PTEP

Prog. Theor. Exp. Phys. **2023** 013A03(15 pages)
DOI: 10.1093/ptep/ptac173

A universal training scheme and the resulting universality for machine learning phases

Yuan-Heng Tseng[1], Fu-Jiun Jiang[1], and C.-Y. Huang[2]

[1]*Department of Physics, National Taiwan Normal University, 88, Sec. 4, Ting-Chou Road, Taipei 116, Taiwan. E-mail: fjjiang@ntnu.edu.tw*
[2]*Department of Applied Physics,Tunghai University, No. 1727, Sec. 4, Taiwan Boulevard, Xitun District, Taichung 40704, Taiwan*

Received October 19, 2022; Revised December 8, 2022; Accepted December 14, 2022; Published December 16, 2022

An autoencoder (AE) and a generative adversarial network (GAN) are trained only once on a one-dimensional (1D) lattice of 200 sites. Moreover, the AE contains only one hidden layer consisting of two neurons, and both the generator and the discriminator of the GAN are made up of two neurons as well. The training set employed to train both the considered unsupervised neural networks (NNs) is composed of two artificial configurations. Remarkably, despite their simple architectures, both the built AE and GAN have precisely determined the critical points of several models, including the three-dimensional classical $O(3)$ model, the two-dimensional generalized classical XY model, the two-dimensional two-state Potts model, and the one-dimensional Bose–Hubbard model. In addition, a factor of several thousands in the speed of calculation is gained for the built AE and GAN when they are compared with the conventional unsupervised NN approaches. The results presented here, as well as those shown previously in the literature, suggest that when phase transitions are considered, an elegant universal neural network that is extremely efficient and is applicable to broad physical systems can be constructed with ease. In particular, since an NN trained with two configurations can be applied to many models, it is likely that when machine learning is concerned, the majority of phase transitions belong to a class having two elements, i.e. the Ising class.

Subject Index A22, A24, A40, A42, A44

1. Introduction

Machine learning (ML) techniques have attracted considerable attention recently in various fields of physics, including astronomy [1–6], particle physics [7–15], computational materials [16–22], and condensed matter physics [23–51]. While these applications are still in the exploratory stage, great improvements over traditional approaches may be expected in the near future. Among the commonly used ML methods, neural networks (NNs), both supervised and unsupervised, have been adopted to distinguish various states of matter. NNs are also considered for reproducing certain classical statistical distributions. Due to the many successful achievements of utilizing ML and NN methods for physical problems, there is optimism that promising breakthroughs in certain research fields of physics may be just around the corner.

One of the applications of NNs in physical problems is to detect different phases of matters. Indeed, for many systems and models, both supervised (e.g. [25,26,28,29,31,33–35,37,38,46]) and unsupervised (e.g. [32,36,41,45,49]) NNs have been used successfully to explore the associated phase diagrams to certain extents. Between the supervised and the unsupervised NNs,

unsupervised ones are typically favored. This is because prior information is needed before one can apply the techniques of supervised NNs. As a result, the majority of NN studies for physical systems usually use unsupervised approaches.

Typically, the process of applying an NN to study the properties of many-body systems has three stages, namely the training, the validation, and the testing (prediction) stages. Among these three stages, training is the most time-consuming one. In particular, with the conventional training strategy usually employed in the literature, new training is required whenever a new model or a different system size is considered. It should be pointed out that the configurations employed for training are based on physical quantities such as the spins or the correlation functions. As a result, a certain amount of computing time is needed to generate the relevant physical data in order to construct the required configurations for the NN training. These technical difficulties usually prevent the NNs from being used in real problems.

We would like to emphasize the fact that in the literature the NNs used to explore physical problems usually have very complicated architectures, such as the deep learning convolutional neural networks (CNNs). Such a strategy of building the NNs would lead to a lot of trainable (tunable) NN parameters. As a result, a huge amount of computing time is required to obtain a working NN that has a dedicated architecture.

To overcome the aforementioned technical difficulties, some approaches, including using artificially constructed configurations as the training set, are considered for supervised NNs [50]. Certain successes have been achieved with these unconventional supervised NNs. As a result, it would be interesting to examine whether these unconventional ideas can be adopted to train unsupervised NNs. In addition, it would also be intriguing to see if one can construct a working unsupervised NN with simple infrastructure rather than complicated ones. Motivated by these ideas, here we train an autoencoder (AE) and a generative adversarial network (GAN) using two artificially made configurations on a one-dimensional (1D) lattice of 200 sites. In particular, besides the input and the output layers, the AE built here contains only one hidden layer consisting of two neurons. Moreover, each of the generator and the discriminator of the constructed GAN has only two neurons as well.

Remarkably, despite their simple architectures, the AE and the GAN obtained here successfully estimate the critical points of various models, including the three-dimensional (3D) classical $O(3)$ model, the two-dimensional (2D) 2-state Potts model, the 2D classical generalized XY model, and the one-dimensional (1D) Bose–Hubbard model. In particular, a factor of few thousands in the speed of calculation is gained for the built AE and GAN compared with the conventional unsupervised NN approaches. We would like to emphasize the fact that both of these two unsupervised NNs have only been trained once, and no new training is conducted whenever new models or different system sizes are considered. It is amazing that two NNs, each of which in principle has merely two neurons and is trained only once on a 1D lattice, can be adopted to calculate the critical points of many 3D, 2D, and 1D models with high precision. In particular, no information of these studied models is needed for these two built unsupervised NNs to detect the associated phase transitions.

We would like to point out that with minor modifications for the testing sets, it is likely that the NNs obtained here can be directly applied to study other models such as the Fermi–Hubbard model as well as the Su–Schrieffer–Heeger (SSH) model.

Finally, it is worth mentioning that the results shown here, in conjunction with some established outcomes in the literature, indicate that an NN trained with only two (artificial) con-

figurations can be applied to many physical models that vary significantly from each other. This implies that when machine learning is concerned, it is likely that the majority of phase transitions belong to a class having two elements, i.e. the Ising class.

