high temperature phases, respectively [40]. In the low-temperature phase, the vortices and anti-vortices stay close to each other to minimize the system's energy and tend to annihilate, resulting in vanishing local winding numbers. In the high-temperature phase, the vortices and anti-vortices become free. The phase transition occurs at $T_c/J = 0.89$, where the unbinding of vortex-antivortex pairs begins as the temperature increases.

3.2 Training data and training process

We utilized MCMC to generate training data for the XY model with system sizes $L \times L$ for $L = 16$ and $L = 32$. To circumvent critical slowing down, we sampled the data in temperature

regions away from the phase transition point, specifically $T/J \in ([0, 0.8] \cup [1.4, 2])$. A total of 10,000 spin configurations were obtained and used as the real samples (database data in Fig.1 (a)) to train the AC-GAN.

For the discriminator, the continuous temperature conditions are divided into 10 classes with intervals of 0.2. Since the generator may produce unreasonable configurations early in the training process, an extra class is introduced to label the conditional classes that are outside the temperature range $T/J \in [0, 2]$. In other words, we assign integer labels from 1 to 10 to the database data and label the generated data with 11 as the condition class, respectively.

For each epoch, we begin by inputting a batch of random noise matrices and temperature conditions (in the temperature range $T \in ([0, 0.8] \cup [1.4, 2])$) into the generator to generate a batch of configurations. Another batch of real configurations is sampled from the database and combined with the generated configurations. Subsequently, the discriminator is trained twice with the mixed data. Then, a new batch of random noise matrices and temperature conditions is sampled and used to train the generator twice.

The optimizer used in both the generator and the discriminator is RMSprop, with a learning rate of 2.5×10^{-4} and a clip value of 0.1. The sigmoid function is employed as the activation function for D_1 in Eqs. (1) and (2), allowing the output value to range between 0 and 1. A linear function is used as the activation function for D_2, with the output value ranging from 0 to 2. The kernel size of the CNN layer is specified in Fig. 7 of Appendix A. The batch size for each epoch is 64, consisting of 32 real input data and 32 generated input data for the discriminator, and 64 random input data for the generator. A total of 5000 epochs are preformed for one complete training cycle.

3.3 Results

To evaluate the performance of the AC-GAN, we sample spin configurations from a well-trained AC-GAN across the entire temperature range and compare the physical quantities measured by the AC-GAN spin configurations with those measured by MCMC spin configurations. Three key physical quantities are of interest to us. Firstly, we examine the energy, as it is a fundamental characteristic of a physical system. Additionally, since the XY model elucidates magnetic dipole-dipole interactions between spins, we also calculate the magnetization as a function of temperature. Lastly, we also measure the vorticity, which quantifies the binding of vortex-antivortex pairs.

For a given spin configuration, the vorticity is characterized by the local winding numbers. In the continuous case, the winding number is defined as the integration over a closed loop γ:

$$W(\gamma) = \frac{1}{2\pi} \oint_{\gamma} (x dy - y dx), \tag{4}$$

where x and y represent the spin components in Cartesian coordinates. In the lattice model, we first select a specific site i and identify the eight neighboring sites around it. These eight sites form a loop, and we calculate the difference in spin orientations between the neighboring spins around this loop. The mean of these differences is then taken as the winding number of the site i, given by:

$$W_i = \frac{1}{8 \sin \frac{\pi}{4}} \sum_{j=1}^{8} \left(\sin \theta_{j+1} \cos \theta_j - \sin \theta_j \cos \theta_{j+1} \right), \tag{5}$$

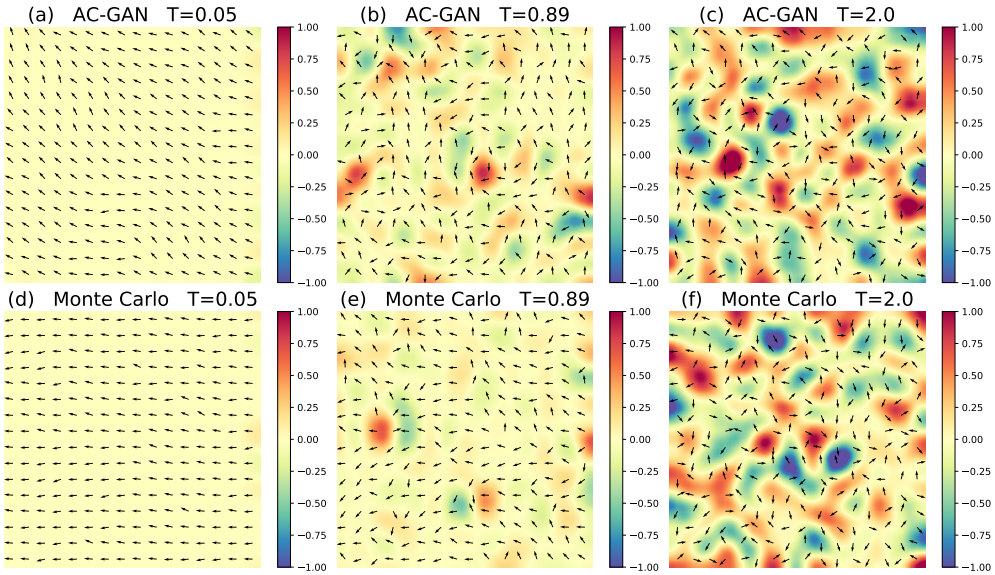

Figure 2: The spin configurations (depicted as black arrows) of the XY model on a square lattice sampled from AC-GAN (top row) and Monte Carlo simulations (bottom row) at various temperatures. The color indicates the local winding number. The spin configurations sampled from both AC-GAN and Monte Carlo simulations exhibit consistent features. At low temperatures, the spins tend to align, resulting in a local winding number close to zero everywhere. As the temperature increases across the Kosterlitz-Thouless transition, the spins become more disordered, leading to the unbinding of vortex-antivortex pairs.

where θ_{j+1} represents the spin orientation adjacent to that of the j-th site (i.e., θ_j) in the counterclockwise direction on the loop.

Figure 2 displays the spin configurations for $L = 16$ generated by AC-GAN and MCMC at low temperature (left panel), phase transition point (middle panel), and high temperature (right panel). The colour indicates the local winding number. With the implementation of the self-attention layer, the AC-GAN model captures the correlation between spins and generates spin configurations with relatively uniform orientation at low temperatures, consistent with those obtained from MCMC. As the temperature approaches the phase transition point and in the high-temperature region, thermal fluctuations become more pronounced, leading to disordered spin configurations generated by MCMC. Similar features are observed in the samples generated by AC-GAN.

