Advanced Review

# Bayesian Vertex Nomination Using Content and Context

Shakira Suwan,[1]* Dominic S. Lee[1] and Carey E. Priebe[2]

Using attributed graphs to model network data has become an attractive approach for various graph inference tasks. Consider a network containing a small subset of interesting entities whose identities are not fully known and that discovering them will be of some significance. Vertex nomination, a subclass of recommender systems relying on the exploitation of attributed graphs, is a task which seeks to identify the unknown entities that are similarly interesting or exhibit analogous latent attributes. This task is a specific type of community detection and is increasingly becoming a subject of current research in many disciplines. Recent studies have shown that information relevant to this task is contained in both the structure of the network and its attributes, and that jointly exploiting them can provide superior vertex nomination performance than either one used alone. We adopt this new approach to formulate a Bayesian model for the vertex nomination problem. Specifically, the goal here is to construct a 'nomination list' where entities that are truly interesting are concentrated at the top of the list. Inference with the model is conducted using a Metropolis-within-Gibbs algorithm. Performance of the model is illustrated by a Monte Carlo simulation study and on the well-known Enron email dataset. 

How to cite this article:
*WIREs Comput Stat* 2015. doi: 10.1002/wics.1365

**Keywords:** vertex nomination; attributed graphs; stochastic blockmodels; Bayesian analysis

## INTRODUCTION

The representation of data as graphs, with the vertices as individuals (articles, people, and neurons) and the edges as interactions (citations, friendships, synapses) between pairs of vertices, have emerged as a powerful formalism across application domains. The majority of networks often inherently contain a rich set of attributes or characteristics attached to each vertex. For example, in social networks, profile information of individuals such as names, genders or ages can be encoded as vertex attributes. This type of graph has been extensively explored in many studies, particularly when information pertaining to a latent class or class membership is embedded with each vertex. Examples include the stochastic blockmodel[1] and its extensions. Similarly, we can have additional information about the relationship of the vertex pairs, such as communication topics and languages, embedded as attributes associated with the edges. Attributed graphs are becoming increasingly prevalent in network modeling for representing a broad variety of data because their use allows the additional intrinsic information to be exploited, thereby potentially leading to improved solutions for various network inference tasks.

In many disciplines such as neuroscience, biology, and social science, discovering hidden community structures by exploiting information encoded in the graph topology is often a primary concern. This task is commonly described as community detection or graph clustering (see Salter-Townshend et al.[2] for a

*Correspondence to: shakira.suwan@pg.canterbury.ac.nz

[1]Department of Mathematics and Statistics, University of Canterbury, Christchurch, New Zealand

[2]Department of Applied Mathematics and Statistics, Johns Hopkins University, Baltimore, MD, USA

Conflict of interest: The authors have declared no conflicts of interest for this article.

review of statistical network modeling for community detection). Here, community refers to a set of vertices that display a similar interaction pattern among themselves compared to the rest of the vertices in the network. A special case of this task is the *vertex nomination* problem. Suppose we have a network containing a small subset of interesting entities whose identities are not fully known; in fact, only a few of them are known. Vertex nomination, introduced by Coppersmith and Priebe[3] is a task that seeks to identify the unknown interesting entities using available vertex- and edge-attribute information, and to do so with a quantifiable measure of being correct. Vertex nomination has connections with the semi-supervised classification problem in the machine learning literature.

The meaning of 'interesting' depends on the application context. For example, in the context of insider commercial fraud, if the identities of a few fraudsters are known, law enforcement might want to find others who may be complicit. Another example in law enforcement is to identify and prioritize child abuse offenders using the logging of peer-to-peer activities on child pornography networks, motivated by evidence of connection between individuals convicted of child pornography possession and child abusers. A country's national security agency may be interested in identifying terrorists hidden within the population, starting from the known identities of a few terrorists. Beyond law enforcement and homeland security, vertex nomination is also relevant in various social and business contexts; for example, targeted marketing using recommender systems.[4]

The most relevant previous works that have an impact on vertex nomination are stochastic blockmodels (SBMs)[1] and latent position models.[5] The SBM assumes that each of $n$ vertices is randomly assigned to one of $K$ blocks, and that the existence of an edge is independent given the block memberships of a pair of vertices. Furthermore, the probability of an edge depends only on the block memberships of the two vertices. The SBM can be considered to be foundational for the vertex nomination task. Specifically, in its simplest form, the vertex nomination problem can be formulated as a two-block SBM where one block contains the interesting vertices and the block membership is observed for a few of these vertices.[6] Proceeding quite in parallel to the development of the SBM, the idea of modeling a network by associating a latent position in a $d$-dimensional Euclidean space to each vertex gave rise to the latent position model.[5] This provided another modeling avenue for vertex nomination, which has been recently pursued by Sussman et al.[7] and Tang et al.[8] using spectral embedding techniques and obtaining promising results.

Other modeling approaches for the vertex nomination task include Lee et al.[9] who developed a multivariate self-exciting point process model that associates the memberships of the vertices to the communication messaging events on the high risk topic. Marchette et al.[10] extended the random dot product graph model of Nickel[11] and Young and Scheinerman[12] a special case of the latent position model, by incorporating edge attributes. When the attribute of interest reflects abnormal behavior, vertex nomination can be conceived as an anomaly detection problem and this was pursued by Priebe et al.[13,14] and Grothendieck et al.[15] Sun et al.[16] made a comparison between graph embedding methods for vertex nomination by using a Wilcoxon rank sum test based algorithm to estimate the power of nominating vertices with a particular attribute of interest. For a comprehensive review of vertex nomination, see Coppersmith.[17]

Recently Fishkind et al.[18] proposed several vertex nomination schemes, namely the canonical, spectral-partitioning, and graph-matching vertex nomination schemes, which operate on partially observed unattributed SBM graphs. The canonical method makes use of the conditional probability of vertices belonging to the interesting block, given the partially observed graph, to rank the vertices. Unfortunately, prior knowledge of all model parameters are needed to implement this scheme, and it is difficult to compute when there are more than 20 vertices. It therefore serves as a comparison benchmark for their other schemes. The spectral-partitioning scheme uses eigen-decomposition of the adjacency matrix to embed the vertices in a low-dimensional Euclidean space, which is similar to Tang et al.[8] Finally, the graph-matching scheme utilizes graph-matching tools to construct the nomination list.

Coppersmith and Priebe[3] formulated a joint statistical model that operates on an attributed graph, with vertices denoting entities and edges representing communications between them. Both vertices and edges have attributes that indicate, respectively, whether an entity or the content of communication is interesting. By defining context (who communicates with who) and content (communication topics) statistics, and making appropriate assumptions, Coppersmith and Priebe proposed a parametric likelihood model that combined both content and context statistics for the vertex nomination task. Even in the simplest two-block SBM case, they showed that useful information for vertex nomination is present in both content and context, and using both can improve performance over either one used alone. Therefore, while it is possible to find solutions for vertex nomination using either content or context alone, the use of both is critical for

achieving optimal performance in real problems. This motivates us to further explore the use of both content and context statistics in a Bayesian perspective.

This paper demonstrates the utility of a Bayesian solution for the vertex nomination problem, which jointly exploits information from content and context statistics derived from an attributed graph. To facilitate this, we extend Coppersmith and Priebe's likelihood model into a Bayesian model by introducing a vector of latent vertex attributes for the unknown entities, together with appropriate prior distributions for parameters of the model. Inference with the model is performed using a Metropolis-within-Gibbs algorithm, which provides posterior sample points that allow us to estimate the posterior probability that each latent vertex is interesting, and thereafter to obtain a 'nomination list'. We demonstrate the performance of our Bayesian model using a Monte Carlo simulation study. Another simulation study compares its performance against the method in Coppersmith and Priebe. An application example is provided using the Enron email corpus (http://www.enron-mail.com/).

This paper is organized as follows. Details of the Bayesian model are described in the next section. The Markov chain Monte Carlo (MCMC) algorithm that implements the Bayesian solution is given in Section *Inference*. Section *Simulation Results* describes the simulation studies and the results obtained. Experiments using the Enron email corpus are presented in Section *Application Results*. Finally, Section *Conclusion* summarizes and concludes the paper.

## MODEL

We proceed by introducing some basic notation before discussing specific definitions and models. Consider an unweighted, undirected attributed graph $G = (V, E)$ with no self-loops, multi-edges, or hyper-edges, where $V$ is the set of vertices with $|V| = n$, and $E$ is set of edges (i.e., a subset of the set of unordered pairs of vertices). The presence of an edge between two vertices indicates that the vertex pair communicates. For the vertex nomination problem, each vertex has an attribute—'uninteresting' or 'interesting'—which is observed for a few vertices but hidden for the rest. Note that what is hidden are the attributes and not the vertices, i.e., there are no missing vertices, but most of the vertex attributes are unobserved except for a few 'interesting' ones. We will refer to a vertex whose attribute is unobserved as a latent vertex. Each edge is also attached with an attribute, again 'uninteresting' or 'interesting', which characterizes the content of the communication. Here, we assume that all edges and edge attributes are observed. In what follows, we associate 'uninteresting' with the color green and the integer 1, and 'interesting' with the color red and the integer 2. The resulting graph may thus be represented as an 'edge-attributed' adjacency matrix, $A \in \{0, 1, 2\}^{n \times n}$, whose entry $A_{uv} = 1$ if there is a green edge between vertices $u$ and $v$, $A_{uv} = 2$ if there is a red edge, and $A_{uv} = 0$ if no edge is present.