The rest of the paper is organized as follows. After the introduction, the studied models are briefly described in Sect. 2. In Sect. 3 the employed unsupervised AE and GAN, the training strategy, and the built configurations for NN predictions are outlined. Benchmark calculations are demonstrated in Sect. 4 and the obtained NN outcomes are presented in Sect. 5. Section 6 contains our discussions and conclusions.

2. The considered models

The Hamiltonians of the models considered here have the following expressions

(1) 3D classical $O(3)$ model [52,53]:

$$\beta H_{O(3)} = -\beta \sum_{<ij>} \vec{s}_i \cdot \vec{s}_j, \tag{1}$$

where β is the inverse temperature and $<ij>$ stands for the nearest-neighbor sites i and j. In addition, $\vec{s}_i$ appearing above is a unit vector belonging to a 3D sphere S^3 and is located at site i.

(2) 2D classical generalized XY model [54,55]:

$$H_{\mathrm{GXY}} = \sum_{<ij>} -\Delta \cos\left(\theta_i - \theta_j\right) - (1-\Delta)\cos\left(q\theta_i - q\theta_j\right), \tag{2}$$

where the summation is over the nearest neighbors i and j, q is some (positive) integer, and $0 \leq \theta_i \leq 2\pi$. We will consider the case of $\Delta = 0.25$ and $q = 3$ in this study.

(3) 2D 2-state Potts model [56]:

$$\beta H_{\mathrm{Potts}} = -\beta \sum_{<ij>} \delta_{\sigma_i,\sigma_j}. \tag{3}$$

Here δ refers to the Kronecker function and the Potts variable σ_i at each site i takes an integer value from $\{1, 2\}$.

(4) 1D Bose–Hubbard model: using the creation and annihilation operators $\hat{a}_i^\dagger$ and $\hat{a}_i$, the Hamiltonian of the 1D Bose–Hubbard model is given by [57,58]

$$H_{\mathrm{BH}} = -t \sum_{i=1}^{L} \left(\hat{a}_i^\dagger \hat{a}_{i+1} + \hat{a}_{i+1}^\dagger \hat{a}_i\right) + \frac{U}{2} \sum_{i=1}^{L} \hat{n}_i \left(\hat{n}_i - 1\right) - \mu \sum_{i=1}^{L} \hat{n}_i, \tag{4}$$

where L is the number of sites, t is the tunneling strength, $U > 0$ is the on-site repulsive interaction strength, μ is the chemical potential, and $\hat{n}_i = \hat{a}_i^\dagger \hat{a}_i$ is the particle number operator at site i.

3. The constructed unsupervised NNs

Typically, an NN has many tunable parameters. In this investigation we use the default values for these parameters unless specified. Apart from this, the two considered unsupervised NNs are trained with double precision by default as well.

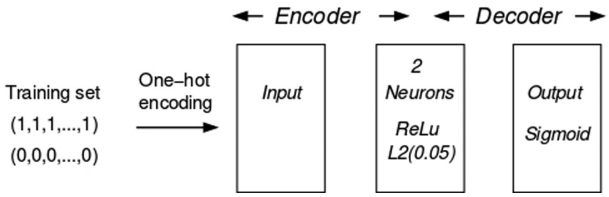

Fig. 1. The AE employed in this study. This AE consists of one input layer, one hidden layer of two neurons, and one output layer. The considered activation functions (and regularizer) are shown inside the boxes.

3.1. *Autoencoder*

Using keras and tensorflow [59], the autoencoder employed here consists of one input layer, one hidden layer having two neurons, and one output layer. In particular, the hidden and the output layers are activated by the ReLU and the sigmoid functions, respectively. The definitions of these two activation functions are as follows

$$\mathrm{ReLU}(x) = \max(0, x), \tag{5}$$

$$\mathrm{sigmoid}(x) = \frac{1}{1 + e^{-x}}. \tag{6}$$

The algorithm used for the training is minibatch. One-hot encoding and L_2 regularizations are also applied. We use 1000 epochs and the batch size is set to 30. Finally, the used loss function and optimizer are the crossentropy C and the adam, respectively. Here the definition of crossentropy C is given by

$$C = -\frac{1}{n}\sum_x y_x \ln b_x + (1 - y_x)\ln(1 - b_x), \tag{7}$$

where n is the total number of objects in the training set, and b are the outcomes obtained after applying all the constructed layers. In addition, x and y are the training inputs and the corresponding designed outputs, respectively. Figure 1 is a cartoon representation of the built AE.

For the training of the constructed AE, 200 copies of two artificially made configurations on a 1D lattice of 200 sites are used as the training set. Specifically, every site of 200 1D configurations takes the value of 1, and each of the other 200 configurations has 0 as the values for all of its elements. Because of the employment of one-hot encoding, the associated output for (1,1,1,...,1,1,1) is a 400-component vector taking the form (1,0,1,0,1,0,...,1,0,1,0,1,0). Similarly, the output for (0,0,0,...,0,0,0) is a 400-component vector having the expression (0,1,0,1,0,1,...,0,1,0,1,0,1). As we will demonstrate shortly, such a simple training setup can lead to a valid NN for learning various phases of all the considered models. Reader who are interested in these machine-learning terminologies associated with AE (and GAN, which will be introduced later) are referred to Refs. [62,63].

The magnitude R of the output (vectors) is employed as the quantity for studying the targeted phase transitions. One expects that when R are considered as functions of temperature T (or β), the associated outcomes will reveal information relevant to T_c (or β_c). Indeed, for the configurations obtained at low temperatures, the outcomes are vary similar to those obtained with the training sets. As a result, the associated outputs R have large magnitude. As the temperature T rises, the magnitude R of the outputs will decrease sharply at the critical point(s). As we will show later, this phenomenon is exactly what we have observed.

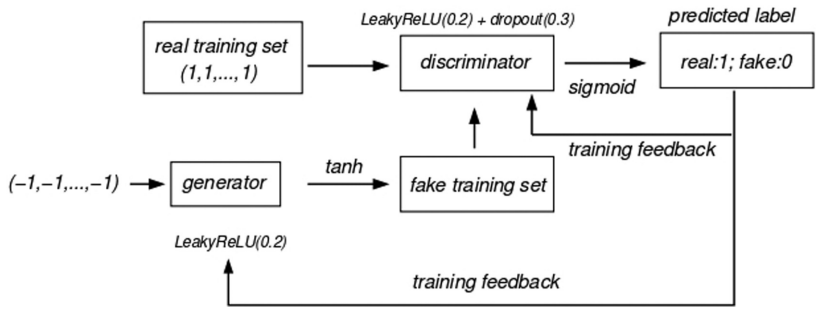

Fig. 2. The GAN employed in this study.

