Phase transitions and optimal algorithms for semisupervised classifications on graphs: From belief propagation to graph convolution network

Pengfei Zhou
CAS Key Laboratory for Theoretical Physics, Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China and School of Physical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China

Tianyi Li
System Dynamics Group, Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA

Pan Zhang[*]
CAS Key Laboratory for Theoretical Physics, Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China; School of Fundamental Physics and Mathematical Sciences, Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou 310024, China; and International Centre for Theoretical Physics Asia-Pacific, Beijing/Hangzhou, China

(Received 9 December 2019; accepted 28 July 2020; published 28 August 2020)

We perform theoretical and algorithmic studies for the problem of clustering and semisupervised classification on graphs with both pairwise relational information and single-point attribute information, upon a joint stochastic block model for synthetic graphs with both item-item edges and item-attribute edges. Asymptotically exact analysis based on the Bayesian inference of the model are conducted, using the cavity method in statistical physics. Analytically, we identify a phase transition of the generative model, which poses fundamental limits on the detectability of the underlying model in the clustering task for all possible algorithms. Algorithmically, we propose a *belief propagation* algorithm that is asymptotically optimal on the generative model, which can be further extended to a *belief propagation graph convolution neural network* (BPGCN) for semisupervised classification on graphs. Well-controlled benchmark data sets of factor graphs accompanied with asymptotically optimal solutions in classification could be produced for the evaluation of graph convolution neural networks and for the theoretical understanding of their strengths and weaknesses. In particular, on these synthetic benchmark networks we observe that existing graph convolution neural networks are subject to an sparsity issue and an overfitting issue in practice, both of which could be successfully overcome by our BPGCN. Moreover, when combined with classic neural network methods, BPGCN yields extraordinary classification performances on real-world data sets that are at least comparable to state-of-the-art graph convolution networks.

DOI: 10.1103/PhysRevResearch.2.033325

I. INTRODUCTION

Learning on graphs is an important task in machine learning and the broader data sciences which triggers a lot of successful applications in various fields, including social sciences (e.g., social network analysis), biology (e.g., protein structure prediction and molecular fingerprints learning), and computer science (e.g., knowledge graph analysis). The key difference between learning with graph data and traditional machine learning on images and natural languages is that, in addition to content features on each item, there are also relational features between items that are encoded by edges in the graph, which adds an extra layer of complexity to the analysis.

One classical problem of learning on graphs is the classification of nodes into groups. Consider a citation network, where each article is represented by a graph node, and the groups of nodes are scientific research fields. In addition to the edges between nodes, which represent the citations between articles, each node is also associated with some *attributes* (i.e., key words in any article), which encode its categorical information of research fields. If the group information is known on a small subset of nodes, then practically, these nodes could serves as a training set, and the learning task is to determine the group membership of the remaining nodes through exploring the direct group information via their attributes, as well as the indirect information via their relationships with the training nodes (edge connectivities of the graph). Essentially, this learning task is *semisupervised classification on graphs*, a problem that recently has drawn much attention in both networks sciences and machine learning communities; for this problem, we witnessed the burst of *graph convolution neural networks* (GCN), which is a powerful neural network architecture that yields ground-breaking performances [1].

[*]panzhang@itp.ac.cn

Deep convolution neural networks have achieved tremendous success in machine learning and artificial intelligence [2]. Since there are many situations in which data are represented as graphs, rather than as voices or images that could be recorded on one- or two-dimensional grids, a lot of effort has been made to extend convolution networks from applying on grid data to applying on graph data, with a heavy focus on constructing linear convolution kernels to extract local node attributes in graphs and on learning effective representations of graph objects. In the past several years, many GCNs have been proposed, utilizing different types of convolution kernels and different network architectures [1,3–5]. Recent studies show that GCNs have quickly dominated among different neural network techniques in the performance on various learning tasks, including text- or graph-object classification, link prediction, forecasting, importance sampling, and are also believed to have big potential for relational reasoning [6]. Nevertheless, although GCNs have achieved the state-of-the-art performance on semisupervised classification, so far there is little theoretical understanding of the mathematical principles behind graph convolutions and of the extent that they may work in a particular problem setting. The main difficulty is that, in previous studies, GCNs are often only tested on real-world data sets which do not have clear theoretical structures, and thus the success or failure of GCNs is hard to pin down in theoretical analysis. A set of network data sets with established mathematical properties and the analysis of GCNs on such data sets are missing and greatly welcomed. Note that some work [7,8] also gave theoretical analyses to show limited expressiveness of GCNs on entire graph learning, and here we offer another perspective to show GCNs' strengths and weaknesses based on semisupervised classification on well-controlled graphs.

The basis for our study is a generative model for both graphs and attributes. In the field of community detection where the task is to detect clusters purely based on edges, many analysis are based on the celebrated stochastic block model (SBM) [9]. However, the SBM is not enough for our purpose because in addition to generating edges, we also need to generate attributes. In this work, we propose to use a variant of the SBM, a joint model consisting of two graph components, characterizing both the relational information and the attribute information of item nodes, which are captured respectively by a standard SBM and a bipartite SBM [10–12]. We call the model the joint stochastic block model (JSBM). This model was originally proposed in Ref. [10] for the problem of link and node predictions through the Markov chain–Monte Carlo method. In this study, we analyze theoretical properties of the JSBM and design techniques for semisupervised learning on graphs based on its desirable properties. Filling the gap discussed in the above, the JSBM produces well-controlled benchmark graphs with continuously tunable parameters for the evaluation of GCNs' classification performance, and for the theoretical understanding of their strengths and weaknesses under certain conditions.

On graphs generated by the JSBM, the clustering and classification problems can be translated into a Bayesian inference problem, which can be solved theoretically with the statistical physics approach in an asymptotically exact manner. This approach leads to a message-passing algorithm, known in computer science as the belief propagation (BP) algorithm [13], which we claim to be asymptotically exact on large random graphs generated by the JSBM. Through analyzing the stability of fixed points of the constructed BP equations on the JSBM, a phase transition—the *detectability transition*—is identified; beyond the phase transition point, no algorithm is able to conduct successful clustering on JSBM graphs in an unsupervised manner. This is an extension of the detectability phase transition [14] in the standard SBM [9], which puts fundamental limits on the ability of algorithms in the clustering tasks on graphs that can be sufficiently modeled by JSBM.

In the semisupervised classification setting, where a small fraction of nodes have ground-truth group labels and could be used as the training data, the BP algorithm for JSBM can be embedded into a graph convolution network architecture. The unknown generative parameters of the JSBM graph can be learned in a standard classification approach, through the forward-passing of (truncated) BP equations together with the backward-passing of the gradients of the loss function. This GCN algorithm, which we term BPGCN, guarantees to yield Bayes optimal classification results [15] on synthetic graphs generated by the JSBM and performs comparably with state-or-the-art GCNs on real-world networks.

This paper is organized as follows. In Sec. II, we introduce the joint stochastic block model. In Sec. III, we formulate the Bayesian inference problem for clustering and classification on the joint stochastic block model and derive the belief propagation equations for the JSBM. In Sec. IV, we study the detectability phase transition of JSBM using stability analysis of the BP algorithm. In Sec. V, we convert the BP equations on JSBM to a graph convolution neural network and propose a GCN algorithm, BPGCN. In Sec. VI, the performance of BPGCN is evaluated and compared with the performance of several state-of-the-art GCNs, on both synthetic and real-world networks. Section VII concludes the study.

II. JOINT STOCHASTIC BLOCK MODEL

The idea of the JSBM, literally the joint of two stochastic block models, is to simultaneously model item nodes and attribute nodes in the network setting by representing both a connectivity graph over item nodes and an attribute graph between item nodes and attribute nodes. The connectivity graph corresponds to the relation network in the traditional sense and the attribute graph is a bipartite graph established on top of the relation network; both graphs associate each node in the graph with a unique group membership. These two graphs, constructed from two SBM processes, constitute the JSBM graph $\mathcal{G}$; a similar framework was proposed in Ref. [10] to study link predictions.

Assume n item nodes in the (undirected) connectivity graph, belonging to κ groups. Each node i has an unknown label t_i^* denoting its independent group membership (i.e., $t_i^* \in \{1, 2, ...\kappa\}$); each group label is chosen at random by nodes, according to a κ-dimensional probability vector α, whose entries sum to 1. Edge connectivities are exclusively determined by their group memberships: For each pair of nodes i and j, there exists an edge $(i, j) \in \mathcal{E}$ with probability $p_{t_i^* t_j^*}$, where $\mathcal{E}$ is the entire edge set of the graph. This stochastic generative process could be understood as a measuring process in which

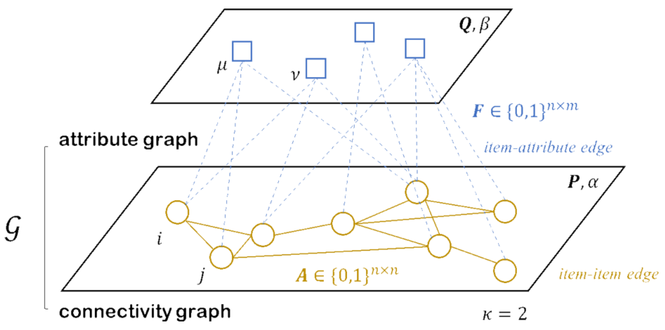

FIG. 1. Illustration of the joint stochastic block model (JSBM). Circles denote item nodes, whose interconnections constitute the connectivity graph, with adjacency matrix **A** defined over edges; boxes denote feature nodes. The attribute graph is the bipartite graph between items and attributes, whose adjacency matrix is **F**. The JSBM graph is generated by parameters $\theta = \{\mathbf{P}, \mathbf{Q}, \alpha, \beta\}$, with κ groups of nodes built in for both items and features.

the ground truth t_i^* is being measured, whose information is encoded implicitly in the edge connectivities as measuring results. The $n \times n$ adjacency matrix of this connectivity graph $\mathbf{A} \in \{0, 1\}^{n\times n}$ follows the generative process and is therefore stochastic, controlled by the deterministic $\kappa \times \kappa$ generation probability matrix $\mathbf{P} = \{p_{t_i^* t_j^*}\}$. If the diagonal elements in **P** are larger than the off-diagonal elements, it corresponds to the situation (known as *assortative* SBM) where there are more edges within node groups than between groups, and vice versa (known as *disassortative* SBM). An intuitive example for an *assortative* SBM is a citation network with nodes denoting research articles (items), which belong to a certain research area (group) and are linked through citations (edges). Conceptually, articles from the same research area are more likely to cite each other (e.g., Ref. [1]).

Besides having membership in a certain research area, moreover, each research article may also be associated with some keywords which further denote its categorical information; expectedly, the research area that an article belongs to could be inferred from these keywords. This notion underlines the idea of making inference on the joint SBM, i.e., inferring hidden group membership of an item node from its relationships to known attribute nodes in the attribute graph. Assume such a graph with m attributes over n item nodes. As with item nodes, each attribute node μ is embedded with an independent label $t_\mu^* \in \{1, 2, ...\kappa\}$, chosen randomly from the κ groups according to probabilities in a κ-dimensional vector β. Relationship between an item node i and an attribute node μ exists with probability $q_{t_i^* t_\mu^*}$, which corresponds to an edge $(i, \mu) \in \mathcal{F}$ in the attribute graph whose edge set is $\mathcal{F}$. This generative process yields a bipartite graph between item nodes and attribute nodes, analogous to the bipartite SBM [11]; the resulting $n \times m$ adjacency matrix of the attribute graph $\mathbf{F} \in \{0, 1\}^{n\times m}$ is stochastic and is governed by a $\kappa \times \kappa$ matrix $\mathbf{Q} = \{q_{t_i^* t_\mu^*}\}$, similar to the case of **A** and **P**. A specific JSBM is thus represented by the two adjacency matrices **A** and **F** for the connectivity graph and the attribute graph, respectively, and is controlled by the generation parameters $\theta = \{\mathbf{P}, \mathbf{Q}, \alpha, \beta\}$ (Fig. 1). It consists of a unipartite graph and a bipartite graph, similar to the structure in semirestricted Boltzmann machines [16]. For convenience, we call edges in unipartite graphs item-item edges and edges in bipartite graphs item-attribute edges. It is also worth noting that in the current model we make no assumption on the relationships between different feature nodes (i.e., the adjacency matrix on the attribute graph); such connectivities may provide information about the group membership of feature nodes (which we do not worry about in the current study), but not directly about the membership of item nodes which is the ground truth under concern.

