Consistent Nonparametric Methods for Network Assisted Covariate Estimation

Xueyu Mao [1] **Deepayan Chakrabarti** [2] **Purnamrita Sarkar** [3]

Abstract

Networks with node covariates are commonplace: for example, people in a social network have interests, or product preferences, etc. If we know the covariates for some nodes, can we infer them for the remaining nodes? In this paper we propose a new similarity measure between two nodes based on the patterns of their 2-hop neighborhoods. We show that a simple algorithm (CN-VEC) like nearest neighbor regression with this metric is consistent for a wide range of models when the degree grows faster than $n^{1/3}$ up-to logarithmic factors, where n is the number of nodes. For "low-rank" latent variable models, the natural contender will be to estimate the latent variables using SVD and use them for non-parametric regression. While we show consistency of this method under less stringent sparsity conditions, our experimental results suggest that the simple local CN-VEC method either outperforms the global SVD-RBF method, or has comparable performance for low rank models. We also present simulated and real data experiments to show the effectiveness of our algorithms compared to the state of the art.

1. Introduction

Suppose we have a social network where each person has a vector of interests, such as desired products, or preferred news topics, or sporting interests. For some people, we know their interest vector from, say, their past tweets or comments. But for others, such data may be unavailable or insufficient. Can we infer their interests from a few people's known interests and the structure of the social network? This basic question is relevant for many applications, such as content and ad targeting, friend and group recommendations, and for investigating privacy in social networks, among others. Thus, a general solution to this problem would be useful in many contexts.

Formally, we consider a network where each node has a vector of node covariates. For some nodes, these covariates are known; we want to predict the covariates for all other nodes. Further, the predictions must have consistency guarantees. That is, the predicted covariates must converge to their actual values as the size of the network grows under some limiting process. In particular, we want consistency even for relatively "sparse" networks, often seen in real-world settings, where the average node degree grows slowly compared to the number of nodes.

To predict node covariates using the network structure, we use latent variable models. Here, the network and the node covariates are "generated" by latent variables associated with the nodes. Such models have been used before for community detection (Yang et al., 2013; Zhang et al., 2016; Weng & Feng, 2016; Binkiewicz et al., 2017; Zhang et al., 2019; Yan & Sarkar, 2020). But the node covariate prediction problem is less well-studied.

A seemingly simple solution is to take the average of the covariates of a node's neighbors in the network. But this is not effective in sparse networks. In sparse network models, with high probability, two nodes with identical latent values will not have an edge or even share a common neighbor. So, a node's neighbors may not be the nodes most similar to it. Random-walk heuristics go beyond a node's direct neighbors, but these lack provable guarantees except in special cases (Li et al., 2019b). Thus, we need a more refined measure of similarity between nodes, which accurately reflects distances in latent space and can be estimated consistently from even sparse networks.

We propose a method (CN-VEC) to predict node covariates by a nearest-neighbor regression using the top-k nodes with the most similar two-hop neighborhoods. The similarity between nodes i and j depends on the number of common neighbors between the node pair (i, h), compared against (j, h), over all nodes h. This goes beyond a simple count of the common neighbors of i and j. Our carefully chosen similarity formula is provably consistent for a wide range of latent variable models and sparsities; to our knowledge, it is the first such algorithm. We do not need to know

[1]Department of Computer Science. [2]Department of Information, Risk, and Operations Management. [3]Department of Statistics and Data Sciences. The University of Texas at Austin, TX, USA. Correspondence to: Xueyu Mao <xmao@cs.utexas.edu>, Deepayan Chakrabarti <deepay@utexas.edu>, Purnamrita Sarkar <purna.sarkar@austin.utexas.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

the function linking the probabilities to the latent variables. Also, the similarity measure has no parameters, so CN-VEC needs no fine-tuning.

If we have some prior knowledge of the underlying model, for example, if it is a "low-rank" model (Udell & Townsend, 2019) like the Generalized Random Dot Product Graph (GRDPG) models (Young & Scheinerman, 2007; Rubin-Delanchy et al., 2020), which include many models like the Stochastic Blockmodel (Holland et al., 1983), the Mixed Membership Stochastic Blockmodel (Airoldi et al., 2008), it will be natural to first do singular value decomposition to estimate the latent variables, and then use those directly in non-parametric regression for estimating an unknown covariate. We denote this method by SVD-RBF.

For both CN-VEC and SVD-RBF, we provide consistency guarantees. For general models, CN-VEC is consistent when the average degree grows faster than $n^{1/3}$ up-to logarithmic factors, where n is the number of nodes. Note that CN-VEC depends on 2-hop connections, but the number of 2-hop paths between two nodes only concentrates when the average degree grows faster than $\sqrt{n}$ (Sarkar & Chakrabarti, 2015). The better convergence guarantee of CN-VEC is due to its specially constructed similarity measure. This similarity measure concentrates even when 2-hop path counts do not concentrate. Thus, the analysis for CN-VEC is quite different to analysis of common neighbors (Sarkar & Chakrabarti, 2015; Sarkar et al., 2010). For low-rank models, we show that SVD-RBF is consistent when the average degree grows faster than polylog of n.

We compare CN-VEC with SVD-RBF and a variety of other methods, including random-walks, regression using Jaccard similarity, node2vec (Grover & Leskovec, 2016), a recent embedding-based algorithm called NOBE (Jiang et al., 2018) and regression with network cohesion (RNC) (Li et al., 2019a; Le & Li, 2020). We run experiments using 4 simulated graph models and 3 real-world networks. Overall, SVD-RBF, CN-VEC, node2vec and NOBE outperform the rest. Among the four, CN-VEC is either the best or close to it. This is a surprising observation, since CN-VEC uses local statistics like 2-hop paths, whereas nearly all other methods use the whole network for inference. Also, CN-VEC is 10x-100x faster than node2vec and NOBE, depending on the sparsity of the network.

Our main contribution is the CN-VEC algorithm, which is based on a novel similarity measure. CN-VEC does not assume a low-rank model, needs no parameters for its similarity measure, and uses only local information, yet mostly outperforms the global SVD-RBF algorithm both in accuracy and time. We provide SVD-RBF mainly to show that CN-VEC does not lose much due to its weaker assumptions.

The paper is organized as follows. We review related work in Section 2. In Section 3, we present our model and describe CN-VEC and SVD-RBF, and provide consistency guarantees. Section 4 shows the empirical results. We conclude in Section 5.

2. Related work

We will survey connections to node similarity measures, node classification, regression with network cohesion, and estimation in latent variable models.

Node similarity measures: There are many existing similarity measures, based on the number of common neighbors (Sarkar et al., 2010) and its weighted variants (Adamic/Adar) (Adamic & Adar, 2003), preferential attachment (Barabási & Albert, 1999), resource allocation (Zhou et al., 2009), Katz index (Katz, 1953), PageRank (Brin & Page, 1998), SimRank (Jeh & Widom, 2002), and graph neural networks (Zhang & Chen, 2018) (see (Lü & Zhou, 2011) for a survey). While these often work well, only a few have consistency guarantees (Sarkar et al., 2010; Sarkar & Chakrabarti, 2015). Our CN-VEC method constructs a similarity measure that provably works in sparser settings.

Node classification: Here the goal is to predict labels of nodes based on the network structure. Many methods are based on **random walks**, such as label propagation (Xiaojin & Zoubin, 2002), personalized PageRank (Page et al., 1999; Kloumann et al., 2017; Jeh & Widom, 2003), partially absorbing random walks (Wu et al., 2012), etc. A weighted version of personalized PageRank has provable guarantees under the degree-corrected Stochastic Blockmodel, but only when there are two communities of nodes (Li et al., 2019b). Another direction is **node embeddings**, which aims to represent nodes by vectors while retaining some network-based properties (Tang et al., 2015; Berberidis & Giannakis, 2019; Tsitsulin et al., 2018; Grover & Leskovec, 2016; Perozzi et al., 2014; Jiang et al., 2018; Qiu et al., 2019). With the embedding vectors, one can train a classifier to predict the unseen labels. These typically lack provable guarantees, but often work well in practice. As a special case of node embedding, SVD-RBF uses the eigenvectors and eigenvalues of the adjacency matrix to embed nodes, which is also known as the spectral embedding. It has been well studied in the statistics literature (Tang et al., 2013; Sussman et al., 2014; Rubin-Delanchy et al., 2020). However theoretical consistency of spectral embedding is typically studied under low-rank GRDPG models, while CN-VEC does not need any such model restriction.