We further compare the mean energy per site, mean magnetization, and the mean vorticity (average of the absolute value of the local winding number) measured by AC-GAN and MCMC. The results are shown in Fig. 3. In the temperature region of $T/J \in ([0, 0.8] \cup [1.4, 2])$, where we use a large amount of training data to train AC-GAN, the mean and variance of the observables measured by AC-GAN exhibit a trend similar to those measured by MCMC. In contrast, while Implicit-GAN [32] and AC-GAN show similar performance in magnetization measurement, our AC-GAN outperforms Implicit-GAN in the measurement of the other two quantities, especially the vorticity (see Appendix B for a quantitative comparison).

In Fig. 3, we observe that the AC-GAN results do not fall within the error range of the MCMC results for $L = 16$ in the low-temperature region, but they are consistent with the

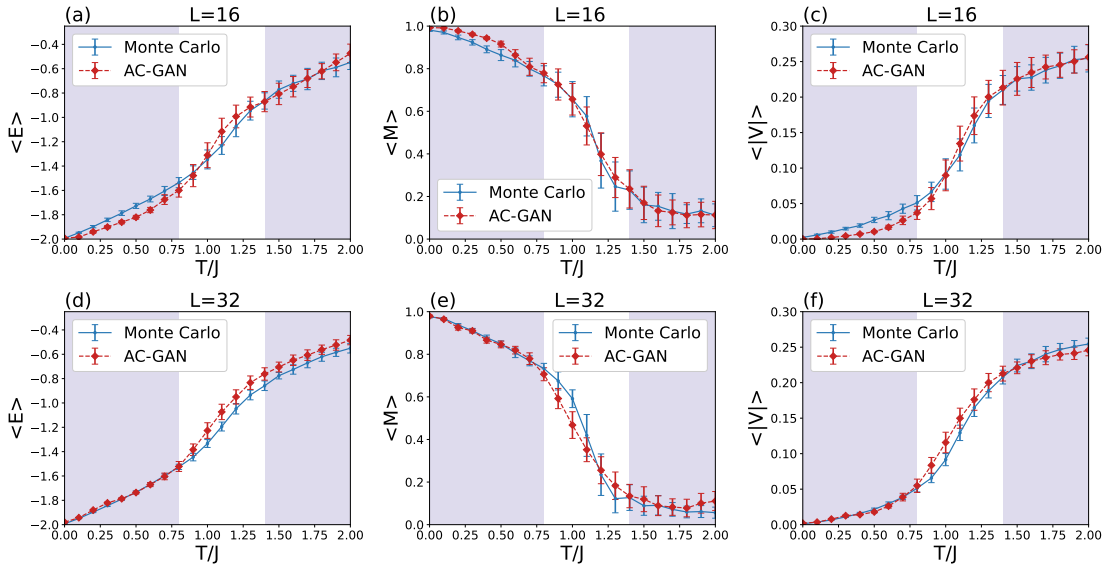

Figure 3: The mean energy density (left column), the mean magnetization (middle column), and the mean vorticity (right column) as a function of the temperature of the XY model with a system size of $L = 16$ (top row) and $L = 32$ (bottom row) measured from AC-GAN and MCMC. The error bars show the standard deviation of 100 data sampled by AC-GAN or MCMC. The results from the two methods agree reasonably well with each other.

MCMC results in the high-temperature region. This discrepancy arises from the fact that in small systems, fluctuations from a few spins at low temperatures can significantly affect the measured macroscopic quantity. However, this situation improves in larger systems. As shown in Fig. 3 (bottom panel), where the system size is doubled, the AC-GAN results fall within the error range of the MCMC results across the entire temperature range.

Moreover, it is worth noting that AC-GAN performs well at high temperatures for $L = 16$ and at low temperatures for $L = 32$. This suggests that the AC-GAN can effectively learn the spin configuration distribution at any temperature in principle. However, learning the spin configuration distribution across the entire temperature range in a single training session can be challenging, and is still an open question undergoing active investigation in generative learning research. Employing more sophisticated training techniques, such as inserting spectral normalization layers between CNN layers, using multiple generators to generate data, or replacing the cross-entropy loss of the discriminator with Wasserstein loss [41–43], may improve the results across the entire temperature range. Nevertheless, our results here demonstrate the potential of using GANs to accelerate MCMC sampling.

In the vicinity of phase transition, even without the training data, the AC-GAN results are similar to those of MCMC, suggesting that the deep learning model can extract information about the phase transition by learning configurations outside the critical region. Implicit-GAN obtains similar results for system sizes of 8×8 and 16×16 [32]. Additionally, from Fig. 3, it is worth noting that the performance of the AC-GAN does not deteriorate with the increase in the system size, indicating its capability to generate configurations in larger-sized systems.

As a remark, our deep learning model also suffers from a certain degree of mode collapse depending on the system's driving parameters. Nevertheless, as shown in Fig. 2, Fig. 3, and

Figure 8 in Appendix B, the mode collapse does not affect the machine's ability to sample physical quantities of interest. Furthermore, we also compare the difference in the distribution of AC-GAN and MCMC samples in Appendix C as quantified by the Earth Mover's Distance [44] and the Jensen-Shannon divergence [45, 46]. The results further suggest that the sampled distribution from AC-GAN is sufficiently close to that from MCMC.

4 Learning with a few raw data

Using MCMC to obtain samples from large-sized 2D and 3D condensed matter models requires a lot of computational time. In these cases, obtaining a large amount of raw data through MCMC for training GANs is challenging. In this section, we present a method to generate a large number of training samples from just a few raw MCMC data using symmetry operations.

4.1 The models and the training data

We tested our method on the 2D XY and 3D constrained Heisenberg models. For the XY model, a total of 16 spin configurations in the temperature range $T/J \in ([0, 0.8] \cup [1.4, 2])$ with an interval of 0.1 are sampled with MCMC. We then applied the following symmetry operations to each spin configuration: (1) randomly shifting all spins in the xy plane, (2) reflecting a spin configuration along the x-axis with a half probability, (3) reflecting a spin configuration along the y-axis with a half probability, (4) transposing a spin configuration with a half probability, and (5) randomly rotating all spins by an angle θ. A training dataset of 10,000 samples was obtained. The 2D XY model serves as a benchmark for comparing the results of learning with a few raw data to that with a large amount of data in the previous section, allowing us to gain a deeper understanding of the advantages and disadvantages of the method presented here.

We also explored whether the method is applicable to models in higher dimensions, where obtaining a large number of configurations through MCMC becomes computationally expensive. Specifically, we considered the 3D constrained Heisenberg model on a simple cubic lattice, whose Hamiltonian is given by

$$H = -J \sum_{\langle i,j \rangle} \mathbf{S}_i \cdot \mathbf{S}_j, \tag{6}$$

where J represents the interaction strength between nearest neighboring spins, $\mathbf{S}_i$ denotes the spin on the i-th site, which can be characterized by the polar angle ϕ and the azimuthal angle θ as $\mathbf{S}_i = (|S_i| \sin\phi_i \cos\theta_i, |S_i| \sin\phi_i \sin\theta_i, |S_i| \cos\phi_i)$, where $|S_i|$ is taken to be 1 in the simulations. In the Monte Carlo simulation, we update θ and ϕ using a uniform distribution from 0 to π and 2π, respectively. The update of θ here is different from the conventional practice [47] to challenge the AC-GANs in sampling spin configurations under different constraints. We sampled a total of 31 spin configurations from MCMC in the temperature range $T/J \in ([0, 1.5] \cup [3.5, 5])$ with an interval of 0.1. Then, we applied the same symmetry operations mentioned above for the 2D XY model to obtain 10,000 training samples.

