

Graph neural networks unveil universal dynamics in directed percolation

Ji-Hui Han(韩继辉), Cheng-Yi Zhang(张程义), Gao-Gao Dong(董高高), Yue-Feng Shi(石月凤), Long-Feng Zhao(赵龙峰), and Yi-Jiang Zou(邹以江)

Citation: Chin. Phys. B, 2025, 34 (**8**): 080702. DOI: 10.1088/1674-1056/add505
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Analysis and design of multivalued many-to-one associative memory driven by external inputs

Qiang Fang(方强) and Hao Zhang(张浩)
Chin. Phys. B, 2025, 34 (**8**): 080701. DOI: 10.1088/1674-1056/add24b

Deep learning-enabled inverse design of polarization-selective structural color based on coding metasurface

Haolin Yang(杨昊霖), Bo Ni(倪波), Junhong Guo(郭俊宏), Hua Zhou(周华), and Jianhua Chang(常建华)
Chin. Phys. B, 2025, 34 (**5**): 050702. DOI: 10.1088/1674-1056/adb94a

Physical information-enhanced graph neural network for predicting phase separation

Yaqiang Zhang(张亚强), Xuwen Wang(王煦文), Yanan Wang(王雅楠), and Wen Zheng(郑文)
Chin. Phys. B, 2024, 33 (**7**): 070702. DOI: 10.1088/1674-1056/ad4328

Dynamic analysis of a novel multilink-spring mechanism for vibration isolation and energy harvesting

Jia-Heng Xie(谢佳衡), Tao Yang(杨涛), and Jie Tang(唐介)
Chin. Phys. B, 2024, 33 (**5**): 050706. DOI: 10.1088/1674-1056/ad21f6

Ultrafast electron microscopy in material science

Huaixin Yang(杨槐馨), Shuaishuai Sun(孙帅帅), Ming Zhang(张明), Zhongwen Li(李中文), Zian Li(李子安), Peng Xu(徐鹏), Huanfang Tian(田焕芳), Jianqi Li(李建奇)
Chin. Phys. B, 2018, 27 (**7**): 070703. DOI: 10.1088/1674-1056/27/7/070703

Graph neural networks unveil universal dynamics in directed percolation

Ji-Hui Han(韩继辉)[1], Cheng-Yi Zhang(张程义)[1], Gao-Gao Dong(董高高)[2,†], Yue-Feng Shi(石月凤)[3], Long-Feng Zhao(赵龙峰)[4], and Yi-Jiang Zou(邹以江)[5]

[1] *School of Computer Science and Technology, Zhengzhou University of Light Industry, Zhengzhou* 450000, *China*
[2] *School of Mathematical Sciences, Jiangsu University, Zhenjiang* 212013, *China*
[3] *Information Management Center, Zhengzhou University of Light Industry, Zhengzhou* 450000, *China*
[4] *School of Management, Northwestern Polytechnical University, Xi'an* 710129, *China*
[5] *School of Economics, Anyang Normal University, Anyang* 455000, *China*

(Received 21 March 2025; revised manuscript received 21 April 2025; accepted manuscript online 7 May 2025)

Recent advances in statistical physics highlight the significant potential of machine learning for phase transition recognition. This study introduces a deep learning framework based on graph neural network to investigate non-equilibrium phase transitions, specifically focusing on the directed percolation process. By converting lattices with varying dimensions and connectivity schemes into graph structures and embedding the temporal evolution of the percolation process into node features, our approach enables unified analysis across diverse systems. The framework utilizes a multi-layer graph attention mechanism combined with global pooling to autonomously extract critical features from local dynamics to global phase transition signatures. The model successfully predicts percolation thresholds without relying on lattice geometry, demonstrating its robustness and versatility. Our approach not only offers new insights into phase transition studies but also provides a powerful tool for analyzing complex dynamical systems across various domains.