Let $\mathcal{M}$ be the set of red vertices, with $|\mathcal{M}| = m \ll n$, leaving $n - m$ green vertices in $V \backslash \mathcal{M}$. We assume that we observe the vertex attributes of only a few red vertices and none of the green ones. Let $\mathcal{M}' \subseteq \mathcal{M}$, with $|\mathcal{M}'| = m' > 1$, contain those observed red vertices. Thus the latent vertices whose attributes are unobserved are in $V \backslash \mathcal{M}'$ and there are $n - m'$ of them ($n - m$ green ones and $m - m'$ red ones). Put together, we have the constraints, $1 < m' \leq m \ll n$. Note that it is possible to have no latent red vertices.

Figure 1 illustrates the ideas underlying the model. Vertices with filled circles are those whose vertex attributes are observed, while hollow circles represent vertices with unobserved vertex attributes. All existing edges are observed as well as their edge attributes.

We adopt the likelihood model proposed by Coppersmith and Priebe[3] but with slightly different assumptions. While Coppersmith and Priebe assumed that $0 < m' < m \ll n$, i.e., there is at least 1 observed red vertex and at least 1 latent red vertex, we assume that there are at least 2 observed red vertices but allow

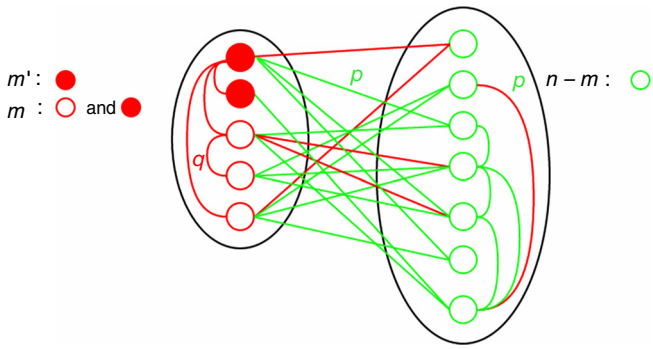


**FIGURE 1** | Illustrative attributed graph with 12 vertices. Here, $m' = 2$ vertices are observed to be red, $m - m' = 3$ are latent red vertices and $n - m = 7$ are latent green vertices. Edges represent communication between connected vertices and edge attributes, denoting content of communication, are assumed to be binary: green or red (1 or 2, respectively). The frequency of communication and distribution of content among red vertices are governed by $q = (q_0, q_1, q_2)$, while $p = (p_0, p_1, p_2)$ quantifies these for the rest of the graph, i.e., among green vertices as well as between a red vertex and a green one. Assuming that all edges and their attributes are observed, these are used together with the observed red vertices to find the remaining red vertices.

the number of latent red vertices to be 0. These new assumptions remain reasonable for our intended applications. It is plausible that the difference between their assumptions and ours will have diminishing consequence as graph size grows. For the insider commercial fraud example, they translate to having two known fraudsters who had colluded in the crime and possibly other (or no other) yet unidentified perpetrators. More importantly, they allow simpler prior distributions that yield a simpler MCMC algorithm for implementing our Bayesian solution. If required, switching to Coppersmith and Priebe's original assumptions can be accommodated through a different choice of prior distribution that enforces the information that there is at least 1 latent red vertex. More details about this are given in the next section.

Let $Y(v)$ be the vertex attribute of vertex $v$,

$$Y(v) = \begin{cases} 1 & \text{if vertex is green,} \\ 2 & \text{if vertex is red.} \end{cases}$$

We define the two statistics as $T(v) = (R(v), S(v))$, where $R(v)$ is the number of observed red vertices connected to $v$ and $S(v)$ is the number of red edges incident to $v$. These are, respectively, the context and content statistics defined in Coppersmith and Priebe and who showed that the use of both statistics resulted in better vertex nomination performance than either one used alone. The advantage of using both context and content was also reported by Qi et al.[19] who modeled multimedia objects and their user-generated tags as a graph with context and content links for the purpose of multimedia annotation. This is analogous to vertex nomination where the vertices are multimedia objects. For the Bayesian approach adopted here, there is potential to use the number of green edges incident to a vertex as an additional statistic, but at the cost of greater model and computational complexity. The cost–benefit of this added complexity is currently being investigated by the authors.

For the vertex pair $u$, $v$, the edge attribute between two green vertices or between a green and a red one (i.e., $Y(u) = Y(v) = 1$ or $Y(u) \neq Y(v)$), is controlled by the probability vector $p = (p_0, p_1, p_2)$, while $q = (q_0, q_1, q_2)$ is the probability vector for edge attributes between two red vertices (i.e., $Y(u) = Y(v) = 2$). They can be expressed as follows:

$$\begin{aligned} \text{(i)}\quad p_1 &= P(A_{uv} = 1 | Y(u) = Y(v) = 1) \\ &= P(A_{uv} = 1 | Y(u) \neq Y(v)), \\ p_2 &= P(A_{uv} = 2 | Y(u) = Y(v) = 1) \\ &= P(A_{uv} = 2 | Y(u) \neq Y(v)), \end{aligned}$$

$$\begin{aligned} \text{(ii)}\quad q_1 &= P(A_{uv} = 1 | Y(u) = Y(v) = 2), \\ q_2 &= P(A_{uv} = 2 | Y(u) = Y(v) = 2), \end{aligned}$$

where in both cases, $p_1$ and $q_1$ denote the probabilities of a green edge, and $p_2$ and $q_2$ denote the probabilities of a red edge. Since $\sum_{i=0}^{2} p_i = 1$ and $\sum_{i=0}^{2} q_i = 1$, $p_0$ and $q_0$ are the probabilities of having no edge in each case.

The probability vector $q$ quantify both the frequency $(q_1 + q_2)$ of communication and distribution $(q_1, q_2)$ of content among red vertices. Likewise, the probability vector $p$ quantify these for the rest of the graph, i.e., among green vertices as well as between a red vertex and a green one—see Figure 1. It is also evident from this figure that the underlying random graph is an edge-attributed two-block SBM, with one block containing red vertices, and another block containing green vertices. Ignoring the edge attributes, the probability of an edge between two red vertices in this SBM would be $q_1 + q_2$. The probability of an edge between two green vertices is $p_1 + p_2$, which is also the probability of an edge between a red vertex and a green one.

Two key assumptions underpinning Coppersmith and Priebe's model are (1) pairs of red vertices, both observed and latent, communicate with a different frequency from other pairs; and (2) the distribution of content among red vertices is different from the rest of the graph. More specifically, it is assumed that $p_1 = q_1$ and $p_2 < q_2$, where the latter prescribes that red edges are more likely between red vertices, and hence a higher frequency of communication among red vertices $(p_1 + p_2 < q_1 + q_2)$.

The joint distribution of the context and content statistics depends on each vertex can be described as follows. Given that a latent green vertex $v \in V\setminus\mathcal{M}$, the number of observed red vertices connected to $v$ has a binomial distribution with parameters $m'$ and $p_1 + p_2$, since there are $m'$ observed red vertices and these can connect to $v$ via green or red edges; hence,

$$f_1(R(v) | p_1, p_2) = \text{Bin}(m', p_1 + p_2), \tag{1}$$

where $\text{Bin}(n, p)$ represents a binomial mass function with parameters $n$ and $p$. The subscript in $f_1$ reminds us that this is for a latent green vertex. Likewise, the number of red edges incident to $v$ has a binomial distribution with parameters $n - 1$ and $p_2$, i.e.,

$$f_1(S(v) | p_2) = \text{Bin}(n - 1, p_2). \tag{2}$$

The joint distribution can be written as

$$\begin{aligned} f_1(T(v)|p_1,p_2) &= f_1(S(v)|R(v),p_1,p_2) f_1(R(v)|p_1,p_2) \\ &= \left\{ \text{Bin}(n-m'-1,p_2) * \text{Bin}\left(R(v), \frac{p_2}{p_1+p_2}\right)\right\} \\ &\quad \cdot \text{Bin}(m', p_1+p_2). \end{aligned} \tag{3}$$

The first term on the RHS is the conditional distribution for the number of red edges incident to $v$ given that there are $R(v)$ observed red vertices connected to it. Knowing the number of observed red vertices connected to $v$ allows us to partition the number of incident red edges into those from the connected observed red vertices and those from the other latent vertices. The distribution for the number of red edges from the other latent vertices is $\text{Bin}(n-m'-1, p_2)$, since there are $n-m'-1$ other latent vertices. The distribution for the number of red edges from the connected observed red vertices is $\text{Bin}(R(v), p_2/(p_1+p_2))$, since the probability that a connecting edge is red is $p_2/(p_1+p_2)$. As these two contributions must sum to $S(v)$, the required conditional distribution is a convolution of these two binomial distributions. Here, $g * h$ denotes the discrete convolution such that $g*h(y) = \sum_z g(y-z)h(z)$.