III. BAYESIAN INFERENCE ON JSBM

On the graph $\mathcal{G}$ generated by the JSBM, the problem of *clustering* is defined as recovering ground-truth labels $\{t_i^*\}$ exclusively using edge information in the connectivity graph and the attribute graph (i.e., adjacency matrix **A** and **F**); the problem of *semisupervised classification*, in a slightly different manner, asks to conduct the same recovery but utilizes as extra information a small number of training labels $\{t_{\tilde{i}}^*\}$ where $\tilde{i}$ belongs to the training set Ω. If the parameters θ in generating the JBSM are known, this classification task can essentially be translated into an inference problem on group labels $\{t_i^*\}$, which could be viewed as hidden parameters in the model, provided with measurements on a specific type of outcomes of these parameters $\{t_i^*\}$ (the location of edges in the two graphs). Bayesian inference, which amounts to computing the posterior distribution, is typically used for such an inference task. For our problem, under Bayesian rules, the posterior is written as

$$P(\{t_i\}, \{t_\mu\}|\mathcal{G}, \theta) = \frac{P(\mathcal{G}|\{t_i\}, \{t_\mu\}, \theta)P_0(\{t_i\}, \{t_\mu\})}{\sum_{\{t_i\}} P(\mathcal{G}|\{t_i\}, \{t_\mu\}, \theta)P_0(\{t_i\}, \{t_\mu\})}, \quad (1)$$

where $P_0(\{t_i\}, \{t_\mu\})$ represents prior information on the labels (e.g., information regarding generative parameters α and β), and $P(\mathcal{G}|\{t_i\}, \{t_\mu\}, \theta)$ is the likelihood of observing graph $\mathcal{G}$ given labels $\{t_i\}, \{t_\mu\}$ and parameters θ, which is the product-form probability of generating existent item-item edges and item-attribute edges in $\mathcal{G}$:

$$P(\mathcal{G}|\{t_i\}, \{t_\mu\}, \theta) = \prod_{(ij)\in\mathcal{E}} p_{t_i,t_j} \prod_{(ij)\notin\mathcal{E}} (1 - p_{t_i,t_j})$$
$$\times \prod_{(i\mu)\in\mathcal{F}} q_{t_i,t_\mu} \prod_{(i\mu)\notin\mathcal{F}} (1 - q_{t_i,t_\mu}). \quad (2)$$

The clustering problem corresponds to adopting a flat prior [i.e., $P_0(\{t_i\}, \{t_\mu\}) = P_0(\{t_i\}, \{t_\mu\}) =$ const. or α and β are uniform-entry (probability) vectors], and semisupervised classification corresponds to adopting strong prior on item nodes that belong to the training set, such that the probability marginals of item nodes belonging to the training set are pinned in the direction of training labels [15].

It is well known that computing the normalization of the posterior distribution [i.e., the denominator of Eq. (1) is a #P problem], and thus efficient and accurate approximations are needed for the inference. In the language of statistical physics, the representation of Bayesian inference on the posterior distribution Eq. (1) corresponds to a Boltzmann distribution at unit temperature: The negative log-likelihood $-\log P(\mathcal{G}|\{t_i\}, \{t_\mu\}, \theta)$ represents the energy; the normalization constant $\sum_{\{t_i\}} P(\mathcal{G}|\{t_i\}, \{t_\mu\}, \theta)P_0(\{t_i\}, \{t_\mu\})$ for the posterior is the partition function; and the prior information P_0 plays the role of external fields acting on item nodes and feature nodes. For the clustering problem with a flat prior, the external field is zero for all nodes; for semisupervised classification, the external field is infinity for nodes in the training set and zero for unlabelled nodes [15].

For random sparse graphs, the inference could be studied at the thermodynamic limit using the cavity method from statistical physics [13,17]. If the parameters used in generating the SBM is known *a priori*, the system is on the *Nishimori line* [18,19] and no spin glass phase could appear. Moreover, the replica symmetry cavity method naturally translates into the *belief propagation* (BP) algorithm, a well-known algorithm for computation on SBM. In BP, cavity messages are passed along directed edges of the factor graph; when the propagation converges, they are used to compute the posterior marginals. Inheriting the message-passing idea, for JSBM, where the factor graph $\mathcal{G}$ is threefold (i.e., there are three types of directed edges: item-item, item-attribute, and attribute-item), we formulate the iterative equations of BP for three types of messages (see Appendix A for derivations):

$$\begin{aligned}
\psi_{t_i}^{i\to j} &= \alpha_{t_i}\frac{e^{-h_{t_i}}}{Z^{i\to j}}\prod_{k\in\partial i\backslash j}\sum_{t_k}p_{t_it_k}\psi_{t_k}^{k\to i}\prod_{\mu\in\partial i}\sum_{t_\mu}q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i},\\
\psi_{t_i}^{i\to\mu} &= \alpha_{t_i}\frac{e^{-h_{t_i}}}{Z^{i\to\mu}}\prod_{k\in\partial i}\sum_{t_k}p_{t_it_k}\psi_{t_k}^{k\to i}\prod_{\nu\in\partial i\backslash\mu}\sum_{t_\nu}q_{t_it_\nu}\psi_{t_\nu}^{\nu\to i},\\
\psi_{t_\mu}^{\mu\to i} &= \beta_{t_\mu}\frac{e^{-h_{t_\mu}}}{Z^{\mu\to i}}\prod_{j\in\partial\mu\backslash i}\sum_{t_j}q_{t_\mu t_j}\psi_{t_j}^{j\to\mu}.
\end{aligned}\tag{3}$$

Here $\psi_{t_i}^{i\to j}$ are the cavity marginals (messages) passing through item node i to item node j, representing the probability of item node i taking label t_i when the item node j is removed from the graph. Similarly, $\psi_{t_i}^{i\to\mu}$ represents the probability of item node i taking label t_i when attribute node μ is removed from the graph, and $\psi_{t_\mu}^{\mu\to i}$ represents the probability of attribute node μ taking label t_μ when item node i is removed. $Z^{i\to\mu}$, $Z^{\mu\to i}$, and $Z^{i\to j}$ are normalizing factors; ∂i denotes the set of neighbors of item node i in the graph. In the three equations, variables h_{t_i} and h_{t_μ} are adaptive fields contributed by nonexistent edges of the graph, which are formulated as (see Appendix A)

$$\begin{aligned}
h_{t_i} &= \sum_k\sum_{t_k}p_{t_it_k}\psi_{t_k}^k+\sum_\mu\sum_{t_\mu}q_{t_it_\mu}\psi_{t_\mu}^\mu,\\
h_{t_\mu} &= \sum_j\sum_{t_j}q_{t_\mu t_j}\psi_{t_j}^j.
\end{aligned}\tag{4}$$

Once the above iterative equations converge (i.e., messages do not change significantly), using the determined cavity messages, the posterior marginals on the two graphs (connectivity, attribute) could be calculated by

$$\begin{aligned}
\psi_{t_i}^i &= \alpha_{t_i}\frac{e^{-h_{t_i}}}{Z^i}\prod_{k\in\partial i}\sum_{t_k}p_{t_it_k}\psi_{t_k}^{k\to i}\prod_{\mu\in\partial i}\sum_{t_\mu}q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i},\\
\psi_{t_\mu}^\mu &= \beta_{t_\mu}\frac{e^{-h_{t_\mu}}}{Z^\mu}\prod_{j\in\partial\mu}\sum_{t_j}q_{t_\mu t_j}\psi_{t_j}^{j\to\mu},
\end{aligned}\tag{5}$$

where $\psi_{t_i}^i$ is the marginal probability of item node i taking label t_i and $\psi_{t_\mu}^\mu$ is the marginal probability of attribute node μ taking label t_μ. In the end, based on these computed marginals, one is able to estimate the label of each item node, which is the specific label that maximizes the item node's marginal:

$$\widetilde{t_i} = \underset{t_i\in\{1,2,\ldots,\kappa\}}{\text{argmax}}\ \psi_{t_i}^i.\tag{6}$$

In Bayesian inference, $\widetilde{t_i}$ is the *maximum posterior estimate*, representing the optimal result with the minimum mean square error (MMSE) [19]. In terms of algorithm design, BP equations essentially adopt the Bethe approximation [13,20], which is a variational distribution formulated as

$$Q(\{t_i\},\{t_\mu\}|\mathcal{G},\theta)=\frac{\prod_{ij}\prod_{i\mu}\Phi_{t_i,t_\mu}^{i,\mu}\Phi_{t_i,t_j}^{i,j}}{\prod_i\left(\psi_{t_i}^i\right)^{|\partial i|-1}\prod_\mu\left(\psi_{t_\mu}^\mu\right)^{|\partial\mu|-1}},\tag{7}$$

where i,μ represent item nodes and attribute nodes respectively, $|\partial i|$ and $|\partial\mu|$ denote degrees of node i and μ respectively; ψ represents single-point marginal, and Φ represents two-point marginal. By optimizing the Kullback-Leibler (KL) divergence Eq. (8) [13,21] between the variational distribution Eq. (7) and Boltzmann posterior distribution Eq. (1), with the constraint induced by normalization of marginals, one can arrive at belief propagation equations (3).

$$\begin{aligned}
&D_{KL}(Q(\{t_i\},\{t_\mu\}|\mathcal{G},\theta)||P(\{t_i\},\{t_\mu\}|\mathcal{G},\theta))\\
&\quad=\sum_{\{t_i\},\{t_\mu\}}Q(\{t_i\},\{t_\mu\}|\mathcal{G},\theta)\log\left(\frac{Q(\{t_i\},\{t_\mu\}|\mathcal{G},\theta)}{P(\{t_i\},\{t_\mu\}|\mathcal{G},\theta)}\right),
\end{aligned}\tag{8}$$

We notice that the two-point marginals $\Phi_{t_i,t_\mu}^{i,\mu}$, $\Phi_{t_i,t_j}^{i,j}$ can be computed using the cavity messages in BP equations after they converge; we refer to Ref. [17] for more details of belief propagation in the factor graphs. The core assumption is the conditional independence assumption, which is exact on trees. Thus, the Bethe approximation (7) is always correct for a tree graph in describing joint probabilities, in which case BP algorithms yield exact posterior marginals. Empirically, BP results are shown to be good approximations to true posterior marginals, if the graph is sparse and of locally treelike structures; hence the BP algorithm is widely applied to inference problems in sparse systems [17].

IV. DETECTABILITY TRANSITIONS OF JSBM

For graphs generated by the JSBM with parameters θ, the cavity method provides asymptotically exact analysis, and the belief propagation algorithm (almost) always converges, according to the Nishimori line property [18,19]. Thus, asymptotically exact properties of the JSBM, such as the phase diagram, can be studied directly at the thermodynamic limit by analyzing the messages in the belief propagation.

From (4) and (6), observe that there is a trivial fixed point of BP equations (3)

$$\begin{aligned}
\psi_{t_i}^{i\to j} &= \psi_{t_i}^{i\to\mu}=\psi_{t_i}^i=\alpha_{t_i},\\
\psi_{t_\mu}^{\mu\to i} &= \psi_{t_\mu}^\mu=\beta_{t_\mu}.
\end{aligned}\tag{9}$$

This fixed point corresponds to the situation where every node in the graph has equal probability of belonging to every group;

therefore, it is known as the *paramagnetic fixed point* or *liquid fixed point*. In this case, the marginals do not provide any information about the ground-truth group labels, whereas only reflecting the permutational symmetry of the system. When this paramagnetic fixed point is stable, the system is in the paramagnetic state, where it is believed that no algorithm can do better than a random guess in revealing planted group structures. This scenario is known as the nondetectable phase for SBM, whose existence has been mathematically proved in Ref. [22]; in this study, we extend the analysis to the JSBM with two SBM components. Conceptually, the nondetectable phase in the JSBM is analogous to the ferromagnetic Ising model in the paramagnetic phase where the underlying ground-truth labels correspond to the all-one configuration, as well as the Hopfield model where underlying ground-truth labels refer to the stored patterns. From the viewpoint of statistical inference, item-item edges $\{(ij)\}$ and item-attribute edges $\{(i\mu)\}$ are observations of the signal (i.e., the ground-truth labels), so the paramagnetic phase denotes the situation where the number of observations is too few to reveal any valid information of the signal, such that the system evolves to the paramagnetic fixed point where the label assignment is of equal probability for any node.

When the number of observations increases, the paramagnetic fixed point will eventually become unstable, leading to a nontrivial fixed point of BP (3) whose values are correlated with the ground truth. Where the paramagnetic fixed point of BP becomes unstable indicates the position of the *detectability transition* for the JSBM, which poses fundamental limits on the ability of algorithms in revealing information of the ground truth, independent of the specific algorithm being used.

This phase transition point can be determined by the stability analysis of the paramagnetic fixed point of the BP (9). Assume that the JSBM graph has $n \to \infty$ item nodes and $m \to \infty$ attribute nodes. Each item node is connected to on average c_1 item nodes and c_2 attribute nodes, and each attribute node is connected to on average c_3 item nodes. So degree distribution of item nodes in the connectivity graph (only counting neighbors of item nodes) and degree distribution of item nodes in the attribute graph (only counting neighbors of attribute nodes) are Poisson distributions, as in the SBM, and c_1 and c_2 also equal to the average excess degree (see Appendix B for definition of average excess degree). Consider putting random noises with zero mean and unit variance on every node (both items and attributes) of the graph. Assuming a local-tree topology of the graph, after one iteration of the BP equations, the noises will be propagated to on average c_1 item nodes through edges, and c_2c_3 item nodes through attribute nodes (Fig. 2). If every leaf node of the tree is associated with a random noise of zero mean and unit variance, after $l \to \infty$ iterations of BP equations (3), the aggregated variance of noises on the root node i can be computed as (see Appendix B for details of derivations)

$$\mathcal{V} = \lim_{l\to\infty} \left(c_1\lambda_A^2 + c_2c_3\lambda_F^4\right)^l. \tag{10}$$

λ_A and λ_F are the largest eigenvalues of the Jacobian matrices $\mathbf{T}^{i\to j}$ and $\mathbf{T}^{\mu\to i}$, related to the connectivity graph and the attribute graph (i.e., the adjacency matrices $\mathbf{A}$ and $\mathbf{F}$),

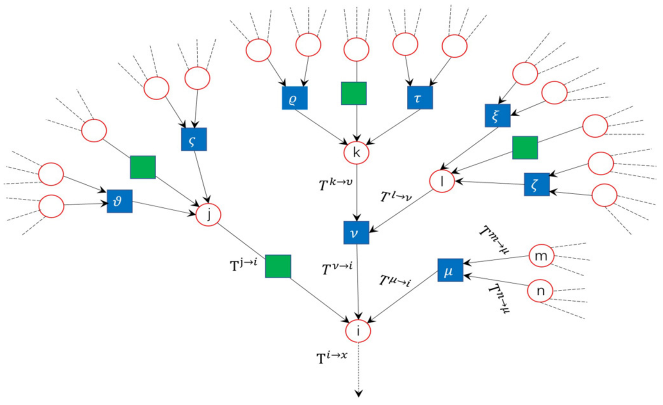

FIG. 2. Illustration of noise propagation on JSBM. Noises are transmitted from leaves to the root node i, with each transmission weighted by the eigenvalues of the Jacobian matrices $\mathbf{T}$, determined by BP message passing equations [Eq. (10)]. Red circles denote item nodes, green boxes represent item-item edges connecting item nodes, and blue boxes denote attribute nodes.