Keywords: graph neural networks, non-equilibrium phase transition, directed percolation model, nonlinear dynamics

PACS: 07.05.Mh, 64.60.Ht, 64.60.A–, 05.45.–a

DOI: 10.1088/1674-1056/add505 **CSTR:** 32038.14.CPB.add505

1. Introduction

Machine learning (ML)[1] has rapidly advanced within the field of physics[2] and is widely applied in various areas such as phase transition recognition,[3–6] particle collision data analysis in high energy physics experiments,[7] astronomical observation data analysis,[8] and quantum phase transitions in condensed matter physics.[9,10] In the realm of statistical physics, ML techniques have successfully been combined with Monte Carlo simulations[11] to identify phase transitions in equilibrium systems.[12,13] Since then, many researchers have proposed supervised and unsupervised methods,[14] advancing the study of physical phase transitions. For example, Carrasquilla,[15] Tanaka,[16] VanderPlas,[8] and Horani[17] employed supervised learning techniques to determine the phase transition points of material systems by classifying model configurations at different temperatures and phases; Wang *et al.*[18] used unsupervised learning to perform principal component analysis on Ising model[19] spin configurations, successfully identifying the order parameters for phase transitions; Van *et al.*[20] proposed a confusion scheme to identify phase transition points by obtaining a universal "W"-shaped performance curve on the Kitaev chain model and two-dimensional classical Ising model datasets, without prior knowledge of phase classifications.

Although significant progress has been made in the study of equilibrium phase transitions, research on nonequilibrium phase transitions[21,22] is still in its early stages, especially for models with complex dynamic processes such as directed percolation (DP).[23] Unlike equilibrium phase transitions, nonequilibrium phase transitions involve systems far from thermodynamic equilibrium, accompanied by time evolution and complex dynamics,[24,25] such as fluid turbulence,[26] chemical reaction oscillations,[27] and epidemic spread. Machine learning for equilibrium systems often relies on static configurations (*e.g.*, Ising spin snapshots), whereas nonequilibrium systems require explicit modeling of time-dependent dynamics and absorbing states.

Current approaches to studying nonequilibrium transitions are mostly limited to specific models and lack generality. For instance, while the DP model is widely used to study nonequilibrium phase transitions, it still lacks a fully established analytical solution, particularly in the (1+1)-dimensional case.[28] Monte Carlo simulations perform well in low-dimensional systems but struggle with larger-scale systems due to the additional time dimension.[29] This presents a significant challenge in analyzing complex, large-scale

†Corresponding author. E-mail: gago999@126.com
© 2025 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

nonequilibrium systems. Recent studies have further extended graph neural networks (GNNs)[30] to analyze quantum many-body systems,[31] turbulence modeling,[32] and critical phenomena in complex networks.[33] It is shown that GNN has an excellent ability to capture spatiotemporal correlations in non-equilibrium systems.

In this study, we propose a universal machine learning framework that applies GNNs to the DP model. By transforming DP lattice configurations of varying dimensions into graph structures, we embed the temporal evolution of the percolation process directly into the node features. This allows for the unified processing of diverse system topologies without dependence on lattice geometry. Our approach not only improves the prediction of percolation thresholds but also offers a new, versatile tool for studying nonequilibrium systems. Furthermore, by combining this framework with a confusion scheme, we demonstrate the ability to identify phase transition points without manual labeling, marking a significant step forward in the application of ML to nonequilibrium dynamics.