In a similar way, given a latent red vertex $v \in \mathcal{M} \backslash \mathcal{M}'$ and remembering that $q_1 = p_1$, the distribution for the number of observed red vertices connected to $v$ is,

$$f_2(R(v)|p_1,q_2) = \text{Bin}(m', p_1+q_2), \tag{4}$$

where the subscript in $f_2$ indicates that this is for a latent red vertex. The red edges that are incident to $v$.

$$f_2(S(v)|m,p_2,q_2) = \text{Bin}(n-m,p_2) * \text{Bin}(m-1,q_2), \tag{5}$$

where $S(v)$ in Eq. (5) can be expressed as the sum of two independent discrete variables (i.e., the number of red edges connecting $v$ to latent green vertices and to latent red vertices). Given $R(v)$, $S(v)$ in Eq. (5) can further be divided. Thus, the double convolution in Eq. (6) of the three independent discrete random variables, consists of the number of red edges connecting $v$ to: (1) the observed red vertices, (2) the latent red vertices, and (3) the latent green vertices. Together with Eq. (4), we have

$$\begin{aligned} & f_2(T(v)|m,p_1,p_2,q_2) \\ &= f_2(S(v)|R(v),m,p_1,p_2,q_2) f_2(R(v)|p_1,q_2) \\ &= \left\{ \text{Bin}(n-m,p_2) * \text{Bin}(m-m'-1,q_2) * \text{Bin}\left(R(v), \frac{q_2}{p_1+q_2}\right)\right\} \\ &\quad \cdot \text{Bin}(m', p_1+q_2). \end{aligned} \tag{6}$$

Note that $f * g * h$ is the double convolution, where $f*g*h(x) = \sum_y \left[ f(x-y) \sum_z g(y-z)h(z) \right]$.

Given that $v \in \mathcal{M}'$ is an observed red vertex,

$$f'(R(v)|p_1,q_2) = \text{Bin}(m'-1, p_1+q_2), \tag{7}$$

$$f'(S(v)|m,p_2,q_2) = \text{Bin}(n-m,p_2) * \text{Bin}(m-1,q_2), \tag{8}$$

$$\begin{aligned} & f'(T(v)|m,p_1,p_2,q_2) \\ &= f'(S(v)|R(v),m,p_1,p_2,q_2) f'(R(v)|p_1,q_2) \\ &= \left\{ \text{Bin}(n-m,p_2) * \text{Bin}(m-m',q_2) * \text{Bin}\left(R(v), \frac{q_2}{p_1+q_2}\right)\right\} \\ &\quad \cdot \text{Bin}(m'-1, p_1+q_2). \end{aligned} \tag{9}$$

Let $\mathbf{T}' = \{T'(1), \ldots, T'(m')\}$ be the statistics for those vertices whose attributes are observed to be red. Similarly, let $\mathbf{T} = \{T(1), \ldots, T(n-m')\}$ be the statistics for those vertices whose attributes, $\mathbf{Y} = \{Y(1), \ldots, Y(n-m')\}$, are unknown. By making the simplifying assumption that the statistics are conditionally independent given $\mathbf{Y}$, $p_1$, $p_2$, and $q_2$, the likelihood function is given by

$$\begin{aligned} & f(\mathbf{T}, \mathbf{T}' | \mathbf{Y}, p_1, p_2, q_2) \\ &= \prod_{i:Y(i)=1} f_1(T(i)|p_1,p_2) \prod_{j:Y(j)=2} f_2(T(j)|m,p_1,p_2,q_2) \\ &\quad \prod_{k=1}^{m'} f'(T'(k)|m,p_1,p_2,q_2), \end{aligned} \tag{10}$$

where $m = m' + \sum_{i=1}^{n-m'} I_{\{2\}}(Y(i))$. The first product contains terms from latent green vertices, the second from latent red vertices, and the third from observed red vertices.

By Bayes rule, the posterior distribution for the unknown quantities, $\mathbf{Y}$, $p_1$, $p_2$, and $q_2$, is given by

$$\begin{aligned} & f(\mathbf{Y}, p_1, p_2, q_2 | \mathbf{T}, \mathbf{T}') \\ &\propto f(\mathbf{T}, \mathbf{T}' | \mathbf{Y}, p_1, p_2, q_2) f(\mathbf{Y}, p_1, p_2, q_2), \end{aligned} \tag{11}$$

where $f(\mathbf{Y}, p_1, p_2, q_2)$ is a prior distribution that must be specified. For our problem, the latent attribute vector, $\mathbf{Y}$, is the quantity of interest while $p_1$, $p_2$, and $q_2$ may be regarded as nuisance parameters.

We assume, for the prior distribution, that $\mathbf{Y}$ is independent of $(p_1, p_2, q_2)$, and choose conditionally independent Bernoulli($\psi$) distributions for the components of $\mathbf{Y}$ with the hyperparameter, $\psi = P(Y_i = 2)$,

chosen to follow a beta distribution with parameters $\alpha, \beta > 0$. For $(p_1, p_2, q_2)$, we choose a Dirichlet distribution with parameters $\alpha_0 = \alpha_1 = \alpha_2 = 1$ for $(p_1, p_2)$, and a uniform distribution for $q_2$ conditional on $p_1$ and $p_2$. To summarize, the prior distributions on the model parameters $\mathbf{Y}, p_1, p_2, q_2$ are

$$\begin{aligned}\mathbf{Y}|\psi &\sim \prod_{i=1}^{n-m'} \text{Bernoulli}(\psi),\\ \psi &\sim \text{Beta}(\alpha,\beta),\\ q_2|p_1,p_2 &\sim \text{Uniform}(p_2, 1-p_1),\\ (p_1,p_2) &\sim \text{Dirichlet}(\alpha_0,\alpha_1,\alpha_2).\end{aligned}$$

Hence, the posterior distribution can now be factorized as

$$\begin{aligned}&f(\mathbf{Y},p_1,p_2,q_2,\psi|\mathbf{T},\mathbf{T}')\\ &\propto f(\mathbf{T},\mathbf{T}'|\mathbf{Y},p_1,p_2,q_2) f(\mathbf{Y}|\psi) f(q_2|p_1,p_2) f(p_1,p_2) f(\psi|\alpha,\beta).\end{aligned} \tag{12}$$

It is worth noting that, in addition to using latent attributes, $\mathbf{Y}$, our Bayesian model also leverages observed edge- and vertex-attributes through the content and context statistics contained within $\mathbf{T}$ and $\mathbf{T}'$.

## INFERENCE

Posterior inference can proceed via Markov Chain Monte Carlo (MCMC) using a Metropolis within Gibbs algorithm. Since the components of $\mathbf{Y}$ are binary, they can be updated sequentially using Gibbs sampling as follows. Let $\mathbf{Y}_{-i} = \mathbf{Y} \backslash Y(i)$ denote the vertex attributes for all but vertex $i$ and let $\gamma_i$ be the conditional posterior probability that the latent vertex $i$ is red given $\mathbf{Y}_{-i}$. Thus,

$$\begin{aligned}\gamma_i(\mathbf{Y}_{-i},p_1,p_2,q_2,\psi) &= P\left(Y(i)=2|\mathbf{Y}_{-i},\mathbf{T},\mathbf{T}',p_1,p_2,q_2,\psi\right)\\ &= \frac{f\left(Y(i)=2,\mathbf{Y}_{-i},p_1,p_2,q_2,\psi|\mathbf{T},\mathbf{T}'\right)}{f\left(Y(i)=1,\mathbf{Y}_{-i},p_1,p_2,q_2,\psi|\mathbf{T},\mathbf{T}'\right)+f\left(Y(i)=2,\mathbf{Y}_{-i},p_1,p_2,q_2,\psi|\mathbf{T},\mathbf{T}'\right)}.\end{aligned} \tag{13}$$

See Appendix A for details.

The conditional posterior density of $\psi$ given $\mathbf{Y}$ and nuisance parameters $(p_1, p_2, q_2)$ is

$$\begin{aligned}&f(\psi|\,\mathbf{T},\mathbf{T}',\mathbf{Y},p_1,p_2,q_2)\\ &\quad\propto \psi^{m-m'+\alpha-1}(1-\psi)^{n-m+\beta-1} I_{(0,1)}(\psi),\end{aligned} \tag{14}$$

which is the beta$(m - m' + \alpha, n - m + \beta)$ density. Thus, $\psi$ can easily be updated using a Gibbs update step. Unfortunately, the conditional posterior distribution of $(p_1, p_2, q_2)$ given $\mathbf{Y}$ and $\psi$ does not have a standard form that we can generate from exactly. As such, we use random-walk Metropolis-Hastings to update each parameter in turn, using the conditional distributions, $f(p_1|p_2, q_2)$, $f(p_2|p_1, q_2)$, and $f(q_2|p_1, p_2)$ as the proposal distributions (see Appendix A for details).

Let the state at iteration $h$ be denoted by $\left(\mathbf{Y}^{(h)},p_1^{(h)},p_2^{(h)},q_2^{(h)},\psi^{(h)}\right)$, our Metropolis-within-Gibbs sampler proceeds according to Algorithm 1.

The ability to update $\psi$ using a Gibbs step that generates from a beta distribution is the motivation for allowing the number of latent red vertices to be 0. This makes the independent Bernoulli model a possible choice as prior for $\mathbf{Y}$. Together with the beta hyperprior for $\psi$, we end up with a conjugate conditional posterior in Eq. (14) that facilitates the Gibbs step for updating $\psi$. With Coppersmith and Priebe's assumption that the number of latent red vertices is at least 1, however, the specified prior is no longer appropriate. Letting $\Sigma_{\mathbf{Y}} = Y(1) + \cdots + Y(n - m')$, a possible alternative is

$$f(\mathbf{Y}|\psi) = \begin{cases} 0 & \Sigma_{\mathbf{Y}} = 0,\\ \dfrac{\psi^{m-m'}(1-\psi)^{n-m}}{1-(1-\psi)^{n-m'}} & \Sigma_{\mathbf{Y}} > 0,\end{cases} \tag{15}$$

which no longer admits a conjugate hyperprior. Updating of $\psi$ will therefore require an additional Metropolis-Hastings step within the Gibbs sampler. The cost–benefit of using this alternative prior model is currently being investigated by the authors.

