Edge Role Discovery via Higher-order Structures

Nesreen K. Ahmed[1], Ryan A. Rossi[2], Theodore L. Willke[1], and Rong Zhou[2]

[1] Intel Labs
{nesreen.k.ahmed,ted.willke}@intel.com
[2] Palo Alto Research Center (Xerox PARC)
{rrossi,rzhou}@parc.com

Abstract. Previous work in network analysis has focused on modeling the roles of nodes in graphs. In this paper, we introduce edge role discovery and propose a framework for learning and extracting edge roles from large graphs. We also propose a general class of higher-order role models that leverage network motifs. This leads us to develop a novel edge feature learning approach for role discovery that begins with higher-order network motifs and automatically learns deeper edge features. All techniques are parallelized and shown to scale well. They are also efficient with a time complexity of $\mathcal{O}(|E|)$. The experiments demonstrate the effectiveness of our model for a variety of ML tasks such as improving classification and dynamic network analysis.

Keywords: role discovery, edge roles, higher-order network analysis, graphlets, network motifs, latent space models, transfer learning

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 [9,15,6,5,28,19] including descriptive network modeling [30], classification [14], anomaly detection [30], and exploratory analysis [29]. See [28] for other applications. Despite the importance of role discovery, existing work has only focused on discovering node roles (*e.g.*, see [5,7,11,23]). 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 have many other limitations, which significantly reduces the practical utility of such methods in real-world networks. One such example is that the existing work has been limited to mainly simple degree and egonet features [14,30], see [28] for other possibilities. Instead, we leverage higher-order network motifs (induced subgraphs) of size $k \in \{3, 4, \ldots\}$ computed from [1,2] and other graph parameters such as the largest clique in a node (or edge) neighborhood, triangle core number, as well as the neighborhood chromatic, among other efficient and highly discriminative graph features. 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 role discovery models:** Proposed a general class of higher-order role models that leverage network motifs and higher-order network features for learning both node and edge roles. This work is also the first to use higher-order network motifs[3] for role discovery in general.
- **Edge feature representation learning:** Proposed a novel deep graph representation learning framework that begins with higher-order network motifs and automatically learns deeper edge features.
- **Efficient and scalable:** The proposed feature and role discovery methods are efficient (linear in the number of edges) for modeling large networks. In addition, all methods are parallelized and shown to scale to massive networks.

2 Related Work

Related research is categorized into the following parts: (1) role discovery, (2) higher-order network analysis, (3) graph representation learning, (4) sparse graph features, and (5) parallel role discovery.

Role discovery: There has been a lot of work on role discovery in general [9,15,6,5,28,19,14,30]. However, all existing approaches have focused on learning roles of *nodes* in graphs. See [28] for a recent survey on role discovery. In contrast, this work introduces the problem of *edge role discovery* and presents a computational framework for learning and extracting edge roles from large networks. Additional key differences are as follows: (1) our approach uses higher-order graphlets for discovering more intuitive and meaningful roles, and (2) the proposed role methods are parallelized and thus able to scale to extremely large real-world networks. Moreover, our approach supports graphs that are directed/undirected/bipartite, attributed, typed/heterogeneous, and signed.

[3] 4-vertex induced subgraphs (graphlets, motifs) and larger

Higher-order network analysis: Small induced subgraphs called graphlets (motifs) have recently been used for graph classification [36], link prediction [25], and visualization and exploratory analysis [1]. However, this work focuses on using graphlets for learning and extracting more useful and meaningful roles from large networks. Furthermore, previous feature-based role methods have been learned based on simple degree and egonet-based features. Thus, another contribution of this work is the use of higher-order network motifs (based on small k-vertex subgraph patterns called graphlets) for role discovery of nodes and edges — a key and fundamental difference between existing work.

Graph representation learning: While a lot of work has engineered features by hand (or manually selected them) for various ML applications, not much work has been done on learning a set of useful features automatically. Our approach is different from previous work in four fundamental ways: (1) the proposed approach learns important and useful *edge features* automatically, whereas existing approaches were designed for learning *node features*, (2) our approach is space-efficient as it learns sparse features and fast/efficient with a time complexity that is linear in the number of edges. (3) an efficient parallel implementation with strong scaling results as shown in Section 4 and thus well-suited for large-scale networks, and finally, (4) most graph representation learning methods were used in SRL systems for classification [12], whereas we use the proposed approach for edge role discovery.