2. Method

2.1. DP model

The DP model is a widely studied framework for non-equilibrium phase transitions, particularly for understanding phenomena such as epidemic spread and chemical reaction diffusion. In this model, each node represents an individual, and edges represent possible infection paths. Infected individuals spread the disease to their neighbors through these paths. The DP model can be viewed as a random spreading process in both spatial and temporal dimensions, with key features such as the propagation of the "infection" state and the recovery of the "self-healing" state. The core parameter of the model is the spreading probability p . When p exceeds a critical value p_c, the infection in the system will continue to spread; when p is smaller than p_c, the infection will gradually decay and eventually stop. The DP model can be defined as the following dynamic process: the model evolves from two possible initial states, one being a fully occupied state and the other starting with a single active seed. Then, the system evolves according to the following dynamic rules ((1+1)-dimensional case):

$$s_i(t+1) = \begin{cases} 1 & \text{if } s_{i-1}(t) = 1 \text{ and } z_i^- < p, \\ 1 & \text{if } s_{i+1}(t) = 1 \text{ and } z_i^+ < p, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $s_i(t+1)$ represents the state of node i at time $t+1$, and $z_i^{\pm} \in (0,1)$ are two random numbers. When the spreading probability p exceeds a critical value p_c, the active particles in the system will continue to expand. Conversely, when p is smaller than p_c, the active particles gradually disappear, eventually reaching an absorbing phase. Figure 1 provides an intuitive illustration of the percolation process of $(1+1)$-dimensional DP under two different initial states. From left to right, the node infection probabilities are 0.55, 0.6447, and 0.8, respectively.

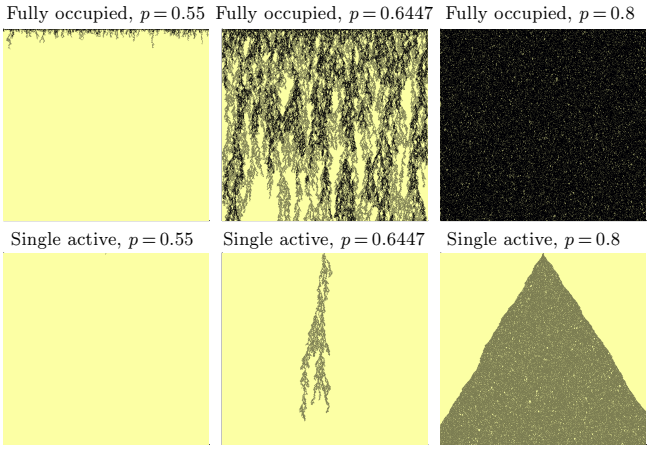


Fig. 1. The simulation process of $(1+1)$-dimensional bond-DP is shown for a fully occupied lattice (top) and a single active particle (bottom), with system size $L = 500$. The bond probabilities p are set to 0.55, 0.6447, and 0.8, respectively.

2.2. DP-GNN framework

Our primary idea is to leverage the representational power of graph structures to model lattices with varying dimensions and topologies as graph-based representations. Figure 2 illustrates the workflow of the DP-GNN framework. Initially, diverse DP configurations are generated using Monte Carlo simulations and transformed into graph structures. In this representation, nodes correspond to lattice positions, while edges signify the connections between neighboring nodes. Temporal evolution is embedded as node features to capture the spatiotemporal dependencies of the system. By applying multiple graph convolutional operations, both local and global structural characteristics of the system are effectively captured, enabling accurate phase classification of various DP configurations. Using supervised learning, an "X"-shaped curve is obtained, while unsupervised learning results in a "W"-shaped curve.

2.2.1. Data input

Before training, we generate different DP configurations and represent them as graph structures. The graph representation preserves the original lattice topology by encoding each bond as a bidirectional edge. For higher-dimensional lattices, edges connect all nearest neighbors, ensuring no loss of spatial connectivity. In this process (Fig. 3), lattice sites are represented as nodes v_i, and the bonds between sites, corresponding to the lattice edges, are represented as edges e_{ij}, where i and j are the indices of the lattice sites. In the graph, each node represents a position in the lattice. The original configurations are simulated using DP with a bond probability $p \in [p_{\min}, p_{\max}]$.

After T steps of simulation, the feature of node v_i is denoted as Tr_i^T, where T represents the simulation time steps. Node features explicitly track temporal evolution, enabling the model to learn both local and global dynamics. Finally, the graph G and its node features $\{\mathrm{Tr}_1^T, \mathrm{Tr}_2^T, \ldots, \mathrm{Tr}_i^T\}$ are input into the model.