### Performance Measures

Similar to recommender systems, vertex nomination is predominantly concerned with a few suggestions (or interesting vertices) instead of a complete classification of vertices. Choosing an appropriate performance

**Algorithm 1** Metropolis-Hasting within Gibbs sampling procedure

Gibbs step:

At iteration $h$;

**for** $i = 1$ to $n - m'$ **do**

Compute $\gamma_i(Y^{(h)}(1), ..., Y^{(h)}(i-1), Y^{(h-1)}(i+1), Y^{(h-1)}(n-m'), p_1^{(h-1)}, p_2^{(h-1)}, q_2^{(h-1)}, \psi^{(h-1)})$,

Set

$$Y^{(h)}(i) = \begin{cases} 1 & \text{with probability } 1-\gamma_i, \\ 2 & \text{with probability } \gamma_i. \end{cases}$$

**end for**

Compute $m^{(h)} = m' + \sum_{i=1}^{n-m'} I_2(Y^h(i))$ ,

Generate $\psi^{(h)} \sim \text{beta}\,(m^{(h)} - m' + \alpha, n - m^{(h)} + \beta)$.

Metropolis-Hasting step:

Generate $p_1^* \sim f(p_1 \mid p_2^{(h-1)}, q_2^{(h-1)})$ ,

Compute $\pi(p_1) = \min\{1, \frac{f(\mathbf{T},\mathbf{T}'|\mathbf{Y}^{(h)},p_1^*,p_2^{(h-1)},q_2^{(h-1)})}{f(\mathbf{T},\mathbf{T}'|\mathbf{Y}^{(h)},p_1^{(h-1)},p_2^{(h-1)},q_2^{(h-1)})}\}$

Set

$$p_1^{(h)} = \begin{cases} p_1^* & \text{with probability } \pi(p_1), \\ p_1^{(h-1)} & \text{with probability } 1-\pi(p_1), \end{cases}$$

Generate $p_1^* \sim f(p_2 \mid p_1^{(h)}, q_2^{(h-1)})$,

Compute $\pi(p_2) = \min\{1, \frac{f(\mathbf{T},\mathbf{T}'|\mathbf{Y}^{(h)},p_1^{(h)},p_2^{(h-1)},q_2^{(h-1)}}{f(\mathbf{T},\mathbf{T}'|\mathbf{Y}^{(h)},p_1^{(h)},p_2^{(h-1)},q_2^{(h-1)})}\}$,

Set

$$p_2^{(h)} = \begin{cases} p_2^* & \text{with probability } \pi(p_2), \\ p_2^{(h-1)} & \text{with probability } 1-\pi(p_2), \end{cases}$$

Generate $q_2^* \sim f(q_2 \mid p_1^{(h)}, p_2^{(h)})$ ,

Compute $\pi(q_2) = \min\{1, \frac{f(\mathbf{T},\mathbf{T}'|\mathbf{Y}^{(h)},p_1^{(h)},p_2^{(h)},q_2^*}{f(\mathbf{T},\mathbf{T}'|\mathbf{Y}^{(h)},p_1^{(h)},p_2^{(h)},q_2^{(h-1)})}\}$,

Set

$$q_2^{(h)} = \begin{cases} q_2^* & \text{with probability } \pi(q_2), \\ q_2^{(h-1)} & \text{with probability } 1-\pi(q_2). \end{cases}$$

measure for vertex nomination depends on the exploitation task at hand. Two common exploitation tasks are to identity from among the latent vertices (1) as many vertices as possible that are likely to be interesting, and (2) one vertex that is most likely to be interesting. The first task seeks to position as many truly interesting vertices as possible near the top of a ranked nomination list. A performance measure that is suitable for quantifying this is mean average precision (MAP).[3] Average precision (AP) evaluates the precision, at each rank, of the positions of all truly interesting vertices in a ranked list, and then averages the precision values at the rank of each vertex that is truly interesting.

Let $\nu_{(1)}, \nu_{(2)}, \ldots, \nu_{(n-m')}$ be the ordered latent vertices in a ranked list. Following Coppersmith and Priebe[3] we define precision at rank $r$ as

$$\pi(r) = \frac{\sum_{i=1}^{r} I_{\{2\}}\left(Y\left(\nu_{(i)}\right)\right)}{r}, \tag{16}$$

and taking the average of the precision values obtained from Eq. (16) as

$$\bar{\pi} = \frac{\sum_{i=1}^{n-m'} I_{\{2\}}\left(Y\left(\nu_{(i)}\right)\right)\pi(i)}{m-m'}. \tag{17}$$

The MAP for a random experiment is defined as

$$\text{MAP} = \mathbb{E}[\bar{\pi}]. \tag{18}$$

Note that the closer MAP is to 1 the better the model is able to position truly interesting vertices near the top of a ranked nomination list.

When the exploitation task is to find only one interesting vertex, an appropriate measure is the probability of correct nomination, which focuses on getting a correct result at the top of the nomination list:

$$\Pr\left(v_{(1)} \in \mathcal{M} \setminus \mathcal{M}'\right) = \mathbb{E}\left[I_{\{2\}}\left(Y\left(v_{(1)}\right)\right)\right]. \tag{19}$$

The reader is referred to Coppersmith and Priebe[3] and Manning et al.[20] for additional information on these performance measures. Here, we will use both MAP and probability of correct nomination to evaluate the performance of our model.

## SIMULATION RESULTS

Consider an illustrative example where $n = 12$, $m = 5$, $m' = 2$, $p_1 = 0.25$, $p_2 = 0.15$, and $q_2 = 0.25$. A particular graph realization is shown in Figure 1. Labeling the observed red vertices as 1 and 2, the latent red vertices as 3, 4, and 5, and the latent green vertices from 6 to 12, the observed edge attributes are given in Table 1.

To use the MCMC sampler developed, values must first be specified for the parameters of the beta hyperprior for $\psi$. Our choice is motivated by the goal of finding a *single* vertex as being red. Hence, it is desirable that the hyperprior be chosen to induce sparsity in the potential nominees. One way to achieve this is to select a beta density with mode at $1/(n - m')$; a convenient choice being $\alpha = 2$ and $\beta = n - m'$. The results given in this section and the next are based on this choice. We have also studied other choices, including (1) a flat prior (i.e., beta(1, 1) or $U(0, 1)$); (2) a flat prior on the interval $(0, 0.5)$ (i.e., truncated beta(1, 1) on (0, 0.5) or $U(0, 0.5)$) motivated by our expectation that $n \gg m$; and (3) a beta density with $\alpha = m'$ and $\beta = n - 2m'$, and thus having mean at $m'/(n - m')$. Results for these other choices are not reported here but we observed that inference about the probability of correct nomination and nuisance parameters was insensitive to the choice of hyperprior, although there were variations in posterior inference about latent vertex attributes and the hyperparameter $\psi$.

To check and monitor MCMC convergence, we computed Gelman-Rubin statistics for two parallel Markov chains by calculating the percentage of misassigned vertices per iteration. Convergence was assessed based on the potential scale reduction, which is a measure of how much the two parallel chains overlap, with a value below 1.2 or 1.1 indicating convergence.[21] Posterior inference was based on the last 500 iterations of each chain after convergence, giving a total of 1000 MCMC sample points. Figure 2 shows trace plots of the moving average estimates of the marginal posterior probabilities that each of the 10 latent vertices is red. For latent vertex $v \in V \setminus \mathcal{M}'$, for example, this is estimated at iteration $h$ by

**TABLE 1** | Edge-attributed Adjacency Matrix $A$ Whose Entries $A_{uv} = 1$ If Edge Between Entity $u$ and $v$ Is Green, $A_{uv} = 2$ If Edge Is Red and $A_{uv} = 0$ If No Edge Presence, for the Attributed Graph Shown in Figure 1

| | | ● 2 | ○ 3 | ○ 4 | ○ 5 | ○ 6 | ○ 7 | ○ 8 | ○ 9 | ○ 10 | ○ 11 | ○ 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ● | 1 | 1 | 2 | 1 | 0 | 0 | 2 | 0 | 1 | 0 | 1 | 2 |
| ● | 2 | | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| ○ | 3 | | | 0 | 0 | 2 | 2 | 0 | 2 | 0 | 1 | 0 |
| ○ | 4 | | | | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 2 |
| ○ | 5 | | | | | 0 | 2 | 0 | 0 | 2 | 0 | 1 |
| ○ | 6 | | | | | | 0 | 1 | 1 | 2 | 0 | 1 |
| ○ | 7 | | | | | | | 1 | 0 | 0 | 2 | 0 |
| ○ | 8 | | | | | | | | 2 | 2 | 0 | 0 |
| ○ | 9 | | | | | | | | | 1 | 1 | 1 |
| ○ | 10 | | | | | | | | | | 0 | 2 |
| ○ | 11 | | | | | | | | | | | 0 |