3.2. *Generative adversarial network*

The GAN considered here consists of a generator and a discriminator, each with two neurons. Each of the 640 configurations for the generator consists of 200 sites and every site has -1 as its element. In addition, the training set is made up of 640 identical copies of 200 sites with all the elements being 1. Adam and crossentropy are the optimizer and the loss function employed, respectively. A total of 2000 epochs are carried out and we use 128 as the batch size. The activation functions used are LeakyReLU, tanh, and sigmoid [62,63]. The new activation functions LeakyReLU and tanh are defined as

$$\text{LeakyReLU}(x) = \begin{cases} \alpha x \text{ if } x < 0 \\ x \quad \text{if } x \geq 0 \end{cases} \tag{8}$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{9}$$

where α is some constant. Finally, the algorithm used for the training is again minibatch and we also apply dropout at the discriminator. The predicted label is either 1 (true) or 0 (fake). Figure 2 is a cartoon representation of the GAN considered in this study.

Due to the data used for the fake and real training sets as well as the employed predicted label(s), the standard deviations (STD) of the outputs (scalars) will be employed to explore the targeted phase transitions. It is expected that the behavior of STD with respect to T (or β) will disclose the information relevant to T_c (or β_c). In particular, for configurations obtained at low temperatures, the related GANs outputs are either 1 or 0 with almost equal probabilities. Hence the associated outputs have large STD. As temperature T rises, the magnitude of STD will diminish dramatically at T_c.

4. The benchmark calculations

To demonstrate the efficiency of the AE constructed here, we have carried out certain benchmark calculations. In particular, the deep learning autoencoder (DLAE) built in Ref. [45] is employed here for the benchmark calculations (Fig. 3). The model considered is the two-state ferromagnetic Potts model on the square lattice. In addition, the epochs and batch size used are 2000 and 30, respectively. Finally, the benchmark calculations are conducted on a server with two opetron 6344 and 96G memory.

The time required to train the AE with one hidden layer having two neurons is about 110 seconds, while the training using 30,200 real (and full) Potts configurations obtained on 128 by 128 lattices as the training set for the DLAE shown in Fig. 3 takes about 302,114 seconds to finish (it is anticipated that it will take longer to conduct all the DLAE trainings if vari-

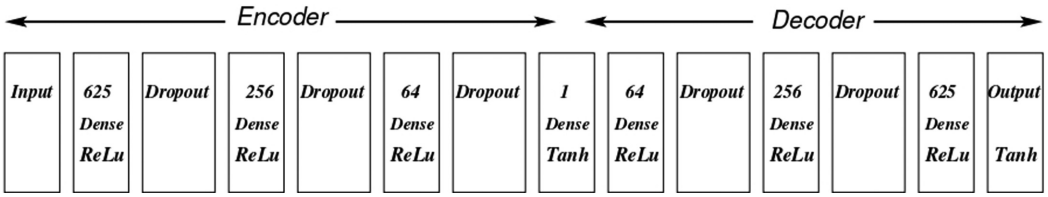

Fig. 3. The deep learning AE employed in Ref. [45]. The activation functions considered and the number of neurons used are shown inside the boxes.

ous system sizes are considered). Here the number of configurations used for the conventional training is about the same as that used in Ref. [45]. Of course, the exact numbers of the benchmark calculations may depend on some factors such as whether CPU or GUP is considered for the execution of the calculations. Nevertheless, the results provide certain useful information regarding the performance of both the deep learning AE of Ref. [45] and the shallow AE used in this study. In any case, based on the benchmark outcomes, it is beyond doubt that the unconventional training strategy adopted here is more efficient than that typically used in the literature. Moreover, as we will demonstrate shortly, the precision of the determined critical points using the unsupervised NN built here is impressive as well.

5. The numerical results

To examine the validity as well as the efficiency of the constructed AE and GAN, certain configurations of the considered models are required. For the classical spin models the required configurations are generated using Monte Carlo simulations with the Wolff cluster algorithm [60]. For the 1D Bose–Hubbard model, the worm algorithm of ALPS is considered [61].

5.1. *Constructing the required configurations for predictions*

Since the configurations used for the NN predictions should have the same dimensionality as those of the training sets, the configurations required for the NN predictions using the AE and GAN constructed here should consist of 1D lattices of 200 sites.

For the $O(3)$ model, the configurations built for the predictions are based on the spin variable ϕ (ϕ is the azimuthal angle) . Specifically, for a given $O(3)$ spin configuration obtained from the Monte Carlo simulations, 200 spins are chosen randomly and uniformly. In addition, the ϕ mod π of these 200 picked spins are used to construct the 1D configuration for the NN predictions associated with AE.

The same procedure is applied for the 2D generalized XY model. In particular, the θ mod π (θ is the angle from the positive x-axis) of 200 spins, which are randomly picked from an original full spin configuration, are used as the elements of a 1D configuration for the predictions related to AE.

For every produced full Potts configuration of the two-state Potts model, 200 Potts variables σ are chosen randomly and the associated results of $\sigma - 1$ are employed to build the 1D configuration required for the AE predictions.

Finally, for the 1D Bose–Hubbard model, the configurations used for the NN predictions are based on the local density.

A similar strategy is utilized to build the configurations required for the predictions related to GAN as well.