4.2 Results

Figure 4 shows the results obtained from the 2D XY model with a system size of 32×32 and the 3D constrained Heisenberg model with a system size of $16 \times 16 \times 16$. The training process

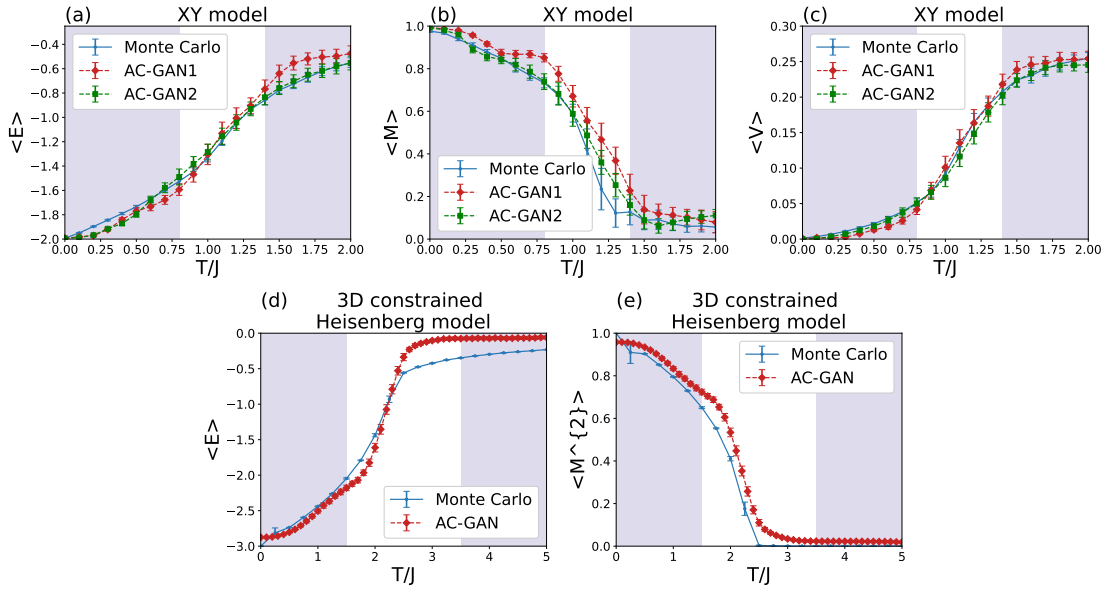

Figure 4: (a-c) show the mean energy density, mean magnetization, and mean vorticity as functions of temperature for the 2D XY model with a system size of 32×32, measured by AC-GAN and MCMC. The AC-GAN1 and AC-GAN2 results were obtained using only 1 and 10 raw data points, respectively, for each temperature interval. (d) and (e) show the mean energy and the mean squared magnetization as functions of temperature for the 3D constrained Heisenberg model with a system size of $16 \times 16 \times 16$. The error bars represent the standard deviation of 100 data points sampled by AC-GAN or Monte Carlo.

follows the same procedure described in Section 3. The observables, as measured from AC-GAN, exhibit a qualitatively similar behavior to those from MCMC, indicating that a small set of raw MCMC configurations can effectively train an AC-GAN. This suggests that such a data augmentation scheme could potentially be extended to other tasks involving deep learning in the study of many-body systems, thereby reducing the computational time required for collecting training data.

Comparing the results shown in Fig. 4 (a-c) to those shown in Fig. 3 (bottom row), we observe that the AC-GAN trained with a small dataset did not perform as well as that trained with a large number of training samples. However, we can reduce 99.84% of the time required for collecting the training data using this data augmentation scheme, while still achieving acceptable results. This will be particularly useful when sampling a system with a large size or strong auto-correlation, where the time needed to collect training data will be much longer than the time required to train GANs.

The main reason for the deteriorated performance is that although we obtain a large amount of training samples through data augmentation, these augmented data possess similar features, which can be easily captured by the discriminator. However, it is preferable that the generator and the discriminator gradually improve their performance during the training process to achieve the goal of adversarial learning. The use of augmented data greatly reduces the training difficulty of the discriminator and enables it to converge to a good local minimum more rapidly than the generator, which makes the training process unstable.

Here we demonstrate an extreme case where only one training sample is provided at each

temperature interval. As we increase the number of raw data from only 1 to 10 in each temperature interval, the performance of the AC-GAN improves, and the sampling results from AC-GAN generally agree better with those from MCMC, as shown in Fig. 4 (a-c). When applying this scheme to real research tasks, one can effectively mitigate the drawback on performance by increasing the amount of raw data, striking a balance between required computational resources, simulation time, and performance.

5 Detecting phase transitions by distribution difference

When a system is in a particular phase, the high-probability states following the Boltzmann distribution form a subspace in the entire phase space. Upon undergoing a phase transition and transitioning to another phase, the distribution's subspace undergoes a drastic change. From the perspective of spin configurations, the correlation length experiences a significant variation. For instance, in the classical 2D Ising model at low temperatures, spontaneous symmetry breaking leads to all spins in the system aligning in one direction, resulting in a large correlation length. Conversely, at high temperatures, due to the restoration of symmetry, spins in the system randomly point up or down, leading to a small correlation length. Similarly, in the classical XY model, although true spontaneous symmetry breaking doesn't occur due to the presence of vortices, phenomena resembling spontaneous symmetry breaking emerge in finite system sizes at low temperatures. In this case, the correlation length of spins becomes substantial. When employing AC-GANs to learn the spin configuration distribution at different temperatures, these distribution characteristics are captured. This capability allows us to use data generated by AC-GANs to identify phase transitions effectively.

5.1 The distribution overlap of spin configurations

We trained our AC-GAN for 5000 epochs using the full temperature range of MCMC spin configurations. The AC-GAN converged after 2000 epochs. For the next 3000 epochs, we sampled spin configurations from the AC-GAN with temperature intervals of 0.1. To measure the distribution difference, we calculated the overlap of the distribution sampled from the AC-GAN and that from MCMC. To collect a sufficiently diverse dataset, we repeatedly restarted the Markov chain and employed various random initial spin configurations in each temperature interval.