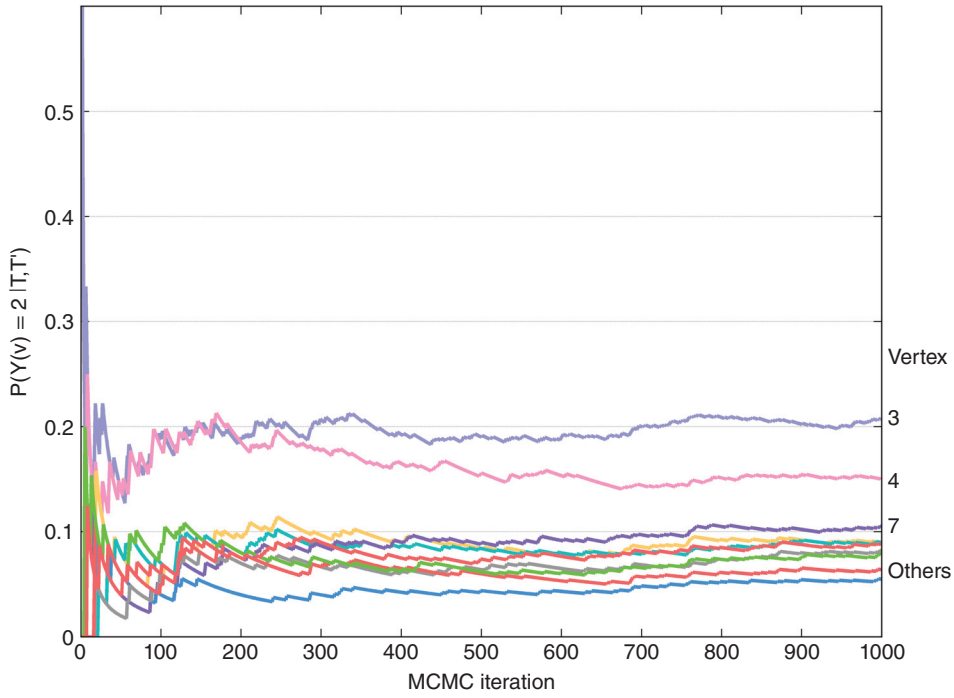


**FIGURE 2** | Trace plots of the moving average estimates of the marginal posterior probabilities that each of the latent vertices is red. The top-three ranking vertices (3, 4, and 7) are labeled as shown; the others (5, 6, 12, 9, 11, 10, and 8) are clustered together at the bottom. Recall that the three latent red vertices are 3, 4, and 5, and so we have a correct nomination in this case.

$$\widehat{P}_h(Y(v)=2|T,T')=\frac{1}{h}\sum_{j=1}^{h} I_{\{2\}}\left(Y^{(j)}(v)\right). \qquad (20)$$

Estimates of the marginal posterior probabilities that each of the latent vertices is a red vertex are given in Table 2. Consequently, vertex number 3, which has the maximum probability, will be posited at the list's beginning. In this case, this turns out to be a correct nomination (recall that the latent red vertices are 3, 4, and 5). We advise caution in interpreting these marginal posterior probabilities at face values because the relationship between them and the probability of correct nomination is not so straightforward. Again, even though we observed that the rankings of these posterior probabilities were quite insensitive to the hyperprior for $\psi$, the values of the posterior probabilities do vary with different hyperpriors. Fortunately, we will show later on that there is evidence of a trend of increasing probability of correct nomination with increasing maximum marginal posterior probability of a latent vertex being red.

Although inference about the nuisance parameters and hyperparameter is not required, it is interesting to look at their prior and posterior distributions. The marginal prior and posterior densities are shown in Figure 3. We used a Gaussian kernel density estimator with diffusion-based bandwidth selection (Algorithm 1 in Botev et al.[22]).

**TABLE 2** | Posterior Probabilities That Latent Vertex Is Red for the Illustrative Attributed Graph with 12 Vertices

| Vertex Number | $\widehat{P}\left(Y(v)=2\|\boldsymbol{T},\boldsymbol{T}'\right)$ |
|---|---|
| 3 | 0.2080 |
| 4 | 0.1510 |
| 5 | 0.0900 |
| 6 | 0.0900 |
| 7 | 0.1060 |
| 8 | 0.0550 |
| 9 | 0.0830 |
| 10 | 0.0640 |
| 11 | 0.0800 |
| 12 | 0.0890 |

Figure 3 shows that concentration of the posterior densities near the true parameter values, indicated by red points on the horizontal axis, is evident for $p_1$ and $p_2$ but less so for $q_2$ because of the smaller number of red vertices. We observed that posterior inference for

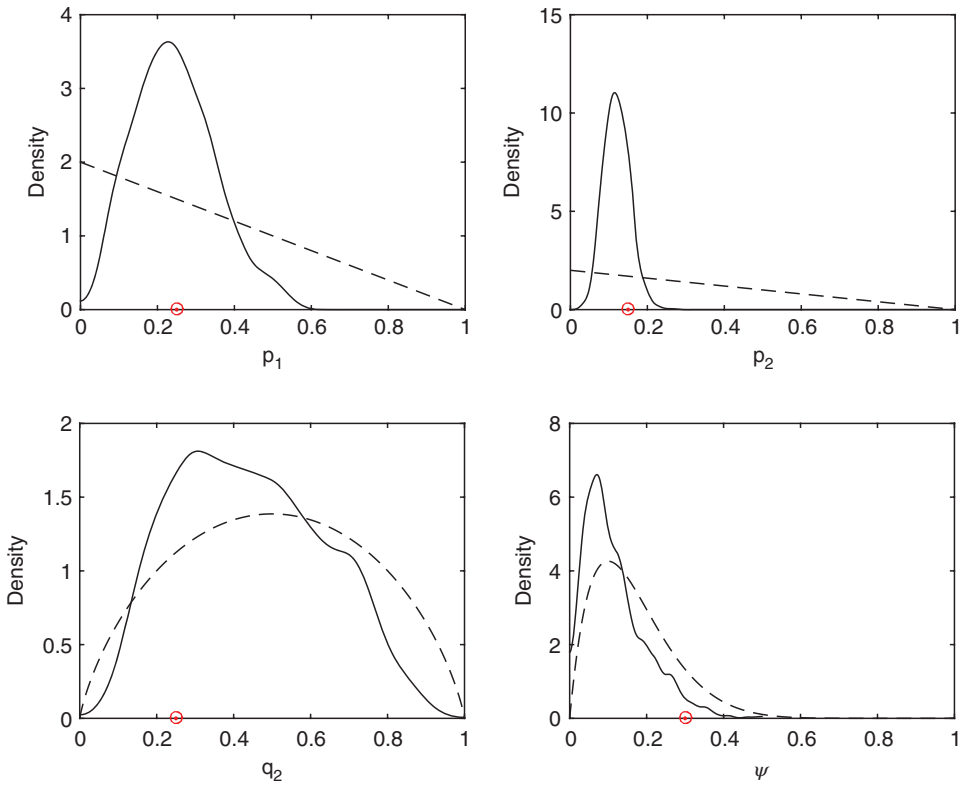


**FIGURE 3** | Marginal prior densities (dashed curves) and posterior densities (solid curves) for $p_1$, $p_2$, $q_2$, and $\psi$. Red points on the horizontal axis indicate the true parameter values.

these nuisance parameters were quite insensitive to the choice of hyperprior for $\psi$, even though the posterior distribution for $\psi$ itself obviously did depend on the hyperprior.

To quantify nomination performance, we repeat the simulation for 1000 graphs using the same parameter settings, i.e., $n = 12$, $m = 5$, $m' = 2$, $p_1 = 0.25$, $p_2 = 0.15$, and $q_2 = 0.25$. We obtain MAP of 0.5413 with 95% bootstrap confidence interval of (0.5409, 0.5417) computed using a method suggested by Park.[23] Notably, MAP of chance is 0.3. Moreover, the estimated probability of correct nomination based on these 1000 graphs is 0.44, with equal-tail 95% confidence interval estimated by the BCA bootstrap[24] as (0.41, 0.47). Recall that for this toy example, the probability of correct nomination purely by chance is 0.3. Hence, both performance measures show that our model performs significantly better than chance. The odds ratio for correct nomination relative to chance is (0.44/0.56)/(0.3/0.7) = 1.8.

Since we have, for any given graph, the marginal posterior probability that the nominated vertex is red, we can estimate the conditional probability of correct nomination given that this marginal posterior probability exceeds $p$. This is shown in Figure 4 where, for example, if the marginal posterior probability that the nominated vertex is red exceeds 0.4, then the conditional probability of correct nomination is estimated to be 0.53, with 95% confidence interval of (0.48, 0.59). There is evidence of a trend of increasing probability of correct nomination with increasing posterior probability that the nominated vertex is red.

The marginal distributions of the posterior means obtained from the 1000 graphs, for the nuisance parameters and hyperparameter, are illustrated in Figure 5. They show concentration of probability mass around the true values of the nuisance parameters.

