A Higher-order Latent Space Network Model

Nesreen K. Ahmed
Intel Labs
nesreen.k.ahmed@intel.com

Ryan A. Rossi
Palo Alto Research Center
rrossi@parc.com

Theodore L. Willke
Intel Labs
ted.willke@intel.com

Rong Zhou
Palo Alto Research Center
rzhou@parc.com

Abstract

Previous work in network analysis has focused on modeling node roles in the graph. In this work, we introduce edge role discovery and develop a general framework for modeling edge roles in large networks. In addition, a general class of higher-order role discovery methods are proposed that leverage features based on induced subgraphs (graphlets, motifs) for learning better and more useful roles. All methods are fast with a runtime that is linear in the number of edges and able to scale to large real-world networks via an effective parallel implementation. The experimental results demonstrate the utility of edge roles for network analysis tasks on a variety of graphs from various problem domains.

1 Introduction

In the traditional graph-based sense, roles represent node-level connectivity patterns such as star-center, star-edge nodes, near-cliques or nodes that act as bridges to different regions of the graph. Intuitively, two nodes belong to the same role if they are "similar" in the sense of graph structure. Our proposed research will broaden the framework for defining, discovering and learning network roles, by drastically increasing the degree of usefulness of the information embedded within rich graphs.

Recently, role discovery has become increasingly important for a variety of application and problem domains (Borgatti, Everett, and Johnson 2013; HollandKathryn Blackmond and Leinhardt 1983; Arabie, Boorman, and Levitt 1978; Anderson, Wasserman, and Faust 1992; Rossi and Ahmed 2015b; Lorrain and White 1971; White and Reitz 1983) including dynamic network analysis (Fu, Song, and Xing 2009), classification (Henderson et al. 2012), anomaly detection (Rossi et al. 2013), and sensemaking/exploratory analysis (Airoldi et al. 2008). Despite the importance of role discovery, existing work has only focused on discovering node roles (Anderson, Wasserman, and Faust 1992; Batagelj et al. 2004; Doreian, Batagelj, and Ferligoj 2005; Nowicki and Snijders 2001). We posit that discovering the roles of edges may be fundamentally more important and able to capture, represent, and summarize the key behavioral roles in the network better than existing methods that have been limited to learning only the roles of nodes in the graph. For instance, a person with malicious intent may appear normal by maintaining the vast majority of relationships and communications with individuals that play normal roles in society. In this situation, techniques that reveal the role semantics of nodes would have difficulty detecting such malicious behavior since most edges are normal. However, modeling the roles (functional semantics, intent) of individual edges (relationships, communications) in the rich graph would improve our ability to identify, detect, and predict this type of malicious activity since we are modeling it directly. Nevertheless, existing work also has many other limitations, which significantly reduces the practical utility of such methods in real-world networks. For instance, most existing work on node roles are based on simple degree and egonet features (Henderson et al. 2012; Rossi et al. 2013). Instead, this work leverages small induced subgraphs called graphlets (motifs) (Ahmed et al. 2016) to learn more meaningful and useful roles in large networks.

The main contributions are as follows:

- **Edge role discovery:** This work introduces the problem of edge role discovery and proposes a computational framework for learning and modeling edge roles in both static and dynamic networks.
- **Higher-order latent space model:** Introduced a higher-order latent role model that leverages higher-order network features for learning and modeling node and edge roles. We also introduced graphlet-based roles and proposed feature and role learning techniques.
- **Efficient and scalable:** All proposed algorithms are parallelized. Moreover, the feature and role learning and inference algorithms are linear in the number of edges.

2 Edge Role Discovery Framework

This section introduces our higher-order edge role model and a flexible framework for computing edge roles based on higher-order network features.

2.1 Extracting Higher-order Graphlet Features

Given the graph $G = (V, E)$, we first decomposes G into its smaller subgraph components called graphlets (motifs). For this, we use parallel edge-centric graphlet decomposition

Algorithm 1 A flexible parallel edge feature representation learning framework for large networks. Given a graph $G = (V, E)$, the framework outputs a matrix $\mathbf{X} \in \mathbb{R}^{m \times f}$ of edge features.

Input:

a directed and possibly weighted/labeled/attributed graph $G = (V, E)$, a set of relational edge kernels/operators $\mathbf{\Phi}$, a feature similarity function $\mathbb{K}\langle \cdot, \cdot \rangle$, an upper bound on the number of feature layers to learn T, a feature similarity threshold λ, and bin size α, $0 \leq \alpha \leq 1$.