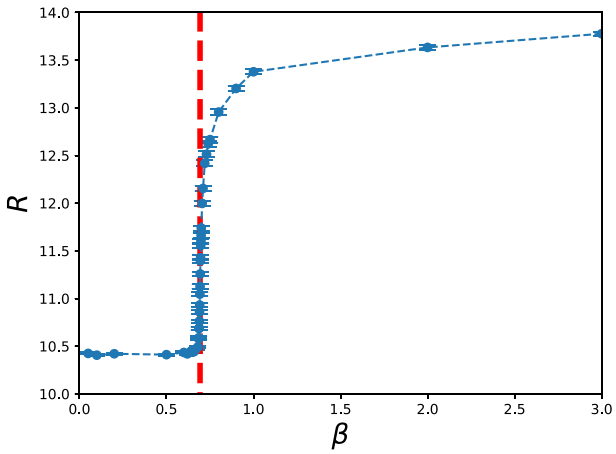

Fig. 4. The magnitude R of the AE outputs as a function of the inverse temperature β for the 3D classical $O(3)$ model. The system size is $L = 48$ and the vertical dashed line is the expected β_c. The value of R begins to rise significantly at β_c.

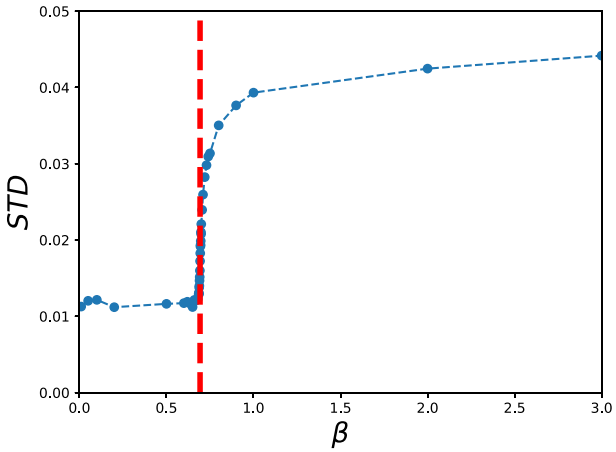

Fig. 5. The STD of the GAN outputs as a function of the inverse temperature β for the 3D classical $O(3)$ model. The system size is $L = 48$ and the vertical dashed line is the expected β_c. The value of STD begins to rise significantly at β_c.

5.2. *3D classical O(3) model*

5.2.1. *AE results.* The magnitude R of the AE outputs (in this study all the AE outputs are 400-component vectors) as a function of the inverse temperature β for the 3D classical $O(3)$ model is depicted in Fig. 4. The system size is $L = 48$ and the vertical dashed line in Fig. 4 is the expected critical inverse temperature β_c. The outcomes given in Fig. 4 imply that the value of R begins to rise significantly at β_c. In other words, how R behaves with respect to β reveals relevant information regarding the critical point. In particular, β_c can be estimated to be the location in the associated parameter space where a dramatic change in R takes place.

5.2.2. *GAN results.* The STD of the GAN outputs (the GAN outputs are numbers between 0 and 1) as a function of β for the 3D classical $O(3)$ model is shown in Fig. 5. The linear system size L for the data in the figure is $L = 48$. As expected, the figure demonstrates that the value of

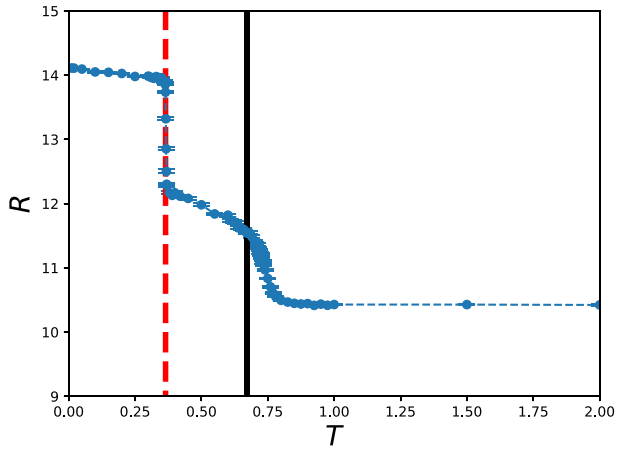

Fig. 6. The magnitude R of the AE outputs as a function of the temperature T for the 2D generalized classical XY model. The system size is $L = 128$. The vertical dashed and solid lines are the expected T_c associated with the three-state Potts and the Berezinskii–Kosterlitz–Thouless (BKT) universalities, respectively. The value of R drops markedly at $T_{c,\,\mathrm{Potts}}$ and $T_{c,\,\mathrm{BKT}}$.

STD rises dramatically at β_c. The shown outcomes indicate that the STD of the NN outputs can indeed be employed to locate the critical point β_c.

5.3. *2D generalized XY model*

5.3.1. *AE results.* The magnitude R of the AE outputs as a function of the temperature T for the 2D generalized classical XY model is shown in Fig. 6. The system size is $L = 128$. The vertical dashed and solid lines in Fig. 6 are the expected $T_{c,\,\mathrm{Potts}}$ and $T_{c,\,\mathrm{BKT}}$ associated with the three-state Potts and the Berezinskii–Kosterlitz–Thouless (BKT) universalities, respectively. As can be seen from Fig. 6, a sudden jump of R occurs at $T_{c,\,\mathrm{Potts}}$. Moreover, close to $T_{c,\,\mathrm{BKT}}$ the behavior of R shows an apparent drop as well. These observed phenomena clearly indicate that the locations of critical points in the parameter space can be estimated by the T-dependence of R. It is also interesting to notice that the drop of R at $T_{c,\,\mathrm{BKT}}$ is less sharp than that at $T_{c,\,\mathrm{Potts}}$. This observation is consistent with the fact that BKT transitions receive certain logarithmic corrections, hence only with extremely large system size will the signal of the transitions appear transparently.

5.3.2. *GAN results.* The STD of the GAN outputs as a function of T for the 2D generalized classical XY model is shown in Fig. 7. The linear system size L for the data in Fig. 7 is $L = 128$. As expected, Fig. 7 demonstrates that the value of STD diminishes dramatically at $T_{c,\,\mathrm{Potts}}$ and $T_{c,\,\mathrm{BKT}}$. We would like to emphasize that when the outcomes in Fig. 7 are compared with those in Fig. 6, the drop of STD at $T_{c,\,\mathrm{BKT}}$ is less abrupt. This indicates that when BKT transitions are concerned, AE has a better performance for detecting their existence than that of GAN.