Theoretically, one should calculate the overlap of the two distributions in all dimensions of the configuration space. However, for a many-body system, the dimension of the configuration space is usually high. One needs to sample a large number of configurations from AC-GAN and MCMC to calculate the overlap accurately, but that would require a lot of computational time. Instead of calculating the overlap between the two distributions in the whole space, we randomly select two sites from the spin configurations and calculate the overlap of their distributions using the formula:

$$\text{CrossArea}(S_i, S_j) = \frac{\min(F_{MC}(S_i, S_j), F_{GAN}(S_i, S_j))}{F_{MC}(S_i, S_j)}. \tag{7}$$

Here, S_i is the randomly selected spin on site i, and F_{MC} and F_{GAN} are the distributions of the spin configurations from MCMC and AC-GAN, respectively. To minimize the error of the overlap calculated by the two sites compared to the full-dimensional space, we repeated the

random selection of two sites 100 times at each temperature and calculated the averaged overlap.

Since the order parameters that detect the phase transitions in condensed matter models may not always be known and can be difficult to obtain, in most cases, even with extensive Monte Carlo data, accurately locating the transition points can be challenging. On the other hand, the method proposed here relies only on the Monte Carlo data, AC-GANs data, and the equation above, but not on the knowledge of the order parameter, to locate the phase transition point.

5.2 Results

We applied the above-mentioned scheme for detecting phase transitions to the 2D square lattice Ising model with $L \in \{16, 32, 64\}$ and the XY model with $L \in \{16, 32\}$, and the results are shown in Fig. 5. The data collection process and the training process of the AC-GAN are similar to that described in Sec. 3, except that we are now using training data across the full temperature range. In Fig. 5(a), the overlap (CrossArea in Eq. (7)) between the distribution of the AC-GAN and MCMC is about 0.5 at low temperatures, indicating that half of the spin configurations from MCMC are pointing up and the other half are pointing down, while all spin configurations from AC-GAN are pointing up (or all pointing down). As the temperature increases, the spin configurations of MCMC eventually become disordered, and the distribution transforms into a uniform one. Consequently, the overlap between the AC-GAN and MCMC distributions increases to a value close to one. Figure 5(c) shows the first derivative of the CrossArea with respect to the temperature. The derivative exhibits a significant change around the transition. From the maximum of the gradient, we estimated the transition temperature to be 2.5, 2.4, and 2.3 (in the unit of J) for $L = 16$, 32, and 64, respectively. The estimated transition temperature tends to the theoretical value as the system size increases.

Figure 5(b) shows the results of the 2D XY model. When the system is at low temperature, the configurations sampled by MCMC can have spins aligning in any direction, which can cause AC-GAN to experience a symmetry-breaking-like phenomenon. Consequently, the configurations sampled by AC-GAN will have spins aligning in one direction, while the input random noise matrix only causes small local fluctuations in the spins. Therefore, the overlap between MCMC data and AC-GAN data is low. As the temperature of the system gradually increases across the phase transition, the distribution of MCMC data gradually tends to a uniform distribution, which is a friendly and easy-to-fit distribution for AC-GAN. Thus, the AC-GAN will not experience the symmetry-breaking-like phenomenon, and the resulting overlap between the MCMC and AC-GAN data is high. The overlap converges to about 0.5 above $T/J = 1.2$.

It is worth noting that the overlap tending to 0.5 at high temperatures does not mean that AC-GAN experiences the symmetry-breaking-like phenomenon. Instead, it is caused by an insufficient number of samples when we calculate the overlap of two distributions. While the spins of the Ising model are binary and we can accurately calculate the overlap, the spins of the XY model can have orientations ranging from $(0, 2\pi]$. This requires more samples to better reflect the distribution of MCMC data and AC-GAN data on the 2-dimensional plane. If the number of samples increases, the overlap is expected to converge to a value closer to one. Nevertheless, to detect the phase transition, we only care about whether there is a significant change in the overlap as a function of the driving parameter, not the value to which the overlap converges.

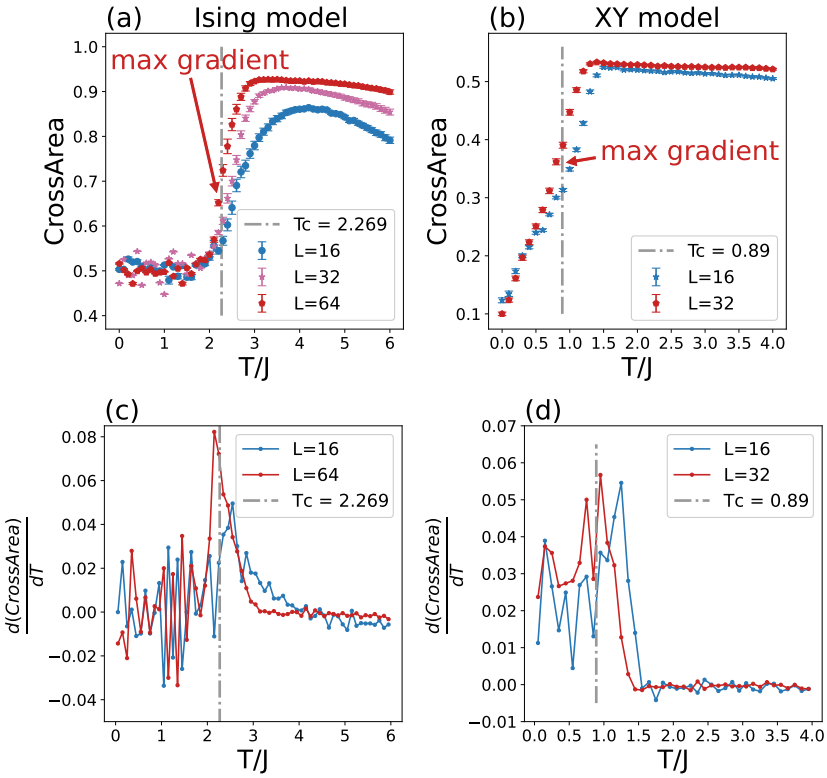

Figure 5: Cross-area as a function of temperature in (a) the 2D Ising model and (b) the 2D XY model with various system sizes. (c) and (d) depict the first derivative of the Cross-Area in (a) and (b) with respect to temperature, respectively. The transition temperature estimated by the maximum of the gradients for the largest system agrees well with the theoretical transition temperature (indicated by the vertical dash-dotted line). The error bars represent standard deviations of 100 samples.

Figure 5(d) shows the first derivative of the CrossArea with respect to the temperature, and the estimated phase transition points from the maximum gradient are $T_c/J = 1.0$ and $T_c/J = 0.9$ for $L = 16$ and $L = 32$, respectively. Note that similar to other unsupervised machine learning methods for phase transition detection [18, 19, 21, 48–52], a detailed error estimation on the predicted T_c is not available. However, the convergence of the predicted T_c as the system size increases suggests the validity of the method.