1: Set $\tau \leftarrow 1$
2: **parallel for each** $e_i \in E$ **and** subgraph $H_k \in \mathcal{H}$ **do**
3: Set X_{ik} to the number of instances of H_k that contain $e_i \in E$
4: Construct in/out/total/weighted *edge egonet* and *edge degree* features (feature layer $\mathcal{F}_1$ which includes the graphlet features as well). Append these to $\mathbf{X}$ and set $\mathcal{F} \leftarrow \mathcal{F}_1$.
5: **repeat** ▷ feature layers $\mathcal{F}_\tau$ for $\tau = 1, 2, ..., \mathrm{T}$
6: **if** $\tau > 1$ **then**
7: Derive candidate features for feature layer $\mathcal{F}_\tau$ using the set of relational operators $\mathbf{\Phi}$ over each of the novel features $f_i \in \mathcal{F}_{\tau-1}$ learned in previous layers. Append the candidate features to $\mathbf{X}$ and the feature definitions to $\mathcal{F}_\tau$.
8: For each feature $f_i \in \mathcal{F}_\tau$, sort the feature values in ascending order and then map the feature values using logarithmic binning (with a bin size of α). Given feature $f_i \in \mathcal{F}_\tau$, we set the αm edges with smallest feature values to 0, then α edges remaining are set to 1, and so on.
9: Initialize the feature graph $\mathcal{G}_F = (V_F, E_F)$ for feature layer $\mathcal{F}_\tau$ where V_F is the set of features from $\mathcal{F} \cup \mathcal{F}_\tau$ and $E_F = \varnothing$
10: **parallel for each** edge feature $f_i \in \mathcal{F}_\tau$ **do**
11: **for each** edge feature $f_j \in \mathcal{F}$ **do**
12: **if** $\mathbb{K}(\mathbf{x}_i, \mathbf{x}_j) \geq \lambda$ **then**
13: Add edge (f_i, f_j) to E_F
14: Partition $\mathcal{G}_F$ using connected components $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots\}$
15: **parallel for each** $\mathcal{C}_k \in \mathcal{C}$ **do** ▷ Prune features
16: Find the earliest feature f_i s.t. $\forall f_j \in \mathcal{C}_k : i < j$.
17: Remove $\mathcal{C}_k$ from $\mathcal{F}_\tau$ and set $\mathcal{F}_\tau \leftarrow \mathcal{F}_\tau \cup \{f_i\}$
18: Discard features not in $\mathcal{F}_\tau$ from $\mathbf{X}$ and set $\mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{F}_\tau$
19: Set $\tau \leftarrow \tau + 1$ and initialize $\mathcal{F}_\tau$ to $\varnothing$ for next feature layer
20: **until** no new features emerge **or** max layers reached
21: **return** $\mathbf{X}$ and the set of feature definitions $\mathcal{F}$

methods such as (Ahmed et al. 2015) to compute a variety of graphlet edge features of size $k = \{3, 4, \ldots\}$ (Alg. 1 Line 2). Moreover, our approach can leverage directed, undirected, and weighted/typed graphlet counts (among other useful and discriminative graphlet edge statistics) using either exact or estimation methods. For instance, we could have used a recent method for estimating local subgraph (graphlet, motif) counts (Ahmed, Willke, and Rossi 2016). These graphlet features are then used to learn deeper higher-order edge features (see below for further details).

2.2 Edge Feature Learning

This section presents our deep edge feature representation learning framework (Alg. 1). Recall that our approach leverages the previous higher-order graphlet counts as a basis for learning deeper and more discriminative higher-order edge features (Line 2-3). Next, primitive edge features are computed in Line 4, including in/out/total/weighted *edge egonet* and *edge degree* features. After computing the initial feature layer $\mathcal{F}_1$ (Line 2-4), redundant features are pruned (Line 5-20). The framework proceeds to learn a set of feature layers where each successive layer represents increasingly deeper higher-order edge features (Line 5-20), *i.e.*, $\mathcal{F}_1 < \mathcal{F}_2 < \cdots < \mathcal{F}_\tau$ such that if $i < j$ then $\mathcal{F}_j$ is said to be a deeper layer than $\mathcal{F}_i$.

The feature layers $\mathcal{F}_2, \cdots, \mathcal{F}_\tau$ are learned as follows (Line 5-20): For each layer $\mathcal{F}_\tau$, we first construct and search candidate features using the set of relational edge feature operators $\mathbf{\Phi}$ (See Line 7), which include mean, sum, product, $\min$, $\max$, variance, L1, L2, and even parameterized relational kernels based on RBF, polynomial functions, among others. Now, we compute the similarity between all pairs of features and prune edges between features that are *not* significantly correlated (Line 9-13): $E_F = \{(f_i, f_j) \mid \forall (f_i, f_j) \in |\mathcal{F}| \times |\mathcal{F}| \text{ s.t. } \mathbb{K}(f_i, f_j) > \lambda\}$. This process results in a feature similarity graph where large edge weights indicate strong similarity/correlation between two features. Next, the feature similarity graph $\mathcal{G}_F$ from Line 9-13 is used to prune all redundant edge features from $\mathcal{F}_\tau$. Features are pruned by first partitioning the feature graph (Line 14) using connected components, though our approach is flexible and allows other possibilities (*e.g.*, largest clique). Intuitively, each connected component is a set of redundant edge features since edges in $\mathcal{G}_F$ represent strong dependencies between features. For each connected component $\mathcal{C}_k \in \mathcal{C}$ (Line 15-17), we identify the earliest feature in $\mathcal{C}_k = \{..., f_i, ..., f_j, ...\}$ (Line 16) and remove all others from $\mathcal{F}_\tau$ (Line 17). After pruning the feature layer $\mathcal{F}_\tau$, Line 18 ensures the pruned features are removed from $\mathbf{X}$ and updates the set of edge features learned thus far by setting $\mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{F}_\tau$. Line 19 increments τ and set $\mathcal{F}_\tau \leftarrow \varnothing$. Finally, Line 20 checks for convergence, and if the stopping criterion is not satisfied, then the approach tries to learn an additional feature layer (Line 5-20).