When only features are given, semi-supervised learning (Zhu et al., 2003; Nadler et al., 2009; El Alaoui et al., 2016; Calder & Slepčev, 2019) constructs the similarity matrix from the features. Note that features are analogous to

the latent positions of nodes in our problem, and these are unknown for us.

Regression with network cohesion: In regression with network cohesion (Li et al., 2019a; Le & Li, 2020; Jung & Tran, 2019; Jung, 2019), (x_i, y_i) pairs are observed for each node, and the network is used as a regularizer. In Network Lasso (Hallac et al., 2015), the network structure is used to enforce smoothness much like (Li et al., 2019a). Since node features are assumed to be observed in the latter, it cannot be applied directly to our setting. As Network Lasso requires edge weights, our similarity matrix can also be potentially used as the edge weight matrix.

Latent variable inference: Consistent latent inference algorithms have been developed for the Latent Space Model (Ma et al., 2020), Stochastic Blockmodel (Rohe et al., 2011; Lei & Rinaldo, 2015; Abbe, 2017) and its degree-corrected version (Zhao et al., 2012; Jin, 2015; Gao et al., 2018), Mixed Membership Stochastic Blockmodel (Mao et al., 2017; Panov et al., 2017; Mao et al., 2020) and its degree-corrected version (Jin et al., 2017; Mao et al., 2018), Stochastic Blockmodel with Overlaps (Kaufmann et al., 2016), Generalized Random Dot Product Graph models (Sussman et al., 2014; Athreya et al., 2017; Rubin-Delanchy et al., 2020), and so on. However, one needs specialized algorithms for different models, and the true model may be unknown for real world networks. For low rank models, our SVD-RBF estimates the latent variables via a singular value decomposition. However, the low rank assumption is not required in our CN-VEC algorithm, which works for a broad range of latent variable models.

For latent variable models, there is also related work on estimating distances/dot-products in latent space. When the latent variables represent positions in a random geometric graph, spectral methods (Araya Valdivia & Yohann, 2019), shortest path lengths (Arias-Castro et al., 2021), and common neighbor counts (Sarkar et al., 2010) have been used. Recently (Parthasarathy et al., 2017) recovers the shortest path metric from a noisy neighborhood graph. However, those methods are specially designed for different link functions, while CN-VEC does not require prior knowledge on the form of the link function. A more in depth discussion of related works on latent distance estimation can be found in (Arias-Castro et al., 2021).

Other related problems: In **matrix completion**, we try to fill in matrix entries given a partially observed noisy matrix. (Song et al., 2016; Li et al., 2019c) introduce a framework to estimate the missing values using nearest neighbors under a latent variable matrix generation model. In **graphon estimation**, we try to estimate underlying edge probabilities of a random graph from the observed adjacency matrix. Some recent work includes sorting-and-smoothing (Chan & Airoldi, 2014), Stochastic Blockmodel approximation (Airoldi et al., 2013), and neighborhood smoothing (Zhang et al., 2017); see also (Gao et al., 2015; Borgs & Chayes, 2017; Xu, 2018). Our problem is different; we want to predict node covariates.

3. Proposed Work

We are given an undirected and unweighted network between n nodes, represented by a binary adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$. We are also given the node covariates $\{\mathbf{X}_i \in \mathbb{R}^p; i \in S\}$ for a subset of nodes S. Our goal is to infer the node covariates of the remaining nodes $\{\mathbf{X}_i; i \in [n] \setminus S\}$. We will present our notation and model, followed by our two algorithms for the model-agnostic and low-rank cases.

Model. We consider networks generated from general latent variable models. Each node $i \in [n]$ in the network has a latent vector $\mathbf{z}_i \in \mathbb{R}^d$, with $\|\mathbf{z}_i\|$ bounded by a constant C. The probability that there is an edge between node i and j depends solely on $\mathbf{z}_i$ and $\mathbf{z}_j$:

$$\begin{aligned}\mathbf{P}_{ij} &:= \mathrm{P}(\mathbf{A}_{ij} = 1 | \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n) \\ &= \rho_n f(\mathbf{z}_i, \mathbf{z}_j; \mathbf{\Theta}) \quad \text{for all } i \neq j, \end{aligned} \tag{1}$$

where $f(\cdot)$ is bounded in $[0,1]$ and has parameters $\mathbf{\Theta}$, and $\rho_n = o(1)$ controls the sparsity of the graph. For simplicity, we will drop the subscript on ρ_n for the rest of the paper. Thus, the matrix $\mathbf{P}$ denotes the conditional expectation of $\mathbf{A}$ given the latent variables; we set the diagonal of $\mathbf{A}$ to zero. The node covariates are also generated from the latent vectors:

$$\mathbf{X}_i = g(\mathbf{z}_i) + \boldsymbol{\epsilon}_i, \tag{2}$$

where $g : \mathbb{R}^d \to \mathbb{R}^p$ is bounded, and $\boldsymbol{\epsilon}_i$ are i.i.d. noise random vectors with uncorrelated elements, whose mean is 0 and variance is σ^2. We assume that $p = \Theta(1)$, $\sigma = \Theta(1)$, and $g(\cdot)$ is suitably smooth, which is a standard assumption in nonparametric regression (Györfi et al., 2006; Wied & Weißbach, 2012; Duchi, 2019):

Assumption 3.1. $g(\cdot)$ is Lipschitz, that is, there is a constant $L_g > 0$ such that

$$\|g(\mathbf{v}_1) - g(\mathbf{v}_2)\| \leq L_g \cdot \|\mathbf{v}_1 - \mathbf{v}_2\|.$$

We denote by the column vector $\mathbf{a}_i$ the i^{th} column of the adjacency matrix $\mathbf{A}$. We denote by $\mathbf{e}_i$ a vector such that $\mathbf{e}_i(j) = 1(i = j)$, with the vector size being evident from the context. We use the standard o, O and ω, Ω order notations, with $\tilde{O}$ hiding poly-logarithmic factors, and o_P and O_P probabilistic order notations (Vaart, 1998).

3.1. Model-Agnostic Algorithm

The node covariate $\mathbf{X}_i$ of a node i depends on the latent $\mathbf{z}_i$ and the function $g(.)$, both of which are unknown. If we

knew the latents but not $g(.)$, we could still estimate $\mathbf{X}_i$ using a non-parametric estimator:

$$\hat{\mathbf{X}}_i = \frac{\sum_{j \in top_k(i)} \mathbf{W}_{ij} \cdot \mathbf{X}_j}{\sum_{j \in top_k(i)} \mathbf{W}_{ij}}, \tag{3}$$

where $\mathbf{W}_{ij}$ is a measure of similarity between $\mathbf{z}_i$ and $\mathbf{z}_j$, and $top_k(i)$ is a set of k nodes $j \in S$ with the largest $\mathbf{W}_{ij}$ values. Under a smooth $g(.)$ (Assumption 3.1) and mild conditions on $\mathbf{W}$, this is asymptotically consistent (Györfi et al., 2006; Wied & Weißbach, 2012). But in our case, we do not know the latents.

Our approach is to use the network to find $top_k(i)$, and we will show how this is possible even without knowing the latents. The underlying idea is that the latents generate the $\mathbf{P}$ matrix, which in turn generates the adjacency matrix $\mathbf{A}$. So, similarities between latents should be reflected in the network structure. We will now present a series of methods of increasing complexity for finding $top_k(i)$, culminating in our proposed method.

Adjacency matrix: The simplest idea is to average the covariates of a node's neighbors in the network. For example, consider a Stochastic Blockmodel (SBM) (Holland et al., 1983). Here $\mathbf{z}_i$ are latent memberships to r blocks and the network is generated such that the probability of connection of node i in block a and node j in block b is simply $\mathbf{B}_{ab}$, where $\mathbf{B}$ is a $r \times r$ matrix. For this simple example, assume that $\mathbf{A}$ is generated from an SBM and the node covariates are generated such that nodes in block i have i.i.d covariates from a distribution mean μ_i. The means of different blocks are different. Suppose $\mathbf{P}_{ij}$ is high if i and j belong to the same cluster ($\mathbf{z}_i = \mathbf{z}_j$), and low otherwise. Then, if we could set $\mathbf{W} = \mathbf{P}$, the nodes selected in $top_k(i)$ would be those in the same cluster as i. So they would have the same latent as i. Thus, averaging over the covariates of $top_k(i)$ would give a good prediction for $\mathbf{X}_i$. Now, we do not have $\mathbf{P}$, but the adjacency matrix $\mathbf{A}$, which is a stochastic version of $\mathbf{P}$. However, if we use $\mathbf{W} = \mathbf{A}$, there is no way to distinguish between in-cluster versus out-of-cluster neighbors of i. This leads to a biased prediction, so we cannot use the adjacency matrix as the $\mathbf{W}$ matrix.