Coppersmith and Priebe[3] defined a linear fusion statistic for vertex $v$, combining its context and content statistics, as

$$\tau_\lambda(v) = (1-\lambda)R(v) + \lambda S(v), \tag{21}$$

where $\lambda \propto \in [0, 1]$ is a fusion parameter that determined the relative weight of context and content information. For a given value of $\lambda$, the nominated vertex was a latent vertex with the largest value of $\tau_\lambda$. To compare that approach with the one here, we conducted a simulation study adopting the values that

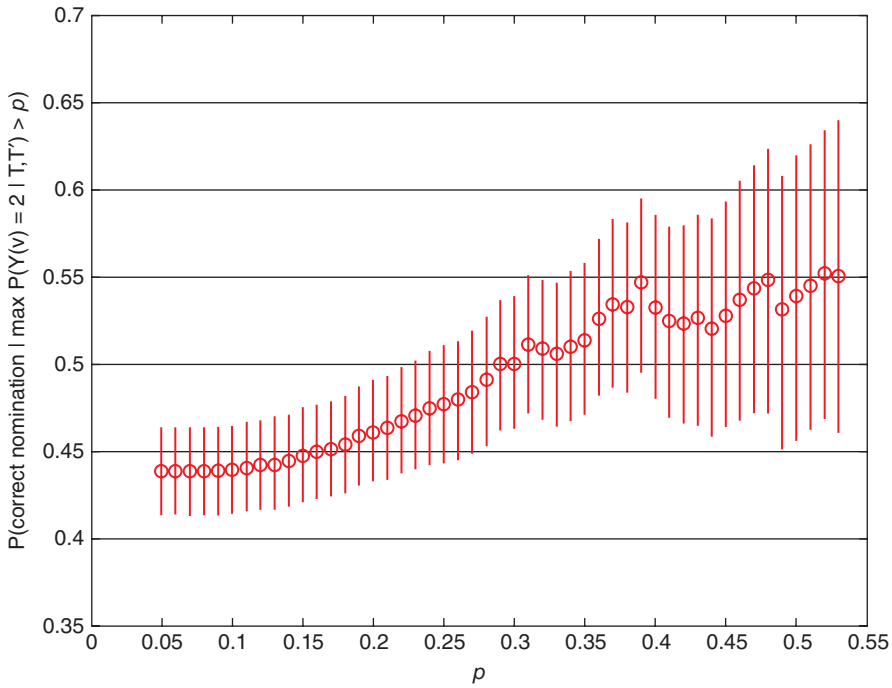


**FIGURE 4** | Conditional probability of correct nomination given that the marginal posterior probability that the nominated vertex is red exceed $p$, with equal-tail 95% BCA bootstrap confidence intervals for 1000 graph realizations.

they had used: $n = 184$, $p_1 = 0.2$, $p_2 = 0.2$, and $q_2 = 0.4$. We investigated two values for $m$, 8 and 32, which represented a 'small' value and a 'large' value of $m$ that they had considered. For each value of $m$, we looked at $m' = m/4$, $m/2$ and $3m/4$, just as they had done. The results, in terms of the probability of correct nomination estimated from 1000 graphs along with 95% BCA bootstrap confidence interval, are given in Table 3.

In Table 3, BVN denotes the approach in this paper and C&P denotes the approach in Coppersmith and Priebe[3] for which the estimated probabilities of correct nomination that are shown corresponded to optimal values of the fusion parameter that gave the best performance. The last row in each table gives the odds ratio (BVN relative to C&P) for correct nomination. We see, for this simulation study and in terms of the probability of correct nomination, that the two approaches have the same performance when $m' = m/4$, i.e., when the number of latent red vertices, $m - m'$, is big relative to $m$. However, as the number of latent red vertices gets smaller relative to $m$, BVN performs increasingly better than C&P (for $m = 8$, the odds ratio increasing from 1 to 1.10 to 1.55 as $m'$ increases and, for $m = 32$, increasing from 1 to 1.47 to 1.89). Moreover, the rate of improvement appears to be higher for a bigger $m$. Note that in an actual application, C&P approach might not achieve its optimal performance because of the difficulty of finding the optimal value of the fusion parameter.

## APPLICATION RESULTS

The Enron email corpus consists of email communications among Enron employees and their associates, where vertices represent Enron employees and edges represent email communication among them. Some of them were allegedly committing fraud and their fraudulent activity was captured in some emails along with many innocuous ones. The data is available at http://www.enron-mail.com/. Other related inference problems had been investigated using this dataset. Priebe et al.[13] looked at anomaly detection by using scan statistics on graphs derived from the email data. In Zhang et al.[25] an influence model that incorporated both email content information and context information (who emailed who) was used to learn the interaction matrix between people in the Enron corpus, and people were clustered based on this interaction matrix.

Priebe et al.[13] derived a processed version of a subset of the email data, over a period of 189 weeks from 1998 to 2002. This yielded 189 graphs (1 graph per week), each containing the same $n = 184$ email users forming the vertices of the graph. Ten of these users have been found to have committed fraud. Berry et al.[26] indexed the contents of a subset of the email corpus into 32 topics. These same topics were adopted by Coppersmith and Priebe,[3] who introduced a mapping from the topics to a binary edge attribute, {green,

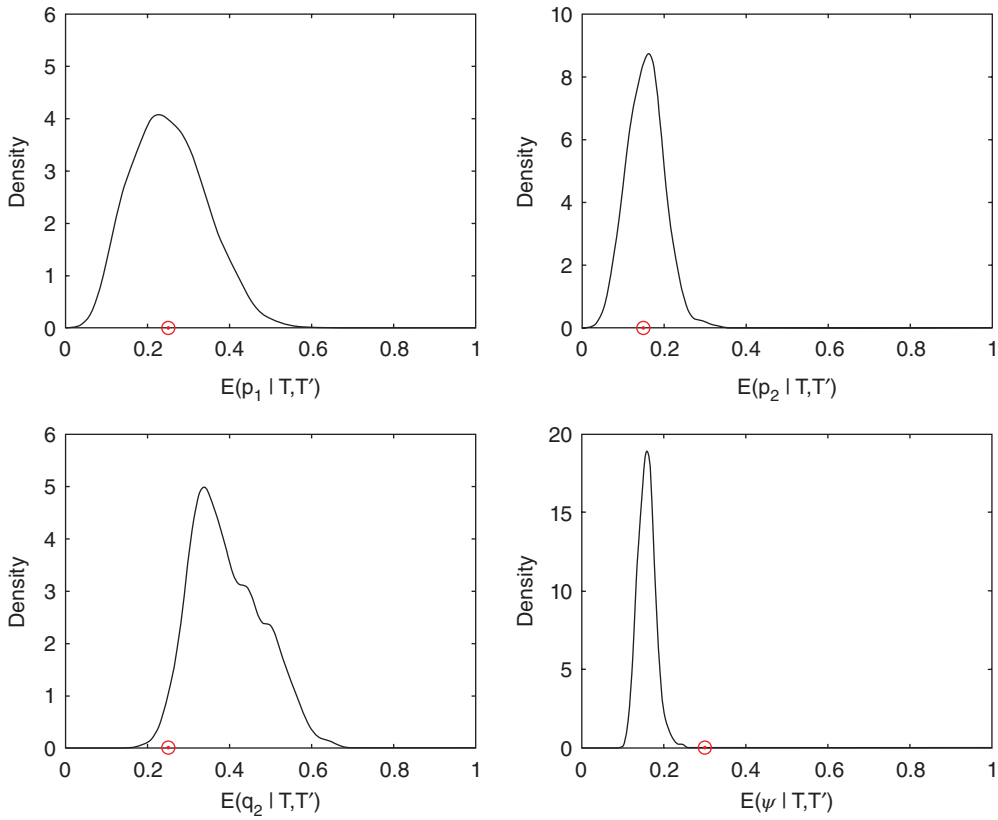


**FIGURE 5** | Kernal densities fitted to posterior means for $p_1$, $p_2$, $q_2$, and $\phi$, obtained from 1000 graph realizations. Red points on the horizontal axis indicate the true parameter values.

red}, denoting content perceived as innocuous and fraudulent, respectively. We use one of the graphs derived by Priebe et al.[13] together with the binary edge attributes from Coppersmith and Priebe[3] for the experiments described here.

For the first experiment, we use Priebe et al.'s Enron graph for week 38. We treat $m' = 5$ of the 10 fraudsters as observed red vertices, the other 5 as latent red vertices and all other remaining users as latent green vertices, to see the nomination performance of our model. The probability of correct nomination and MAP were estimated from all 252 (10 choose 5) combinations of five observed red vertices taken from the 10 fraudsters. For each combination, MCMC iterations for two parallel chains were simulated and the last 500 iterations from each chain were used for estimation once we have reached approximate convergence as indicated by the Gelman-Rubin statistics. Here, we obtain MAP of 0.1623 with 95% bootstrap confidence interval of $(0.1616, 0.1629)$. Note that MAP of chance is 0.03. The estimated probability of correct nomination is 0.12, with 95% BCA bootstrap confidence interval of $(0.11, 0.13)$. Recall that for this experiment, the probability of

**TABLE 3** | Estimated Probability of Correct Nomination Based on 1000 Graphs

| | $m' = 2$ | $m' = 4$ | $m' = 6$ |
|---|---|---|---|
| (a) $m = 8$ | | | |
| BVN | 0.0907 | 0.1001 | 0.1283 |
| | (0.082, 0.10)[1] | (0.10, 0.13) | (0.08, 0.11) |
| C&P[2] | 0.09 | 0.11 | 0.06 |
| OR $\frac{\text{BVN}}{\text{CP}}$ [3] | 1 | 1.10 | 1.55 |

| | $m' = 8$ | $m' = 16$ | $m' = 24$ |
|---|---|---|---|
| (b) $m = 32$ | | | |
| BVN | 0.83 | 0.91 | 0.87 |
| | (0.81, 0.85) | (0.90, 0.93) | (0.86, 0.89) |
| C&P | 0.83 | 0.86 | 0.78 |
| OR $\frac{\text{BVN}}{\text{CP}}$ | 1 | 1.47 | 1.89 |

For $n = 184$, $p_1 = 0.2$, $p_2 = 0.2$, $q_2 = 0.4$. BVN denotes the approach in this paper while C&P is the approach described in Coppersmith and Priebe.[3]
[1] BCA bootstrap confidence interval.
[2] Optimal performance with optimal fusion parameter.
[3] Odds ratio for correct nomination.

correct nomination purely by chance is $5/179 \approx 0.03$. This gives an odds ratio of $(0.12/0.88)/(0.03/0.97) = 4.41$, for correct nomination by the Bayesian model relative to chance.