2.3 Role Assignment

Let $\mathbf{X} = \begin{bmatrix} x_{ij} \end{bmatrix} \in \mathbb{R}^{m \times f}$ be a matrix with m rows representing edges and f columns representing arbitrary features[1]. More formally, given $\mathbf{X} \in \mathbb{R}^{m \times f}$, the edge role discovery optimization problem is to find $\mathbf{U} \in \mathbb{R}^{m \times r}$ and $\mathbf{V} \in \mathbb{R}^{f \times r}$ where $r \ll \min(m, f)$ such that the product of two lower rank matrices $\mathbf{U}$ and $\mathbf{V}^T$ minimizes the divergence between $\mathbf{X}$ and $\mathbf{X}' = \mathbf{U}\mathbf{V}^T$. Intuitively, $\mathbf{U} \in \mathbb{R}^{m \times r}$ represents the latent *role mixed-memberships* of the edges whereas $\mathbf{V} \in \mathbb{R}^{f \times r}$ represents the contributions of the features with respect to each of the roles. Each row $\mathbf{u}_i^T \in \mathbb{R}^r$ of $\mathbf{U}$ can be interpreted as a low dimensional rank-r embedding of the i^{th} *edge* in $\mathbf{X}$. Alternatively, each row $\mathbf{v}_j^T \in \mathbb{R}^r$ of $\mathbf{V}$ represents a r-dimensional role embedding of the j^{th} feature in $\mathbf{X}$ using the same low rank-r dimensional space. Also, $\mathbf{u}_k \in \mathbb{R}^m$ is the k^{th} column representing a "latent feature" of $\mathbf{U}$ and similarly $\mathbf{v}_k \in \mathbb{R}^f$ is the k^{th} column of $\mathbf{V}$.

[1]For instance, the columns of $\mathbf{X}$ represent arbitrary features such as graph topology features, non-relational features/attributes, and relational neighbor features, among other possibilities.

For the higher-order latent network model, we solve:

$$\underset{(\mathbf{U},\mathbf{V})\in\mathcal{C}}{\arg\min} \Big\{ \mathbb{D}_\phi(\mathbf{X} \| \mathbf{U}\mathbf{V}^T) + \mathcal{R}(U,V) \Big\} \tag{1}$$

where $\mathbb{D}_\phi(\mathbf{X}\|\mathbf{U}\mathbf{V}^T)$ is an arbitrary Bregman divergence (Bregman 1967) between $\mathbf{X}$ and $\mathbf{U}\mathbf{V}^T$. Furthermore, the optimization problem in (1) imposes hard constraints $\mathcal{C}$ on $\mathbf{U}$ and $\mathbf{V}$ such as non-negativity constraints $\mathbf{U}, \mathbf{V} \geq 0$ and $\mathcal{R}(U,V)$ is a regularization penalty. In this work, we mainly focus on solving $\mathbb{D}_\phi(\mathbf{X}\|\mathbf{U}\mathbf{V}^T)$ under non-negativity constraints:

$$\underset{\mathbf{U}\geq 0,\mathbf{V}\geq 0}{\arg\min} \Big\{ \mathbb{D}_\phi(\mathbf{X} \| \mathbf{U}\mathbf{V}^T) + \mathcal{R}(U,V) \Big\} \tag{2}$$

Given the edge feature matrix $\mathbf{X} \in \mathbb{R}^{m\times f}$, the edge role discovery problem is to find $\mathbf{U} \in \mathbb{R}^{m\times r}$ and $\mathbf{V} \in \mathbb{R}^{f\times r}$ such that

$$\mathbf{X} \approx \mathbf{X}' = \mathbf{U}\mathbf{V}^T \tag{3}$$

To measure the quality of our edge mixed membership model, we use Bregman divergences:

$$\sum_{ij} \mathbb{D}_\phi(x_{ij} \| x'_{ij}) = \sum_{ij} \big(\phi(x_{ij}) - \phi(x'_{ij}) - \ell(x_{ij}, x'_{ij}) \big)$$

where ϕ is a univariate smooth convex function and

$$\ell(x_{ij}, x'_{ij}) = \nabla\phi(x'_{ij})(x_{ij} - x'_{ij}),$$

where $\nabla^p\phi(x)$ is the p-order derivative operator of ϕ at x. Furthermore, let $\mathbf{X} - \mathbf{U}\mathbf{V}^T = \mathbf{X}^{(k)} - \mathbf{u}_k\mathbf{v}_k^T$ denote the residual term in the approximation (3) where $\mathbf{X}^{(k)}$ is the k-residual matrix defined as:

$$\mathbf{X}^{(k)} = \mathbf{X} - \sum_{h\neq k} \mathbf{u}_h\mathbf{v}_h^T \tag{4}$$

$$= \mathbf{X} - \mathbf{U}\mathbf{V}^T + \mathbf{u}_k\mathbf{v}_k^T, \quad \text{for } k = 1,\ldots,r \tag{5}$$

We use a fast *scalar block coordinate descent approach* that easily generalizes for heterogeneous networks (Rossi and Zhou 2016). The approach considers a single element in $\mathbf{U}$ and $\mathbf{V}$ as a block in the block coordinate descent framework. Replacing $\phi(y)$ with the corresponding expression from Table 1 gives rise to a fast algorithm for each Bregman divergence. Table 1 gives the updates for Frobenius norm (Fro.), KL-divergence (KL), and Itakura-Saito divergence (IS). Note that Beta divergence and many others are also easily adapted for our higher-order network modeling framework.