Sparse graph features: We also make a significant contribution in terms of space-efficient role discovery. In particular, this work proposes the first practical space-efficient approach for feature-based role discovery by learning sparse graph features automatically. In contrast, feature-based node role methods [14,30] store hundreds/thousands of dense features in memory, which is impractical for any relatively large network, *e.g.*, they require more than 2TB of memory for a 500M node graph with 1,000 features.

Parallel role discovery: The existing role discovery methods are sequential, despite the practical importance of parallel role discovery algorithms that scale to massive real-world networks. This work is the first parallel role discovery approach. Furthermore, the proposed edge feature learning techniques are also parallelized and designed to be both efficient in terms of space and communication.

3 Framework

This section introduces edge role discovery along with higher-order edge role models and a computational framework for learning and extracting roles based on higher-order structures.

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 methods such as [1] 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

Algorithm 1 A framework for learning deep edge feature representations from graphs

Input:

a directed and possibly weighted/labeled/attributed graph $G = (V, E)$
a set of relational edge kernels/operators Φ
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: Compute X_{ik}, the number of instances of graphlet H_k that contain edge $e_i \in E$
4: Given G and $\mathbf{X}$, compute 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 using the set of relational operators Φ 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: Let $\mathcal{G}_F = (V_F, E_F)$ be the initial feature graph 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 $f_j \in (\mathcal{F}_\tau \cup \mathcal{F})$ **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 the feature graph $\mathcal{G}_F$ 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 from $\mathbf{X}$ that were pruned (not in $\mathcal{F}_\tau$) 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** feature layer $\mathcal{F}_{\tau-1} = \varnothing$ (no new features emerged) **or** max layers reached ($\tau = \mathrm{T}$)
21: **return** $\mathbf{X}$ and the set of feature definitions $\mathcal{F}$

weighted/typed graphlet counts (among other useful and discriminative graphlet edge statistics) using either exact or estimation methods. These graphlet features are then used to learn deeper higher-order edge features (see below for further details).

Edge Feature Representation Learning Framework: 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 $\mathcal{F}_2, \mathcal{F}_3, \cdots, \mathcal{F}_\tau$ 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 Φ (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. See Table 1 for a few examples. 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. Now, 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).

Table 1. Relational edge feat. operators

Operator		Definition
Hadamard	$\boxplus$	$\prod_{e_j \in \Gamma(e_i)} f_k(e_j)$
mean	$\boxdot$	$\frac{1}{d_i} \sum_{e_j \in \Gamma(e_i)} f_k(e_j)$
sum	$\otimes$	$\sum_{e_j \in \Gamma(e_i)} f_k(e_j)$
Wt. L_p	$\|\cdot\|_{\bar{p}}$	$\sum_{e_j \in \Gamma(e_i)} \lvert f_k(e_i) - f_k(e_j)\rvert^p$

Learning Higher-order Edge Roles: Let $\mathbf{X} = [x_{ij}] \in \mathbb{R}^{m \times f}$ be an edge feature matrix with m rows representing edges and f columns representing higher-order graph features learned from our edge feature representation learning approach. 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 $\mathbf{V}$. For learning higher-order edge roles, we solve:

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

where $\mathbb{D}_\phi(\mathbf{X}\|\mathbf{U}\mathbf{V}^T)$ is an arbitrary Bregman divergence [10] 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 $\mathcal{R}(\mathbf{U}, \mathbf{V})$ 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:

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

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 $\mathbf{X} \approx \mathbf{X}' = \mathbf{U}\mathbf{V}^T$. To measure the quality of our edge mixed membership model, we use Bregman divergences:

$$\sum_{ij} \mathbb{D}_\phi(x_{ij}\|x'_{ij}) = \sum_{ij} \big(\phi(x_{ij}) - \phi(x'_{ij}) - \ell(x_{ij}, x'_{ij})\big) \tag{3}$$

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 $\mathbf{X} \approx \mathbf{X}' = \mathbf{U}\mathbf{V}^T$ where $\mathbf{X}^{(k)}$ is the k-residual matrix defined as:

$$\mathbf{X}^{(k)} = \mathbf{X} - \sum_{h\neq k} \mathbf{u}_h\mathbf{v}_h^T = \mathbf{X} - \mathbf{U}\mathbf{V}^T + \mathbf{u}_k\mathbf{v}_k^T, \quad \text{for } k = 1, \ldots, r \tag{4}$$

We use a fast *scalar block coordinate descent approach* that easily generalizes for heterogeneous networks [32]. 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 2 gives rise to a fast algorithm for each Bregman divergence. Table 2 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 edge role discovery framework.