The distributions of the marginal posterior means of the nuisance parameters, from the 252 combinations, are shown in Figure 6. The sample means of the posterior means are $\bar{p}_1 = 0.0175$, $\bar{p}_2 = 0.0112$, and $\bar{q}_2 = 0.1303$. These estimates will be used in the second experiment.

For the second experiment, we treat the estimates of $p_1, p_2$, and $q_2$ from the first experiment as true values in a Monte Carlo simulation involving $n = 184$, $m = 10$, and $m' = 5$. Vertices 1–5 were known red vertices, 6–10 were latent red vertices and the rest were latent green vertices. 1000 graphs are generated and similar procedure is carried out for inference as described in Section *Simulation Results*. We obtain MAP of 0.2957 with (0.2953, 0.2960) as 95% bootstrap confidence interval, and the probability of correct nomination is estimated to be 0.50, with 95% BCA bootstrap confidence interval of (0.47, 0.52). Hence, the odds ratio for correct nomination relative to chance is $(0.50/0.50)/(0.03/0.97) = 32.3$.

Estimates of the probability of correct nomination given that the posterior probability that the nominated vertex is red exceeds $p$ are illustrated in Figure 7. Once again, there is a clear trend of increasing probability of correct nomination with increasing posterior probability. Furthermore, since we have more data (larger graph with more vertices), the probability of correct nomination is higher. For example, recall that for the graph with 12 vertices in the previous section, the probability of correct nomination given that the posterior probability exceeds 0.4 was between 0.48 and 0.59 with 95% confidence interval. For the current graph with 184 vertices, the same probability of correct nomination is between 0.73 and 0.83.

## CONCLUSION

We have formulated a Bayesian model for the vertex nomination task using content and context statistics derived from an attributed graph. Our methodology is motivated by the joint statistical approach of Coppersmith and Priebe.[3] It is important to note that, in addition to using latent vertex attributes in the model,

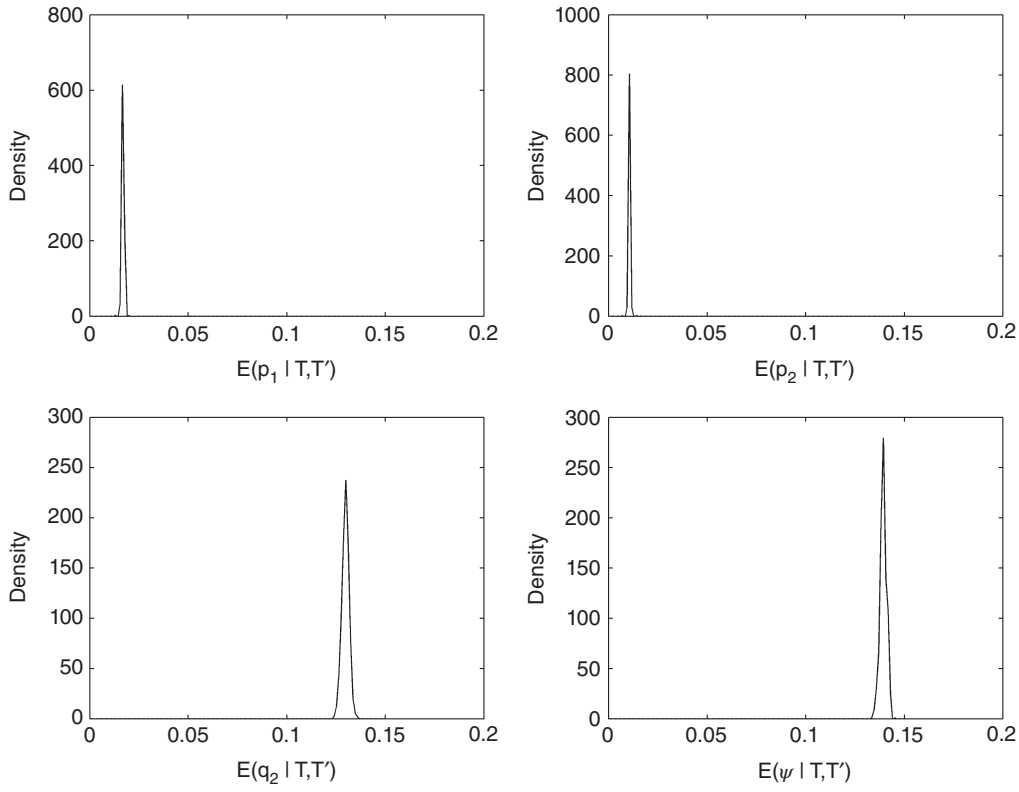


**FIGURE 6** | Kernel densities fitted to posterior means for $p_1$, $p_2$, and $q_2$, obtained from the 252 combinations in Experiment 1.

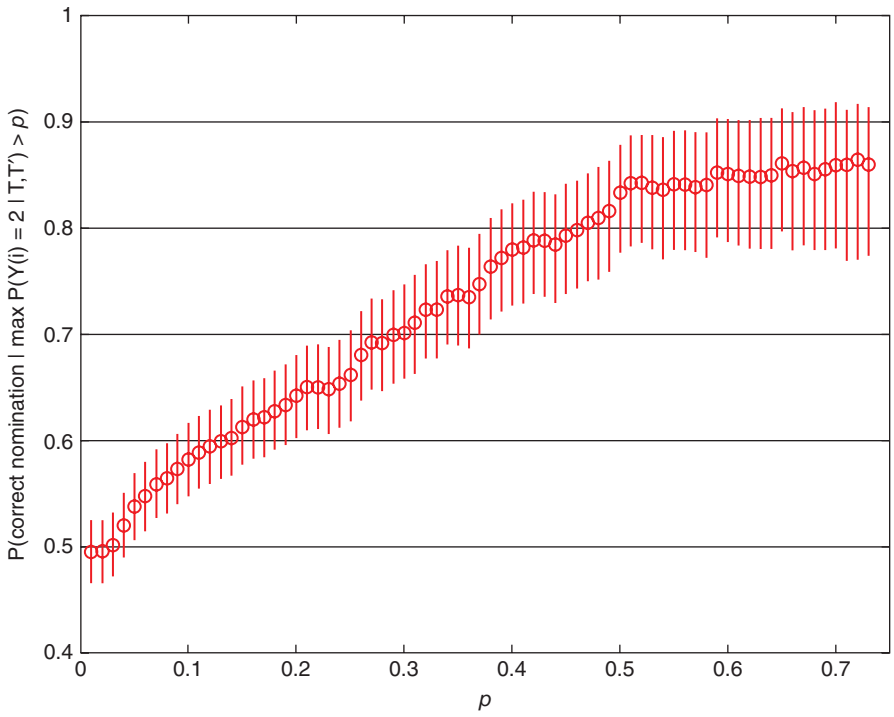


**FIGURE 7** | Conditional probability of correct nomination given that the marginal posterior probability that the nominated vertex is red exceeds $p$, with equal-tail 95% BCA bootstrap confidence intervals. This is obtained from 1000 graphs, each with 1000 MCMC iterations from the two parallel chains after convergence.

our approach leverages observed edge and vertex attributes by harnessing information from content and context. Inference with the model proceeds via a Metropolis-within-Gibbs algorithm for generating sample points from the posterior distribution. Inference about the probability of correct nomination appeared to be insensitive to the choice of hyperprior for $\psi$ even though there were variations in posterior inference about latent vertex attributes.

Results from simulation studies, using a toy example and Coppersmith and Priebe's[3] simulation setting, show that our Bayesian model performs significantly better than chance. Moreover, comparing with the approach in Coppersmith and Priebe[3] our Bayesian model performs increasingly better as the number of unknown interesting vertices decreases relative to the total number of interesting vertices, and with a higher rate of improvement as the total number of interesting vertices gets larger. In the case of nominating only one interesting vertex, there is evidence to indicate a trend of increasing probability of correct nomination when the posterior probability that the nominated vertex is interesting increases. Similar results are observed from experiments with the Enron email corpus.

Our results were focused on identifying interesting vertices based on their marginal posterior probabilities of being interesting. In practice, it can be of interest to identify two or more vertices as being jointly interesting. For example, in the context of insider commercial fraud, one may wish to identify two or more individuals who are complicit in committing fraud. It would be relatively straightforward to estimate the required posterior joint probabilities given that we have MCMC sample points from the full joint posterior distribution, but this might require a greater number of points.

In reality, connections between vertices as well as attributes of edges may not be observed perfectly, leading to missing edges and missing edge attributes respectively. One way to extend our model to handle missing data is to use similar ideas as in Aicher et al.[27] to distinguish between (1) no edge: observed absence of connection between vertices, (2) missing edge: unobserved connection between vertices, and (3) missing edge attribute: observed connection between vertices but unobserved edge attribute. These distinctions can then be explicitly incorporated into the model to handle missing data.