2.4 Model Selection

Since it is unrealistic to expect a domain expert to manually select the appropriate model, this section introduces an approach for learning the appropriate model given an arbitrary graph. The approach leverages the Minimum Description Length (MDL) (Grünwald 2007; Rissanen 1978) principle for automatically selecting the "best" higher-order network model. The MDL principle is a practical formalization

Table 1: Summary of update rules

	$\phi(y)$	$\nabla^2\phi(y)$	$\mathbb{D}_\phi(x\|x')$	**Update**
Fro.	$y^2/2$	1	$(x-x')^2/2$	$v_{jk} = \frac{\sum_{i=1}^m x_{ij}^{(k)} u_{ik}}{\sum_{i=1}^m u_{ik}u_{ik}}$
KL	$y\log y$	$1/y$	$x\log\frac{x}{x'} - x + x'$	$v_{jk} = \frac{\sum_{i=1}^m x_{ij}^{(k)} u_{ik}/x'_{ij}}{\sum_{i=1}^m u_{ik}u_{ik}/x'_{ij}}$
IS	$-\log y$	$1/y^2$	$\frac{x}{x'} - \log\frac{x}{x'}$	$v_{jk} = \frac{\sum_{i=1}^m x_{ij}^{(k)} u_{ik}/{x'_{ij}}^2}{\sum_{i=1}^m u_{ik}u_{ik}/{x'_{ij}}^2}$

of Kolmogorov complexity (Li and Vitányi 2009). More formally, the approach finds the model $\mathcal{M}_\star = (\mathbf{V}_r, \mathbf{U}_r)$ that leads to the best compression by solving:

$$M_\star = \underset{M\in\mathcal{M}}{\arg\min}\ \mathcal{L}(M) + \mathcal{L}(\mathbf{X} \,|\, M) \tag{6}$$

where $\mathcal{M}$ is the model space, $M_\star$ is the model given by the solving the above minimization problem, and $\mathcal{L}(M)$ as the number of bits required to encode M using code Ω, which we refer to as the description length of M with respect to Ω. Recall that MDL requires a lossless encoding. Therefore, to reconstruct $\mathbf{X}$ *exactly* from $M = (\mathbf{U}_r, \mathbf{V}_r)$ we must explicitly encode the error $\mathbf{E}$ such that

$$\mathbf{X} = \mathbf{U}_r\mathbf{V}_r^T + \mathbf{E}$$

Hence, the total compressed size of $M = (\mathbf{U}_r, \mathbf{V}_r)$ with $M \in \mathcal{M}$ is simply $\mathcal{L}(X, M) = \mathcal{L}(M) + \mathcal{L}(\mathbf{E})$. Given an arbitrary model $M = (\mathbf{U}_r, \mathbf{V}_r) \in \mathcal{M}$, the description length is decomposed into:

- Bits required to describe the model
- Cost of describing the approximation errors $\mathbf{X} - \mathbf{X}_r = \mathbf{U}_r\mathbf{V}_r^T$ where $\mathbf{X}_r$ is the rank-r approximation of $\mathbf{X}$,

$$\mathbf{U}_r = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_r \end{bmatrix} \in \mathbb{R}^{m\times r}, \quad \text{and} \tag{7}$$

$$\mathbf{V}_r = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_r \end{bmatrix} \in \mathbb{R}^{f\times r} \tag{8}$$

The model $M_\star$ is the model $M \in \mathcal{M}$ that minimizes the total description length: the model description cost X and the cost of correcting the errors of our model. Let $|\mathbf{U}|$ and $|\mathbf{V}|$ denote the number of nonzeros in $\mathbf{U}$ and $\mathbf{V}$, respectively. Thus, the model description cost of M is: $\kappa r(|\mathbf{U}| + |\mathbf{V}|)$ where κ is the bits per value. Similarly, if $\mathbf{U}$ and $\mathbf{V}$ are dense, then the model description cost is simply $\kappa r(m+f)$ where m and f are the number of edges and features, respectively. Assuming errors are non-uniformly distributed, one possibility is to use KL divergence (see Table 1) for the error description cost[2]. The cost of correcting a single element in the approximation is $\mathbb{D}_\phi(x\|x') = x\log\frac{x}{x'} - x + x'$ (assuming KL-divergence), and thus, the total reconstruction cost is:

$$\mathbb{D}_\phi(\mathbf{X}\|\mathbf{X}') = \sum_{ij} X_{ij}\log\frac{X_{ij}}{X'_{ij}} - X_{ij} + X'_{ij} \tag{9}$$

where $\mathbf{X}' = \mathbf{U}\mathbf{V}^T \in \mathbb{R}^{m\times f}$. Other possibilities are given in Table 1. The above assumes a particular representation scheme for encoding the models and data. Recall

[2]The representation cost of correcting approximation errors

that the optimal code assigns $\log_2 p_i$ bits to encode a message (Shannon 1948). Lloyd-Max quantization (Max 1960; Lloyd 1982) with Huffman codes (Huffman and others 1952; Van Leeuwen 1976) are used to compress the model and data (Oliver, Pierce, and Shannon 1948; Bennett 1948). Notice that we require only the length of the description using the above encoding scheme, and thus we do not need to materialize the codes themselves. This leads to the improved model description cost: $\bar{\kappa} r(|\mathbf{U}| + |\mathbf{V}|)$ where $\bar{\kappa}$ is the mean bits required to encode each value[3].

3 Dynamic Edge Role Membership Model

This section introduces the *dynamic edge role mixed-membership model* (DERM) and proposes a computational framework for computing edge roles in dynamic networks.