Table 2. Role divergences and update rules

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

Model Selection: In this section, we introduce an approach that automatically learns the appropriate role mixed-membership model. The approach is based on the Minimum Description Length (MDL) [13,26] principle; a practical formalization of Kolmogorov complexity [17]. More formally, we find the model $M_\star = (\mathbf{V}_r, \mathbf{U}_r)$ that leads to the best compression by solving:

$$M_\star \;=\; \underset{M\in\mathcal{M}}{\arg\min}\; \mathcal{L}(M) \;+\; \mathcal{L}(\mathbf{X}\,|\,M) \tag{5}$$

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 $\mathcal{L}(X \mid M) = \mathcal{L}(M) + \mathcal{L}(\mathbf{E})$. Given a role mixed-membership model with r roles $M = (\mathbf{U}_r, \mathbf{V}_r) \in \mathcal{M}$, the description length is decomposed into: (1) bits required to describe the model, and (2) cost of describing the approximation errors $\mathbf{X} - \mathbf{X}_r$ where $\mathbf{X}_r = \mathbf{U}_r\mathbf{V}_r^T$ is the rank-r approximation of $\mathbf{X}$,

$$\mathbf{U}_r = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_r \end{bmatrix} \in \mathbb{R}^{m \times r}, \quad \text{and} \quad \mathbf{V}_r = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_r \end{bmatrix} \in \mathbb{R}^{f \times r} \tag{6}$$

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 2) for the error description cost[4]. 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:

$$\mathbb{D}_\phi(\mathbf{X} \| \mathbf{X}') = \sum_{ij} X_{ij} \log \frac{X_{ij}}{X'_{ij}} - X_{ij} + X'_{ij} \tag{7}$$

where $\mathbf{X}' = \mathbf{U}\mathbf{V}^T \in \mathbb{R}^{m \times f}$. Other possibilities are given in Table 2. The above assumes a particular representation scheme for encoding the models and data. Recall that the optimal code assigns $\log_2 p_i$ bits to encode a message [34]. Lloyd-Max quantization [22,18] with Huffman codes [16,35] are used to compress the model and data [24,8]. 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[5]. In general, the higher-order (edge) role discovery framework can easily leverage other model selection techniques such as AIC [4] and BIC [33].

4 Experiments

This section investigates the effectiveness and scalability of the proposed edge role discovery framework (Section 3). All network data is available at NR [27].

Higher-order Model Selection: We now validate our model learning approach. Figure 1

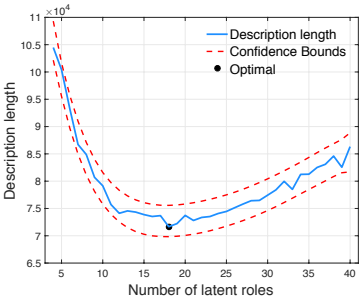
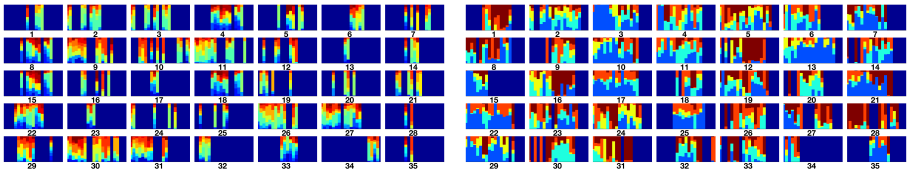


Fig. 1. The valley identifies the correct number of latent roles.

[4] The representation cost of correcting approximation errors

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



(a) Evolving *edge role* memberships (b) Evolving *node role* memberships

Fig. 3. Temporal changes in the edge and node mixed-membership vectors. 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.

demonstrates the effectiveness of our approach for automatically selecting the "best" model from the space of models expressed in the framework (Section 3). In particular, our approach finds the best model with $r = 18$ roles by minimizing the description length (in bits)[6]. As expected, the model description cost is inversely proportional to the error description cost. We also demonstrate the efficiency of our approach in Figure 2. Furthermore, Figure 4 demonstrates the impact on the learning time, number of novel features discovered, and their sparsity, as the tolerance (ε) and bin size (α) varies.

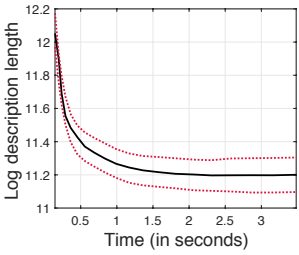


Fig. 2. Runtime of our edge role model selection. The curve is the average over 50 experiments and the dotted lines represent three standard deviations. The result reported above is from a laptop with a single core.