respectively, which are evaluated at the paramagnetic fixed point (together with a third matrix $\mathbf{T}^{i\to\mu}$):

$$\begin{aligned}
T^{i\to j}_{t_it_j} &= \frac{\partial\psi^{j\to x}_{t_j}}{\partial\psi^{i\to j}_{t_i}}\bigg|_{\alpha_{t_i}} = \alpha_{t_i}\left(\frac{np_{t_it_j}}{c_1} - 1\right),\\
T^{\mu\to i}_{t_it_\mu} &= \frac{\partial\psi^{i\to x}_{t_i}}{\partial\psi^{\mu\to i}_{t_\mu}}\bigg|_{\alpha_{t_i},\beta_{t_\mu}} = \alpha_{t_i}\left(\frac{mq_{t_it_\mu}}{c_2} - 1\right),\\
T^{i\to\mu}_{t_\mu t_i} &= \frac{\partial\psi^{\mu\to j}_{t_\mu}}{\partial\psi^{i\to\mu}_{t_i}}\bigg|_{\alpha_{t_i},\beta_{t_\mu}} = \beta_{t_\mu}\left(\frac{mq_{t_\mu t_i}}{c_2} - 1\right).
\end{aligned} \tag{11}$$

Since $l \to \infty$, the paramagnetic fixed point is unstable under random perturbations whenever $\mathcal{V} > 1$. As a result, the detectability phase transition is located at

$$c_1\lambda_A^2 + c_2c_3\lambda_F^4 = 1. \tag{12}$$

This kind of stability conditions are known in the spin glass literature as the Almeida-Thouless local stability condition [23], and in computer sciences as the Kesten-Stigum bound on reconstruction on trees [24,25] and the robust reconstruction threshold [26].

We demonstrate the phase transition Eq. (12) through a simple case of JSBM. Consider a $\mathbf{P}$ matrix with p_{in} being the diagonal and p_{out} being the off-diagonal elements, and a similar $\mathbf{Q}$ matrix with q_{in} on the diagonal and q_{out} on the off-diagonal. Notice that p_{in} and p_{out}, as well as q_{in} and q_{out}, are related by

$$\begin{aligned}
p_{\text{in}} + (\kappa-1)p_{\text{out}} &= \frac{\kappa c_1}{n},\\
q_{\text{in}} + (\kappa-1)q_{\text{out}} &= \frac{\kappa c_2}{m},
\end{aligned} \tag{13}$$

and hence there is only one free parameter for each matrix. We introduce $\epsilon_1 = \frac{p_{\text{in}}}{p_{\text{out}}}$ and $\epsilon_2 = \frac{q_{\text{in}}}{q_{\text{out}}}$. For this simple JSBM, the first eigenvalues of the two Jacobian matrices are

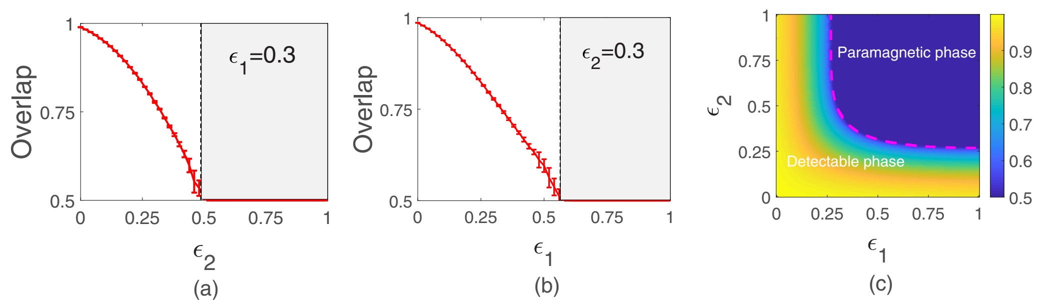

FIG. 3. Detectability phase transition of JSBM. Synthetic JSBM graphs are generated with $n = 2 \times 10^5$, $m = 2 \times 10^5$, group number $\kappa = 2$, and average degree $c_1 = c_2 = 3$. [(a), (b)] Overlap as a function of ϵ_1 or ϵ_2, with the other parameter fixed to 0.3 [(a) ϵ_1 fixed; (b) ϵ_2 fixed]. The numerical detectability phase transition point agrees with the theoretical calculation [Eq. (11)], except for small fluctuations due to the finite size effect. (c) Overlap on the $\epsilon_1 - \epsilon_2$ plane. In all three plots, dashed lines indicate the theoretical detectability phase transition point, separating the detectable phase [white in panels (a) and (b)] from the paramagnetic phase [gray in panels (a) and (b)]. Numerical and analytical results are consistent.

expressed as

$$\lambda_A = \frac{1-\epsilon_1}{1+(\kappa-1)\epsilon_1}, \qquad \lambda_F = \frac{1-\epsilon_2}{1+(\kappa-1)\epsilon_2}. \tag{14}$$

Numerical experiments are conducted to verify our theoretical results. The performance of BP is evaluated using the overlap $\mathcal{O}$ between the BP results $\{\widetilde{t_i}\}$ obtained from Eq. (6) and the ground truth $\{t_i^*\}$:

$$\mathcal{O}(\{\widetilde{t_i}\}, \{t_i^*\}) = \max_{\pi} \frac{1}{n} \sum_{i=1}^{n} \delta_{\pi(\widetilde{t_i}), t_i^*}, \tag{15}$$

where $\delta_{a,b}$ is the Kronecker function, and the overlap function is maximizing over all permutations of groups π (i.e., $|\pi| = \kappa!$). In Eq. (15), if the inferred labels $\{\widetilde{t_i}\}$ are chosen randomly, which has nothing to do with the ground truth $\{t_i^*\}$, the overlap $\mathcal{O} = 1/\kappa$ (lower bound); if there is an exact match between $\{\widetilde{t_i}\}$ and $\{t_i^*\}$, $\mathcal{O} = 1$ (upper bound). For a small number of groups, overlap is a commonly used metric for estimating the similarity between two group assignments. When the number of groups is large, or with different group sizes, more advanced measures are expected, such as the normalized mutual information (NMI) and its variants [27,28].

Results are shown in Fig. 3. In Fig. 3(a), ϵ_1 is fixed at 0.3 and ϵ_2 varies; in Fig. 3(b) it is the other way around. In both Figs. 3(a) and 3(b), the overlap between BP results and the (synthetic) ground truth are plotted against the varied ϵ (ϵ_1 or ϵ_2), which are optimal in the thermodynamic limit among using all possible algorithms. The overlap is close to 1.0 for small ϵ, indicating near-perfect reconstructions of the ground truth; with increased values of ϵ, the accuracy of BP inference decreases, which eventually downgrades to 0.5. This is consistent with the theoretical limit of the detectability phase transition Eq. (12) (indicated by dashed lines). For a large ϵ, the system goes beyond the phase transition point and lies in the paramagnetic phase, where the overlap is always 0.5, i.e., results of the BP detection are indistinguishable from that of a 50:50 random guess. In Fig. 3(c), the accuracy of BP for JSBM is shown on the ϵ_1-ϵ_2 plane, where the overlap decays from large values (yellow) to the noninformative 0.5 (blue). The dashed line represents the theoretical phase transition point, consistent with the numerical results.

V. FROM BELIEF PROPAGATION TO GRAPH CONVOLUTION NETWORK

When applying BP algorithms [Eq. (3)] to real-world graphs or synthetic graphs without *a priori* knowledge of the generative process, a critical problem is to determine the parameters θ of the underlying JSBM. Iterative schemes need to be called for to locate the optimal parameter set; for a pure clustering problem, classical approaches for this task are extensively used, such as the expectation maximization (EM) algorithm [29], in which parameters are updated through maximizing the total log-likelihood of data (i.e., minimizing the total free energy). In practice, however, the established approaches are often prone to the problem of overfitting [14] and may be trapped in local minima, which is clearly undesired in parameter estimation. A different approach could be taken to determine the parameters, however, if a small number of ground truth labels are available, as is the setting of the current study, since now the parameters could be determined in a semisupervised fashion. Such a semisupervised update of parameters could possibly be achieved through back-propagation on a neural network structure, which also facilitates the message passing of BP since it needs to be done in an iterative manner. With this idea in mind, we propose a solution framework for the JSBM which contains two steps in each epoch: In the forward step, BP equations propagate to finite time steps (layers) and conduct messages passing, and in the backward step, the parameters θ of the underlying JSBM are determined in a supervised approach, and back-propagate to BP layers.

Essentially, the graph convolution network (GCN) [1] structure is adopted in the above framework, which is an outstanding neural network model that triggers a large number of variants (see the next section). In a GCN, information on item nodes $\mathbf{X}$ propagate forward among layers; in a high-level description, the propagation takes the following form,

$$\mathbf{X}_{l+1} = \sigma(\mathbf{U}\mathbf{X}_l\mathbf{W}), \tag{16}$$

where $\sigma(\cdot)$ represents an activation function and $\mathbf{X}_l \in \mathbb{R}^{n\times\kappa_l}$ is the neural network state at layer l (κ_l denoting the dimension of network state at layer l, which is not necessarily equal to κ, the number of groups in the network). The matrix $\mathbf{W} \in$

$\mathbb{R}^{\kappa_l,\kappa_{l+1}}$ is the trainable weight matrix at the lth layer, and the propagator $\mathbf{U}$ is responsible for convolving the neighborhood of nodes, a kernel shared over the entire graph.

Based on the GCN structure, we propose the belief propagation graph convolution network (BPGCN), as our solution framework of the JSBM. Marginals and cavity messages ψ are network states; kernels take the job of the products and summations in Eqs. (3) and (5); and parameters of the JSBM $\mathbf{P}$ and $\mathbf{Q}$ become the weight matrices of the neural network, which are to be gradually learnt through back-propagation. Under these conventions, the propagation from the lth layer to the $(l+1)$-th layer on a BPGCN is formulated as

$$\begin{aligned}
\boldsymbol{\Psi}^{\mu}_{l+1} &= \mathrm{Softmax}\big[\mathbf{U_I}\log\big(\boldsymbol{\Psi}^{i\to\mu}_{l}\mathbf{P}\big)\big],\\
\boldsymbol{\Psi}^{i}_{l+1} &= \mathrm{Softmax}\big[\mathbf{U}_{II}\log\big(\boldsymbol{\Psi}^{\mu\to i}_{l}\mathbf{Q}\big)+\mathbf{U}_{III}\log\big(\boldsymbol{\Psi}^{i\to j}_{l}\mathbf{P}\big)\big],\\
\boldsymbol{\Psi}^{i\to j}_{l+1} &= \mathrm{Softmax}\big[\mathbf{B}_{I}\log\big(\boldsymbol{\Psi}^{\mu\to i}_{l}\mathbf{Q}\big)+\mathbf{B}_{II}\log\big(\boldsymbol{\Psi}^{i\to j}_{l}\mathbf{P}\big)\big],\\
\boldsymbol{\Psi}^{i\to \mu}_{l+1} &= \mathrm{Softmax}\big[\mathbf{B}_{III}\log\big(\boldsymbol{\Psi}^{i\to j}_{l}\mathbf{P}\big)+\mathbf{B}_{IV}\log\big(\boldsymbol{\Psi}^{\mu\to i}_{l}\mathbf{Q}\big)\big],\\
\boldsymbol{\Psi}^{\mu\to i}_{l+1} &= \mathrm{Softmax}[\mathbf{B}_{V}\log(\boldsymbol{\Psi}^{i\to\mu}\mathbf{Q})],
\end{aligned}\tag{17}$$

where $\mathrm{Softmax}(z_t)=\frac{e^{z_t}}{\sum_{s=1}^{\kappa}e^{z_s}}$ is used as the activation function, inherited naturally from BP, which asks to normalize marginal probabilities with κ components. If $\kappa=2$, the Softmax activation function reduces to the logistic function. Also note the logarithm on $\boldsymbol{\Psi}$ inside the Softmax, which might be considered as a part of the activation function in a strict sense. Kernel matrices $\mathbf{U_I}$ to $\mathbf{U_{III}}$, $\mathbf{B_I}$ to $\mathbf{B_V}$ are nonbacktracking matrices [30] that encode adjacency information of cavity messages and marginals (see Appendix D for details). In practice, random values are used to initialize the input layer of the network, for marginals $\boldsymbol{\Psi}^{i}_{0}\in\mathbb{R}^{n\times\kappa}$ and $\boldsymbol{\Psi}^{\mu}_{0}\in\mathbb{R}^{m\times\kappa}$, as well as messages $\boldsymbol{\Psi}^{i\to j}_{0}\in\mathbb{R}^{2M_A\times\kappa}$, $\boldsymbol{\Psi}^{i\to\mu}_{0}\in\mathbb{R}^{2M_F\times\kappa}$, and $\boldsymbol{\Psi}^{\mu\to i}_{0}\in\mathbb{R}^{2M_F\times\kappa}$, where M_A and M_F are the number of item-item edges (in the connectivity graph) and item-attribute edges (in the attribute graph), respectively.