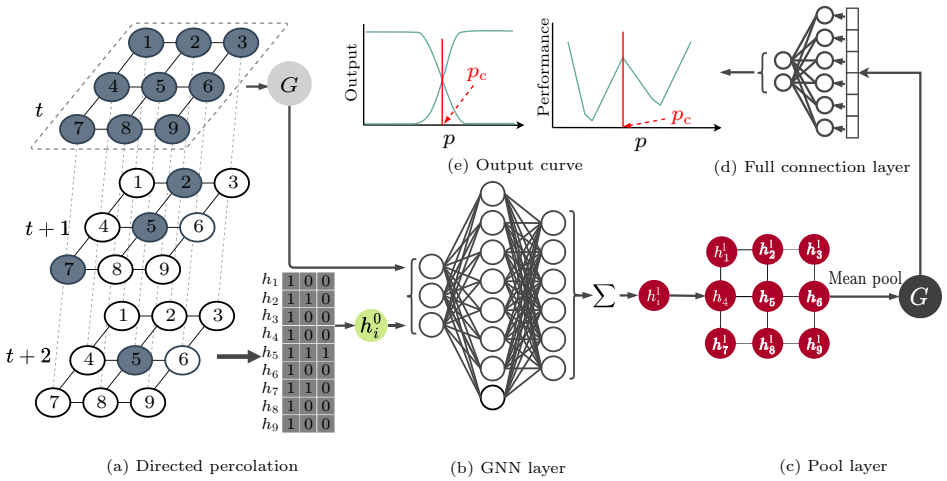


Fig. 2. The main workflow of the DP-GNN framework: (a) constructing DP configurations as graphs, (b) graph convolution layer, (c) pooling layer, (d) fully connected layer, and (e) supervised learning produces an "X"-shaped curve, while unsupervised learning produces a "W"-shaped curve.

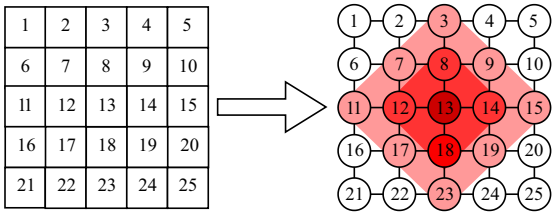


Fig. 3. The original DP configuration of size 5×5 (left) is represented as a graph structure (right). In the graph on the right, the red section illustrates the process by which node 13 gathers information from its neighbors.

2.2.2. Convolution and pooling

In Fig. 2(b), we use a graph attention network (GAT)[34] for graph convolution. The GAT adjusts the weights of edges adaptively to capture both local and global structural features. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the set of nodes, $\mathcal{E}$ represents the set of edges, and $h_i \in \mathbb{R}^T$ is the feature vector of node i with dimension T. The goal is to update node features by aggregating information from neighboring nodes. This is achieved through the following steps: First, a linear transformation is applied to the features of node i and its neighbors $j \in \mathcal{N}(i)$, using a shared weight matrix $W \in \mathbb{R}^{T' \times T}$ to project the original features into a new space:

$$h_i' = W h_i, \quad h_j' = W h_j. \tag{2}$$

Next, an attention coefficient e_{ij} is calculated, reflecting the importance of node j to node i. The computation combines the content of the node features and the edge weights:

$$e_{ij} = \mathrm{ReLU}(\alpha^\top [h_i' \| h_j']), \tag{3}$$

where, $\alpha \in \mathbb{R}^{2T'}$ is a learnable parameter, and $\|$ denotes concatenation. The attention coefficients are normalized using the softmax function across all neighbors of node i:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})}. \tag{4}$$

Using the normalized attention coefficients α_{ij}, the features of neighboring nodes are aggregated to update the feature of node i

$$h_i'' = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} h_j'\right). \tag{5}$$