Common neighbor matrix: The previous idea of using the adjacency matrix $\mathbf{A}$ failed because it did not accurately reflect the probability matrix $\mathbf{P}$. To remedy this, we can set $\mathbf{W} = \mathbf{C}$, where $\mathbf{C}_{ij} = \mathbf{a}_i^T \mathbf{a}_j$ is the number of common neighbors of nodes i and j (for $i \neq j$). The off-diagonal entries of $\mathbf{C}$ concentrate around those of $\mathbf{P}^2$ when the average degree of nodes grows faster than $\tilde{O}(\sqrt{n})$ (Rohe et al., 2011; Sarkar & Chakrabarti, 2015). For the stochastic blockmodel under appropriate conditions, the nodes selected in $top_k(i)$ are again those in the same cluster as i. Thus, setting $\mathbf{W} = \mathbf{C}$ works in dense networks where nodes have high degree. However, this method will not work for sparse networks seen in real-world settings. For sparse matrices, one may need to use more complex similarity matrices like the personalized pagerank (Jeh & Widom, 2003) matrix, which also uses information from long paths.

We will show experimentally that prediction accuracy matches the above discussion. Using the adjacency matrix directly is worse than the matrix of common neighbors, which in turn is worse than matrices based on personalized pagerank. However, we can do much better, and provably so, by extending the common-neighbors idea. We describe this next.

Distances between rows of C: Using $\mathbf{W} = \mathbf{C}$ allowed us to use the "rest of the network" in computing the similarity between i and j. However, only nodes that are common neighbors of both i and j contributed to this measure. Our key observation is that if i and j have similar latents, then we should also expect $\mathbf{P}_{i\ell} \approx \mathbf{P}_{j\ell}$ for any node $\ell \neq i, j$. If the same also holds for $\mathbf{P}^2$ (i.e., $(\mathbf{P}^2)_{i\ell} \approx (\mathbf{P}^2)_{j\ell}^2$), then $\mathbf{C}_{i\ell} \approx (\mathbf{P}^2)_{i\ell}^2 \approx (\mathbf{P}^2)_{j\ell}^2 \approx \mathbf{C}_{j\ell}$ by concentration. So, instead of just considering $\mathbf{C}_{ij}$ as the similarity between i and j, we should use a measure that compares $\mathbf{C}_{i\ell}$ to $\mathbf{C}_{j\ell}$ for all ℓ. In other words, we set $\mathbf{W}_{ij}$ to be the similarity between rows i and j of the matrix $\mathbf{C}$. This goes beyond just the common neighbors of i and j, and hence can work even in sparse networks.

We need the following assumption:

Assumption 3.2. There exist positive constants ℓ and L, and $\Delta_n = o(1)$, such that

$$\ell \|\mathbf{z}_i - \mathbf{z}_j\| - \Delta_n \leq \frac{1}{n^{1.5}\rho^2} \|(\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{P}^2 \left(\mathbf{I} - \mathbf{e}_i\mathbf{e}_i^T - \mathbf{e}_j\mathbf{e}_j^T\right) \| \leq L \|\mathbf{z}_i - \mathbf{z}_j\|.$$

The middle term equals the square root of $\sum_{k \neq i,j} \left((\mathbf{P}^2)_{ik} - (\mathbf{P}^2)_{jk}\right)^2$, normalized by its order. So, the assumption states that $\mathbf{z}_i$ is far from $\mathbf{z}_j$ *iff* $(\mathbf{P}^2)_{ik}$ is far enough from $(\mathbf{P}^2)_{jk}$ for one or more $k \in [n] \setminus \{i, j\}$. That is, for some nodes k in the 2-hop neighborhood of i or j, there should be significant differences.

Remark 3.1. The second inequality of Assumption 3.2 can be derived from the piece-wise Lipschitz condition that is commonly used in graphon estimation literature (Airoldi et al., 2013; Zhang et al., 2017; Chan & Airoldi, 2014; Gao et al., 2018; Xu, 2018). The LHS ensures that each node has enough nearest neighbors in latent space. While the condition looks technical, we show that it is satisfied for Generalized Random Dot Product Graph (GRDPG) models (Young & Scheinerman, 2007; Rubin-Delanchy et al., 2020) which include Stochastic Blockmodel and Mixed Membership Stochastic Blockmodel. The details are in the supplementary material.

Algorithm 1 CN-VEC: model-agnostic algorithm

Input: Adjacency matrix $\mathbf{A}$, Set S of nodes with known covariates, number of neighbors k
Output: Estimated node covariates $\hat{\mathbf{X}}_i$, $i \in [n] \setminus S$
1: **for** $i \in [n] \setminus S$ **do**
2: $dist(j) \leftarrow \mathbf{K}_{ij}$ (by Eq. (4)), for $j \in S$
3: $top_k(i) \leftarrow k$ nodes with the smallest values of $dist(j)$
4: $\hat{\mathbf{X}}_i \leftarrow \frac{1}{k}\sum_{j \in top_k(i)} \mathbf{X}_j$
5: **end for**

By Assumption 3.2, the similarity between i and j can be inferred from $\sum_{k \neq i,j} \left((\mathbf{P}^2)_{ik} - (\mathbf{P}^2)_{jk}\right)^2$. But $\mathbf{P}$ is unknown. So, we need a statistic that converges to this quantity, up to a constant. Recall that $\mathbf{C}_{ij} = \mathbf{a}_i^T \mathbf{a}_j$ denotes the number of common neighbors between nodes i and j. Now, it is easily shown that $\mathbf{E}[\mathbf{C}_{ik}] = (\mathbf{P}^2)_{ik}$. So it may seem that $\sum_{k \neq i,j} (\mathbf{C}_{ik} - \mathbf{C}_{jk})^2$ will work. But $\mathbf{C}_{ik}$ converges to $(\mathbf{P}^2)_{ik}$ only for "dense" networks, where the average degree grows faster than $\tilde{O}(\sqrt{n})$. In sparse networks, $\mathbf{C}_{ik} = 0$ for most (i,k) pairs. For a given k, it is very unlikely that both $\mathbf{C}_{ik} > 0$ and $\mathbf{C}_{jk} > 0$. So, paradoxically, $|\mathbf{C}_{ik} - \mathbf{C}_{jk}| = \mathbf{C}_{ik} + \mathbf{C}_{jk}$ for many k. This means that $\sum_{k \neq i,j} (\mathbf{C}_{ik} - \mathbf{C}_{jk})^2$ may be large even if $\mathbf{z}_i = \mathbf{z}_j$ and $\sum_{k \neq i,j} \left((\mathbf{P}^2)_{ik} - (\mathbf{P}^2)_{jk}\right)^2 = 0$.

Instead, we propose the following statistic to measure the similarity of i and j:

$$\mathbf{K}_{ij} = \sum_{k \neq i,j} \big[(\mathbf{C}_{ik}^2 - 2)1(\mathbf{C}_{ik} \geq 2) + (\mathbf{C}_{jk}^2 - 2)1(\mathbf{C}_{jk} \geq 2) - 2\mathbf{C}_{ik}\mathbf{C}_{jk}\big]. \quad (4)$$

This statistic concentrates around the desired quantity (up to a constant) for both sparse and dense networks, as the next theorem shows.

Theorem 3.1. We have:

$$\mathbf{K}_{ij} = \left(\sum_{k \neq i,j} \left((\mathbf{P}^2)_{ik} - (\mathbf{P}^2)_{jk}\right)^2\right) + e + c,$$

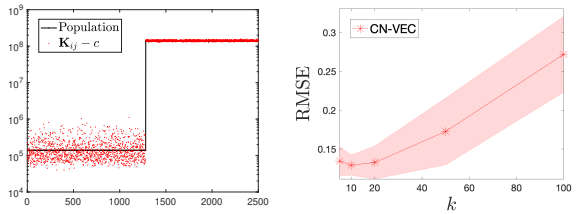
where $e = O_P(n^{2.5}\rho^3\sqrt{\log n})$, $c = -4(n-2)$ if $n\rho^2 = \Omega(\log^\xi n)$, $\xi > 1$; and $e = O_P(n^4\rho^6 + n\rho\sqrt{\log^5 n})$, $c = 0$ if $n\rho^2 = o(1)$, $n^2\rho^3 = \Omega(\log^\xi n)$, $\xi > 2.5$.