The marginals in the last layer of BPGCN $\boldsymbol{\Psi}=\{\boldsymbol{\Psi}^{i}_{L}\}\in[0,1]^{n\times\kappa}$ are the output of a L-layer BPGCN. A loss function on the fraction of marginals belong to training labels is adopted for the supervised learning. A common choice of the loss function for classification is the cross entropy $\mathcal{L}$; in our model, it is defined as

$$\mathcal{L}=-\sum_{i\in\Omega}\sum_{s=1}^{\kappa}(\mathbf{y}_i)_s\ln\big(\Psi^{i}_{L}\big)_s,\tag{18}$$

where Ω denotes the training set of item nodes (a fraction of ground-truth nodes) and $\mathbf{y}_i=\{0,0,..,1_{\mathrm{position}(t_i^*)},...0\}$ is a one-hot vector denoting the ground-truth label of node i in the training set.

Training the BPGCN is the same as training other GCNs: In each epoch, we first do a forward pass to obtain the computed marginals in the last layer, based on which we calculate the loss function on the training set; after that, we use the back-propagation [2] algorithm to compute the gradients of the loss function with respect to elements in $\mathbf{P}$ and $\mathbf{Q}$, and then apply (stochastic) gradient descent or its variants (e.g., ADAM [31]) to update the parameters. The iteration stops when the results converge, or after a finite number of epochs. In the end, the performance of the BPGCN algorithm is evaluated by the accuracy [i.e., overlap, Eq. (15)] of label assignments on the ground-truth item nodes in the test set.

A critical difference between BPGCN and traditional BP (including the semisupervised version [15]) is that BP minimizes the (Bethe) free energy, whereas BPGCN minimizes an loss function evaluated on the training data. On JSBM synthetic graphs with matched parameters, the free energy is theoretically the best loss function to minimize; however, on graphs not generated by JSBM, minimizing the (free) energy may be prone to overfitting [14,32]. More importantly, minimizing the loss function is more flexible since the function could be formulated in different ways; for example, new (informative, or regularization) terms could be added. Notably, one implicit feature of inference on JSBM is that the classification on feature nodes (i.e., Ψ^{μ}_{L}) is left unconstrained, in both BP [Eq. (3)] and BPGCN [Eq. (17)], since the ground truth of attribute labels are often not available. Nevertheless, in specific situations, such constraints could be readily incorporated into the loss function. For example, sometimes it is reasonable to believe that the distribution of classified attribute nodes labels should be close to a uniform distribution (or Gaussian in other cases); thus, in these cases, we are able to append a term in the loss function to constrain the classification performance on features. In particular, using a one-hot vector $\mathbf{w}_\mu=\{1,1,..,1\}$ to characterize the uniform distribution, we could formulate a new loss function as

$$\mathcal{L}'=-\sum_{i\in\Omega}\sum_{s=1}^{\kappa}(\mathbf{y}_i)_s\ln\big(\Psi^{i}_{L}\big)_s-\eta\sum_{\mu\in\Omega_m}\sum_{s=1}^{\kappa}(\mathbf{w}_\mu)_s\ln\big(\Psi^{\mu}_{L}\big)_s,\tag{19}$$

where η is a damping factor and Ω_m denotes the set of all attribute nodes successfully classified. This new loss function may probably help enhance the classification performance of BPGCN on real-world networks, although it is certainly not optimal for synthetic JSBM graphs with nonzero η. It is also worth noting that the adaptive fields [Eq. (4)], which in traditional BP are contributed by nonedges and are indispensable, are not necessary in BPGCN. This is because the existence of adaptive fields helps BP equations avoid the convergence to trivial fixed points where all nodes are assigned to the same group; for BPGCN in the semisupervised classification task, as we have ground-truth group information available which rules out the existence of these unrealistic points already, the adaptive fields are not necessary.

Comparing BPGCN with canonical GCNs, two major differences emerge. First, the activation function in BPGCN (Softmax) is not chosen arbitrarily but rather determined by BP message passing equations; this is in contrast with common GCNs where the activation function may take various forms, such as ReLU, PReLU, or Tanh, but without sufficient physical or mathematical warrants. Second, there are only a few parameters to be trained in BPGCN, which are the elements of $\mathbf{P}$ and $\mathbf{Q}$, and they are shared across all layers of the neural network; therefore, the number of dimension of all layer is fixed to be group numbers κ. Although this may narrow the overall representational power of BPGCN, the problem of overfitting could nevertheless be obviated to a great extent. Indeed, in semisupervised classification, the amount of training data is often much less than that of (completely) supervised classification, whereas the number of observations, in the case of networks, may be proportional to the number of edges; therefore, the sharing of parameters across layers is believed to be extremely helpful in preventing overfit of the training data.

VI. COMPARING BPGCN WITH OTHER GRAPH CONVOLUTION NETWORKS

In this section, we compare the classification performance of BP and BPGCN, constructed on the joint stochastic block model, with several state-of-the-art graph convolution networks, including the (standard) GCN, the approximate personalized propagation of neural predictions (APPNP), the graph attention network (GAT), and the simplified graph convolution network (SGCN).

A. (Standard) graph convolution network (GCN)

The standard GCN [1] is probably the most famous graph convolution network, which drastically outperformed all non-neural-network algorithms when first proposed in 2017. The forward propagation rule of standard GCN is formulated as

$$\mathbf{H}^{(l+1)} = \mathrm{ReLU}(\widetilde{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{20}$$

where $\mathbf{H}^{(l)}$ and $\mathbf{W}^{(l)}$ are states of hidden variables and weight matrices at the lth layer. $\widetilde{\mathbf{A}}$ is the graph convolution kernel, defined as

$$\widetilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}. \tag{21}$$

$\mathbf{A}$ is the connectivity matrix and $\mathbf{D}$ is the diagonal degree matrix with $D_{ii} = 1 + \sum_k A_{ik}$. The propagation rule of the standard GCN was motivated by a first-order approximation of localized spectral filters (see Appendix C). A two-layer standard GCN is written as

$$\mathbf{Z} = \mathrm{Softmax}[\widetilde{\mathbf{A}}\mathrm{Relu}(\widetilde{\mathbf{A}}\mathbf{F}\mathbf{W}^0)\mathbf{W}^1], \tag{22}$$

where $\mathbf{F}$ is the just the adjacency matrix representing attribute graph and $\mathbf{Z} \in \mathbb{R}^{n\times\kappa}$ is the classification result.

B. Approximate personalized propagation of neural predictions (APPNP)

APPNP [3] extracts attribute information (encoded in $\mathbf{F}$) to hidden neuron states $\mathbf{H}$ using a multilayer perceptron (MLP):

$$\mathbf{H} = \mathbf{Z}^{(1)} = \mathrm{MLP}(\mathbf{F}). \tag{23}$$

Similar to the GCN, the hidden states $\mathbf{H}$ are propagated via the personalized PageRank scheme to produce predictions of node labels:

$$\begin{aligned}\mathbf{Z}^{(k+1)} &= (1-\omega)\widetilde{\mathbf{A}}\mathbf{Z}^{(k)} + \omega\mathbf{H},\\ \mathbf{Z} &= \mathrm{Softmax}[(1-\omega)\widetilde{\mathbf{A}}\mathbf{Z}^{(K-1)} + \omega\mathbf{H}],\end{aligned} \tag{24}$$

where $\mathbf{Z}$ is the output probability ($\mathbf{Z} \leftarrow \mathbf{Z_K}$, where the subscript K indicates the last layer) of item nodes used to construct training loss and predict labels, K is the layer of APPNP, and $\omega \in [0, 1]$ is a weight factor. In the recent benchmarking study on the performance of various GCNs, APPNP produces the best results on multiple datasets [33].

C. Graph attention network (GAT)

GAT [4] adopts the *attention mechanism* [34] in which attention coefficients between pairs of connected nodes are regarded as weights. This scheme can be viewed as based on the adjacency matrix of the graph, but with adjustable weights on edges.

D. Simplified graph convolution networks (SGCN)

SGCN [35] tries to remove redundant and unnecessary computations from GCN by adopting a low-pass filter followed by a linear classifier. As a result, SGCN takes a simplified propagation rule using the kth power of the adjacency matrix $\widetilde{\mathbf{A}}$, as in GCN:

$$\mathbf{Z} = \mathrm{Softmax}(\widetilde{\mathbf{A}}^k\mathbf{F}\mathbf{W}), \tag{25}$$

where $\mathbf{W}$ and $\mathbf{Z}$ are weights and classification results, respectively. Empirical results show that the simplification process may yield positive impacts on the accuracy of GCN and could dramatically speed up the computation.

E. Results on synthetic networks

First, we compare these GCN algorithms on synthetic networks generated by the JSBM. Each JSBM graph consists of $n = 10\,000$ item nodes, $m = 10\,000$ attribute nodes; each item node has on average c_1 neighbors in the connectivity graph and c_2 neighbors in the attribute graph; each attribute node is connected to on average c_3 item nodes. In the semisupervised experiments, 5% of randomly chosen item nodes with ground-truth labels are used as the training set, and another 5% of item nodes are used as the validation set. The rest belong to the test set, on which the classification performance is evaluated by the overlap measure [Eq. (15)]. For BP and BPGCN, diagonal matrices are used as the initial values for $\mathbf{P}$ and $\mathbf{Q}$, with p_{in} (and q_{in}) on the diagonal and p_{out} (and q_{out}) on the off-diagonal entries. Define ratio $\epsilon_1 = p_{\mathrm{out}}/p_{\mathrm{in}}$ and $\epsilon_2 = q_{\mathrm{out}}/q_{\mathrm{in}}$, both of which could be viewed as the (inverse) signal-to-noise ratio; for all synthetic networks, we use $\epsilon_1 = \epsilon_2 = 0.5$ as the initial values for BPGCN, while for BP we use the initial values that are used to generate the underlying graphs, which guarantees asymptotically optimal results. The initial values of ϵ_1 and ϵ_2 for BPGCN could be instead determined in a soft manner by the validation set; however, tests show that this treatment does not make much difference. Also, all experiments including below real networks are set with known χ (i.e., the ground truth group numbers), since we have semisupervised information. When χ is unknown (i.e., a pure clustering task), we can determine the χ by optimizing the free energy with respect to χ, as discussed in Refs. [14,15].

Extensive experiments have been carried out on large synthetic graphs with varying average degrees $c_{1,2,3}$ and varying $\epsilon_{1,2}$ (Fig. 4; see also Appendix E). First, it is confirmed that BPGCN results are perfectly aligned with the (asymptotically) optimal BP results, better than all other GCNs. In Fig. 4(a), c_1 is fixed to 4, and c_2 increases from 3 to 20. We can see that all other reference algorithms (GCN, APPNP, GAT, and SGCN) work much worse than BPGCN, even when c_2 is quite large. In Fig. 4(b), c_2 is fixed to 4 and c_1 is varied. Similar

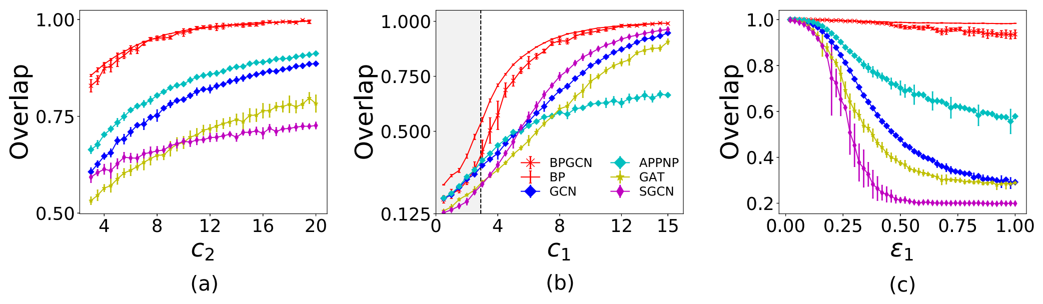

FIG. 4. Classification performance of BP, BPGCN, and other GCNs on synthetic networks generated by JSBM ($n = 10\,000$, $m = 10\,000$). All synthetic nodes have ground-truth group labels; 5% randomly selected item nodes are used as the training set, another 5% used as the validation set, and the rest are the test set, on which overlap [Eq. (15)] is evaluated to indicate different algorithms' classification performance. Each data point in the figure is averaged over 10 instances. (a) $\epsilon_1 = 0.1$, $\epsilon_2 = 0.2$, $\kappa = 5$, $c_1 = 4$; c_2 varies; (b) $\epsilon_1 = 0.1$, $\epsilon_2 = 0.2$, $\kappa = 8$ $c_2 = 4$; c_1 varies; (c) $c_1 = c_2 = 10$, $\kappa = 5$, ϵ_2 is fixed to 0.2, and ϵ_1 ranges from 0 to 1. Dashed line in panel (b) marks the theoretical detectability threshold, separating the detectable phase (white) with the undetectable and/or paramagnetic phase (gray), as in Fig. 3.

to Fig. 4(a), it shows that when c_1 is small, GCN, APPNP, GAT, and SGCN perform far worse than BP and BPGCN. These results on synthetic graphs show clearly the superiority of BPGCN over existing popular GCN algorithms. The poor performance of these reference GCN algorithms is due to the *sparsity* of the graphs (see below). In Fig. 4(b), we mark the detectability threshold [Eq. (12)] by the vertical dashed line, as in Fig. 3, which is located at $c_1 \approx 2.9$. Therefore, in the $c_1 < 2.9$ regime, we almost always cannot detect the graph's community structures without extra information. Notably, there is a very narrow regime when c_1 is slightly smaller than 2.9, that searching over different initial values fed in the algorithms will locate the nontrivial fix point and hence the communities are detectable; yet this search requires brute force and is essentially exponential time. With additional information provided, such as a few more node labels, the threshold (dashed line) gradually moves to the left, toward the y axis, and the narrow unstable regime gradually disappears; i.e., the whole regime is detectable. For these discussions, we refer to Ref. [15] for more details. Nevertheless, even with additional information, any GCN algorithm will not work better than BP with correct model parameters, since BP yields the optimal solutions on these JSBM graphs.