## ACKNOWLEDGMENTS

We thank the Review Editor and two Referees for their comments and useful suggestions. Thanks also to Youngser Park for assisting with the Enron data and for helpful comments on a draft version, and to Celine Cattoen-Gilbert for suggestions for speeding up computations.

## REFERENCES

1. Holland PW, Laskey KB, Leinhardt S. Stochastic blockmodels: first steps. *Social Netw* 1983, 5:109–137.
2. Salter-Townshend M, White A, Gollini I, Murphy T. Review of statistical network analysis: models, algorithms, and software. *Stat Anal Data Mining* 2012, 5:243–264.
3. Coppersmith G, Priebe C. Vertex nomination via content and context. *arXiv:1201.4118v1*, 2012.
4. Resnick P, Varian H. Recommender systems. *Commun ACM* 1997, 40:56–58.
5. Hoff P, Raftery A, Handcock M. Latent space approaches to social network analysis. *J Am Stat Assoc* 2002, 97:1090–1098.
6. Nowicki K, Snijders T. Estimation and prediction for stochastic blockstructures. *J Am Stat Assoc* 2001, 96:1077–1087.
7. Sussman DL, Tang M, Priebe CE. Consistent latent position estimation and vertex classification for random dot product graphs. *IEEE Trans Pattern Anal Mach Intell* 2014, 36:48–57.
8. Tang M, Sussman DL, Priebe CE. Universally consistent vertex classification for latent positions graphs. *Ann Stat* 2013, 41:1406–1430.
9. Lee N, Leung T, Priebe C. Random graphs based on self-exciting messaging activities, 2011.
10. Marchette D, Priebe C, Coppersmith G. Vertex nomination via attributed random dot product graphs. In: *Proceedings of the 57th ISI World Statistics Congress*, Vol. 6, 16, 2011.
11. Nickel C. Random dot product graphs: a model for social networks. PhD Thesis, Johns Hopkins University, 2006.
12. Young S, Scheinerman E. Random dot product graph models for social networks. Workshop on *Algorithms and Models for the Web-graph*, 2007, 138–149.
13. Priebe C, Conroy J, Marchette D, Park Y. Scan statistics on enron graphs. *Comput Math Organ Theory* 2005, 11:229–247.
14. Priebe C, Park Y, Marchette D, Conroy J, Grothendieck J, Gorin A. Statistical inference on attributed random graphs: fusion of graph features and content: an experiment on time series of enron graphs. *Comput Stat Data Anal* 2010, 54:1766–1776.
15. Grothendieck J, Priebe C, Gorin A. Statistical inference on attributed random graphs: fusion of graph features and content. *Comput Stat Data Anal* 2010, 54:1777–1790.
16. Sun M, Tang M, Priebe C. A comparison of graph embedding methods for vertex nomination. In: *2012 11th International Conference on Machine Learning and Applications (ICMLA)*, IEEE, Volume 1, 2012, 398–403.
17. Coppersmith G. Vertex nomination. *WIREs Comput Stat* 2014, 6:144–153.
18. Fishkind DE, Lyzinski V, Pao H, Chen L, Priebe CE. Vertex nomination schemes for membership prediction. *arXiv preprint arXiv:1312.2638*, 2013.
19. Qi G, Aggarwal C, Qi T, Ji H, Huang T. Exploring context and content links in social media: a latent space method. *IEEE Trans Pattern Anal Mach Intell* 2012, 34:850–862.
20. Manning CD, Raghavan P, Schütze H. *Introduction to Information Retrieval*, vol. 1. Cambridge: Cambridge University Press; 2008.
21. Gelman A, Rubin DB. Inference from iterative simulation using multiple sequences. *Statistical Sci* 1992, 7:457–472.
22. Botev Z, Grotowski J, Kroese D. Kernel density estimation via diffusion. *Ann Stat* 2010, 38:2916–2957.
23. Park L. Bootstrap confidence intervals for mean average precision. In: *Proceedings of the Fourth ASEARC Conference*, 2011, 51–54.
24. Efron B. Better bootstrap confidence intervals. *J Am Stat Assoc* 1987, 82:171–185.
25. Zhang D, Gatica-Perez D, Roy D, Bengio S. Modeling interactions from email communication. In: *2006 I.E. International Conference on Multimedia and Expo*, IEEE, 2006, 2037–2040.
26. Berry MW, Browne M, Signer B. *Topic annotated enron email data set*. Philadelphia: Linguistic Data Consortium; 2001.
27. Aicher C, Jacobs AZ, Clauset A. Learning latent block structure in weighted networks. *J Complex Netw* 2015, 3:221–248.

## APPENDIX A: FULL CONDITIONAL POSTERIOR DISTRIBUTION FOR Y

Continuing from Eq. (13), $\gamma_i$ can be computed as follows:

$$\begin{aligned}
&\frac{1}{\gamma_i(Y_{-i},p_1,p_2,q_2,\psi)}\\
&=1+\frac{f(Y(i)=1,Y_{-i},p_1,p_2,q_2,\psi|T,T')}{f(Y(i)=2,Y_{-i},p_1,p_2,q_2,\psi|T,T')}\\
&=1+\frac{\left[(1-\psi)/\psi]f_1(T(i)|p_1,p_2)\prod_{j:j\neq i,Y(j)=2}f_2(T(j)|m_{-i},p_1,p_2,q_2)\prod_{k=1}^{m'}f'(T'(k)|m_{-i},p_1,p_2,q_2\right)}{f_2(T(i)|m_{-i}+1,p_1,p_2,q_2)\prod_{j:j\neq i,Y(j)=2}f_2(T(j)|m_{-i}+1,p_1,p_2,q_2)\prod_{k=1}^{m'}f'(T'(k)|m_{-i}+1,p_1,p_2,q_2)}\\
&=1+\frac{\left[(1-\psi)/\psi]f_1(T(i)|p_1,p_2)\prod_{j:j\neq i,Y(j)=2}f_2(S(j)|R(j),m_{-i},p_1,p_2,q_2)\prod_{k=1}^{m'}f'(S'(k)|R'(k),m_{-i},p_1,p_2,q_2\right)}{f_2(T(i)|m_{-i}+1,p_1,p_2,q_2)\prod_{j:j\neq i,Y(j)=2}f_2(S(j)|R(j),m_{-i}+1,p_1,p_2,q_2)\prod_{k=1}^{m'}f'(S'(k)|R'(k),m_{-i}+1,p_1,p_2,q_2)},
\end{aligned} \tag{A1}$$

where $m_{-i}=m'+\sum_{j:j\neq i}I_{\{2\}}(Y(j))$.

**The conditional prior distributions of the nuisance parameters: $p_1, p_2, q_2$**

Since the joint prior distribution of the nuisance parameters as described in Section 3 is

$$\begin{aligned}
f(p_1,p_2,q_2) &= f(q_2|p_1,p_2)f(p_1,p_2)\\
&= \text{Uniform}(p_2,1-p_1)\cdot\text{Dirichlet}(\alpha_0,\alpha_1,\alpha_2).
\end{aligned} \tag{A2}$$

Choosing $\alpha_0=\alpha_1=\alpha_2=1$ for the Dirichlet distribution gives

$$f(p_1,p_2,q_2)=\frac{2}{1-p_1-p_2}I_{(0,1)}(p_1)I_{(0,1-p_1)}(p_2)I_{(p_2,1-p_1)}(q_2), \tag{A3}$$

and by choice, we specify

$$f(q_2|\,p_1,p_2)=\text{Uniform}(p_2,1-p_1)=(1-p_1-p_2)^{-1}I_{(p_2,1-p_1)}(q_2). \tag{A4}$$

From Eq. (24), the conditional prior for $p_1|p_2,q_2$ is

$$f(p_1|\,p_2,q_2)\propto(1-p_1-p_2)^{-1}I_{(0,1-q_2)}(p_1), \tag{A5}$$

which, for $p_1\in(0,1-q_2)$, can be written as

$$f(p_1|\,p_2,q_2)=\frac{(1-p_1-p_2)^{-1}}{\log(1-p_2)-\log(q_2-p_2)}. \tag{A6}$$

The corresponding conditional distribution function is

$$F(p_1 \mid p_2, q_2) = \frac{\log(1-p_2) - \log(1-p_1-p_2)}{\log(1-p_2) - \log(q_2-p_2)}, \tag{A7}$$

and so the conditional inverse distribution function is, for $u \in [0, 1]$,

$$F^{-1}(u \mid p_2, q_2) = 1 - p_2 - \frac{(q_2-p_2)^u}{(1-p_2)^{u-1}}. \tag{A8}$$

This enables us to generate from $f(p_1|p_2, q_2)$ easily by the inverse distribution function method. Similarly,

$$f(p_2 \mid p_1, q_2) \propto (1-p_1-p_2)^{-1} I_{(0,q_2)}(p_2), \tag{A9}$$

and so for $p_2 \in (0, q_2)$,

$$f(p_2 \mid p_1, q_2) = \frac{(1-p_1-p_2)^{-1}}{\log(1-p_1) - \log(1-p_1-q_2)} \tag{A10}$$

$$F(p_2 \mid p_1, q_2) = \frac{\log(1-p_1) - \log(1-p_1-p_2)}{\log(1-p_1) - \log(1-p_1-q_2)}, \tag{A11}$$

and

$$F^{-1}(u \mid p_1, q_2) = 1 - p_1 - \frac{(1-p_1-q_2)^u}{(1-p_1)^{u-1}}. \tag{A12}$$