Modeling Dynamic Networks: In this section, we investigate the Enron email communication networks using the *higher-order 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 over a 3 year period. The email communications are from 05/11/1999 to 06/21/2002. For learning we use only the first year of emails. A dynamic network $\{G_t\}_{t=1}^{T}$ is constructed from the remaining email communications (approximately 2 years) where each snapshot graph G_t, $t = 1, \ldots, T$ represents a month of communications. Interestingly, our higher-order dynamic *node* role mixed-membership model has 5 latent roles, whereas we learn 18 roles using the *edge* role model. Evolving edge and node mixed-memberships from the Enron email communication network are shown in Figure 3. The set of edges and nodes visualized in Figure 3 are selected using the difference entropy rank (defined below) and correspond to the edges and nodes with largest difference entropy rank **d**. The first role in Figure 3 represents inactivity (dark blue). The above empirical results suggest that edge roles are superior to node roles in three fundamental ways: (1) Edge roles reveal novel behavioral characteristics that are not captured by the node role models. We posit that these novel behavioral roles are intrinsic to the edge semantics (which represent communications in Figure 3). (2) Roles learned on the edges represent behavioral characteristics at a much lower-level of granularity than those learned on nodes. (3) Edge roles are better at modeling dynamic/temporal networks and

[6] We note that MDL is used in Figure 1, though AIC/BIC gave similar results.

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) Num. features learned

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

Fig. 4. Impact on the learning time, number of features, and their sparsity, as the tolerance ε (rows) and bin size α (columns) varies.

avoid many of the unrealistic assumptions that lie at the heart of dynamic node role mixed-membership models.

We define $\mathbf{d} = \max_t H(\mathbf{u}_t) - \min_t H(\mathbf{u}_t)$ as the difference entropy rank 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. Using the difference entropy rank, we are able to reveal important communications between key players involved in the Enron Scandal, such as Kenneth Lay, Jeffrey Skilling, and Louise Kitchen. In particular, anomalous relationships between these individuals appear in the top anomalies from the difference rank. 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 other possibilities. Many results are removed for brevity.

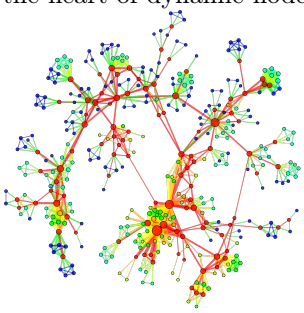


Fig. 5. Edge and node roles for ca-netscience. Link color represents the edge role and node color indicates the corresponding node role.

Exploratory Analysis: Figure 5 visualizes the node and edge roles learned for ca-netscience. While our higher-order role edge discovery method learns a stochastic r-dimensional vector for each edge (and/or node) representing the individual role memberships, Figure 5 assigns a single role to each link and node, *i.e.*, the role with maximum likelihood $k_\star \leftarrow \arg\max_k u_{ik}$. The higher-order edge and node roles from Figure 5 are clearly meaningful. For instance, the red edge role represents a type of bridge relationship.

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 3. 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 feature representations learned by our approach is

Table 3. Higher-order sparse graph feature learning for latent node and edge network modeling. 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)	

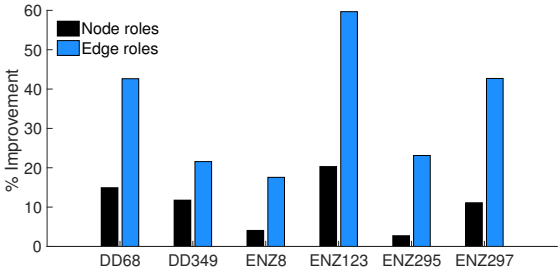


Fig. 6. Relative improvement in label consistency (homophily) — a known proxy for classification performance. In all cases, links predicted using edge roles improves the label consistency over both the initial graph as well as links predicted using node roles.

between $[0.164, 0.318]$ and $[0.162, 0.334]$ for nodes (See $\rho(\mathbf{X})$ and $\rho(\mathbf{Z})$ in Table 3) and up to $6x$ more space-efficient than other approaches.