In Fig. 4(c), ϵ_2 is fixed to 0.1; the average degrees of the synthetic graphs are fixed at $c_1 = 10$ and $c_2 = 10$; ϵ_1 is varied. Quite surprisingly, results show that BPGCN works perfectly in the whole range of ϵ_1, even when ϵ_1 is close to 1; i.e., the $\mathbf{P}$ matrix is homogeneous, whereas conventional GCNs quickly fail when ϵ_1 increases. This phenomenon suggests that these reference GCNs have difficulties in extracting the information on group labels from attributes when the group structures of the graph are noisy, as one would imagine; in contrast, to an extraordinary extent, BPGCN is not subject to this failure. Further checks on the outputs of conventional GCNs reveal that these results all have good overlap in the training set (see Appendix E), but poor overlap in the test set, even worse than a MLP, which indicates a clear *overfitting* to training labels.

Per the above discussion, in what follows, we analyze the two issues emerged in the results (the *sparsity* issue and the *overfitting issue*) and explain how they could be successfully overcome by BPGCN.

(1) The sparsity issue. Properties of the forward-propagation of GCNs are closely related to their linear convolution kernels, and hence the reason to account for the sparsity issue can be uncovered by studying the spectrum of the linear kernels used in different graph convolutions. In GCN, SGCN, and APPNP, the convolution kernel is a variant of the normalized adjacency matrix $\widetilde{\mathbf{A}}$ [Eq. (21)]. It has been established in, e.g., Refs. [30,36] that this type of linear operators have localization problems on large sparse graphs, with leading eigenvectors only encoding local rather than global information on group structures (see Appendix C). By contrast, our BPGCN is immune to the sparsity issue, because the convolution kernels of BPGCN are the nonbacktracking matrices, which naturally overcome the localization problem on large sparse graphs [30]. Therefore, inspired by BPGCN, a straightforward approach to overcome the sparsity issue in classic GCNs might be to consider using a linear kernel that does not trigger the localization problem in sparse graphs, such as the nonbacktracking matrix or the X-Laplacian [36].

(2) The overfitting issue. From Eqs. (20), (24), and (25), one could observe that in these models the linear filters always operate directly on the weight matrices or on the hidden states. This implies an underlying assumption of conventional GCNs: The relational data (i.e., edges in the connectivity graph) must always contain information on item nodes' group structures (labels). This is a natural assumption, yet may not always be true. In an advanced manner, BPGCN relaxes this assumption by learning affinity matrices $\mathbf{P}$ (and $\mathbf{Q}$) that store the learned signal-to-noise ratios ϵ_1 (and ϵ_2), which indicate the relative distribution of edges within versus outside planted communities. Hence, in this case it is not necessary that group information is directly encoded in edge connectivities.

F. Results on real-world data sets

Next we compare BPGCN and reference algorithms on several well-known real-world networks with and without node attributes. The Karate Club network [37] and Political Blogs network [38] are classical network data sets with planted community structures. Since in these two networks there are no node attributes, canonical GCNs commonly use the identity matrix as the feature matrix $\mathbf{F}$ [1]. Given their small sizes, for the Karate Club network we use two item nodes per group as training nodes and two item nodes per group as validation nodes, and for the Political Blogs network

TABLE I. Classification performance of BPGCN and reference GCNs on real-world networks. c denotes the average degree of the connectivity graph. For reference GCNs, results on Karate Club and Political Blogs are carried out using publicly available implementations of these algorithms; results on Cora, Pubmed, and Citeseer are adapted from Ref. [33]. For BPGCN, the reported accuracy values are based on the vanilla version. The preprocessed version of BPGCN greatly enhances the classification accuracy on the Pubmed network, up to 81.7% (see text).

	Karate	Political Blogs	Citeseer	Cora	Pubmed
No. nodes	34	1490	3327	2078	19 717
No. features	0	0	3703	1433	500
No. groups	2	2	6	7	3
No. training	4	20	120	140	60
No. validations	4	20	500	500	500
No. tests	26	1450	1000	1000	1000
c	4.6	22.4	2.78	3.89	4.49
MLP [1]			58.4	52.2	72.7
GCN [1]	**96.5**	86.2	71.1	81.5	79.0
GAT [4]	87.9	88.7	70.8	83.1	78.5
SGCN [35]	91.6	81.3	71.3	81.7	78.9
APPNP [3]	96.3	87.5	**71.8**	**83.5**	**80.1**
BPGCN	95.8	**89.3**	71.1	82.1	70.0

we use 10 item nodes per group for training and 10 item nodes per group for validation. The Citeseer, Cora, and Pubmed networks are standard data sets for semisupervised classification widely used in graph convolution network studies; we follow the splitting rule of training, test, and validation sets introduced in Ref. [1] on these graphs. For all these networks, there are ground-truth labels for item nodes' group membership. Same as the case for synthetic data, the performance of tested algorithms is evaluated by the overlap between classification results and the ground-truth labels.

For BPGCN, a tunable external field is adopted and corresponding hyper parameters γ are introduced, so as to adjust the relative strength of the training labels on nodes (see Appendix D for details). Identical to the case of synthetic graphs, diagonal matrices are used as the initial values for $\mathbf{P}$ and $\mathbf{Q}$, controlled by the two signal-to-noise ratios $\epsilon_1 = p_{\rm out}/p_{\rm in}$ and $\epsilon_2 = q_{\rm out}/q_{\rm in}$. Unlike on synthetic graphs where initial ϵ_1 and ϵ_2 are fixed at 0.5, on real-world graphs we did a coarse search using the validation set to determine proper values for ϵ_1 and ϵ_2 during the preprocessing step, altogether with the search for proper hyper parameters, including the strength of the external field strength γ and the number of layers of BPGCN L (see Appendix D for details).

Classification results are shown in Table I. On the two networks without node attributes, all tested GCN algorithms perform quite well, and label information is successfully extracted from edge connectivities. BPGCN outperforms other GCNs for the Political Blogs network; the good performance may come from its nonbacktracking convolution kernel, which has good spectral properties on large sparse graphs such as the Political Blogs network [30]. We notice that in Ref. [15], parameters $\mathbf{P}$ are updated using the expectation maximization (EM) algorithm, where the accuracy of detection for the Political Blogs network is worse that our result as shown in Table I. This is because with fewer known labels, the EM algorithm often converges to core-periphery structure where nodes are divided according to high or low degrees. On networks with node attributes, the evaluation of classification performance is less straightforward. On Citeseer and Cora, the performance of BPGCN is certainly comparable to other GCNs; in both cases it is superior to the performance of at least one reference algorithm. Nevertheless, BPGCN works poorly on the Pubmed network, yielding the worse performance among all tested GCNs. The reason is that the Pubmed network contains 19 717 item nodes, but only 500 attributes; when attributes are densely connected to item nodes, the attribute graph significantly deviates from being a sparse random graph, a critical assumption our BPGCN algorithm and the underlying JSBM rely on. Indeed, it is verified that, when completely ignoring attributes and only using edge connectivities in classification, BPGCN yields a classification accuracy at 71.0%, better than 70.0%. If we try to increase the number of group labels of attribute nodes, unfortunately it does not work. Since the multilayer percetron (MLP) method, which only uses the attribute information, already achieves a high accuracy, a possible remedy for BPGCN in situations where the attribute graph is far from having a locally treelike topology is to use the classification results yielded by MLP in place of the original attribute graph, as the external field acting on BPGCN (i.e., adopting MLP as a preclassification step). Tests confirm that this preprocessed version of BPGCN greatly enhances the classification accuracy on the Pubmed network, up to 81.7%, which is significantly better than the state-of-the-art results yielded by APPNP. This preprocessing step does not improve the results on Citeseer and Cora, as expected, since the attribute graphs are sparse of these two networks.

VII. CONCLUSIONS

In this study, we constructed the joint stochastic block model (JSBM) that generates a twofold graph which simultaneously models item nodes and attribute nodes in an aggregated network setting, based on the celebrated SBM. Utilizing the cavity method in statistical physics and the corresponding belief propagation (BP) algorithm which is asymptotically exact in the thermodynamic limit, theoretical results on the detectability phase transition point and the phase diagram for JSBM are uncovered. Expectedly, JSBM could be used to generate benchmark networks with continuously tunable parameters, which might be particularly useful in evaluating the classification performance of graph convolution neural networks.

Based on the BP equations established on JSBM, we proposed an algorithm for semisupervised classification, adopting the graph convolution network structure, which we termed as BPGCN. In contrast to most existing graph convolution networks, the convolution kernel and activation function of BPGCN are determined mathematically from the Bayesian inference on the JSBM. We show that on synthetic networks generated by JSBM, BPGCN clearly outperforms several well-known existing graph convolution networks, and obtains extraordinary classification accuracy in the parameter regime where conventional GCNs fail to work; on real-world networks, BP also displays comparable performance to state-of-the-art GCNs. Compared with conventional GCNs, BPGCN is quite powerful in extracting label information from edge connectivities; this advantage is rooted in its nonbacktracking convolution kernel inherited from the BP algorithm. The weakness of BPGCN is demonstrated by its performance on the Pubmed dataset, in which case there are too few attributes for the attribute graph to be approximated by a SBM; a remedy for applying BPGCN on such graphs is proposed, which uses MLP as a preprocessing step, and the corresponding advanced version of BPGCN is implemented and tested. Based on the fact that BPGCN is immune to the sparsity issue and the overfitting issue exposed by conventional GCNs, we discussed possible ways inspired by BPGCN to improve current GCN techniques. It would be interesting to combine successful features of BPGCN and state-of-the-art GCNs in greater depth and design new architectures for graph convolution neural networks; we leave this idea for future work.

PYTORCH and C++ implementations of our BPGCN and other GCNs on JSBM together with the real-world data sets used in our experiments are available at Ref. [39].

ACKNOWLEDGMENTS

Useful discussions with Yifan Qu and Haijun Zhou are acknowledged. This work is supported by Project No. QYZDB-SSW-SYS032 of Chinese Academy of Sciences and the National Natural Science Foundation of China under Grants No. 11947302 and No. 11975294. Part of the computation was carried out at the HPC cluster of ITP, CAS.

APPENDIX A: BELIEF PROPAGATION EQUATIONS

Combine the Boltzmann distribution Eq. (1), the likelihood function Eq. (2), and the variational distribution Eq. (7), by minimizing the Bethe free energy with respect to constraints subject to the normalizations of marginals, and one arrives at the standard form of belief propagation equations [13] for JSBM:

$$\begin{aligned}
\psi_{t_i}^{i\to j} &= \frac{\alpha_{t_i}}{Z^{i\to j}} \prod_{k\in\partial i\setminus j}\sum_{t_k} p_{t_it_k}\psi_{t_k}^{k\to i} \prod_{k\notin\partial i\setminus j}\sum_{t_k}(1-p_{t_it_k})\psi_{t_k}^{k\to i}\cdot \prod_{\mu\in\partial i}\sum_{t_\mu} q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i}\prod_{\mu\notin\partial i}\sum_{t_\mu}(1-q_{t_it_\mu})\psi_{t_\mu}^{\mu\to i},\\
\psi_{t_i}^{i\to \mu} &= \frac{\alpha_{t_i}}{Z^{i\to \mu}} \prod_{k\in\partial i}\sum_{t_k} p_{t_it_k}\psi_{t_k}^{k\to i} \prod_{k\notin\partial i}\sum_{t_k}(1-p_{t_it_k})\psi_{t_k}^{k\to i}\cdot \prod_{\nu\in\partial i\setminus\mu}\sum_{t_\nu} q_{t_it_\nu}\psi_{t_\nu}^{\nu\to i}\prod_{\nu\notin\partial i\setminus\mu}\sum_{t_\nu}(1-q_{t_it_\nu})\psi_{t_\nu}^{\nu\to i},\\
\psi_{t_\mu}^{\mu\to i} &= \frac{\beta_{t_\mu}}{Z^{\mu\to i}} \prod_{j\notin\partial\mu\setminus i}\sum_{t_j} q_{t_\mu t_j}\psi_{t_j}^{j\to\mu} \prod_{j\in\partial\mu\setminus i}\sum_{t_j}(1-q_{t_\mu t_j})\psi_{t_j}^{j\to\mu},
\end{aligned} \tag{A1}$$

where $\psi_{t_i}^{i\to j}$, $\psi_{t_i}^{i\to\mu}$, and $\psi_{t_\mu}^{\mu\to i}$ are cavity probabilities. Marginal probabilities are estimated as a function of cavity probabilities as

$$\begin{aligned}
\psi_{t_i}^{i} &= \frac{\alpha_{t_i}}{Z^{i\to j}} \prod_{k\in\partial i}\sum_{t_k} p_{t_it_k}\psi_{t_k}^{k\to i} \prod_{k\notin\partial i}\sum_{t_k}(1-p_{t_it_k})\psi_{t_k}^{k\to i}\\
&\quad\times \prod_{\mu\in\partial i}\sum_{t_\mu} q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i}\prod_{\mu\notin\partial i}\sum_{t_\mu}(1-q_{t_it_\mu})\psi_{t_\mu}^{\mu\to i},\\
\psi_{t_\mu}^{\mu} &= \frac{\beta_{t_\mu}}{Z^{\mu\to i}} \prod_{j\notin\partial\mu}\sum_{t_j} q_{t_\mu t_j}\psi_{t_j}^{j\to\mu} \prod_{j\in\partial\mu}\sum_{t_j}(1-q_{t_\mu t_j})\psi_{t_j}^{j\to\mu}.
\end{aligned} \tag{A2}$$

Since we have nonzero interactions between every pair of nodes, in Eq. (A1) we have in total $n(n-1)+2mn$ messages. This results to an algorithm where even a single update takes $O(n^2)$ time, making it suitable only for networks of up to a few thousand nodes. Fortunately, for large sparse networks, i.e., when n, m is large, $p_{t_it_j}=O(1/n)$ and $q_{t_it_\mu}=O(1/n)$, we can neglect terms of subleading order in the equations. In this case, it is assumed that i or μ sends the same message to all its non-neighbors j, and these messages are viewed as representing an external field. By this means, now we only need to keep track of $2M$ messages, where M is the total number of all edges, and each update step takes $O(n+m)$ time.