Proof Sketch. We may understand the intuition for $\mathbf{K}_{ij}$ by separately considering the cases of dense and sparse networks. In the case of a dense network ($n\rho^2 \to \infty$), we expect $\mathbf{C}_{ik}$ be large, so the indicators may be safely ignored. Thus, we expect $\mathbf{K}_{ij} \approx \sum_{k \neq i,j} (\mathbf{C}_{ik} - \mathbf{C}_{jk})^2 + c$. Now, $\mathbf{C}_{ik} = \sum_h \mathbf{A}_{ih}\mathbf{A}_{hk}$, so it is a sum of independent random variables. Hence, by Bernstein's inequality, $\mathbf{C}_{ik}$ concentrates around its expectation $(\mathbf{P}^2)_{ik}$. This leads to the desired concentration result in the dense case.

This reasoning does not hold for the sparse case ($n\rho^2 \to 0$) because $\mathrm{E}[\mathbf{C}_{ik}] \approx 0$ and $\mathbf{C}_{ik}$ does not concentrate. In this case, $\mathbf{C}_{ik}$ is well-approximated by a Poisson random variable with rate $\lambda_{ik} = (\mathbf{P}^2)_{ik} = O(n\rho^2)$. Thus, the indicator $1(\mathbf{C}_{ik} \geq 2)$ is true when $\mathbf{C}_{ik} = 2$ with probability $\approx \lambda_{ik}^2/2$, and $\mathbf{C}_{ik} > 2$ can be ignored since its probability is of a lower order. Similarly, $\mathbf{C}_{ik}$ and $\mathbf{C}_{jk}$ can be treated as nearly independent since it is very unlikely that a node h is connected to i, j, and also k. So $\mathbf{C}_{ik}\mathbf{C}_{jk} = 1$ with probability $\approx \lambda_{ik}\lambda_{jk}$, with higher values having probabilities of a lower order. Thus, we expect $\mathbf{K}_{ij} \approx \sum_k (2 \cdot (\lambda_{ik}^2/2 + \lambda_{jk}^2/2) - 2\lambda_{ik}\lambda_{jk}) = \sum_k (\lambda_{ik} - \lambda_{jk})^2$, which again gives the desired concentration result. The detailed proof is more involved, and is presented in the supplementary material. □

Remark 3.2. When $\mathbf{z}_i = \mathbf{z}_j$, we have $\mathbf{e}_i^T\mathbf{P} = \mathbf{e}_j^T\mathbf{P}$, so $\mathbf{K}_{ij} - c = e$. But when $\|\mathbf{z}_i - \mathbf{z}_j\| \gg \Delta_n/\ell$, $\mathbf{K}_{ij} - c \approx \left(\sum_{k \neq i,j} \left((\mathbf{P}^2)_{ik} - (\mathbf{P}^2)_{jk}\right)^2\right) = \Omega(n^3\rho^4) \gg e$. So for both sparse and dense networks, the node pairs with small $\mathbf{K}_{ij}$ are also the node pairs with small $\|\mathbf{z}_i - \mathbf{z}_j\|$.

Remark 3.3. We also want to emphasize that the above theoretical result makes use of the fact that our common-neighbor based metric is looking at an *ensemble* of common neighbors, and hence it concentrates in a broader range of sparsity parameters compared to pairwise common neighbors (Sarkar & Chakrabarti, 2015). Our analysis is also completely different from (Sarkar & Chakrabarti, 2015), and requires finer analysis.



(a) $\mathbf{K}_{ij}$ on SBM (log-scale). (b) Changing # of neighbors k.

Figure 1. Simulations.

Remark 3.4. Note that although the c term in Theorem 3.1 may be large, for any graph, it is a constant and does not affect the ordering of $\mathbf{K}_{ij}$. CN-VEC only needs this ordering to pick nearest neighbors for any node i. So, the goal of Theorem 3.1 is to show that ordering by $\mathbf{K}_{ij}$ matches the ordering by the population quantity $\sum_{k \neq i,j} \left((\mathbf{P}^2)_{ik} - (\mathbf{P}^2)_{jk}\right)^2$. The error term e is of a smaller order (Remark 3.2). Figure 1(a) shows this for a Stochastic Blockmodel (SBM) with two communities. The y-axis shows, in log-scale, the value of $(\mathbf{K}_{ij} - c)$ and its population counterpart for a random

Algorithm 2 SVD-RBF: nonparametric regression for low rank models with the RBF kernel $K_\theta(\mathbf{v}_1, \mathbf{v}_2)$

Input: Adjacency matrix $\mathbf{A}$, Set S of nodes with known covariates, bandwidth θ, rank of matrix d
Output: Estimated node covariates $\hat{\mathbf{X}}$

1: $\hat{\mathbf{U}} \leftarrow$ top-d eigenvector matrix for $\mathbf{A}$
2: $\hat{\mathbf{v}}_i \leftarrow i^{th}$ row of $\hat{\mathbf{U}}|\hat{\mathbf{E}}|^{1/2}$
3: **for** $i \in [n] \setminus S$ **do**
4: $\quad dist(j) \leftarrow \|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j\| \quad$ for $j \in S$
5: $\quad \hat{\mathbf{X}}_i \leftarrow \dfrac{\sum_{j\in S} K_\theta(\hat{\mathbf{v}}_i, \hat{\mathbf{v}}_j)\mathbf{X}_j}{\sum_{j\in S} K_\theta(\hat{\mathbf{v}}_i, \hat{\mathbf{v}}_j)}$,
$\quad$ where $K_\theta(\mathbf{v}_1, \mathbf{v}_2) = \exp\left(-\dfrac{||\mathbf{v}_1 - \mathbf{v}_2||^2}{2\theta^2}\right)$
6: **end for**

node i. The x-axis shows nodes grouped by their communities. The figure shows that $(\mathbf{K}_{ij} - c)$ concentrates well around the population. Varying the graph's sparsity yields qualitatively similar results.

Thus, given a node $i \in [n] \setminus S$, ordering the nodes $j \in S$ according to $\mathbf{K}_{ij}$ is equivalent to ordering them according to $\sum_{k\neq i,j}\left((\mathbf{P}^2)_{ik} - (\mathbf{P}^2)_{jk}\right)^2$. We find the top $\log(n)$ nodes among S with the smallest values of $\mathbf{K}_{ij}$ (call this set $top_k(i)$), and average their node covariates to estimate the covariates for node i. So the $\mathbf{W}$ matrix in Eq. (3) can be thought of as a binary matrix with $\mathbf{W}_{ij} = 1$, if $j \in top_k(i)$. We call this algorithm CN-VEC; Algorithm 1 shows the details. Theorem 3.1 coupled with the following theorem shows that the CN-VEC algorithm is consistent.

Theorem 3.2. Suppose in Eq. (2) each element of the random noise vector $\boldsymbol{\epsilon}_i$ has same variance σ^2, $|S| = \Theta(n)$, and Assumptions 3.1 and 3.2 hold, then for any sequence k_n such that $k_n \to \infty$, $k_n/n \to 0$, k_n-nearest-neighbors regression using $\|(\mathbf{e}_i - \mathbf{e}_j)^T\mathbf{P}^2\left(\mathbf{I} - \mathbf{e}_i\mathbf{e}_i^T - \mathbf{e}_j\mathbf{e}_j^T\right)\|$ as the distance metric yields weakly consistent estimates for node covariates when ties occur with probability 0:

$$\mathrm{E}[\|\hat{\mathbf{X}}_i - g(\mathbf{z}_i)\|^2] = o(1) \quad \text{for } i \in [n] \setminus S.$$

Remark 3.5. It is possible to relax $|S|$ to be $o(n)$ as long as $|S| \to \infty$. Intuitively, to predict the covariates for node i, we need to only consider $\mathbf{K}_{ij}$ for $j \in S$. So we can apply Theorem 3.2 by replacing n by the effective size $|S|$.