Finally, it is important to note that the method we propose here does not require any data pre-processing and is universally applicable for identifying phase transitions. This approach also provides a physical intuition for interpreting the deep learning model, as it correlates phase transitions with changes in the configuration distribution.

6 Conclusion

In this study, we explored the application of GANs in many-body systems. We found that AC-GANs with self-attention layers could effectively capture spin-spin correlations in many-body systems and produce high-quality configuration samples. We validated our deep learning model using the 2D XY model. The spin configurations generated by AC-GAN closely

resembled those obtained by MCMC. Moreover, the mean energy, magnetization, and vorticity calculated from AC-GAN also exhibited strong agreement with those derived from MCMC.

We further investigated the performance of AC-GAN trained with only a few raw data samples. Using MCMC, we sampled a limited number of spin configurations and applied data augmentation through symmetry operations. Remarkably, the AC-GAN was successfully trained with the augmented data and demonstrated strong performance. The success of this data augmentation method not only enables the study of condensed matter models with challenges in MCMC sampling via AC-GAN but also suggests its applicability to other deep learning tasks requiring a large number of configurations.

We also observed that the symmetry-breaking-like phenomenon in AC-GAN can serve as a tool for locating phase transitions. We trained AC-GAN using MCMC data spanning the full temperature range. By computing the overlap between the distributions of configurations sampled by AC-GAN and MCMC, we quantified the distribution difference and successfully identified the transition temperatures of the 2D Ising and XY models on a square lattice. This approach to phase transition detection provides physical insights into the information encoded in the configuration distributions, rather than relying solely on the order parameters. One could extend this method to more complex systems, such as the random bond model [53], where either the order parameter is unknown or accurately calculating the order parameter using Monte Carlo simulations is challenging.

Acknowledgements

We acknowledge financial support from Research Grants Council of Hong Kong (Grant No. CityU 11318722), National Natural Science Foundation of China (Grant No. 12005179, 12204130), City University of Hong Kong (Grant No. 9610438, 7005610, 9680320), and Harbin Institute of Technology Shenzhen (Grant No. ZX20210478, X20220001).

References

[1] P. W. Anderson, *More is different: broken symmetry and the nature of the hierarchical structure of science.*, Science **177**(4047), 393 (1972), doi:10.1126/science.177.4047.393.

[2] D. P. Kroese, T. Brereton, T. Taimre and Z. I. Botev, *Why the monte carlo method is so important today*, WIREs Comput. Stat. **6**(6), 386 (2014), doi:10.1002/wics.1314.

[3] J. Kolorenč and L. Mitas, *Applications of quantum monte carlo methods in condensed systems*, Rep. Prog. Phys. **74**(2), 026502 (2011), doi:10.1088/0034-4885/74/2/026502.

[4] R. H. Swendsen and J.-S. Wang, *Replica monte carlo simulation of spin-glasses*, Phys. Rev. Lett. **57**(21), 2607 (1986), doi:10.1103/PhysRevLett.57.2607.

[5] U. Wolff, *Collective monte carlo updating for spin systems*, Phys. Rev. Lett. **62**(4), 361 (1989), doi:10.1103/PhysRevLett.62.361.

[6] B. A. Berg and T. Neuhaus, *Multicanonical ensemble: A new approach to simulate first-order phase transitions*, Phys. Rev. Lett. **68**(1), 9 (1992), doi:10.1103/PhysRevLett.68.9.

[7] N. Prokof'ev, B. Svistunov and I. S. Tupitsyn, *"worm" algorithm in quantum monte carlo simulations*, Phys. Lett. A **238**(4-5), 253 (1998), doi:10.1016/S0375-9601(97)00957-2.

[8] J. Liu, Y. Qi, Z. Y. Meng and L. Fu, *Self-learning monte carlo method*, Phys. Rev. B **95**(4), 041101 (2017), doi:10.1103/PhysRevB.95.041101.

[9] L. Del Debbio, J. M. Rossney and M. Wilson, *Efficient modeling of trivializing maps for lattice ϕ 4 theory using normalizing flows: A first look at scalability*, Phys. Rev. D **104**(9), 094507 (2021), doi:10.1103/PhysRevD.104.094507.

[10] P. Białas, P. Korcyl and T. Stebel, *Analysis of autocorrelation times in neural markov chain monte carlo simulations*, Phys. Rev. E **107**(1), 015303 (2023), doi:10.1103/PhysRevE.107.015303.

[11] N. M. Ball and R. J. Brunner, *Data mining and machine learning in astronomy*, Int. J. Mod. Phys. D **19**(07), 1049 (2010), doi:10.1142/S0218271810017160.

[12] D. Baron, *Machine learning in astronomy: A practical overview*, arXiv:1904.07248 (2019), doi:10.48550/arXiv.1904.07248.

[13] A. Radovic, M. Williams, D. Rousseau, M. Kagan, D. Bonacorsi, A. Himmel, A. Aurisano, K. Terao and T. Wongjirad, *Machine learning at the energy and intensity frontiers of particle physics*, Nature **560**(7716), 41 (2018), doi:10.1038/s41586-018-0361-2.

[14] M. S. Albergo, D. Boyda, D. C. Hackett, G. Kanwar, K. Cranmer, S. Racanière, D. J. Rezende and P. E. Shanahan, *Introduction to normalizing flows for lattice field theory*, arXiv:2101.08176 (2021), doi:10.48550/arXiv.2101.08176.

[15] E. Bedolla, L. C. Padierna and R. Castaneda-Priego, *Machine learning for condensed matter physics*, J. Phys.: Condens. Matter **33**(5), 053001 (2020), doi:10.1088/1361-648X/abb895.

[16] J. F. Rodriguez-Nieva and M. S. Scheurer, *Identifying topological order through unsupervised machine learning*, Nat. Phys. **15**(8), 790 (2019), doi:10.1038/s41567-019-0512-x.

[17] J. Carrasquilla, *Machine learning for quantum matter*, Adv. Phys.: X **5**(1), 1797528 (2020), doi:10.1080/23746149.2020.1797528.

[18] L. Wang, *Discovering phase transitions with unsupervised learning*, Phys. Rev. B **94**(19), 195105 (2016), doi:10.1103/PhysRevB.94.195105.

[19] W. Hu, R. R. Singh and R. T. Scalettar, *Discovering phases, phase transitions, and crossovers through unsupervised machine learning: A critical examination*, Phys. Rev. E **95**(6), 062122 (2017), doi:10.1103/PhysRevE.95.062122.

[20] K. Kottmann, P. Huembeli, M. Lewenstein and A. Acín, *Unsupervised phase discovery with deep anomaly detection*, Phys. Rev. Lett. **125**(17), 170603 (2020), doi:10.1103/PhysRevLett.125.170603.

[21] E. P. Van Nieuwenburg, Y.-H. Liu and S. D. Huber, *Learning phase transitions by confusion*, Nat. Phys. **13**(5), 435 (2017), doi:10.1038/nphys4037.