3.1 Dynamic Graph Model & Representation

Given a graph stream $G = (V, E)$ where $E = \{e_1, \ldots, e_k, e_{k+1}, \ldots, e_m\}$ is an ordered set of edges in the graph stream such that $\tau(e_1) \leq \tau(e_2) \leq \cdots \leq \tau(e_m)$. Note that $\tau(e_i)$ is the edge time for $e_i \in E$ (which may be the edge activation time, arrival time, among other possibilities). Intuitively, E is an infinite edge streaming network where edges arrive continuously over time. From this edge stream, we derive a dynamic network $\mathcal{G} = \{G_t\}_{t=1}^{T}$ where $G_t = (V, E_t)$ represents a snapshot graph at time t. Note that time t is actually a discrete time interval $[a, b)$ where a and b are the start and end time, respectively. Therefore, $E_t = \{e_t \in E \mid a \leq \tau(e_i) < b\}$ and $E = E_1 \cup E_2 \cup \cdots \cup E_T$.

3.2 Dynamic Edge Role Learning

We start by learning a time series of features automatically. Let $G_{1:k} = (V, E_{1:k})$ be the initial dynamic training graph where $E_{1:k} = E_1 \cup \cdots \cup E_k$ and k represents the number of snapshot graphs to use for learning the initial set of (representative) dynamic features. Given $\{G_t\}_{t=1}^{T}$ and $G_{1:k} = (V, E_{1:k})$, the proposed approach automatically learns a set of features $\mathcal{F} = \{f_1, f_2, \ldots, f_d\}$ where each $f_i \in \mathcal{F}$ represents a learned feature definition from $G_{1:k}$. Given the learned role definitions $\mathbf{V} \in \mathbb{R}^{r \times d}$ using a subset of past temporal graphs, we then estimate the edge role memberships $\{\mathbf{U}_t\}_{t=1}^{T}$ for each $\{G_t\}_{t=1}^{T}$ (and any future graph snapshots $G_{t+1}, \ldots, G_{t+p}$) where $\mathbf{U}_t \in \mathbb{R}^{m \times r}$ is an edge by role membership matrix. The dynamic edge role model is selected using the approach proposed in Section 2.4.

4 Experiments

This section investigates the effectiveness and scalability of the higher-order latent space modeling framework (Section 2). All network data is from Network Repository[4] (Rossi and Ahmed 2015a).

Scalability: We investigate the scalability of the parallel framework for modeling higher-order latent edge roles. To evaluate the effectiveness of the parallel modeling framework, we measure the speedup defined as simply $S_p = T_1/T_p$ where T_1 is the execution time of the sequential algorithm, and T_p is the execution time of the parallel algorithm with p processing units. Overall, the methods show strong scaling (See Figure 1). Similar results were observed for other networks. As an aside, the experiments in Figure 1 used a 4-processor Intel Xeon E5-4627 v2 3.3GHz CPU.

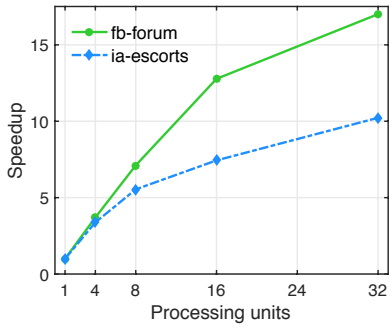


Figure 1: Higher-order role discovery shows strong scaling as we increase the number of processing units.

Higher-order Model Selection: MDL is used to automatically learn the appropriate edge role model. In Figure 2, description length (in bits) is minimized when $r = 18$. Intuitively, too many roles increases the model description cost, whereas too few roles increases the cost of describing errors. In addition, Figure 3 shows the runtime of our approach. Furthermore, Figure 5 demonstrates the impact on the learning time, number of novel features discovered, and their sparsity, as the tolerance (ε) and bin size (α) varies.

Modeling Dynamic Networks: In this section, we investigate the Enron email communication networks using the proposed *dynamic edge role mixed-membership model*. The Enron email data consists of 151 Enron employees whom have sent 50.5k emails to other Enron employees. The email communications are from 05/11/1999 to 06/21/2002. For learning edge roles (and a set of representative edge features), we leverage the first year of emails. Note that other

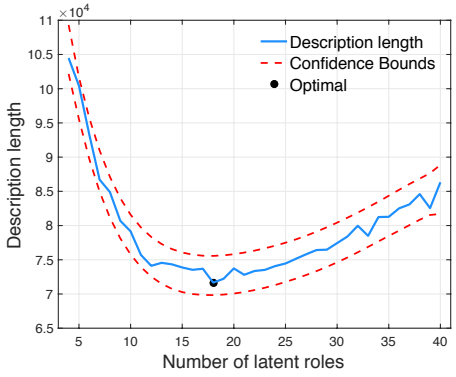


Figure 2: In the example shown, the valley identifies the correct number of latent roles.