Remark 3.6. Theorem 3.2 suggests that we can set the number of nearest neighbors in Algorithm 1 as $k = O(\log n)$, with the constant chosen by cross validation. Figure 1(b) shows the RMSE of simulations of Stochastic Blockmodel (SBM) when we change k. This shows a sweet spot for $k \in [10, 20]$.

3.2. Algorithm for Low-Rank Models

One popular class of network models assumes that the probability matrix $\mathbf{P}$ is low-rank. This results from a bilinear form for f, that is, $f(\mathbf{z}_i, \mathbf{z}_j) = \mathbf{z}_i^T\boldsymbol{\Theta}\mathbf{z}_j$ for some model parameters $\boldsymbol{\Theta} \in \mathbb{R}^{d\times d}$. For example, in the Stochastic Blockmodel (SBM) (Holland et al., 1983), the d-dimensional latent vector $\mathbf{z}_i$ for node i is of the form $\mathbf{z}_i = \mathbf{e}_a$ for some $a \in [d]$. Here, we say that node i belongs to "community" a. The probability of a link between nodes i and j is given by $\mathbf{P}_{ij} = \rho f(\mathbf{z}_i, \mathbf{z}_j) = \rho\mathbf{z}_i^T\boldsymbol{\Theta}\mathbf{z}_j = \rho\boldsymbol{\Theta}_{ij}$. That is, link probabilities are solely dependent on the community memberships of nodes, and $\boldsymbol{\Theta}$ represents the community interconnections. The Mixed Membership Stochastic Blockmodel (MMSB) (Airoldi et al., 2008) generalizes this to allow "soft" community memberships. Here, $\mathbf{z}_i$ is a probability vector, representing a distribution over communities for node i.

The Generalized Random Dot Product Graph (GRDPG) model (Young & Scheinerman, 2007; Rubin-Delanchy et al., 2020) allows for more general $\mathbf{z}_i$ and sets $\mathbf{P}_{ij} = \rho\mathbf{z}_i^T\mathbf{I}_{q,d-q}\mathbf{z}_j$, where $\mathbf{I}_{q,d-q}$ is a diagonal matrix with first q elements on the diagonal as 1 and the rest as -1. Let $\hat{\mathbf{U}}\hat{\mathbf{E}}\hat{\mathbf{U}}^T$ be the top-d eigen-decomposition of $\mathbf{A}$, where $\hat{\mathbf{E}}$ is a diagonal matrix, and both $\hat{\mathbf{U}}$ and $\hat{\mathbf{E}}$ have rank d (typically, $d \ll n$). Then for large enough n, the latent vectors $\mathbf{z}_i$ are arbitrarily close to a linear transformation of the rows of $\hat{\mathbf{U}}|\hat{\mathbf{E}}|^{1/2}$ (call them $\hat{\mathbf{v}}_i$) (Rubin-Delanchy et al., 2020). So if the Assumption 3.1 holds for $g(\mathbf{z}_i)$, then it also holds for $g(\hat{\mathbf{v}}_i)$. Hence, we can use $\hat{\mathbf{v}}_i$ as the latent positions in place of $\mathbf{z}_i$. In practice, the number of eigenvectors can be chosen via the USVT estimator (Chatterjee, 2015). For a node i with unknown covariates, we calculate its distances $\|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j\|$ to other nodes $j \in S$ and put them to an RBF kernel to get the weights for nonparametric regression. The estimated covariates $\hat{\mathbf{X}}_i$ is then the weighted average of the covariates $\mathbf{X}_j$. We call this algorithm SVD-RBF; Algorithm 2 shows the details. We prove that Algorithm 2 gives consistent results.

Proposition 3.1. Consider a sequence of networks generated from a GRDPG model with bounded supported where the density exists and its infimum is positive. If Assumptions 3.1 holds, $|S| = \Theta(n)$, $\rho n = \omega(\log^{4\xi} n)$ for some constant $\xi > 0$, $d = \Theta(1)$, and the smallest singular value of $\mathbf{P}$ grows linearly with $n\rho$, then for bandwidth $\theta = \tilde{\Theta}(n^{-\frac{1}{2(d+1)}})$, and $\hat{\mathbf{X}}_i$ returned by Algorithm 2, we have, with probability tending to one,

$$\max_{i\in[n]\setminus S} \|\hat{\mathbf{X}}_i - g(\mathbf{z}_i)\| = o(1).$$

Proof Sketch. The proof follows from an analysis of the Nadaraya–Watson estimator using an RBF kernel. The

bandwidth is chosen using the bound

$$\max_{i\in[n]} \|\mathbf{O}_n\hat{\mathbf{v}}_i - \sqrt{\rho}\mathbf{z}_i\| = O_P\left(\frac{(\log n)^{\xi}}{n^{1/2}}\right)$$

in (Rubin-Delanchy et al., 2020), for some full rank matrix $\mathbf{O}_n \in \mathbb{R}^{d\times d}$ whose spectral norm is almost surely bounded. The details are presented in the supplementary material. □

Proposition 3.1 gives a guidance on choosing the bandwidth θ for Algorithm 2, e.g., setting $\theta = \Theta\left((\log n)^{\frac{3}{d}}/n^{\frac{1}{2(d+1)}}\right)$, while the constant can be fine-tuned by cross-validation.

Complexity: SVD-RBF needs $O((n^2+E)d)$ time to predict the covariates for all nodes in a network with n nodes, E edges, and rank d for $\mathbf{P}$. CN-VEC needs $O(nE)$ time to perform three matrix-matrix multiplications involving $\mathbf{A}$. Both have a space complexity of $O(n^2)$ to store the pairwise node similarities.

4. Experiments

We evaluate the accuracy and speed of CN-VEC and SVD-RBF on several simulated and real-world networks. Since both CN-VEC and SVD-RBF are based on non-parametric regression using our proposed similarity measures, we mainly compare against other similarity measures. So, each method constructs a similarity $\mathbf{W}_{ij}$ between each pair of nodes i and j. Then, given a node i, it picks the top-10 most similar nodes according to the $\mathbf{W}$, and calculates the weighted average of their node covariates, with $\mathbf{W}$ as the weights. We consider the following similarity measures:

- NBR: This predicts the missing covariates for a node using the average of covariates of the neighbors of the node. This simply uses the adjacency matrix $\mathbf{A}$ as $\mathbf{W}$. We use all neighbors of a node instead of selecting top-10 neighbors.

- W-PPR: This is based on personalized pagerank, which can be interpreted as similarity based on random walks (Jeh & Widom, 2003; Kloumann et al., 2017; Li et al., 2019b). The similarity weights are given by $\mathbf{W} = (\mathbf{M}+\mathbf{M}')/2$, where $\mathbf{M} = (1-\gamma)(I-\gamma\mathbf{A}\mathbf{D}^{-1})^{-1}$, and $\mathbf{D}$ is the diagonal matrix of degrees. We set $\gamma = \exp(-.25)$ as recommended in (Li et al., 2019b).

- JACCARD: Here we use the Jaccard index as the similarity matrix $\mathbf{W}$. The Jaccard score between two nodes i and j is defined as $\mathbf{C}_{ij}/(d_i+d_j-\mathbf{C}_{ij})$, where d_i is the degree of node i.

- CN: Here we use the number of common neighbors $\mathbf{C}$ as the similarity matrix $\mathbf{W}$.

- NODE2VEC: This constructs a node embedding $\mathbf{u}_i$ for each node i in the graph (Grover & Leskovec, 2016). We use the default setting of the code[1] provided by the authors. The similarity between i and j is then constructed as for SVD-RBF. That is, we set $\mathbf{W}_{ij} = \exp(-\|\mathbf{u}_i - \mathbf{u}_j\|^2/(2\theta^2))$ for a bandwidth θ.

- NOBE: This is another recent node embedding algorithm (Jiang et al., 2018). We use the default setting of the code[2] provided by the authors and construct $\mathbf{W}$ similar to node2vec.

We also compare with a method that is not based on similarity measure: regression with network cohesion (RNC) (Li et al., 2019a; Le & Li, 2020). Their response variables are *linear* functions of *observed* independent variables $\mathbf{Z}$ and unobserved node-wise effects that are learned from a network-based regularizer. Unlike us, the network is *fixed and not random*. To apply it to our setting, we set $\mathbf{Z}$ to zero and predict the unobserved $\mathbf{X}$ values using their semi-supervised method.