[22] J. Carrasquilla and R. G. Melko, *Machine learning phases of matter*, Nat. Phys. **13**(5), 431 (2017), doi:10.1038/nphys4035.

[23] J. Gui, Z. Sun, Y. Wen, D. Tao and J. Ye, *A review on generative adversarial networks: Algorithms, theory, and applications*, IEEE Trans. Knowl. Data Eng. (2021), doi:10.1109/TKDE.2021.3130191.

[24] A. Aggarwal, M. Mittal and G. Battineni, *Generative adversarial network: An overview of theory and applications*, IJIMDI **1**(1), 100004 (2021), doi:10.1016/j.jjimei.2020.100004.

[25] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta and A. A. Bharath, *Generative adversarial networks: An overview*, IEEE Signal Process. Mag. **35**(1), 53 (2018), doi:10.1109/MSP.2017.2765202.

[26] Y. Li, O. Vinyals, C. Dyer, R. Pascanu and P. Battaglia, *Learning deep generative models of graphs*, arXiv:1803.03324 (2018), doi:10.48550/arXiv.1803.03324.

[27] W. P. Walters and M. Murcko, *Assessing the impact of generative ai on medicinal chemistry*, Nat. Biotechnol. **38**(2), 143 (2020), doi:10.1038/s41587-020-0418-2.

[28] Y. Mao, Q. He and X. Zhao, *Designing complex architectured materials with generative adversarial networks*, Sci Adv. **6**(17), eaaz4169 (2020), doi:10.1126/sciadv.aaz4169.

[29] Y. Dan, Y. Zhao, X. Li, S. Li, M. Hu and J. Hu, *Generative adversarial networks (gan) based efficient sampling of chemical composition space for inverse design of inorganic materials*, npj Computational Materials **6**(1), 1 (2020), doi:10.1038/s41524-020-00352-0.

[30] S. Kim, J. Noh, G. H. Gu, A. Aspuru-Guzik and Y. Jung, *Generative adversarial networks for crystal structure prediction*, ACS central science **6**(8), 1412 (2020), doi:10.1021/acscentsci.0c00426.

[31] A. Nouira, N. Sokolovska and J.-C. Crivello, *Crystalgan: learning to discover crystallographic structures with generative adversarial networks*, arXiv:1810.11203 (2018), doi:10.48550/arXiv.1810.11203.

[32] J. Singh, M. Scheurer and V. Arora, *Conditional generative models for sampling and phase transition indication in spin systems*, SciPost Phys. **11**(2), 043 (2021), doi:10.21468/SciPostPhys.11.2.043.

[33] A. Odena, C. Olah and J. Shlens, *Conditional image synthesis with auxiliary classifier gans*, In *ICML*, pp. 2642–2651. PMLR, doi:10.48550/arXiv.1610.09585 (2017).

[34] H. Zhang, I. Goodfellow, D. Metaxas and A. Odena, *Self-attention generative adversarial networks*, In *ICML*, pp. 7354–7363. PMLR, doi:10.48550/arXiv.1805.08318 (2019).

[35] A. Brock, J. Donahue and K. Simonyan, *Large scale gan training for high fidelity natural image synthesis*, arXiv:1809.11096 (2018), doi:10.48550/arXiv.1809.11096.

[36] T. T. Wu, B. M. McCoy, C. A. Tracy and E. Barouch, *Spin-spin correlation functions for the two-dimensional ising model: Exact theory in the scaling region*, Phys. Rev. B **13**(1), 316 (1976), doi:10.1103/PhysRevB.13.316.

[37] D. X. Nui, L. Tuan, N. D. T. Kien, P. T. Huy, H. T. Dang and D. X. Viet, *Correlation length in a generalized two-dimensional xy model*, Phys. Rev. B **98**(14), 144421 (2018), doi:10.1103/PhysRevB.98.144421.

[38] Y.-H. Li and S. Teitel, *Finite-size scaling study of the three-dimensional classical xy model*, Phys. Rev. B **40**(13), 9122 (1989), doi:10.1103/PhysRevB.40.9122.

[39] W. Luo, Y. Li, R. Urtasun and R. Zemel, *Understanding the effective receptive field in deep convolutional neural networks*, Adv. Neural Inf. Process. Syst. **29** (2016), doi:10.48550/arXiv.1701.04128.

[40] J. M. Kosterlitz and D. J. Thouless, *Ordering, metastability and phase transitions in two-dimensional systems*, J. Phys. C **6**(7), 1181 (1973), doi:10.1088/0022-3719/6/7/010.

[41] M. Arjovsky, S. Chintala and L. Bottou, *Wasserstein generative adversarial networks*, In *International conference on machine learning*, pp. 214–223. PMLR, doi:10.48550/arXiv.1701.07875 (2017).

[42] T. Miyato, T. Kataoka, M. Koyama and Y. Yoshida, *Spectral normalization for generative adversarial networks*, arXiv:1802.05957 (2018), doi:10.48550/arXiv.1802.05957.

[43] S. Zhu, R. Urtasun, S. Fidler, D. Lin and C. Change Loy, *Be your own prada: Fashion synthesis with structural coherence*, In *Proceedings of the IEEE international conference on computer vision*, pp. 1680–1688, doi:10.1109/ICCV.2017.186 (2017).

[44] Y. Rubner, C. Tomasi and L. J. Guibas, *A metric for distributions with applications to image databases*, In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, pp. 59–66. IEEE, doi:10.1109/ICCV.1998.710701 (1998).

[45] I. Dagan, L. Lee and F. Pereira, *Similarity-based methods for word sense disambiguation*, arXiv (1997), doi:10.48550/arXiv.cmp-lg/9708010.

[46] J. Lin, *Divergence measures based on the shannon entropy*, IEEE Transactions on Information theory **37**(1), 145 (1991), doi:10.1109/18.61115.

[47] P. Peczak and D. Landau, *Monte carlo study of critical relaxation in the 3d heisenberg model*, J. Appl. Phys. **67**(9), 5427 (1990), doi:10.1063/1.344578.

[48] S. J. Wetzel, *Unsupervised learning of phase transitions: From principal component analysis to variational autoencoders*, Phys. Rev. E **96**(2), 022140 (2017), doi:10.1103/PhysRevE.96.022140.

[49] A. Canabarro, F. F. Fanchini, A. L. Malvezzi, R. Pereira and R. Chaves, *Unveiling phase transitions with machine learning*, Phys. Rev. B **100**(4), 045129 (2019), doi:10.1103/PhysRevB.100.045129.

[50] J. Wang, W. Zhang, T. Hua and T.-C. Wei, *Unsupervised learning of topological phase transitions using the calinski-harabaz index*, Phys. Rev. Res **3**(1), 013074 (2021), doi:10.1103/PhysRevResearch.3.013074.