As shown in Fig. 3 (right), after one round of learning, each node obtains information from its neighbors. At this stage, the features of the network are $\{h_1^1, h_2^1, \ldots, h_n^1\}$. After a pooling layer operation,[35] a supernode feature is obtained to represent the entire graph:

$$h_{\mathrm{global}} = \frac{1}{n} \sum_{i=1}^{n} h_i''. \tag{6}$$

The GAT architecture consists of 3 layers with hidden dimensions [64, 64, 64]. The attention heads are set to 4, and the learning rate is 10^{-3} with Adam optimizer. Monte Carlo simulations run for $T = 64$ steps, and node states are sampled every steps. Training uses a batch size of 16 for 50 epochs with early stopping based on validation loss.

2.2.3. Output

The graph processed by mean pooling is fed into fully connected layer, which maps it into classification probabilities. (Fig. 2(d)), the model outputs two values representing the probabilities of the current configuration belonging to the active phase and the absorbing phase, respectively. In Fig. 2(e), for the supervised learning method, we obtain an "X"-shaped

curve, where the intersection corresponds to the predicted p_c. For the unsupervised method, the true p_c is determined by the position of the peak in the "W"-shaped performance curve.

3. Result

3.1. Supervised learning

In the DP model, all configurations generated by a single bond percolation probability belong to the same category. The objective of the supervised learning method is to classify the phase of DP configurations generated with different bond probabilities. Our dataset consists of training, testing, and corresponding label sets. The generation mechanism for the training and testing datasets is identical. Since the evolution characteristics of a fully occupied initial state are more pronounced and conducive to model learning and training, we select fully occupied one-dimensional and (2D) lattices of different sizes as training data. The sizes of the one-dimensional (1D) lattices are $L = [8, 16, 32, 48, 64]$, while the 2D lattice sizes are $L \times L = [4 \times 4,\ 8 \times 8,\ 16 \times 16,\ 32 \times 32]$. For 1D data, configurations are generated using 51 bond probabilities in the range $(0.4, 0.9)$ with an interval of 0.01, while 2D configurations are generated using a similar scheme in the range $(0.1, 0.5)$, with each bond probability corresponding to 100 samples. The ratio of training to testing samples is 7:3. Using Monte Carlo simulation, we record the dynamic evolution of the percolation process as node features. At time t, a node i is recorded as "1" if it is in the active state, and "0" otherwise.

Before learning, the generated configuration data must be labeled, and several hyperparameters of the neural network need to be defined. When the bond probability $p < p_c$, the generated input configurations are labeled as "0" indicating that the system has entered the absorbing phase. When $p > p_c$, the input configurations are labeled as "1" representing the percolating state. In the input layer of the neural network, the number of neurons equals the dimension of the input network features, *i.e.*, the simulation time steps T. We set $T = L_{\max} = 64$ for both (1+1)-dimensional and (2+1)-dimensional simulations.

We tested the trained model using (1+1)-dimensional DP configurations with $L = [20, 40, 60, 80]$ (Fig. 4, top) and (2+1)-dimensional configurations with $L = [20, 40, 60, 80]$ (Fig. 4, bottom). The results (Figs. 4(a) and 4(c) show that the model can accurately identify phase transitions despite processing data from different dimensions. For instance, in Fig. 4, the blue pair of curves ($L = 20$) corresponds to the two outputs of the neural network. As the bond probability increases, the predicted probability of the active phase gradually rises, while that of the absorbing phase decreases. This trend is consistent across data from different dimensions, demonstrating that the model effectively captures the phase transition features of the DP model.

Additionally, we extracted the predicted critical thresholds at four system sizes (Figs. 4(b) and 4(d)) and plotted them as a function of $1/L$. By performing finite-size scaling to extrapolate from $L \to 0$ to $L \to \infty$, we obtained the critical points: $p_c = 0.642$ for the (1+1)-dimensional DP and $p_c = 0.287$ for the (2+1)-dimensional DP. These results further validate the accuracy of the model. Notably, considering the time cost of model training, we employed the simplest neural network architecture and used a limited amount of training data.