5.4. *2D two-state Potts model*

5.4.1. *AE results.* The magnitude R of the AE outputs as a function of the temperature T for the 2D two-state Potts model is shown in Fig. 8. The system size is $L = 120$ and the vertical dashed line is the expected T_c. Clearly the outcomes demonstrated in Fig. 8 imply that R drops significantly at T_c.

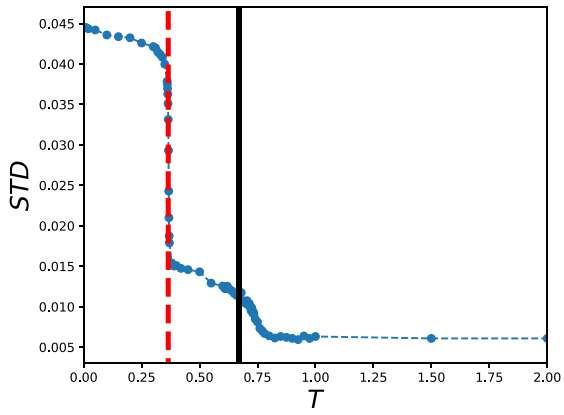

Fig. 7. The STD of the NN (GAN) outputs as a function of the temperature T for the 2D generalized classical XY model. The system size is $L = 128$. The vertical dashed and solid lines are the expected T_c associated with the three-state Potts and the BKT universalities, respectively. The value of STD drops markedly at $T_{c,\,\mathrm{Potts}}$ and $T_{c,\,\mathrm{BKT}}$.

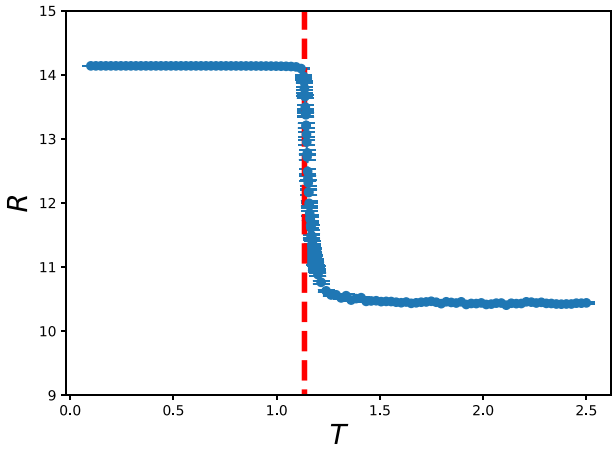

Fig. 8. The magnitude R of the AE outputs as a function of the temperature T for the 2D two-state Potts model. The system size is L=120. The vertical dashed line is the expected T_c. The value of R drops dramatically at T_c.

5.4.2. *GAN results.* The STD of the GAN outputs as a function of T for the two-state Potts model is shown in Fig. 9. The linear system size L for the data in Fig. 9 is $L = 120$ and the vertical dashed line is the expected T_c. As expected, Fig. 9 demonstrates that the value of STD drops dramatically at T_c.

5.5. *1D Bose–Hubbard model*

The associated simulations are conducted with system size $L = 256$, $t = 0.1$, $U = 1.0$, $T = 0.0025$, and various values of μ. As μ varies, one expects to see a transition from superfluidity to a Mott insulator. In addition, the maximum number of bosons per site is set to 5, and the naive configurations used for the NN prediction are built from the local densities which are generated based on the means and the mean errors of the outcomes from Monte Carlo simulations. Finally, in the process of one-hot encoding, if the local density n_i of an original site i is greater (smaller) than 0.99 (i.e. we set a density cutoff to be 0.01), then integer 1 (0) is assigned to the associated spot.

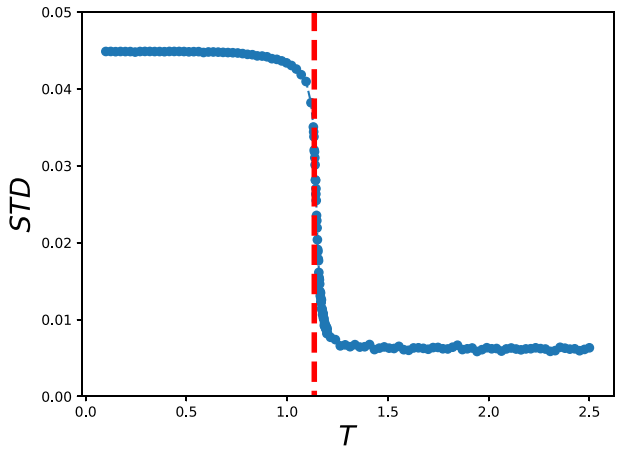

Fig. 9. The STD of the GAN outputs as a function of the temperature T for the 2D two-state Potts model. The system size is $L = 120$ and the vertical dashed line is the expected T_c. The value of STD drops dramatically at T_c.

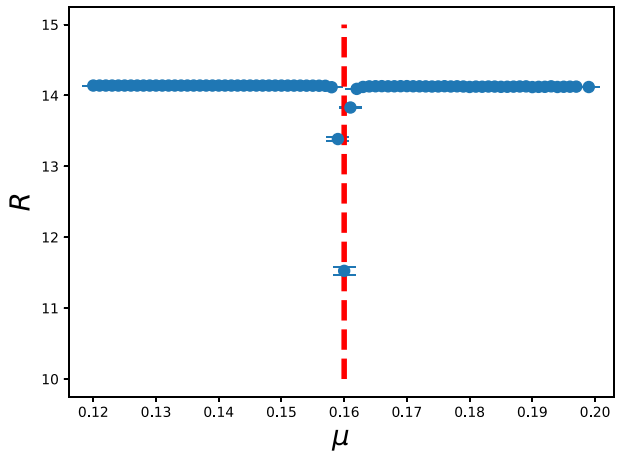

Fig. 10. The magnitude R of the AE outputs as a function of μ for the 1D Bose–Hubbard model. The vertical dashed line is the expected μ_c.