[3]Note $\log_2(m)$ quantization bins are used

[4]http://networkrepository.com

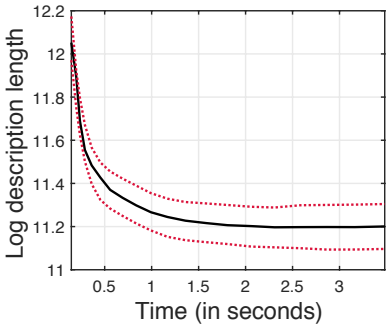
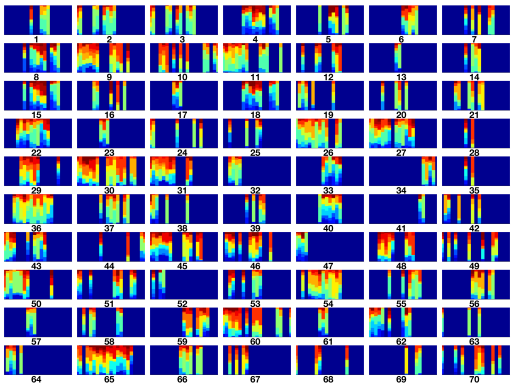


Figure 3: The running time of our approach. The x-axis is time in seconds and the y-axis is the log description cost. The curve is the average over 50 experiments and the dotted lines represent three standard deviations. The result reported above is from running on a single core.

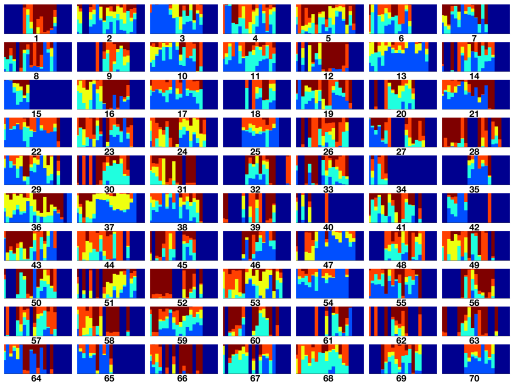
work such as dMMSB (Fu, Song, and Xing 2009) use email communications from 2001 only, which corresponds to the time period that the Enron scandal was revealed (October 2001). We instead study a more difficult problem. In particular, given only past data, can we actually uncover and detect the key events leading up to the downfall of Enron? A dynamic network $\{G_t\}_{t=1}^{T}$ is constructed from the remaining email communications (approximately 2 years) where each snapshot graph G_t represents a month of communications. Interestingly, we learn a *dynamic node role mixed-membership model* with 5 latent roles, which is exactly the number of *latent node roles* learned by dMMSB (Fu, Song, and Xing 2009). However, we learn a dynamic *edge* role mixed-membership model with 18 roles. Evolving edge and node mixed-memberships from the Enron email communication network are shown in Figure 4. Note that the first role in Figure 4 represents inactivity (dark blue). The set of edges and nodes visualized in Figure 4 are selected using the difference entropy rank in Eq. (10) and correspond to the edges and nodes with largest difference entropy rank $\mathbf{d}$ defined as:

$$\mathbf{d} = \max_{t \in T} H(\mathbf{u}_t) - \min_{t \in T} H(\mathbf{u}_t) \tag{10}$$

where $H(\mathbf{u}_t) = -\mathbf{u}_t \cdot \log(\mathbf{u}_t)$ and $\mathbf{u}_t$ is the r-dimensional mixed-membership vector for an edge (or node) at time t. The difference entropy rank reveals important communications between key players involved in the Enron Scandal, such as Kenneth Lay and Jeffrey Skilling. In particular, Kenneth Lay is the former CEO and Chairman of Enron, and was found guilty of 10 counts of securities fraud, whereas Jeffrey Skilling is the former COO and CEO of Enron, and was convicted of federal felony charges relating to Enron's collapse. As an aside, the Enron data is a standard graph-based anomaly detection data set (Akoglu, McGlohon, and Faloutsos 2010; Chen, Hendrix, and Samatova 2012; Fu, Song, and Xing 2009). Notice that when node roles are used for identifying dynamic anomalies in the graph, we are only provided with potentially malicious employees, whereas using edge roles naturally allow us to not only detect the key malicious individuals involved, but also the important relationships between them, which can be used for further analysis, among



(a) Evolving *edge role* mixed-memberships



(b) Evolving *node role* mixed-membership

Figure 4: Temporal changes in the edge and node mixed-membership vectors (from Enron). The horizontal axes of each subplot is time, whereas the vertical axes represent the components of each mixed-membership vector. Roles are represented by different colors.

other possibilities.

Exploratory Analysis: Figure 6 visualizes the node and edge roles learned for ca-netscience. While our higher-order latent space model learns a stochastic r-dimensional vector for each edge (and/or node) representing the individual role memberships, Figure 6 assigns a single role to each link and node for simplicity. In particular, given an edge $e_i \in E$ (or node) and its mixed-membership row vector $\mathbf{u}_i$, we assign e_i the role with maximum likelihood $k_\star \leftarrow \arg\ \max_k u_{ik}$. The higher-order edge and node roles from Figure 6 are clearly meaningful. For instance, the red edge role represents a type of bridge relationship as shown in Figure 6.