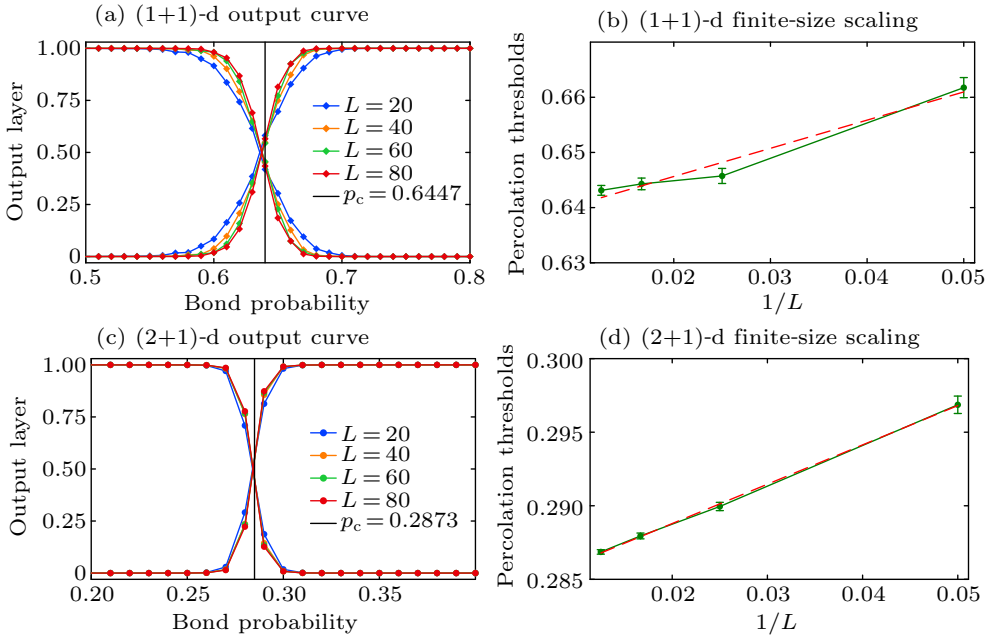


Fig. 4. Supervised learning predicts the phase transition points of (1+1)-dimensional [(1+1)-d] DP (a) and (2+1)-dimensional [(2+1)-d] DP (c) for $L = [20, 40, 60, 80]$. The intersection points correspond to the predicted p_c'. The black line represents the true p_c. The subfigures (b) and (d) display the finite-size scaling fits, yielding a phase transition point of 0.642 for the 1D case and 0.287 for the 2D case.

3.2. Unsupervised learning

Supervised learning in the DP model requires prior knowledge of the percolation threshold of the configurations, which must be manually labeled for the neural network. However, in most cases, the very goal is to identify these labels (*i.e.*, the phase classification). Can our framework automatically identify the phase transition point without manual labeling? Evert *et al*. proposed a confusion scheme that enables the machine to identify phase transition points without manual annotation.

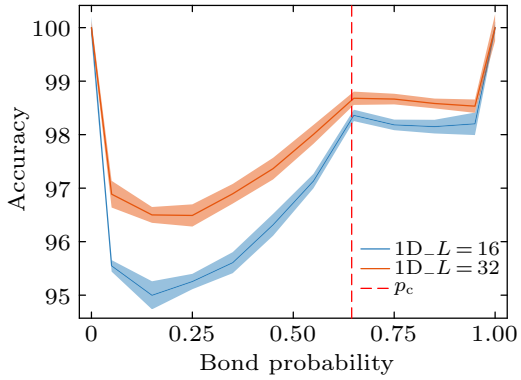


Fig. 5. Unsupervised learning predicts the phase transition point of (1+1)-dimensional DP for $L = [16, 32]$. The peaks of the "W"-shaped curves correspond to the predicted p_c', while the red dashed line represents the true p_c.