5.5.1. *AE results.* The magnitude R of the AE outputs as a function of μ is shown in Fig. 10. The dashed vertical line in Fig. 10 is the critical point μ_c estimated from Refs. [57,58]. As can be seen from Fig. 10, the μ which has the smallest value of R is very close to μ_c. In other words, the constructed AE is capable of detecting the considered phase transition of the 1D Bose–Hubbard model. Here we would like to point out that if one uses another value of density cutoff, then the μ having the minimum R clearly will shift (slightly). This will have an impact on estimating μ_c. It is obvious that such an impact will become much milder if extremely accurate data points (so that the density cutoff can be very tiny) are used for the (NN) calculations.

5.5.2. *GAN results.* For the GAN, when the detection of the phase transition (of the 1D Bose–Hubbard model) is concerned, the performance of the outputs themselves is better than that of the STD. Indeed, as can be seen from Fig. 11, the R drops significantly when one approaches μ_c (the vertical dashed line) from the left. Here again, a density cutoff of 0.01 is considered. It should be pointed out that unlike those used in the previous sections, the GAN

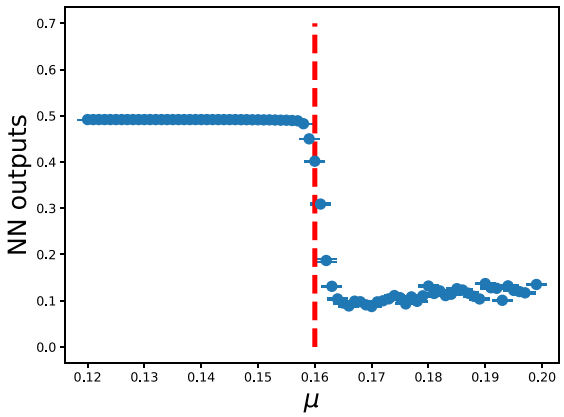

Fig. 11. The GAN outputs as a function of μ for the 1D Bose–Hubbard model. The vertical dashed line is the expected μ_c.

leading to the results presented in Fig. 11 has been trained with the additional step of one-hot encoding.

6. Discussions and conclusions

In this study, an AE with only one hidden layer of two neurons as well as a GAN with a two-neuron generator and a two-neuron discriminator are constructed. In particular, they are trained only once on a 1D lattice of 200 sites. The training set used consists of two artificial configurations. The obtained unsupervised NNs successfully determine the critical points of several models, including the 3D classical $O(3)$ model, the 2D generalized classical XY model, the 2D two-state Potts model, and the 1D Bose–Hubbard model. It is remarkable that an AE and a GAN with such simple architectures can lead to high-precision calculations of the targeted phase transitions. With moderate modifications, the applications of the AE and GAN constructed here can undoubtedly be extended to include the Fermi–Hubbard or SSH models. It is also amazing that any of the two unsupervised NNs built here can be employed to calculate the critical points of various 3D and 2D models accurately. In particular, the constructed AE and GAN are able to successfully detect both the symmetry-breaking and the topology-related phase transitions. It is also surprising that no information of these models is needed for the AE and GAN constructed here to detect the associated phase transitions.

In our calculations, a site can be chosen at most one time. We have performed an investigation so that a site can be picked more than once. The NN outcomes associated with this strategy for the 3D $O(3)$ model (AE) and the 2D two-state Potts model (GAN) are shown in Fig. 12. As can be seen from Fig. 12, this approach also leads to accurate estimations of the critical points.

The idea of sites can be picked more than once can be applied to systems which have fewer sites than 200 sites. We have conducted a calculation for the 2D two-state Potts model and the 2D generalized XY model with linear system size $L = 12$. The related NN outcomes are shown in Fig. 13. While one can expect sizable finite-size effects (as can be seen in the figure), the results shown in Fig. 13 indicate that the two unsupervised NNs constructed can be applied to systems having fewer sites than 200 sites.

When compared with conventional deep-learning AEs typically used in the literature, a benchmark investigation indicates that the AE employed here offers at least a several thousand-fold improvement in speed. An improvement in the prediction calculations is also anticipated.

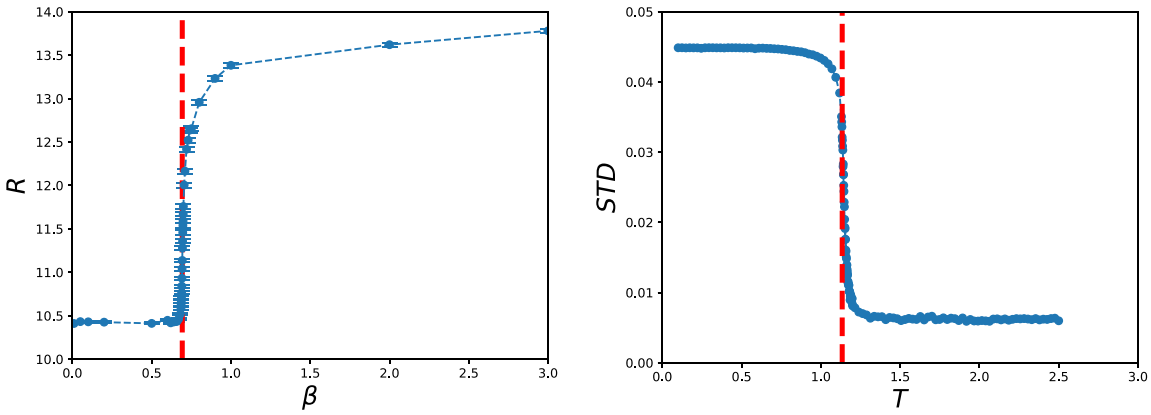

Fig. 12. (Left panel) The AE outputs as a function of β for the 3D $O(3)$ model. The vertical dashed line is the expected β_c. (Right panel) The STD of GAN outputs as a function of T for the 2D two-state Potts model. The vertical dashed line is the expected T_c. For both results, a site can be picked more than once.