Suppose that $j\notin\partial i$, we have

$$\begin{aligned}
\psi_{t_i}^{i\to j} &= \frac{\alpha_{t_i}}{Z^{i\to j}} \prod_{k\in\partial i}\sum_{t_k} p_{t_it_k}\psi_{t_k}^{k\to i} \prod_{k\notin\partial i\setminus j}\left(1-\sum_{t_k}p_{t_it_k}\psi_{t_k}^{k\to i}\right)\\
&\quad\times \prod_{\mu\in\partial i}\sum_{t_\mu} q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i}\prod_{\mu\notin\partial i}\sum_{t_\mu}(1-q_{t_it_\mu})\psi_{t_\mu}^{\mu\to i}\\
&= \psi_{t_i}^{i}+O(1/n).
\end{aligned} \tag{A3}$$

Similarly, we have $\psi_{t_\mu}^{\mu\to i}=\psi_{t_\mu}^{\mu}+O(1/m)$. To the leading order, the messages on nonedges do not depend on the target node. For nodes with $j\in\partial i$, we have

$$\begin{aligned}
\psi_{t_i}^{i\to j} &= \frac{\alpha_{t_i}}{Z^{i\to j}} \prod_{k\in\partial i\setminus j}\sum_{t_k} p_{t_it_k}\psi_{t_k}^{k\to i} \prod_{k\notin\partial i}\left(1-\sum_{t_k}p_{t_it_k}\psi_{t_k}^{k\to i}\right)\\
&\quad\cdot \prod_{\mu\in\partial i}\sum_{t_\mu} q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i}\prod_{\mu\notin\partial i}\left(1-\sum_{t_\mu}q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i}\right)\\
&\simeq \alpha_{t_i}\frac{e^{-h_{t_i}}}{Z^{i\to j}} \prod_{k\in\partial i\setminus j}\sum_{t_k} p_{t_it_k}\psi_{t_k}^{k\to i}\prod_{\mu\in\partial i}\sum_{t_\mu} q_{t_it_\mu}\psi_{t_\mu}^{\mu\to i},
\end{aligned} \tag{A4}$$

where terms having $O(1/n)$ and $O(1/m)$ contributions to $\psi_{t_i}^{i\to j}$ combine to the definition of the auxiliary external field:

$$h_{t_i} = \sum_k\sum_{t_k} p_{t_it_k}\psi_{t_k}^{k} + \sum_\mu\sum_{t_\mu} q_{t_it_\mu}\psi_{t_\mu}^{\mu}. \tag{A5}$$

Applying the same approximations to the external fields acting on attribute nodes, one finally arrives at BP equations (3).

APPENDIX B: DETECTABILITY TRANSITION ANALYSIS USING NOISE PERTURBATIONS

On a graph generated by the JSBM with n item nodes and m attribute nodes, the probability that an item node has k_1 neighboring item nodes $p_1(k_1)$ follows a Poisson distribution with average degree c_1, and the probability of it being associated with k_2 attribute nodes $p_2(k_2)$ follows a Poisson distribution with average degree c_2. Similarly, the probability that an attribute node has k_3 neighboring item nodes $p_3(k_3)$ follows a Poisson distribution with average degree $c_3 = nc_2/m$. Considering a branching process on a graph generated by JSBM with infinite size, the average branching ratio of the process is related to the excess degree which is defined upon the average number of neighbors. The average excess degree of an item node is computed as

$$\widetilde{c}_1 = \frac{\sum_{k_1} k_1(k_1-1)p_1(k_1)}{\sum_{k_1} k_1 p_1(k_1)} = c_1. \tag{B1}$$

Similarly we have $\widetilde{c}_2 = c_2$ as well, and the excess degree of attribute nodes is

$$\widetilde{c}_3 = \frac{\sum_{k_2} p_2(k_2)k_2(k_2-1)n/m}{\sum_{k_2} k_2 p_2(k_2)} = c_2 n/m = c_3. \tag{B2}$$

Consider the noise propagation process on a tree graph as depicted in Fig. 2, with depth $l \to \infty$. In the tree, odd layers contain exclusively item nodes, and even layers contain attribute nodes (blue boxes) and item-item edges (green boxes) which are used to connect item nodes in odd layers. Assume that on the leaves of the tree (nodes on the lth layer) the paramagnetic fixed point is perturbed as

$$\psi_{t_{v_l}}^{v_l \to v_{l-1}} = \alpha_{t_{v_l}} + \eta_{t_{v_l}}^{v_l \to v_{l-1}}, \tag{B3}$$

where v_l represent an item node on the l layer, t_{v_l} is label of v_l, $\eta_{t_{v_l}}^{v_l \to v_{l-1}}$ is the perturbation, v_0 corresponds to the root node i in Fig. 2. Now let us investigate the influence of the perturbation, from the message on any leaf, to the message on the root node. For simplicity, first choose one path only containing item nodes (i.e., not connected via any feature node) and latter generalize to paths containing both item nodes and attribute nodes. We define the Jacobian matrix $\mathbf{T}^{i\to j}$ with respect to the message passing from an item node i to another item node j along an edge (ij), and the Jacobian matrix $\mathbf{T}^{\mu\to i}$ corresponding to the message passing from an attribute node μ to an item node i, and similarly the matrix $\mathbf{T}^{i\to\mu}$ for the backward passage. Elements of these Jacobian matrices are formulated as

$$T_{t_i t_k}^{i\to j} = \frac{\partial \psi_{t_j}^{j\to k}}{\partial \psi_{t_i}^{i\to j}}\bigg|_{\alpha_{t_i}} = \alpha_{t_i}\left(\frac{np_{t_i t_k}}{c_1} - 1\right),$$

$$T_{t_i t_\mu}^{\mu\to i} = \frac{\partial \psi_{t_i}^{i\to j}}{\partial \psi_{t_\mu}^{\mu\to i}}\bigg|_{\alpha_{t_i},\beta_{t_\mu}} = \alpha_{t_i}\left(\frac{mq_{t_\mu t_i}}{c_2} - 1\right),$$

$$T_{t_i t_\mu}^{i\to\mu} = \frac{\partial \psi_{t_\mu}^{\mu\to j}}{\partial \psi_{t_i}^{i\to\mu}}\bigg|_{\alpha_{t_i},\beta_{t_\mu}} = \beta_{t_\mu}\left(\frac{mq_{t_i t_\mu}}{c_2} - 1\right). \tag{B4}$$

These matrices represent propagation strength (3) between two messages in the vicinity of the paramagnetic fixed point. We can see that all three matrices are independent of node indices, only depending on the type of the nodes. If the path only contains item nodes, the perturbation $\eta_{t_{v_0}}^{v_0\to x}$ on the root node induced by the perturbation $\eta_{t_{v_l}}^{v_l\to v_{l-1}}$ on the leaf node can be written as

$$\eta_{t_{v_0}}^{v_0\to x} = \sum_{\{t_{v_d}:d=1,\dots,l\}} \prod_{d=0}^{l-1} T_{t_{v_d},t_{v_{d+1}}}^{i\to j}\, \eta_{t_{v_l}}^{v_l\to v_{l-1}}, \tag{B5}$$

or in the vector form $\eta^{v_0\to x} = (\mathbf{T}^{i\to j})^l \eta^{v_l\to v_{l-1}}$. Now consider the path contains both item nodes and attribute nodes: Every time an attribute node is passed through, $\mathbf{T}^{i\to\mu}$ and $\mathbf{T}^{\mu\to i}$ transmits to $\mathbf{T}^{i\to j}$; therefore, the total weight acting on the path is

$$\eta^{v_0\to x} = (\mathbf{T}^{i\to j})^s(\mathbf{T}^{i\to\mu}\mathbf{T}^{\mu\to i})^{(l-s)}\eta^{v_l\to v_{l-1}},$$

where s is the number of edges and $l-s$ is the number of attribute nodes on this path.

For $l\to\infty$, $(\mathbf{T}^{i\to j})^s(\mathbf{T}^{i\to\mu}\mathbf{T}^{\mu\to i})^{(l-s)}$ is dominated by the product of the largest eigenvalues, λ_A, λ_F, and $\lambda_{F'}$ of the three matrices; notice that $\lambda_F = \lambda_{F'}$. So the above equation can be written as

$$\eta^{v_0\to x} = (\lambda_A)^s(\lambda_F)^{2(l-s)}\eta^{v_l\to v_{l-1}}.$$

Then consider the collection of all perturbations on the root node v_0 from all leaves. Obviously the mean is 0, and the variance on the tth component of the perturbation vector is computed as

$$\begin{aligned}\left\langle\left(\eta_t^{v_0\to x}\right)^2\right\rangle &= \left\langle\left(\sum_{s=0}^{l}\sum_{1}^{\binom{l}{s}}\sum_{v_l=0}^{(c1)^s(c2c_3)^{l-s}} {\lambda_A}^s {\lambda_F}^{2(l-s)}\eta^{v_l\to v_{l-1}}\right)^2\right\rangle\\ &= \sum_{s=0}^{l}\sum_{1}^{\binom{l}{s}}\sum_{v_l=0}^{(c1)^s(c2c_3)^{l-s}} {\lambda_A}^{2s}{\lambda_F}^{4(l-s)}\left\langle\left(\eta_t^{v_l\to v_{l-1}}\right)^2\right\rangle\\ &= \sum_{s=0}^{l}\binom{l}{s}\left(c_1{\lambda_A}^2\right)^s\left(c_2c_3{\lambda_F}^4\right)^{l-s}\left\langle\left(\eta_t^{v_l\to v_{l-1}}\right)^2\right\rangle\\ &= \left(c_1{\lambda_A}^2 + c_2c_3{\lambda_F}^4\right)^l\left\langle\left(\eta_t^{v_l\to v_{l-1}}\right)^2\right\rangle.\end{aligned} \tag{B6}$$

Here, we have made use of the property that all perturbations on different leaves are independent. Obviously, the paramagnetic fixed point is locally unstable under random perturbations when $(c_1{\lambda_A}^2 + c_2c_3{\lambda_F}^4)^l > 1$, so the phase transition of detectablity locates at

$$c_1{\lambda_A}^2 + c_2c_3{\lambda_F}^4 = 1. \tag{B7}$$

APPENDIX C: GRAPH CONVOLUTION NETWORKS AND THE SPECTRAL LOCALIZATION PROBLEM OF LINEAR CONVOLUTION KERNELS

Convolution networks [40] have been proven to be one of the most successful models for image classifications [41] and many other machine learning problems [2]. The success of CNN is credited to the convolution kernel, which is a linear operator defined on gridlike structures. However, in recent years, a significant amount of attention has been paid to generalizing convolutional operations to graphs, which do not have grid structures.

In Ref. [42], authors propose to define convolutional layers on graphs that operate on the spectrum of the graph Laplacian:

$$\mathbf{X}_{l+1} = \sigma(\Lambda \star \mathbf{X}_l) = \sigma(\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\mathbf{X}_l), \tag{C1}$$

where $\mathbf{X}_l$ is the state in the lth layer, $\mathbf{V}$ is the eigenvector matrix of the graph Laplacian $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ ($\mathbf{A}$ is adjacency matrix and $\mathbf{D}$ is the diagonal matrix on node degrees), $\sigma(\cdot)$ is an elementwise nonlinear activation function, and $\mathbf{\Lambda}$ is a diagonal matrix representing a kernel in the frequency (graph Fourier) domain. Usually $\mathbf{\Lambda}$ contains only a few nonzero elements, working as cutoffs to the frequency in the Fourier domain, because it is believed that using only a few eigenvectors of the graph Laplacian are sufficient for describing smooth structures of the graph. Only computing leading eigenvectors is also computationally efficient.

Even though computing only a few eigenvectors of the Laplacian of large graphs could be quite cumbersome, in Ref. [43] authors proposed to parametrize the kernel in the frequency domain through a truncated series expansion, using the Chebyshev polynomials up to Kth order:

$$\mathbf{X}_{l+1} = \sigma(\Lambda \star \mathbf{X}_l) \approx \sigma\left[\sum_{k=0}^{K} c_k T_k\left(\frac{2}{\lambda_{\max}}\mathbf{L} - \mathbf{I}\right)\mathbf{X}_l\right], \tag{C2}$$

where c_k denotes coefficients, $T_K(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ are recursively defined Chebyshev polynomials, and $\lambda_{\max}$ is the largest eigenvalue. In Ref. [1], the authors limited the convolution operation to $K = 1$, and approximate $\lambda_{\max} = 2$, and then obtain

$$\mathbf{X}_{l+1} = \sigma(\mathbf{\Lambda} \star \mathbf{X}_l) \approx \theta_1 X - \theta_2 D^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}},$$

with two free parameters θ_1 and θ_2. Further, Ref. [1] restricted the two parameters by specifying $\theta = \theta_1 = -\theta_2$ and introduced a normalization trick

$$\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} \rightarrow \widehat{\mathbf{D}}^{-\frac{1}{2}}\widehat{\mathbf{A}}\widehat{\mathbf{D}}^{-\frac{1}{2}},$$

where $\widehat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\widehat{D}_i = 1 + \sum_j A_{ij}$. Finally, upon generalizing to node features of κ components, that is, to the κ-channel signal $\mathbf{X} \in \mathbb{R}^{n\times\kappa}$, one arrives at the GCN [1], with the forward model taking the form of

$$\mathbf{X}_{l+1} = \sigma(\widetilde{\mathbf{A}}\mathbf{X}_l\mathbf{W}), \tag{C3}$$

where $\sigma(\cdot)$ is an activation function such as ReLU and $\mathbf{X}_l \in \mathbb{R}^{n\times\kappa_l}$ is the network state at layer l (n denotes the number of nodes in the graph and κ_l is the dimension of state at layer l). Matrix $\mathbf{W} \in \mathbb{R}^{\kappa_{l+1},\kappa_l}$ is the trainable weight matrix at the lth layer, and the propagator $\widetilde{\mathbf{A}}$ (21) is responsible for convolving the neighborhood of a node, which is shared over the whole network.