Assume that the percolation threshold $p_c \in [a, b]$. A hypothetical critical point p_c' is artificially assumed, and a neural network N_c is trained. All data with bond probabilities smaller than p_c' are labeled as "0" , while the rest are labeled as "1". Next, the performance of N_c is evaluated across the entire dataset, and the total performance relative to the proposed critical point p_c' is denoted as $P(p_c')$. The function $P(p_c')$ exhibits a universal "W"-shaped curve, with the peak located at the correct critical point p_c.

By combining our framework with the confusion scheme and applying it to (1+1)-dimensional DP configurations, we obtained the expected universal "W"-shaped curve. The "W"-shaped performance curve arises from the competition between critical slowing down and absorbing state stability. Near p_c, the correlation length diverges, leading to ambiguous classifications and reduced model accuracy (valley of the "W"). Peaks correspond to p_c' values where the model effectively distinguishes between subcritical and supercritical dynamics. We set the range of the percolation threshold $p_c \in [0.05, 1]$, taking a hypothetical critical point p_c' at intervals of 0.1. For each p_c', we uniformly sampled 20000 bond probabilities in the range $[0.05, p_c')$ and labeled them as "0" (absorbing phase), and 20000 bond probabilities in the range $[p_c', 1]$ labeled as "1" (percolating). These samples were then fed into our framework for training to evaluate the performance $P(p_c')$. After testing all p_c' values, the peak of the performance curve was located at 0.65, which closely matches the actual p_c.

4. Conclusion

This study presents a universal framework leveraging GNNs to analyze non-equilibrium phase transitions, with a focus on the DP model. By converting lattice configurations into graph structures and embedding temporal evolution into node features, our approach enables unified processing of DP systems with diverse dimensions and topologies. The framework accurately predicts percolation thresholds without depending on lattice geometry, showcasing its adaptability and broad applicability across various systems.

Furthermore, by integrating our framework with the confusion scheme, we enable the automatic identification of phase transition points in an unsupervised learning setup. This research offers a fresh perspective on applying machine learning techniques to nonequilibrium phase transition studies. While the focus is on the DP model, the proposed framework is highly versatile and can be extended to other complex dynamical systems. Future work could explore its application to higher-dimensional and large-scale systems, as well as optimize its computational efficiency to tackle more intricate nonequilibrium phenomena.

While our framework demonstrates versatility, scalability to higher dimensions remains challenging due to the quadratic growth of graph edges with system size. Additionally, training on large-scale systems requires significant GPU memory. Future work could explore sparse graph representations or hierarchical pooling to mitigate these limitations.

Acknowledgements

Project supported by the Fund from the Science and Technology Department of Henan Province, China (Grant Nos. 222102210233 and 232102210064), the National Natural Science Foundation of China (Grant Nos. 62373169 and 72474086), the Young and Mid-career Academic Leader of Jiangsu Province, China (Grant No. Qinglan Project in 2024), the National Statistical Science Research Project (Grant No. 2022LZ03), Shaanxi Provincial Soft Science Project (Grant No. 2022KRM111), Shaanxi Provincial Social Science Foundation (Grant No. 2022R016), the Special Project for Philosophical and Social Sciences Research in Shaanxi Province, China (Grant No. 2024QN018), and the Fund from the Henan Office of Philosophy and Social Science (Grant No. 2023CJJ112).