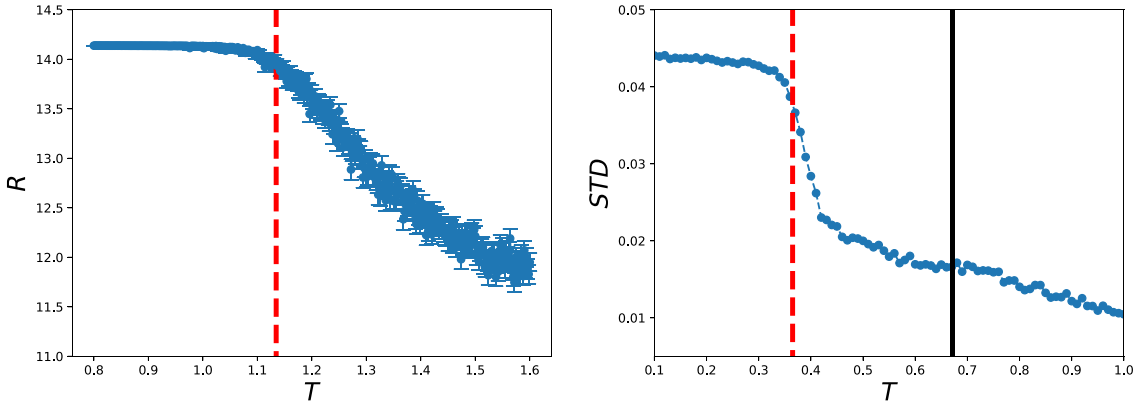

Fig. 13. (Left panel) The AE outputs as a function of T for the 2D two-state Potts model with $L = 12$. The vertical dashed line is the expected T_c. (Right panel) The STD of GAN outputs as a function of T for the 2D generalized XY model with $L = 12$. The vertical dashed and solid lines are the expected critical points.

Finally, it should be noted that the training of the GAN considered in this study is also extremely efficient. Specifically, it takes about 154 seconds to train the GAN with 2000 epochs (It takes about 388 seconds to train the GAN with the step of one-hot encoding).

It should be pointed out that in Ref. [37], a supervised NN with one hidden layer of two neurons has successfully determined the T_c of the 2D Ising model. Here we have gone much further than the results achieved in Ref. [37].

For each of the models studied here, only one system size data is considered. It is very clear, however, that the calculations required to obtain results for several box sizes can also be performed. With these new outcomes, semi-experimental finite-size scaling ansatz can be utilized to obtain an estimate of the bulk critical point(s). Here we do not perform such an investigation because relevant detailed studies are available in the literature [35,48,51].

We would also like to emphasize the fact that the conventional training methods usually used in the literature are computationally demanding and require huge amounts of computer mem-

ory. Hence, most of the calculations are limited to small to moderately large system sizes.The AE and GAN employed here have no such system size restrictions.

In addition to the advantage of extremely efficient training, the method adopted here requires much less storage capacity compared with the standard NN approaches used in the literature. In practice, one only needs to save 200 spins for every generated large system size configuration which usually involve several thousand to over 10,000 spins. This feature is similar to the supervised NN employed in Refs. [50,51]. To demonstrate the high efficiency of the unsupervised NNs employed, we have recorded the total time needed to complete the NN calculations associated with the 3D $O(3)$ model. The 3D $O(3)$ model is chosen because it takes the longest time to perform the related NN investigation for this model by the conventional approaches. For the AE considered in this study, from the start of the training to the end of the prediction, the time needed for these procedures is about 200 seconds (this also includes the time required for reading (loading) the data files). In other words, it takes less than 4 minutes to produce the results shown in Fig. 4. Here, for each of the 49 temperatures, 2000 configurations are used for the NN prediction. In reality, one can only keep 200 relevant quantities (such as the spins) for every generated configuration and use these data to perform the NN prediction with the AE and GAN constructed here. As a result, it is anticipated that if the same amount of data are employed, then the required time to complete the entire NN calculation for each of the other studied models is about the same as that for the 3D $O(3)$ model.

Based on the results obtained here, it is likely that the training schemes introduced here are the most efficient strategy for training unsupervised NNs when studying phase transitions are concerned. In addition, these outcomes show that a NN trained once with two artificial configurations can be applied to many models that are dramatically different from each other. This suggests strongly that when machine learning is considered, many, or even the majority of phase transitions belong to a class having two elements, namely the Ising class.

Finally, it will be interesting to examine whether by simply engineering the configurations for prediction one can apply the unsupervised NN constructed here to study the criticalities of some other models that are not investigated in this study. It will also be interesting to see if similar elegant ideas exist for other research fields of physics.

Acknowledgement

Partial support from the Ministry of Science and Technology of Taiwan (MOST) is acknowledged (MOST 110-2112-M-003-015 and MOST 108-2112-M-029-006-MY3). A preprint version has appeared on arXiv (arXiv:2205.03061).

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

F.-J.J. and C.-Y.H. proposed and supervised the project. F.-J.J. wrote the initial manuscript. Y.-H.T. conducted the calculations and analyzed the data. All authors contributed to the final version of the manuscript.

Data Availability

Data will be made available on request.