Sparse Graph Feature Learning: Recall that the proposed feature learning approach attempts to learn "sparse graph features" to improve learning and efficiency, especially in terms of space-efficiency. This section investigates the effectiveness of our sparse graph feature learning approach. Results are presented in Table 2. In all cases, our approach learns a highly compressed representation of the graph, requiring only a fraction of the space of current (node) approaches. Moreover, the density of edge and node

t/b	0.5	0.6	0.7	0.8	0.9
0.01	1.48	0.95	0.57	0.47	0.41
0.05	1.03	0.55	0.48	0.46	0.45
0.1	0.72	0.57	0.54	0.51	0.48
0.2	0.78	0.58	0.55	0.52	0.49
0.5	0.58	0.56	0.54	0.6	0.56

(a) Learning time

t/b	0.5	0.6	0.7	0.8	0.9
0.01	327	149	81	46	26
0.05	168	73	48	31	18
0.1	111	53	42	26	18
0.2	94	49	36	24	18
0.5	39	33	30	21	16

(b) Number of features discovered

t/b	0.5	0.6	0.7	0.8	0.9
0.01	0.151	0.158	0.136	0.097	0.077
0.05	0.23	0.209	0.169	0.111	0.084
0.1	0.235	0.23	0.186	0.133	0.084
0.2	0.24	0.223	0.222	0.143	0.084
0.5	0.319	0.276	0.242	0.158	0.094

(c) Sparsity of features

Figure 5: Impact on the learning time, number of features, and their sparsity, as the tolerance (ε) and bin size (α) varies.

feature representations learned by our approach is between $[0.164, 0.318]$ and $[0.162, 0.334]$ for nodes (See $\rho(\mathbf{X})$ and $\rho(\mathbf{Z})$ in Table 2) and up to $6x$ more space-efficient than other approaches. While existing feature learning approaches for graphs are unable to learn higher-order graph features (and thus impractical for higher-order network analysis and modeling), they also have another fundamental disadvantage: they return dense features. Learning space-efficient features is critical especially for large networks. For instance, notice that on extremely large networks, storing even a small number of edge (or node) features quickly becomes impractical. Despite the importance of learning sparse graph features, existing work has ignored this problem as most approaches stem from Statistical Relational Learning (SRL) (Getoor and Taskar 2007) and have been designed for extremely small graphs. Moreover, nearly all existing methods focus on node features (Davis et al. 2007; Kok and Domingos 2007; Landwehr et al. 2006; Landwehr, Kersting, and De Raedt 2005), whereas we focus on both and primarily on learning novel and important edge feature representations from large massive networks.

Computational Complexity: Recall that m is the number of edges, f is the number of features, and r is the number of latent roles. The total computational complexity of the *higher-order latent space model* is $\mathcal{O}\big(f(mf + mr)\big)$. The computational complexity is decomposed into the following main parts: Edge feature learning takes $\mathcal{O}(f(m+mf))$. Model learning takes $\mathcal{O}(mfr)$ in the worst case (which arises when $\mathbf{U}$ and $\mathbf{V}$ are completely dense). The quantization and Huffman coding terms are very small and therefore ignored. Role assignment using scalar element-wise coordinate descent has worst case complexity of $\mathcal{O}(mfr)$ per iteration which arises when $\mathbf{X}$ is completely dense. We assume the initial graphlet features are computed using fast and accurate estimation methods, see Ahmed *et al.* (2016).

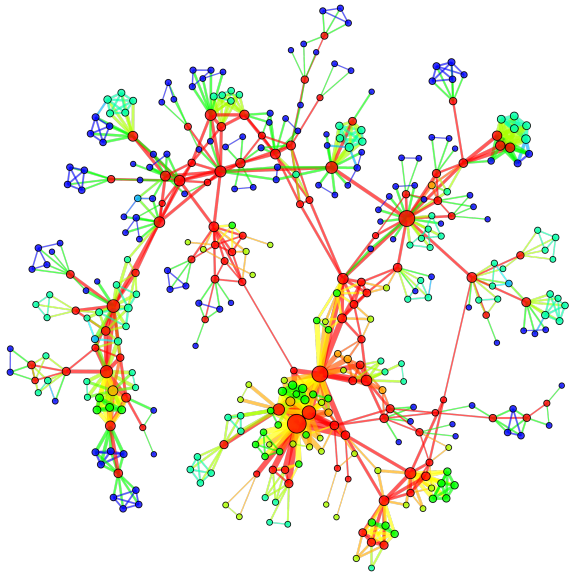


Figure 6: Edge and node roles for ca-netscience. Link color represents the edge role and node color indicates the corresponding node role.

Table 2: Higher-order sparse graph feature learning for node and edge role discovery. The approach is clearly space-efficient and requires significantly less memory than existing approaches. Recall that f is the number of features, L is the number of layers, and $\rho(\mathbf{X})$ is the sparsity of the feature matrix. Edge values are bold.

graph	f		L		$\rho(\mathbf{X})$	$\rho(\mathbf{Z})$
socfb-MIT	**2080**	(912)	**8**	(9)	**0.318**	(0.334)
yahoo-msg	**1488**	(405)	**7**	(7)	**0.164**	(0.181)
enron	**843**	(109)	**5**	(4)	**0.312**	(0.320)
Facebook	**1033**	(136)	**7**	(5)	**0.187**	(0.162)
bio-DD21	**379**	(723)	**6**	(6)	**0.215**	(0.260)

5 Conclusion

This paper introduced the *edge role discovery* problem and presented a computational framework for learning and extracting edge roles from large networks. In addition, higher-order role discovery methods were proposed that leverage network motifs (including all induced subgraphs of size 3, 4, and larger) for learning more discriminative and useful roles. This work also presented an edge feature learning approach for edge role discovery. All methods are fast with a runtime that is linear in the number of edges and able to scale to large real-world networks via an effective parallel implementation. Furthermore, the proposed class of higher-order role models naturally support directed, undirected, and bipartite graphs that are attributed, typed, and/or signed. Finally, the higher-order role discovery methods are effective for a variety of descriptive and predictive modeling tasks.