References

[1] Mahesh B 2020 *IJSR* **9** 381
[2] Carleo G, Cirac I, Cranmer K, Daudet L, Schuld M, Tishby N, Vogt-Maranto L and Zdeborová L 2019 *Rev. Mod. Phys.* **91** 045002
[3] Broecker P, Assaad F F and Trebst S 2017 arXiv:1707.00663 [cond-mat]
[4] Carrasquilla J and Melko R G 2017 *Nat. Phys.* **13** 431
[5] Xu R, Fu W and Zhao H 2019 arXiv:1901.00774 [cond-mat.stat-mech]
[6] Zhang W, Liu J and Wei T C 2019 *Phys. Rev. E* **99** 032142

[7] Carleo G and Troyer M 2017 *Science* **355** 602
[8] VanderPlas J, Connolly A J and Gray A 2012 *2012 Conference on Intelligent Data Understanding* **18** 47
[9] Mehta P, Bukov M, Wang C H, Day A G, Richardson C, Fisher C K and Schwab D J 2019 *Phys. Rep.* **810** 1
[10] Carleo G, Cirac I, Cranmer K, Daudet L, Schuld M, Tishby N, Vogt-Maranto L and Zdeborová L 2019 *Rev. Mod. Phys.* **91** 045002
[11] Landau D and Binder K 2021 *A guide to Monte Carlo simulations in statistical physics* (Cambridge: Cambridge University Press)
[12] Shen J, Li W, Deng S and Zhang T 2021 *Phys. Rev. E* **103** 052140
[13] Hinrichsen H 2000 *Adv. Phys.* **49** 815
[14] Takeuchi K A, Kuroda M, Chate H and Sano M 2007 *Phys. Rev. Lett.* **99** 234503
[15] Carrasquilla J and Melko R G 2017 *Nat. Phys.* **13** 431
[16] Tanaka A and Tomiya A 2017 *J. Phys. Soc. Jpn.* **86** 063001
[17] Horani A, Ustione A, Huang T, Firth A L, Pan J, Gunsten S P, Haspel J A, Piston D W and Brody S L 2018 *Proc. Natl. Acad. Sci. USA* **115**
[18] Wang L 2016 *Phys. Rev. B* **94** 195105
[19] Hughes R I 1999 *Phys. Ideas Context* **52** 97
[20] Van Nieuwenburg E P L, Liu Y H and Huber S D 2017 *Nat. Phys.* **13** 435
[21] Marro J and Dickman R 2005 *Nonequilibrium Phase Transitions in Lattice Models*
[22] Anon 2008 *Non-Equilibrium Phase Transitions* (Dordrecht: Springer Netherlands)
[23] Obukhov S P 1980 *Physica A* **101** 145
[24] Battiston F, Amico E, Barrat A, Bianconi G, Ferraz de Arruda G, Franceschiello B, Iacopini I, Kéfi S, Latora V and Moreno Y 2021 *Nat. Phys.* **17** 1093
[25] Carleson L and Gamelin T Complex dynamics 1996 *Complex dynamics* (Springer Science, Business Media)
[26] Bec J, Gustavsson K and Mehlig B 2024 *Annu. Rev. Fluid Mech.* **56** 189
[27] Canet L 2004 *Processus de réaction-diffusion: une approche par le groupe de renormalisation non perturbatif*
[28] Newman M E J 2002 *Phys. Rev. E* **66** 016128
[29] Takeuchi K A, Kuroda M, Chate H and Sano M 2009 *Phys. Rev. E* **80** 051116
[30] Corso G, Stark H, Jegelka S, Jaakkola T and Barzilay R 2024 *Nature Reviews Methods Primers* **4** 17
[31] Vivas D, Madronero J, Bucheli V, Gomez L and Reina J 2022 arXiv:2204.12966 [quant-ph]
[32] Steeven J, Aurelien B, Madiha N, Julie D, Nicolas T and Christian W 2023 arXiv:2302.10803 [cs.LG]
[33] Munikoti S, Das L and Natarajan B 2022 *Neurocomputing* **468** 211
[34] Velickovic P, Cucurull G, Casanova A, Romero A, Lio P and Bengio Y 2018 Graph Attention Networks 2018 arXiv:1710.10903 [stat]
[35] Gholamalinezhad H and Khosravi H 2020 arXiv:2009.07485 [cs]