REFERENCES

[1] B. Hoyle, Astron. Comput. **16**, 34 (2016).
[2] R. Morgan and[DES], Astrophys. J. **901**, 83 (2020).
[3] M. Cabero, A. Mahabal, and J. McIver, Astrophys. J. Lett. **904**, L9 (2020)
[4] S. Schuldt et al., Astron. Astrophys. **651**, A55 (2021).
[5] C. Li et al., Mon. Notices Royal Astron. Soc., **509**, 2289 (2022).
[6] C. Jacobs, K. Glazebrook, A. K. Qin, and T. Collett, Astron. Comput. **38**,100535 (2022).
[7] P. Baldi, P. Sadowski, and D. Whiteson, Phys. Rev. Lett. **114**, 111801 (2015).
[8] P. Baldi, K. Bauer, C. Eng, P. Sadowski, and D. Whiteson, Phys. Rev. D **93**, 094034 (2016).
[9] A. Mott, J. Job, J. R. Vlimant, D. Lidar, and M. Spiropulu, Nature **550**, 375 (2017).
[10] L. G. Pang, K. Zhou, N. Su, H. Petersen, H. Stcker, and X. N. Wang, Nature Commun. **9**, 210 (2018).
[11] P. E. Shanahan, D. Trewartha, and W. Detmold, Phys. Rev. D **97**, 094506 (2018).
[12] A. J. Larkoski, I. Moult, and B. Nachman, Phys. Rep. **841**, 1 (2020).
[13] X. Han and S. A. Hartnoll, Phys. Rev. X **10**, 011069 (2020).
[14] G. Aad[ATLAS] et al., Phys. Rev. Lett. **125**, 131801 (2020).
[15] K. A. Nicoli, C. J. Anders, L. Funcke, T. Hartung, K. Jansen, P. Kessel, S. Nakajima, and P. Stornati, Phys. Rev. Lett. **126**, 032001 (2021).
[16] J. C. Snyder, M. Rupp, K. Hansen, K.-R. Müller, and K. Burke, Phys. Rev. Lett. **108** 253002 (2012).
[17] Z. Li, J. R. Kermode, and A. De Vita, Phys. Rev. Lett. **114**, 096405 (2015).
[18] B. Kolb, L. C. Lentz, and A. M. Kolpak, Sci. Rep. **7**, 1192 (2017).
[19] G. Carleo and M. Troyer, Science **355**, 602 (2017).
[20] S. Lu, Q. Zhou, Y. Ouyang et al., Nat. Commun. **9**, 3405 (2018).
[21] K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, and A. Walsh, Nature **559**, 547 (2018).
[22] A. P. Bartok, J. Kermode, N. Bernstein, and G. Csaǹyi, Phys. Rev. X **8**, 041048 (2018).
[23] G. Torlai and R. G. Melko, Phys. Rev. B **94**, 165134 (2016).
[24] L. Wang, Phys. Rev. B **94**, 195105 (2016).
[25] P. Broecker, J. Carrasquilla, R. G. Melko, and S. Trebst, Sci. Rep. **7**, 8823 (2017).
[26] J. Carrasquilla and R. G. Melko, Nat. Phys. **13**, 431 (2017).
[27] J. Tubiana and R. Monasson, Phys. Rev. Lett. **118**, 138301 (2017).
[28] E. P. L. van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nat. Phys. **13**, 435 (2017).
[29] Y. Nagai, H. Shen, Y. Qi, J. Liu, and L. Fu, Phys. Rev. B **96** 161102 (2017).
[30] D.-L. Deng, X. Li, and S. Das Sarma, Phys. Rev. B **96** 195145 (2017).
[31] Y. Zhang, R. G. Melko, and E.-A. Kim, Phys. Rev. B **96**, 245119 (2017).
[32] W. Hu, R. R. P. Singh, and R. T. Scalettar, Phys. Rev. E **95**, 062122 (2017).
[33] Y. Zhang and E.-A. Kim, Phys. Rev. Lett. **118**, 216401 (2017).
[34] A. Tanaka and A. Tomiya, J. Phys. Soc. Jpn. **86**, 063001 (2017).
[35] M. J. S. Beach, A. Golubeva, and R. G. Melko, Phys. Rev. B **97**, 045207 (2018).
[36] K. Ch'ng, N. Vazquez, and E. Khatami, Phys. Rev. E **97**, 013306 (2018).
[37] D. Kim and D.-H. Kim, Phys. Rev. E **98**, 022138 (2018).
[38] C.-D. Li, D.-R. Tan, and F.-J. Jiang, Ann. Phys. **391**, 312 (2018).
[39] P. Mehta, M. Bukov, C.-H. Wang, A. G. R. Day, C. Richardson, C. K. Fisher, and D. J. Schwab, Phys. Rep. **810**, 1 (2019).
[40] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Rev. Mod. Phys. **91**, 045002 (2019).
[41] W. Zhang, J. Liu, and T.-C. Wei, Phys. Rev. E **99**, 032142 (2019).
[42] J. Greitemann, K. Liu, and L. Pollet, Phys. Rev. B **99**, 060404(R) (2019).
[43] X.-Y. Dong, F. Pollmann, and X.-F. Zhang, Phys. Rev. B **99**, 121104(R) (2019).
[44] K. Kashiwa, Y. Kikuchi, and A. Tomiya, Prog. Theor. Exp. Phys. **2019**, 083A04 (2019).
[45] C. Alexandrou, A. Athenodorou, C. Chrysostomou, and S. Paul, Eur. Phys. J. B **93**, 226 (2020).
[46] T. Ohtsuki and T. Mano, J. Phys. Soc. Jpn. **89**, 022001 (2020).
[47] D.-R. Tan et al. New J. Phys. **22**, 063016 (2020).
[48] D.-R. Tan and F.-J. Jiang, Phys. Rev. B **102**, 224434 (2020).
[49] A. Lidiak and Z. Gong, Phys. Rev. Lett. **125**, 225701 (2020).
[50] D.-R. Tan, J.-H. Peng, Y.-H. Tseng, and F.-J. Jiang, Eur. Phys. J. Plus **136**, 1116 (2021).

[51] Y.-H. Tseng and F.-J. Jiang, Results Phys., **33** 105134 (2022).
[52] C. Holm and W. Janke, Phys. Lett. A **173**, 8 (1993).
[53] M. Campostrini, M. Hasenbusch, A. Pelissetto, P. Rossi, and E. Vicari, Phys. Rev. B **65**, 144520 (2002).
[54] G. A. Canova, Y. Levin, and H. J. Arenzon, Phys. Rev. E **89**, 012126 (2014).
[55] G. A. Canova, Y. Levin, and H. J. Arenzon, Phys. Rev. E **94**, 032140 (2016).
[56] F. Y. Wu, Rev. Mod. Phys. **54**, 235 (1982).
[57] S. Ejima, H. Fehske, and F. Gebhard, Europhys. Lett. **93**, 30002 (2011).
[58] S. Ejima, H. Fehske, F. Gebhard, K. zu Mnster, M. Knap, E. Arrigoni, and W. von der Linden, Phys. Rev. A **85**, 053644 (2012).
[59] https://keras.io; https://www.tensorflow.org.
[60] U. Wolff, Phys. Rev. Lett. **62**, 361 (1989).
[61] B. Bauer et al., J. Stat. Mech. P05001 (2011).
[62] S. Raschka and V. Mirjalili, Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2, 3rd edition, Packt, Birmingham, UK (2019).
[63] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd edition, O'Reilly Media, Sebastopol, CA (2019).