All experiments are performed with Matlab R2018b on servers with 24-core Intel Xeon X5675 and 99GB RAM.

4.1. Simulations

We generate networks from Latent Space Model, Stochastic Blockmodel, Mixed-membership Stochastic Blockmodel, and Random Dot Product Graph model. Each network has $n = 2,500$ nodes and latent dimension $d = 5$ by default. The node covariates are generated by $X_i = \boldsymbol{\beta}^T\mathbf{z}_i + \mathcal{N}(0,.1)$, where $\boldsymbol{\beta}$ is sampled uniformly from the surface of a unit sphere and $\mathbf{z}_i \in \mathbb{R}^d$ is the latent vector of node i.

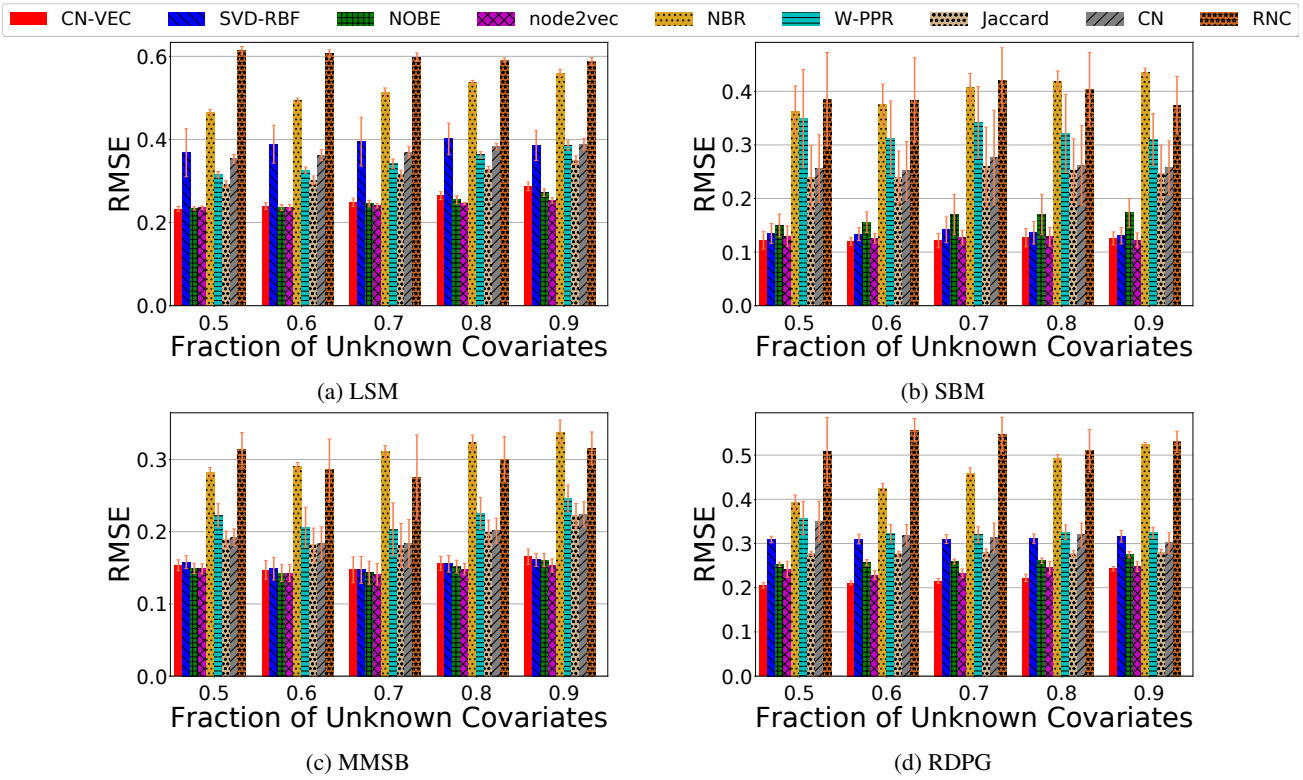
For each network, we vary the fraction of nodes with unknown covariates from 0.5 to 0.9. For each fraction, we randomly select the nodes with unknown covariates and predict their covariates using the various algorithms. We report the mean and variance of root mean square error (RMSE) of the predictions over 10 runs.

Latent Space Model (LSM): The latent vectors $\mathbf{z}_i$ are sampled independently and uniformly between 0 and 1, and $\mathbf{P}_{ij} = \rho\cdot(1+\exp(2.5\times(\|\mathbf{z}_i-\mathbf{z}_j\|)))^{-1}$ with $\rho = 1$. Figure 2(a) shows that CN-VEC, node2vec and NOBE outperform the other methods. Under LSM, the probability matrix $\mathbf{P}$ has full rank, so SVD-RBF is not suited for this model. Indeed, we find that SVD-RBF performs similarly to CN.

Stochastic Blockmodel (SBM): We split the set of n nodes into $d = 5$ equal-sized communities; $\mathbf{z}_i = \mathbf{e}_j$ for $j \in [5]$. The probability of forming a link between i and j is given

[1] https://github.com/aditya-grover/node2vec

[2] https://github.com/Jafree/NOnBacktrackingEmbedding



(a) LSM (b) SBM

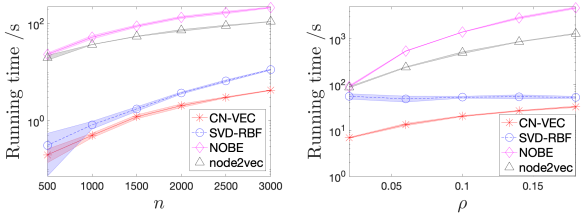
(c) MMSB (d) RDPG

Figure 2. RMSE on recovering hidden node covariates.

by $\mathbf{P}_{ij} = \rho \cdot \mathbf{z}_i^T \mathbf{\Theta} \mathbf{z}_j$ with $\rho = 0.1$, where we sample each cell of $\mathbf{\Theta}$ uniformly from 0 to 1, and then symmetrize $\mathbf{\Theta}$ by $\mathbf{\Theta} = (\mathbf{\Theta} + \mathbf{\Theta}^T + 2 \cdot \mathbf{I})/4$. Since this is a low-rank model, we expect SVD-RBF to perform well. Indeed, Figure 2(b) shows that CN-VEC perform best, followed by SVD-RBF, node2vec and NOBE. The remaining methods are significantly worse.

Mixed-membership Stochastic Blockmodel (MMSB): Here, each $\mathbf{z}_i$ is a 5-dimensional probability vector where the ℓ^{th} component is the probability that node i belongs to community ℓ. The latent variables $\mathbf{z}_i$ are sampled from a Dirichlet distribution that gives equal weight $1/5$ to each of the 5 communities. The link probabilities are given by $\mathbf{P}_{ij} = \rho \cdot \mathbf{z}_i^T \mathbf{\Theta} \mathbf{z}_j$ with $\rho = 0.1$, where $\mathbf{\Theta}$ has a unit diagonal and 0.1 on all off-diagonals. Thus, within-community links are preferred to across-community links. By construction, MMSB leads to a low-rank $\mathbf{P}$, so we expect SVD-RBF to do well. Figure 2(c) shows that CN-VEC, SVD-RBF, node2vec and NOBE are best.