[51] Y.-H. Tsai, K.-F. Chiu, Y.-C. Lai, K.-J. Su, T.-P. Yang, T.-P. Cheng, G.-Y. Huang and M.-C. Chung, *Deep learning of topological phase transitions from entanglement aspects: An unsupervised way*, Phys. Rev. B **104**(16), 165108 (2021), doi:10.1103/PhysRevB.104.165108.

[52] E.-J. Kuo and H. Dehghani, *Unsupervised learning of interacting topological and symmetry-breaking phase transitions*, Phys. Rev. B **105**(23), 235136 (2022), doi:10.1103/PhysRevB.105.235136.

[53] A. Scaramucci, H. Shinaoka, M. V. Mostovoy, R. Lin, C. Mudry and M. Müller, *Spiral order from orientationally correlated random bonds in classical x y models*, Phys. Rev. Res. **2**(1), 013273 (2020), doi:10.1103/PhysRevResearch.2.013273.

A Self-attention layers and detailed machine architecture

In the classical convolutional GAN, the decoder and encoder in the generator are multi-layer convolutional neural networks (CNNs). The performance of this GAN model will be limited by the kernel size of the CNN layers. Even though the receptive field of the kernel increases as the encoder deepens, the kernel only operates on the matrix elements within its size in the feature map, and information from elements far away is ignored. However, long-range correlations can play a significant role in condensed matter models. If the generator is unable to capture this long-range information in the feature map, the generated spin configurations will be of low quality. To overcome this deficiency, we introduced the self-attention layers into the network. The self-attention layers can learn the global relationship between the elements and further adjust the value of each element in the feature map, thus allowing the generator to fine-tune the spin on each site in the generated configurations according to the long-range correlation.

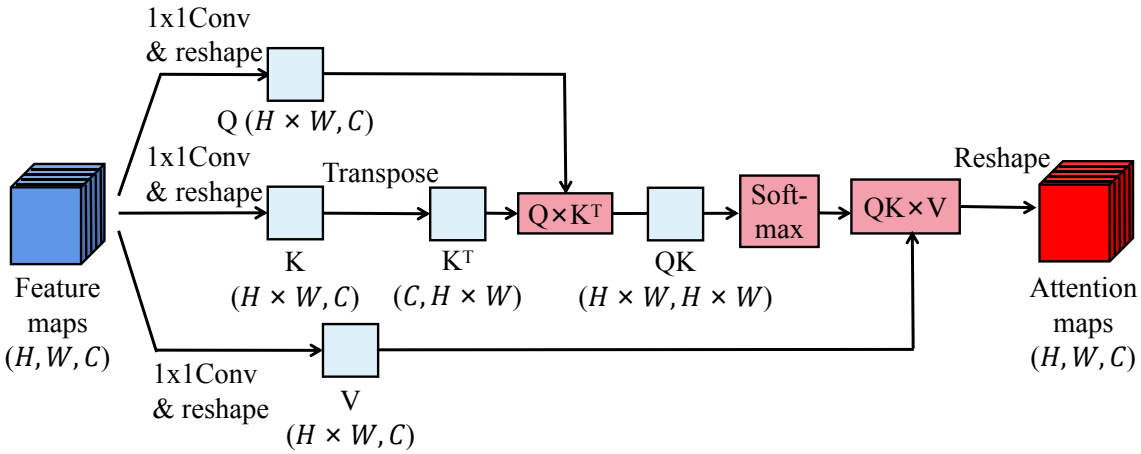

Figure 6: The architecture of the self-attention layer.

The architecture of the self-attention layer is shown in Fig. 6. Specifically, the feature maps input to the self-attention layer are a 3D tensor with shapes of height, width, and channel (H, W, C). These feature maps are fed into three CNN layers with a kernel size of 1x1 and reshaped into $(H \times W, C)$. Consequently, we obtain three matrices named query (Q), key (K), and value (V) respectively. The transpose of K is multiplied by Q to obtain a matrix QK, also known as the energy matrix, with a shape of $(H \times W, H \times W)$. In the energy matrix, the (i, j)-th element represents the relationship between the i-th element in the feature map across all channels and the j-th element across all channels (note that here we only use one index to denote the position of an element in the feature map since we have flattened each feature map into a 1D vector). By applying the softmax activation function on QK, all elements of each row of QK now add up to one. This re-scaling helps the self-attention layer converge faster. Finally, QK is multiplied by V to incorporate the channel information. The resultant matrices are then reshaped into a 3D tensor with the same shape as the input feature maps.

Figure 7 shows the detailed architecture of the machine learning model used in this work. In this figure, the content labeled 'conv 3x3 64' indicates the use of 64 filters in the convolution layer, each being a 3×3 matrix. Pixelwise multiplication implies that each element in every channel of the $16 \times 16 \times 64$ (or $16 \times 16 \times 128$) feature maps is multiplied by the corresponding element in the $16 \times 16 \times 1$ feature map. Figure 7(a) depicts the architecture of the generator. Input 1 and Input 2 represent the condition matrix and the random noise