It has been shown in Ref. [1] that GCN significantly outperforms related standard methods, including manifold regularization (ManiReg) [44], semisupervised embedding (SemiEmb) [45], label propagation (LP) [46], skip-gram-based graph embedding (DeepWalk) [47], the iterative classification algorithm in conjunction with two classifiers taking care of both local node features and aggregations [48], and the recently proposed Planetoid [45].

Of the many developments of GCNs in recent years, almost all of them can be understood as an effective object representation starting from the original feature vector and then projected forward by multiplying finite times of the linear convolution kernel. So it is recognized that the main principle of graph convolution kernels is inspired by the spectral properties of linear operators, such as Laplacians, normalized Laplacians [1,35], random walk matrix [3], etc. The underlying assumption is that the eigevectors of the graph convolution kernel contain global information about group labels, which can be revealed during the forward propagation of GCNs.

However, it is known that this assumption may not hold on large sparse networks, because the eigenvectors of conventional linear operators such as graph Laplacians are subject to the localization problem, induced by the fluctuation of degrees or the local structures of the graph [30,36]. Even on random graphs where the Poisson degree distribution is rather concentrated, the localization problem is still significant on large graphs. For the adjacency matrix, it is well known that the largest eigenvalue is bounded below by the squared root of the largest node degree (which grows as $\frac{\log(n)}{\log\log(n)}$), and diverges on large graphs when the number of nodes $n \to \infty$. Thus, the corresponding eigenvectors only report information of the largest degrees. For normalized matrices such as the normalized Laplacian, there are many eigenvalues that are very close to 0, with corresponding eigenvectors reporting information about local dangling subgraphs rather than global structures related to the group labels. We refer to Refs. [30,36] for detailed analysis of spectrum localizations of graph Laplacians and for the comparison between spectral algorithms using different operators on large sparse graphs. Unfortunately, real-world networks are usually sparse, as we can see from Table I that (although they might not be large enough) the citation networks we used for experiments all have an average degree around 4, which is extremely low.

On large synthetic networks, the sparsity issue for conventional GCNs is revealed more clearly (see main text). We carried out extensive experiments to verify our analysis (in Fig. 5).

APPENDIX D: BELIEF PROPAGATION GRAPH CONVOLUTION NETWORK(BPGCN)

On a graph generated by JSBM with n item nodes, m attribute nodes, $nc_1/2$ item-item edges, and nc_2 item-attribute edges, the average degree of item nodes in the connectivity graph is c_1 and the average degree of the attribute graph is c_2. We define matrices storing cavity messages $\mathbf{\Psi}_l^{i\to j} \in \mathbb{R}^{nc_1\times\kappa}$, $\mathbf{\Psi}_l^{i\to\mu} \in \mathbb{R}^{nc_2\times\kappa}$, and $\mathbf{\Psi}_l^{\mu\to i} \in \mathbb{R}^{nc_2\times\kappa}$, and marginal matrices $\mathbf{\Psi}_l^{i} \in \mathbb{R}^{n\times\kappa}$ and $\mathbf{\Psi}_l^{\mu} \in \mathbb{R}^{m\times\kappa}$, where l indicates the cavity messages after the lth step of iterations, The BP equations (3) can be written in the form of matrix multiplications:

$$\begin{aligned}
\mathbf{\Psi}_{l+1}^{\mu} &= \text{Softmax}\left[\mathbf{U_I}\log\left(\mathbf{\Psi}_l^{i\to\mu}\mathbf{P}\right)\right],\\
\mathbf{\Psi}_{l+1}^{i} &= \text{Softmax}\left[\mathbf{U}_{II}\log\left(\mathbf{\Psi}_l^{l\to i}\mathbf{Q}\right) + \mathbf{U}_{III}\log\left(\mathbf{\Psi}_l^{i\to j}\mathbf{P}\right)\right],\\
\mathbf{\Psi}_{l+1}^{i\to j} &= \text{Softmax}\left[\mathbf{B}_I\log\left(\mathbf{\Psi}_l^{\mu\to i}\mathbf{Q}\right) + \mathbf{B}_{II}\log\left(\mathbf{\Psi}_l^{i\to j}\mathbf{P}\right)\right],\\
\mathbf{\Psi}_{l+1}^{i\to\mu} &= \text{Softmax}\left[\mathbf{B}_{III}\log\left(\mathbf{\Psi}_l^{i\to j}\mathbf{P}\right) + \mathbf{B}_{IV}\log\left(\mathbf{\Psi}_l^{\mu\to i}\mathbf{Q}\right)\right],\\
\mathbf{\Psi}_{l+1}^{\mu\to i} &= \text{Softmax}[\mathbf{B}_V\log(\mathbf{\Psi}^{i\to\mu}\mathbf{Q})],
\end{aligned} \tag{D1}$$

where $\mathbf{P}$, $\mathbf{Q}$ are affinity matrices of size $\kappa \times \kappa$.

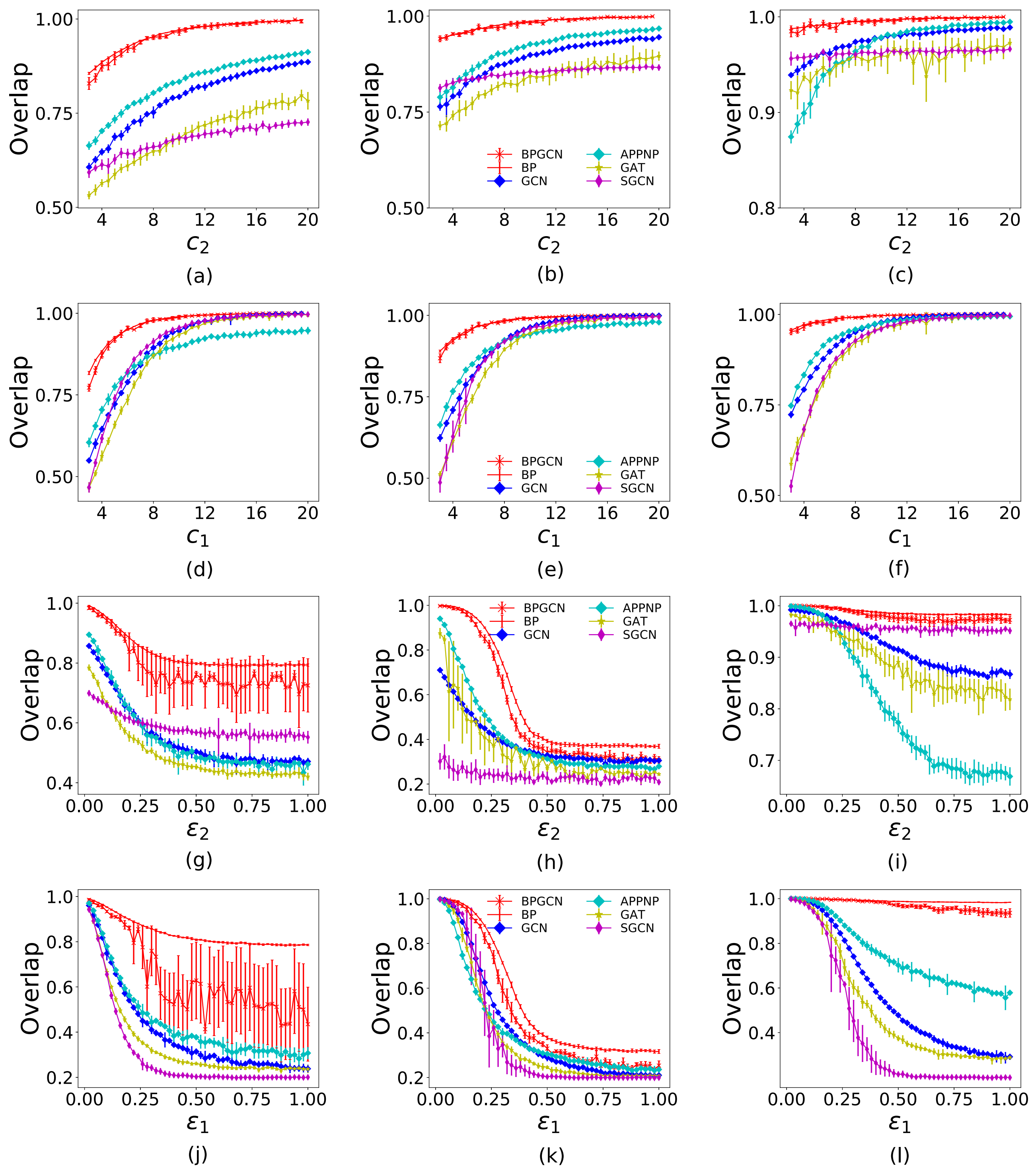

FIG. 5. Overlap [Eq. (15)] obtained by BP, BPGCN, and reference GCNs on synthetic networks generated by JSBM with different parameters. All networks have $n = m = 10\,000$, and group numbers $\kappa = 5$. The fraction of training labels is fixed at $\rho = 0.05$. Each data point is averaged over 10 random instances. In the top row, the average degree of the connectivity graph is fixed to $c_1 = 4, 6$, and 10 from panels (a) to (c). In the second row, the average degree of the attribute graph is fixed to $c_1 = 4, 6$, and 10 from panels (d) to (f). From subfigures (a) to (f), ϵ_1 is fixed to 0.1 and ϵ_2 is fixed to 0.2. In the third row of the figure, c_1, c_2, and ϵ_1 are fixed and ϵ_2 varies. (g) $c_1 = c_2 = 4$, $\epsilon_1 = 0.1$, (h) $c_1 = c_2 = 10$, $\epsilon_1 = 0.4$, and (i) $c_1 = c_2 = 10$, $\epsilon_1 = 0.1$. In the last row of the figure, c_1, c_2, and ϵ_2 are fixed, and the inverse signal-to-noise ratio ϵ_1 varies. (j) $c_1 = c_2 = 4$, $\epsilon_2 = 0.1$, (k) $c_1 = c_2 = 10$, $\epsilon_2 = 0.4$, and (l) $c_1 = c_2 = 10$, $\epsilon_2 = 0.1$.

Kernel matrices $\mathbf{U}_\mathrm{I}$ to $\mathbf{U}_\mathrm{III}$, $\mathbf{B}_\mathrm{I}$ to $\mathbf{B}_\mathrm{V}$ are nonbacktracking matrices [30] that encode adjacency information of cavity messages and marginals, which are defined as

$$\begin{aligned}
\mathbf{U}_I^{\mu,i\to\nu} &= \delta_{\mu\nu} \in \{0,1\}^{m\times nc_2},\\
\mathbf{U}_{II}^{i,\mu\to j} &= \delta_{ij} \in \{0,1\}^{n\times nc_2},\\
\mathbf{U}_{III}^{i,k\to l} &= \delta_{il} \in \{0,1\}^{n\times nc_1},\\
\mathbf{B}_I^{i\to j,\mu\to l} &= \delta_{il}(1-\delta_{\mu j}) \in \{0,1\}^{nc_1\times nc_2},\\
\mathbf{B}_{II}^{i\to j,k\to l} &= \delta_{il}(1-\delta_{kj}) \in \{0,1\}^{nc_1\times nc_1},\\
\mathbf{B}_{III}^{i\to\mu,j\to l} &= \delta_{il}(1-\delta_{\mu j}) \in \{0,1\}^{nc_2\times nc_1},\\
\mathbf{B}_{IV}^{i\to\mu,\nu\to l} &= \delta_{il}(1-\delta_{\mu\nu}) \in \{0,1\}^{nc_2\times nc_2},\\
\mathbf{B}_V^{\mu\to i,j\to\nu} &= \delta_{\mu\nu}(1-\delta_{ij}) \in \{0,1\}^{nc_2\times nc_2}.
\end{aligned} \qquad \text{(D2)}$$

Therefore, the matrix multiplication form of the parallel-updated BP equations can be viewed as a forward model of a neural network. When the above equations propagate and are truncated after L steps of iterations, the resulting algorithm scheme is termed as BPGCN, with L layers. States in the last layer are extracted as the output of BPGCN for computing the cross-entropy loss on training labels (18). Matrices $\mathbf{Q}$ and $\mathbf{P}$ are trainable parameters of the BPGCN and are updated using the back-propagation algorihtm. Input of the neural network are probability-normalized random initial cavities and marginals. In order to accelerate the training process of BPGCN, we use an adjustable external field strength γ as a hyper parameter. For example, suppoese node i is in the training set, i.e., we have label t_i for node i, and a term $\gamma \log(0.1, 0.1, ..., 0.9, ..., 0.1)$ is added to (D1) inside the Softmax function. When γ approaches infinity, node i is pinned to label t_i [15], and is used as an hyper parameter adjusted by the validation set (i.e., when training BPGCN with the training set, we tune these hyper parameters to get better accuracy on the validation set). For nodes in the training set, cavity probabilities and marginals are initialized as (0,1,...,0). Thus, the parameters of BPGCN are $\mathbf{P}$ and $\mathbf{Q}$, and the hyper parameters are ϵ_1, ϵ_2, and γ. The hyper parameters we used in the experiments on real-world data sets (Table I) are for Citeseer and Cora, $L = 5$ and external strength $\gamma = 2.0$; on Cora we set $\epsilon_1 = 0.1$, $\epsilon_2 = 0.6$; on Citeseer we set $\epsilon_1 = 0.1$, $\epsilon_2 = 0.5$. For Pubmed, $L = 2$, external strength $\gamma = 1.5$, $\epsilon_1 = 0.3$, and $\epsilon_2 = 0.8$; when the attribute graph is preprocessed with a MLP to offer BPGCN as a prior or an external field, we set $L = 12$, external strength $\gamma = 0.5$, $\epsilon_1 = 0.1$. For both Karate Club and Political Blogs networks, $L = 5$, external strength $\gamma = 0.5$, and $\epsilon_1 = 0.1$ (ϵ_2 is not applicable).