References

Ahmed, N. K.; Neville, J.; Rossi, R. A.; and Duffield, N. 2015. Efficient graphlet counting for large networks. In *ICDM*, 10.

Ahmed, N. K.; Neville, J.; Rossi, R. A.; Duffield, N.; and Willke, T. L. 2016. Graphlet decomposition: Framework, algorithms, and applications. *Knowledge and Information Systems (KAIS)* 1–32.

Ahmed, N. K.; Willke, T. L.; and Rossi, R. A. 2016. Estimation of local subgraph counts. In *Proceedings of the IEEE International Conference on BigData*, 1–10.

Airoldi, E. M.; Blei, D. M.; Fienberg, S. E.; and Xing, E. P. 2008. Mixed membership stochastic blockmodels. *JMLR* 9(Sep):1981–2014.

Akoglu, L.; McGlohon, M.; and Faloutsos, C. 2010. Oddball: Spotting anomalies in weighted graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 410–421. Springer.

Anderson, C.; Wasserman, S.; and Faust, K. 1992. Building stochastic blockmodels. *Social Networks* 14(1):137–161.

Arabie, P.; Boorman, S.; and Levitt, P. 1978. Constructing blockmodels: How and why. *Journal of Mathematical Psychology* 17(1):21–63.

Batagelj, V.; Mrvar, A.; Ferligoj, A.; and Doreian, P. 2004. Generalized blockmodeling with pajek. *Metodoloski zvezki* 1:455–467.

Bennett, W. R. 1948. Spectra of quantized signals. *Bell System Technical Journal* 27(3):446–472.

Borgatti, S.; Everett, M.; and Johnson, J. 2013. *Analyzing Social Networks*. Sage Publications.

Bregman, L. M. 1967. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Math. and Mathematical Physics* 7(3):200–217.

Chen, Z.; Hendrix, W.; and Samatova, N. F. 2012. Community-based anomaly detection in evolutionary networks. *Journal of Intelligent Information Systems* 39(1):59–85.

Davis, J.; Ong, I. M.; Struyf, J.; Burnside, E. S.; Page, D.; and Costa, V. S. 2007. Change of representation for statistical relational learning. In *IJCAI*, 2719–2726.

Doreian, P.; Batagelj, V.; and Ferligoj, A. 2005. *Generalized Blockmodeling*, volume 25. Cambridge University Press.

Fu, W.; Song, L.; and Xing, E. P. 2009. Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 329–336.

Getoor, L., and Taskar, B., eds. 2007. *Introduction to Statistical Relational Learning*. MIT Press.

Grünwald, P. D. 2007. *The minimum description length principle*. MIT press.

Henderson, K.; Gallagher, B.; Eliassi-Rad, T.; Tong, H.; Basu, S.; Akoglu, L.; Koutra, D.; Faloutsos, C.; and Li, L. 2012. Rolx: Structural role extraction & mining in large graphs. In *SIGKDD*, 1231–1239.

HollandKathryn Blackmond, P., and Leinhardt, S. 1983. Stochastic blockmodels: First steps. *Social Networks* 5(2):109–137.

Huffman, D. A., et al. 1952. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE* 40(9):1098–1101.

Kok, S., and Domingos, P. 2007. Statistical predicate invention. In *ICML*, 433–440. ACM.

Landwehr, N.; Passerini, A.; De Raedt, L.; and Frasconi, P. 2006. kfoil: Learning simple relational kernels. In *AAAI*, volume 6, 389–394.

Landwehr, N.; Kersting, K.; and De Raedt, L. 2005. nFOIL: Integrating naıve bayes and FOIL. In *AAAI*, 795–800.

Li, M., and Vitányi, P. 2009. *An introduction to Kolmogorov complexity and its applications*. Springer Science & Business Media.

Lloyd, S. 1982. Least squares quantization in pcm. *Transactions on Information Theory* 28(2):129–137.

Lorrain, F., and White, H. 1971. Structural equivalence of individuals in social networks. *Journal of Mathematical Sociology* 1(1):49–80.

Max, J. 1960. Quantizing for minimum distortion. *Transactions on Information Theory* 6(1):7–12.

Nowicki, K., and Snijders, T. 2001. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association* 96(455):1077–1087.

Oliver, B.; Pierce, J.; and Shannon, C. E. 1948. The philosophy of pcm. *Proceedings of the IRE* 36(11):1324–1331.

Rissanen, J. 1978. Modeling by shortest data description. *Automatica* 14(5):465–471.

Rossi, R., and Ahmed, N. 2015a. The network data repository with interactive graph analytics and visualization. In *AAAI*, volume 15, 4292–4293.

Rossi, R. A., and Ahmed, N. K. 2015b. Role discovery in networks. *TKDE* 27(4):1112–1131.

Rossi, R. A., and Zhou, R. 2016. Parallel Collective Factorization for Modeling Large Heterogeneous Networks. In *Social Network Analysis and Mining*, 30.

Rossi, R. A.; Gallagher, B.; Neville, J.; and Henderson, K. 2013. Modeling dynamic behavior in large evolving graphs. In *WSDM*, 667–676.

Shannon, C. E. 1948. A mathematical theory of communication. *Bell Syst. Tech. J.* 27(1):623656.

Van Leeuwen, J. 1976. On the construction of huffman trees. In *ICALP*, 382–410.

White, D., and Reitz, K. 1983. Graph and semigroup homomorphisms on networks of relations. *Social Networks* 5(2):193–234.