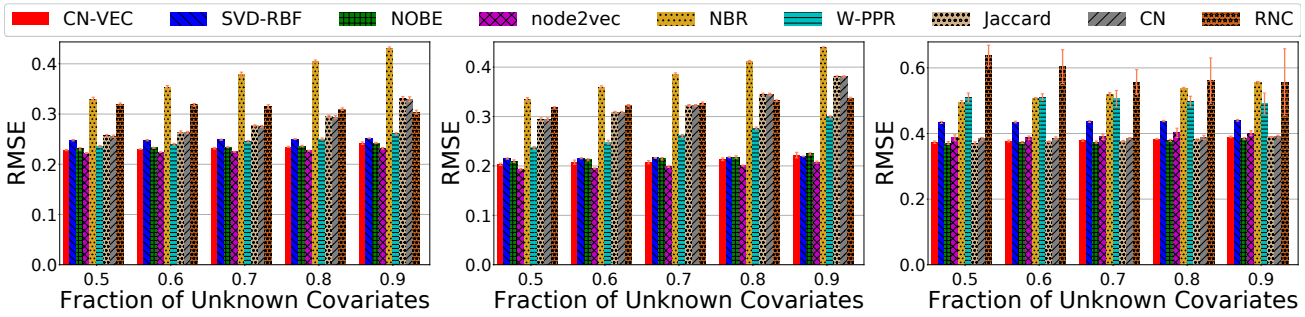
Random Dot Product Graph model (RDPG): We sample the latent variables $\mathbf{z}_i$ from a mixture of d-dimensional Gaussians with means $\mathbf{e}_\ell$ ($\ell = 1, \ldots, 5$) and covariance $0.1 \cdot \mathbf{I}$. The link probabilities are $\mathbf{P}_{ij} = \min(1, \max(0, \rho \cdot \mathbf{z}_i^T \mathbf{z}_j))$ with $\rho = 0.1$. Since $\mathbf{P}_{ij}$ is clipped to $[0, 1]$, $\mathbf{P}$ need not be low-rank. Figure 2(d) shows that CN-VEC significantly outperforms all other methods. Since $\mathbf{P}$ need not be low-rank, SVD-RBF is worse than CN-VEC, and is comparable to W-PPR and CN.



(a) Increasing n with $n\rho$ fixed. (b) Increasing ρ with n fixed.

Figure 3. Running time (log scale).

To summarize, we find that CN-VEC, node2vec, NOBE and SVD-RBF perform better than the other methods. Among them, SVD-RBF works very well for low-rank models, as expected. The model-agnostic CN-VEC algorithm works well in most cases – it outperforms SVD-RBF by all but the MMSB model, NOBE on SBM and RDPG models, and performs comparably with node22ec on all models. However, node2vec and NOBE have no convergence guarantees and takes 10x longer time than CN-VEC, as can be seen from the wall-clock timing results in Figure 3. The timing results are for the SBM graph using Matlab im-



(a) Cora (b) CiteSeer (c) Sinanet

Figure 4. RMSE on recovering hidden topic distributions for each node.

plementations of all algorithms except node2vec, which is implemented in Python. For the first example, we increase n and set ρ such that $n\rho = 250$. In the second plot, we fix n and increase ρ. The same pattern is seen for other network models as well. Thus, for large networks, CN-VEC is more computationally feasible than node2vec and NOBE.

4.2. Real networks

We evaluated our method on two citation networks, namely Cora (McCallum et al., 2000) and CiteSeer (Giles et al., 1998)[3], and one social network, namely Sinanet (Jia et al., 2017)[4]. The citation networks have roughly 3,000 nodes, with average degree 2-4. The nodes in citation networks represent publications and directed edges represents a who-cites-whom relationship. By training a topic model on the words associated with each publication, we obtain a topic distribution for each node, which are then used as node covariates. The number of topics range between 6-7. For this experiment, we remove the directionality of the edges to create an undirected network. Sinanet is a social network extracted from a microblog website[5] (Jia et al., 2017). The nodes are users of the website, and the node covariates are the topic distributions published by Jia et al. (2017). It has roughly 3500 nodes, 10 topics and average degree 16. Table 1 shows the number of nodes, average degree and number of topics for all three datasets.

Table 1. Network statistics.

DATASET	n	AVG. DEGREE	d
CORA	2,708	3.90	7
CITESEER	3,312	2.75	6
SINANET	3,490	16.4	10

For all three datasets, the covariate for a node i is the topic distribution vector $\mathbf{X}_i$. So, our evaluation metric is the RMSE of our estimates $\hat{\mathbf{X}}_i$, measured as $\text{RMSE} = \sqrt{\frac{1}{|U|}\sum_{i\in U}\|\hat{\mathbf{X}}_i - \mathbf{X}_i\|^2}$, where U is the set of unlabeled nodes.

In Figure 4, we see that CN-VEC, node2vec and NOBE are the best on all three datasets. SVD-RBF is comparable for Cora and CiteSeer, but much worse for Sinanet. Since real-world datasets may not follow low-rank models, it is not surprising that SVD-RBF fails in some cases. However, the model-agnostic CN-VEC works well everywhere.

Among the other methods, we find that CN and JACCARD have similar accuracies in all cases. For the citation networks, W-PPR is better than them. But for Sinanet, CN and JACCARD are better than W-PPR, and also SVD-RBF.

5. Conclusion

In this paper, we study the problem of estimating covariates for some nodes in a network, given the covariates for other nodes and the full network structure. This problem has applications in ad targeting and content recommendations, among others. We propose two provably consistent and computationally efficient algorithms. The first, called CN-VEC, applies without knowledge of the underlying model, which is the main contribution of our paper. The second, called SVD-RBF, is aimed at low-rank latent variable models, and works for a more flexible sparsity regime than CN-VEC. Both outperform several popular network statistics in simulated and real-world experiments, with CN-VEC being better overall. CN-VEC is also comparable or better than using a recent node-embedding methods while being 10x-100x faster.

Acknowledgements

P.S. gratefully acknowledges support from NSF (DMS 1713082 and 1934932). D.C. thanks Facebook's Research and Academic Relations Program for a faculty award.

[3] https://linqs.soe.ucsc.edu/data
[4] https://github.com/smileyan448/Sinanet
[5] http://www.weibo.com

References

Abbe, E. Community detection and stochastic block models: recent developments. *Journal of Machine Learning Research*, 18(1):6446–6531, 2017.

Adamic, L. A. and Adar, E. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

Airoldi, E. M., Costa, T. B., and Chan, S. H. Stochastic blockmodel approximation of a graphon: Theory and consistent estimation. In *Advances in Neural Information Processing Systems*, pp. 692–700, 2013.

Araya Valdivia, E. and Yohann, D. C. Latent distance estimation for random geometric graphs. In *Advances in Neural Information Processing Systems*, pp. 8721–8731, 2019.

Arias-Castro, E., Channarond, A., Pelletier, B., and Verzelen, N. On the estimation of latent distances using graph distances. *Electronic Journal of Statistics*, 15(1):722–747, 2021.

Athreya, A., Fishkind, D. E., Tang, M., Priebe, C. E., Park, Y., Vogelstein, J. T., Levin, K., Lyzinski, V., and Qin, Y. Statistical inference on random dot product graphs: a survey. *Journal of Machine Learning Research*, 18(1): 8393–8484, 2017.

Barabási, A.-L. and Albert, R. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

Berberidis, D. and Giannakis, G. B. Node embedding with adaptive similarities for scalable learning over graphs. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

Binkiewicz, N., Vogelstein, J. T., and Rohe, K. Covariate-assisted spectral clustering. *Biometrika*, 104(2):361–377, 2017.

Borgs, C. and Chayes, J. Graphons: A nonparametric method to model, estimate, and design algorithms for massive networks. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pp. 665–672, 2017.

Brin, S. and Page, L. The anatomy of a large-scale hypertextual web search engine. 1998.

Calder, J. and Slepčev, D. Properly-weighted graph laplacian for semi-supervised learning. *Applied Mathematics & Optimization*, pp. 1–49, 2019.

Chan, S. and Airoldi, E. A consistent histogram estimator for exchangeable graph models. In *International Conference on Machine Learning*, pp. 208–216, 2014.

Chatterjee, S. Matrix estimation by universal singular value thresholding. *The Annals of Statistics*, 43(1):177–214, 2015.

Duchi, J. Nonparametric regression: minimax upper and lower bounds, 2019.

El Alaoui, A., Cheng, X., Ramdas, A., Wainwright, M. J., and Jordan, M. I. Asymptotic behavior of\ell_p-based laplacian regularization in semi-supervised learning. In *Conference on Learning Theory*, pp. 879–906, 2016.

Gao, C., Lu, Y., and Zhou, H. H. Rate-optimal graphon estimation. *The Annals of Statistics*, 43(6):2624–2652, 2015.

Gao, C., Ma, Z., Zhang, A. Y., and Zhou, H. H. Community detection in degree-corrected block models. *The Annals of Statistics*, 46(5):2153–2185, 2018.

Giles, C. L., Bollacker, K. D., and Lawrence, S. Citeseer: An automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries*, pp. 89–98, 1998.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, 2016.

Györfi, L., Kohler, M., Krzyzak, A., and Walk, H. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

Hallac, D., Leskovec, J., and Boyd, S. Network lasso: Clustering and optimization in large graphs. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 387–396, 2015.