APPENDIX E: MORE COMPARISONS ON SYNTHETIC NETWORKS

We carried out extensive numerical experiments on synthetic networks generated by JSBM with various parameters. The overlap of BP, BPGCN, and reference algorithms BPGCN, GCN, APPNP, GAT, and SGCN are compared in Fig. 5. In the top row, the average degree of the connectivity graph is fixed to $c_1 = 4$, 6, and 10 from left to right. Due to the spectrum localization problem of graph Laplacians, as we have discussed earlier, when c_1 is fixed to 4, GCN, APPNP, GAT, and SGCN work much worse than BPGCN, even when c_2 is large. We also see that when c_1 is large, most of these tested GCNs work well, even for a very low c_2. In the second row, the average degree of the attribute graph c_2 is fixed and c_1 varies. Performances of reference GCNs approaches the performance of BPGCN, when c_1 gradually increases.

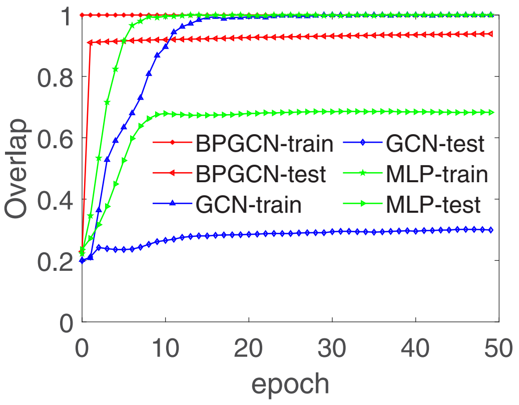

FIG. 6. Training and test overlap [Eq. (15)] as a function of epoch on a synthetic graph generated by JSBM, with $n = m = 10\,000$, $c_1 = c_2 = 10$, $\epsilon 1 = 1$, and $\epsilon_2 = 0.1$.

In the third row of the figure, c_1, c_2 and ϵ_1 are fixed and ϵ_2 varies. On the left, $c_1 = c_2 = 4$, $\epsilon_1 = 0.1$. The graphs are in the sparse regime so conventional GCNs do not work well due to the sparsity issue. In the middle, $c_1 = c_2 = 10$, $\epsilon_1 = 0.4$. Graphs are not sparse, but the edges contains relatively little information about group labels. Results show that SGCN has trouble in this regime even when ϵ_2 is very small. On the right $c_1 = c_2 = 10$, and $\epsilon_1 = 0.1$. The graphs are not in the sparse regime, and edges contain enough information about labels. We see that all GCNs except APPNP perform reasonably well; APPNP has trouble with low ϵ_2, probably because an nonoptimal ω parameter was learned.

In the last row of the figure, c_1, c_2, and ϵ_2 are fixed, and the inverse signal-to-noise ratio ϵ_1 varies. On the left, $c_1 = c_2 = 4$, $\epsilon_2 = 0.1$. We see that BPGCN starts to have a large variance, but the overlap is on average much higher than other GCNs, due to the sparsity of the graphs. In the middle, $c_1 = c_2 = 10$, $\epsilon_2 = 0.4$, meaning that the item-attribute edges in the attribute graph contain little information of labels, and the major information comes from the connectivity graph which is not sparse. In this case, we see that the performances of reference GCNs are comparable to BP and BPGCN, since there is no sparsity issue. On the right, $c_1 = c_2 = 10$, and $\epsilon_2 = 0.1$. There is no sparsity issue for conventional GCNs, and the attribute graph contains enough information about the labels. However, the result is quite surprising: While BP and BPGCN gives almost 100% classification accuracy in the full range of ϵ_1, conventional GCNs yield very low accuracy with ϵ_1 larger than 0.5. This phenomenon implies that when the item-item edges of the connectivity graph are quite noisy, information from the attributes is not well extracted by conventional GCNs.

To understand the reason for this observation, in Fig. 6 we plot the training process of GCN, MLP, and BPGCN on a graph generated with $c_1 = c_2 = 10$, $\epsilon_1 = 1$, and $\epsilon_2 = 0.1$. In this case, it means that the connectivity graph is a pure random graph, with edges containing no information of the label at all ($\epsilon_1 = 1$), while the attribute graph contains adequate information about the ground truth ($\epsilon_2 = 0.1$). We can see that BPGCN yields very high accuracy on the training set and the test set, even from the beginning, because ϵ_2 is so small that the initial values for BPGCN are already good enough. At the begining of the training process, MLP has low training accuracy as well as low test accuracy. But after several epochs of training, its training accuracy goes to 1, and it also generalizes very well to the test set on which the accuracy gradually increases to around 0.6. However, this generalization does not happen for GCN, as we can see that although GCN fits very well to the training set, the overlap for the test set is only slightly above 0.2, i.e., result of a random guess. Obviously, GCN overfits to the training set; the reason is that GCN assumes that edges always contain information of the labels, even when the underlying graph is purely random. Moreover, we can deduct directly from the formula of APPNP (24) that, since APPNP uses ratios ω and $1 - \omega$ to weight the contributions of item-item edges and item-attribute edges, the best possible result given by APPNP on graphs of the above type, where item-item edges contain no information and item-attribute edges contain all the information, is almost equal to the result of MLP. Indeed, we have tested with $\epsilon_1 = 1$, $\epsilon_2 = 0.1$, $c_1 = c_2 = 10$; APPNP achieves an overlap around 0.6, while MLP falls around 0.66.

[1] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, in *International Conference on Learning Representations, Palais des Congrès Neptune, Toulon, France* (2017), arXiv:1609.02907.

[2] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning* (MIT Press, Cambridge, MA, 2016), Vol. 1.

[3] J. Klicpera, A. Bojchevski, and S. Günnemann, Predict then propagate: Graph neural networks meet personalized PageRank, in *International Conference on Learning Representations, New Orleans, LA, USA* (2019), arXiv:1810.05997.

[4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, Graph attention networks, in *International Conference on Learning Representations, Vancouver, BC, Canada* (2018), arXiv:1710.10903.

[5] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, Neural message passing for quantum chemistry, in *Proceedings of the 34th International Conference on Machine Learning* (PMLR, Sydney, 2017), Vol. 70. pp. 1263–1272.

[6] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner *et al.*, Relational inductive biases, deep learning, and graph networks, arXiv:1806.01261.

[7] N. Dehmamy, A.-L. Barabási, and R. Yu, Understanding the representation power of graph neural networks in learning graph topology, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Montréal, Quebec, Canada, 2019), pp. 15387–15397.

[8] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, Weisfeiler and Leman go neural: Higher-order graph neural networks, in *Proceedings of the AAAI Conference on Artificial Intelligence* (AAAI, Hawaii, 2019), Vol. 33, pp. 4602–4609.

[9] P. W. Holland, K. B. Laskey, and S. Leinhardt, Stochastic blockmodels: First steps, Social Netw. **5**, 109 (1983).

[10] D. Hric, T. P. Peixoto, and S. Fortunato, Network Structure, Metadata, and the Prediction of Missing Nodes and Annotations, Phys. Rev. X **6**, 031038 (2016).

[11] L. Florescu and W. Perkins, Spectral thresholds in the bipartite stochastic block model, in *29th Annual Conference on Learning Theory*, edited by V. Feldman, A. Rakhlin, and O. Shamir (PMLR, New York, 2016), Vol. 49, pp. 943–959.

[12] R. Guimerà and M. Sales-Pardo, Justice blocks and predictability of us supreme court votes, PLoS One **6**, e27188 (2011).

[13] J. S. Yedidia, W. T. Freeman, and Y. Weiss, Understanding belief propagation and its generalizations, *Exploring Artificial Intelligence in the New Millennium* (Morgan Kaufmann Publishers Inc., San Francisco, 2003).

[14] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications, Phys. Rev. E **84**, 066106 (2011).

[15] P. Zhang, C. Moore, and L. Zdeborová, Phase transitions in semisupervised clustering of sparse networks, Phys. Rev. E **90**, 052802 (2014).

[16] S. Osindero and G. E. Hinton, Modeling image patches with a directed hierarchy of Markov random fields, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Vancouver, British Columbia, Canada, 2008), pp. 1121–1128.

[17] M. Mezard and A. Montanari, *Information, Physics, and Computation* (Oxford University Press, Oxford, UK, 2009).

[18] H. Nishimori, Exact results and critical properties of the Ising model with competing interactions, J. Phys. C: Solid State Phys. **13**, 4071 (1980).

[19] Y. Iba, The Nishimori line and Bayesian statistics, J. Phys. A: Math. Gen. **32**, 3875 (1999).

[20] H. A. Bethe, Statistical theory of superlattices, Proc. R. Soc. London, Ser. A **150**, 552 (1935).

[21] D. J. C. MacKay and D. J. C. Mac Kay, *Information Theory, Inference, and Learning Algorithms* (Cambridge University Press, Cambridge, UK, 2003).

[22] E. Mossel, J. Neeman, and A. Sly, A proof of the block model threshold conjecture, Combinatorica **38**, 665 (2018).

[23] J. R. L. De Almeida and D. J. Thouless, Stability of the Sherrington-Kirkpatrick solution of a spin glass model, J. Phys. A: Math. Gen. **11**, 983 (1978).

[24] H. Kesten and B. P. Stigum, Additional limit theorems for indecomposable multidimensional Galton-Watson processes, Ann. Math. Stat. **37**, 1463 (1966).

[25] H. Kesten and B. P. Stigum, Limit theorems for decomposable multi-dimensional Galton-Watson processes, J. Math. Anal. Appl. **17**, 309 (1967).

[26] S. Janson and E. Mossel, Robust reconstruction on trees is determined by the second eigenvalue, Ann. Probab. **32**, 2630 (2004).

[27] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas, Comparing community structure identification, J. Stat. Mech.: Theory Exp. (2005) P09008.

[28] P. Zhang, Evaluating accuracy of community detection using the relative normalized mutual information, J. Stat. Mech.: Theory Exp. (2015) P11006.

[29] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc.: Ser. B (Methodol.) **39**, 1 (1977).

[30] F. Krzakala, C. Moore, E. Mossel, J. Neeman, A. Sly, L. Zdeborová, and P. Zhang, Spectral redemption in clustering sparse networks, Proc. Natl. Acad. Sci. USA **110**, 20935 (2013).

[31] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, *Third International Conference for Learning Representations, San Diego, CA, 2015*, arXiv:1412.6980.

[32] P. Zhang and C. Moore, Scalable detection of statistically significant communities and hierarchies, using message passing for modularity, Proc. Natl. Acad. Sci. USA **111**, 18144 (2014).

[33] M. Fey and J. E. Lenssen, Fast graph representation learning with PyTorch Geometric, in *ICLR Workshop on Representation Learning on Graphs and Manifolds, New Orleans, LA, USA* (2019), arXiv:1903.02428.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, Attention is all you need, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Long Beach, CA, 2017), pp. 5998–6008.

[35] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, Simplifying graph convolutional networks, in *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97 (PMLR, Long Beach, California, 2019), pp. 6861–6871, arXiv:1902.07153.

[36] P. Zhang, Robust spectral detection of global structures in the data by learning a regularization, in *Advances in Neural Information Processing Systems* (Curran Associates, Barcelona, Spain, 2016), pp. 541–549.

[37] W. W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. **33**, 452 (1977).

[38] L. A. Adamic and N. Glance, The political blogosphere and the 2004 US election: Divided they blog, in *Proceedings of the Third International Workshop on Link Discovery* (ACM, Chicago, Illinois, 2005), pp. 36–43.

[39] https://github.com/pengfzhou/BPGCN.

[40] Y. LeCun and Y. Bengio, Convolutional networks for images, speech, and time series, in *The Handbook of Brain Theory and Neural Networks* (MIT, Cambridge, Massachusetts, London, 1995), p. 3361.

[41] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Lake Tahoe, Nevada, 2012), pp. 1097–1105.

[42] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, Spectral networks and locally connected networks on graphs, in *Second International Conference on Learning Representations, Banff, AB, Canada* (2014), arXiv:1312.6203.

[43] D. K. Hammond, P. Vandergheynst, and R. Gribonval, Wavelets on graphs via spectral graph theory, Appl. Comput. Harmon. Anal. **30**, 129 (2011).

[44] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, Deep learning via semi-supervised embedding, in *Neural Networks: Tricks of the Trade* (Springer, Berlin, 2012), pp. 639–655.

[45] Z. Yang, W. W. Cohen, and R. Salakhutdinov, Revisiting semi-supervised learning with graph embeddings, *Proceedings of the 33rd International Conference on Machine Learning*, Vol. 48 (JMLR, New York, 2016), pp. 40–48.

[46] X. Zhu, Z. Ghahramani, and J. D. Lafferty, Semi-supervised learning using gaussian fields and harmonic functions, in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)* (AAAI, Washington, DC, 2003), pp. 912–919.

[47] B. Perozzi, R. Al-Rfou, and S. Skiena, Deepwalk: Online learning of social representations, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, 2014), pp. 701–710.

[48] Q. Lu and L. Getoor, Link-based classification, in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)* (Washington, DC, 2003), pp. 496–503.