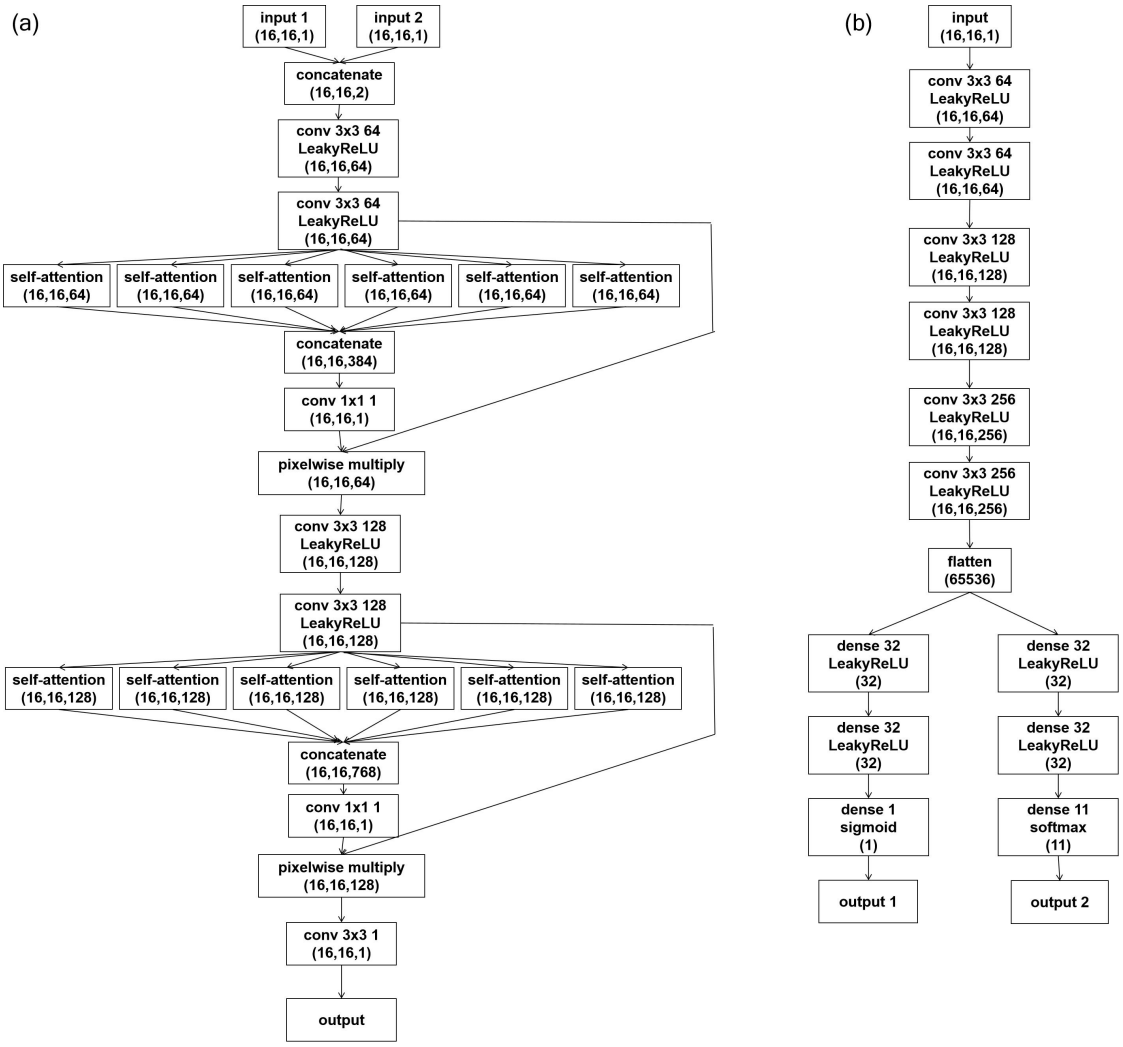

Figure 7: The detailed architecture of (a) the generator and (b) the discriminator of the deep learning model used.

matrix, respectively, with the output being the generated spin configuration. Figure 7(b) illustrates the discriminator's architecture. The input is the spin configurations obtained either by Monte Carlo simulation or from the generator. Output 1 determines whether the input spin configuration originates from Monte Carlo data or generative data, while Output 2 specifies the condition of the input spin configuration.

B Comparing the performance between AC-GAN and Implicit-GAN

To compare the performance of AC-GAN and Implicit-GAN quantitatively, we computed the percentage difference between the GAN-generated results and the MCMC results in the XY model. Figure 8 illustrates the results for measurements on various physical quantities. While both GANs perform similarly in magnetization measurements (Fig. 8(b)), the AC-GAN surpasses the Implicit-GAN in energy and vorticity measurements. Specifically, in the energy measurement, the Implicit-GAN yields a percentage difference that is about twice that of AC-GAN in the high-temperature regime (Fig. 8(a)). Additionally, in the calculation of vorticity,

the Implicit-GAN performs poorly in the low-temperature regime, where the percentage difference is generally higher, and can be up to 10 times higher compared to the case of using AC-GAN (Fig. 8(c)). This provides strong evidence that the AC-GAN with the self-attention layer outperforms the Implicit-GAN.

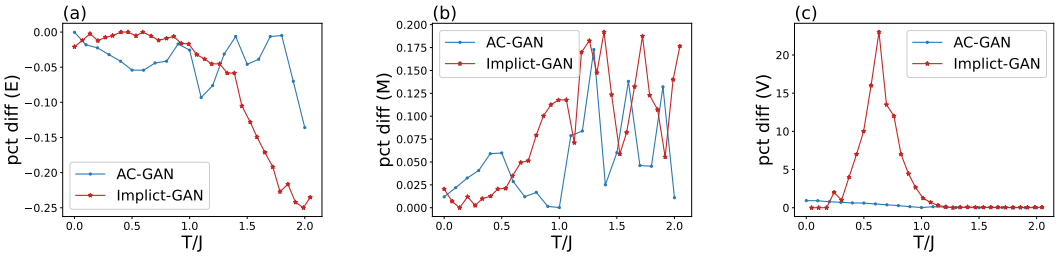

Figure 8: The percentage difference in (a) the mean energy, (b) the mean magnetization, and (c) the mean vorticity between the GANs and MCMC results for the XY model with a system size of $L = 16$. The AC-GAN generally yields significantly smaller percentage differences in the energy measurement at high temperatures and vorticity measurement at low temperatures.

C The distribution difference in the AC-GAN and MCMC samples

In Figure 3 of the main text, we compare the physical quantities obtained from AC-GAN samples and from MCMC simulations. The calculated quantities agree well in the entire simulated temperature range. This indicates that the AC-GAN has effectively learned the Monte Carlo (MC) sampling distribution. As supplementary analysis, we examined the distribution differences in the energy of the XY model for $L = 16$ and $L = 32$ sampled via MC and AC-GANs, quantified by the Earth Mover's Distance [44] and Jensen-Shannon divergence [45, 46].

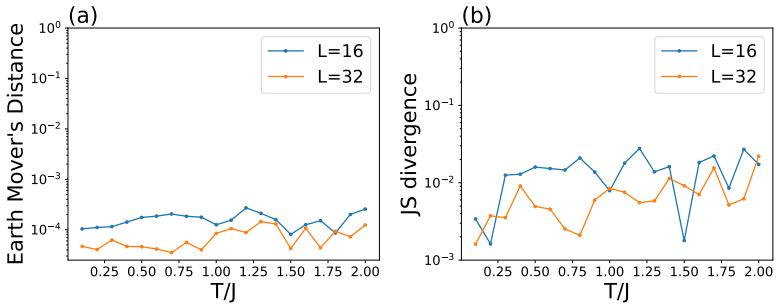

Figure 9: (a) Earth Mover's Distance and (b) Jensen-Shannon divergence measuring the distribution differences in the energy sampled by AC-GANs and Monte Carlo methods for the XY model.

The Earth Mover's Distance, also known as the Wasserstein distance, quantifies the minimum amount of work required to transform one distribution into another. It measures the distance between distributions as the cost of transporting mass in one distribution to match the other. On the other hand, the Jensen-Shannon divergence is based on the Kullback-Leibler divergence, providing a symmetric and smoothed measure of the difference between two probability distributions. It calculates the divergence between each distribution and the average

of both, offering a bounded score (typically between 0 and 1) that indicates how similar or different the distributions are. Both measures yield smaller values for distributions that are more similar to each other.

Figure 9 shows the Earth Mover's Distance and JS divergence between the AC-GAN and MCMC samples as a function of temperature. It is evident that, despite fluctuations in the two measures, the dissimilarity between the two distributions remains consistently low across the entire temperature range. This further substantiates the validity of the machine learning model used.