Holland, P. W., Laskey, K. B., and Leinhardt, S. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, June 1983. ISSN 0378-8733.

Jeh, G. and Widom, J. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 538–543, 2002.

Jeh, G. and Widom, J. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*, pp. 271–279, 2003.

Jia, C., Li, Y., Carson, M. B., Wang, X., and Yu, J. Node attribute-enhanced community detection in complex networks. *Scientific reports*, 7(1):1–15, 2017.

Jiang, F., He, L., Zheng, Y., Zhu, E., Xu, J., and Yu, P. S. On spectral graph embedding: A non-backtracking perspective and graph approximation. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pp. 324–332. SIAM, 2018.

Jin, J. Fast community detection by score. *The Annals of Statistics*, 43(1):57–89, 2015.

Jin, J., Ke, Z. T., and Luo, S. Estimating network memberships by simplex vertex hunting. *arXiv preprint arXiv:1708.07852*, 2017.

Jung, A. Learning networked exponential families with network lasso. *arXiv preprint arXiv:1905.09056*, 2019.

Jung, A. and Tran, N. Localized linear regression in networked data. *IEEE Signal Processing Letters*, 26(7): 1090–1094, 2019.

Katz, L. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

Kaufmann, E., Bonald, T., and Lelarge, M. A spectral algorithm with additive clustering for the recovery of overlapping communities in networks. In *International Conference on Algorithmic Learning Theory*, pp. 355–370. Springer, 2016.

Kloumann, I. M., Ugander, J., and Kleinberg, J. Block models and personalized pagerank. *Proceedings of the National Academy of Sciences*, 114(1):33–38, 2017.

Le, C. M. and Li, T. Linear regression and its inference on noisy network-linked data. *arXiv preprint arXiv:2007.00803*, 2020.

Lei, J. and Rinaldo, A. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1): 215–237, 2015.

Li, T., Levina, E., Zhu, J., et al. Prediction models for network-linked data. *The Annals of Applied Statistics*, 13 (1):132–164, 2019a.

Li, T., Ying, N., Yu, X., and Jing, B.-Y. Semi-supervised learning in unbalanced and heterogeneous networks. *arXiv preprint arXiv:1901.01696*, 2019b.

Li, Y., Shah, D., Song, D., and Yu, C. L. Nearest neighbors for matrix estimation interpreted as blind regression for latent variable model. *IEEE Transactions on Information Theory*, 2019c.

Lü, L. and Zhou, T. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170, 2011.

Ma, Z., Ma, Z., and Yuan, H. Universal latent space model fitting for large networks with edge covariates. *Journal of Machine Learning Research*, 21(4):1–67, 2020.

Mao, X., Sarkar, P., and Chakrabarti, D. On mixed memberships and symmetric nonnegative matrix factorizations. In *International Conference on Machine Learning*, pp. 2324–2333. JMLR. org, 2017.

Mao, X., Sarkar, P., and Chakrabarti, D. Overlapping clustering models, and one (class) svm to bind them all. In *Advances in Neural Information Processing Systems*, pp. 2126–2136, 2018.

Mao, X., Sarkar, P., and Chakrabarti, D. Estimating mixed memberships with sharp eigenvector deviations. *Journal of the American Statistical Association*, 0(0):1–13, 2020. doi: 10.1080/01621459.2020.1751645.

McCallum, A. K., Nigam, K., Rennie, J., and Seymore, K. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

Nadler, B., Srebro, N., and Zhou, X. Semi-supervised learning with the graph laplacian: The limit of infinite unlabelled data. *Advances in neural information processing systems*, 22:1330–1338, 2009.

Page, L., Brin, S., Motwani, R., and Winograd, T. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Panov, M., Slavnov, K., and Ushakov, R. Consistent estimation of mixed memberships with successive projections. In *International Workshop on Complex Networks and their Applications*, pp. 53–64. Springer, 2017.

Parthasarathy, S., Sivakoff, D., Tian, M., and Wang, Y. A Quest to Unravel the Metric Structure Behind Perturbed Networks. In *33rd International Symposium on Computational Geometry (SoCG 2017)*, volume 77, pp. 53:1–53:16, 2017. ISBN 978-3-95977-038-5.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, 2014.

Qiu, J., Dong, Y., Ma, H., Li, J., Wang, C., Wang, K., and Tang, J. Netsmf: Large-scale network embedding as sparse matrix factorization. In *The World Wide Web Conference*, pp. 1509–1520, 2019.

Rohe, K., Chatterjee, S., and Yu, B. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, pp. 1878–1915, 2011.

Rubin-Delanchy, P., Priebe, C. E., Tang, M., and Cape, J. A statistical interpretation of spectral embedding: the generalised random dot product graph. *arXiv preprint arXiv:1709.05506*, 2020.

Sarkar, P. and Chakrabarti, D. The consistency of common neighbors for link prediction in stochastic blockmodels. In *Advances in Neural Information Processing Systems*, pp. 3016–3024, 2015.

Sarkar, P., Chakrabarti, D., and Moore, A. W. Theoretical justification of popular link prediction heuristics. In *Conference on Learning Theory*, 2010.

Song, D., Lee, C. E., Li, Y., and Shah, D. Blind regression: Nonparametric regression for latent variable models via collaborative filtering. In *Advances in Neural Information Processing Systems*, pp. 2155–2163, 2016.

Sussman, D. L., Tang, M., and Priebe, C. E. Consistent latent position estimation and vertex classification for random dot product graphs. *IEEE transactions on pattern analysis and machine intelligence*, 36(1):48–57, 2014.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pp. 1067–1077, 2015.

Tang, M., Sussman, D. L., and Priebe, C. E. Universally consistent vertex classification for latent positions graphs. *The Annals of Statistics*, 41(3):1406–1430, 2013.

Tsitsulin, A., Mottin, D., Karras, P., and Müller, E. Verse: Versatile graph embeddings from similarity measures. In *Proceedings of the 2018 World Wide Web Conference*, pp. 539–548, 2018.

Udell, M. and Townsend, A. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.

Vaart, A. W. v. d. *Asymptotic Statistics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998. doi: 10.1017/CBO9780511802256.

Weng, H. and Feng, Y. Community detection with nodal information. *arXiv preprint arXiv:1610.09735*, 2016.

Wied, D. and Weißbach, R. Consistency of the kernel density estimator: a survey. *Statistical Papers*, 53(1):1–21, 2012.

Wu, X.-M., Li, Z., So, A. M., Wright, J., and Chang, S.-F. Learning with partially absorbing random walks. In *Advances in neural information processing systems*, pp. 3077–3085, 2012.

Xiaojin, Z. and Zoubin, G. Learning from labeled and unlabeled data with label propagation. *Tech. Rep., Technical Report CMU-CALD-02–107, Carnegie Mellon University*, 2002.

Xu, J. Rates of convergence of spectral methods for graphon estimation. In *International Conference on Machine Learning*, 2018.

Yan, B. and Sarkar, P. Covariate regularized community detection in sparse graphs. *Journal of the American Statistical Association*, 0(0):1–12, 2020. doi: 10.1080/01621459.2019.1706541.

Yang, J., McAuley, J., and Leskovec, J. Community detection in networks with node attributes. In *13th International Conference on Data Mining*, pp. 1151–1156. IEEE, 2013.

Young, S. J. and Scheinerman, E. R. Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, pp. 138–149. Springer, 2007.

Zhang, M. and Chen, Y. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, pp. 5165–5175, 2018.

Zhang, Y., Levina, E., and Zhu, J. Community detection in networks with node features. *Electronic Journal of Statistics*, 10(2):3153–3178, 2016.

Zhang, Y., Levina, E., and Zhu, J. Estimating network edge probabilities by neighbourhood smoothing. *Biometrika*, 104(4):771–783, 2017.

Zhang, Y., Chen, K., Sampson, A., Hwang, K., and Luna, B. Node features adjusted stochastic block model. *Journal of Computational and Graphical Statistics*, 28(2):362–373, 2019.

Zhao, Y., Levina, E., and Zhu, J. Consistency of community detection in networks under degree-corrected stochastic block models. *The Annals of Statistics*, 40(4):2266–2292, 2012.

Zhou, T., Lü, L., and Zhang, Y.-C. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, 2009.

Zhu, X., Ghahramani, Z., and Lafferty, J. D. Semi-supervised learning using gaussian fields and harmonic functions. In *International conference on Machine learning*, pp. 912–919, 2003.