Improving Classification via Link Prediction: This section demonstrates the effectiveness of edge roles for improving relational classification by predicting links between nodes in the graph. For consistency, we first construct node features from the edge role memberships using a set of relational operators (e.g., relational `mean`, `sum`, `var`, `max`, among others), as introduced in [31]. Thus, let us assume $\mathbf{x}_i$ is a k-dimensional feature vector for node $v_i \in V$. Given $\mathbf{x}_i$ and $\mathbf{x}_j$, and a positive semidefinite kernel function $K\langle\cdot,\cdot\rangle$, the relationship strength between v_i and v_j is defined as:

$$\mathbf{S} = [S_{ij}],\ \forall i, j \quad \textit{and} \quad S_{ij} = \begin{cases} K\langle \mathbf{x}_i, \mathbf{x}_j \rangle & \text{if } (v_i, v_j) \notin E \wedge K\langle \mathbf{x}_i, \mathbf{x}_j \rangle > \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where $K\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ represents the "closeness" between node v_i and v_j in the latent lower-dimensional subspace, $\mathbf{S} \in \mathbb{R}^{n \times n}$ is the (implicit) "similarity" matrix (which can be thought of as the weighted adjacency matrix for a graph G') and S_{ij} represents the relationship strength between node v_i and v_j such that $(v_i, v_j) \notin E$, and 0 otherwise. Note ϵ is a small scalar that controls sparsity. In this work, we use $K\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \exp(-\|\mathbf{x}_i - \mathbf{z}_j\|^2 / 2\sigma^2)$. Given $\mathbf{S}$, let $G' = (V, E')$ denote the predicted latent graph where E' is the set of k predicted links with the largest relationship strength weights. By definition $|E| + |E'| = m + k$ and thus $E \cap E' = \varnothing$.

For quantitative evaluation of the edge roles, we use a measure of *homophily* called label consistency [21]. Let $\xi(v_i)$ be the class of v_i, then the label consistency of G is defined as: $\mathbb{L}(G) = {}^1/_{|E|} \sum_{(v_i,v_j)\in E} \mathbb{L}(v_i, v_j)$ where $\mathbb{L}(v_i, v_j) = 1$ if $\xi(v_i) = \xi(v_j)$ and 0 otherwise. Hence, label consistency measures how often two connected nodes belong to the same class. It is a good proxy measure for classification performance since most existing statistical relational learning (SRL) [12] methods assume the labels of neighbors are highly correlated, *i.e.*, the network exhibits high relational autocorrelation (or homophily) [12,20]. To determine the effectiveness of edge roles for link prediction, we measure $\mathbb{L}(G)$ and $\mathbb{L}(G')$. Notice that if the higher-order edge roles (and node roles for that matter) are useful and effective, one would expect that $\mathbb{L}(G) < \mathbb{L}(G')$, that is, the predicted links resulted in higher homophily among the connected nodes since the class labels of the connected nodes in G' are more consistent than G. Results are provided in Figure 6 for six

different networks. In particular, Figure 6 demonstrates the effectiveness of the higher-order edge roles (and node roles) for link prediction. In all cases, both the higher-order node and edge roles significantly outperform the baseline. Further, the edge role models always perform significantly better than the node roles.

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, seel [3].

Scalability: To evaluate the scalability of the parallel framework for modeling higher-order latent edge roles, we measure the speedup defined as $S_p = T_1/T_p$ 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 7). Similar results were observed for other networks. The experiments used a machine with 4 Intel Xeon E5-4627 v2 3.3GHz CPUs.

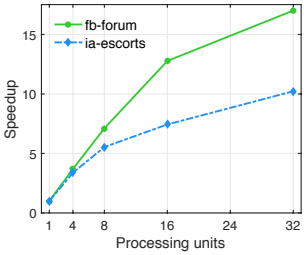


Fig. 7. Strong parallel scaling is observed.

5 Conclusion

In this paper, we introduced the *edge role discovery* problem and presented a computational framework for learning and extracting edge roles from large networks. In addition, we proposed higher-order role discovery methods that leverage network motifs (including all motifs of size 3,4, and larger) for learning more meaningful and discriminative roles. We also proposed a novel edge feature learning approach, which was used for our feature-based edge roles. Furthermore, all methods are space-efficient (by learning sparse features) and efficient with a runtime that is linear in the number of edges. Finally, the approach also supports graphs that are directed/undirected/bipartite, attributed, typed, and signed.

References

1. Ahmed, N.K., Neville, J., Rossi, R.A., Duffield, N.: Efficient graphlet counting for large networks. In: ICDM. p. 10 (2015)
2. Ahmed, N.K., Neville, J., Rossi, R.A., Duffield, N., Willke, T.L.: Graphlet decomposition: Framework, algorithms, and applications. KAIS pp. 1–32 (2016)
3. Ahmed, N.K., Willke, T.L., Rossi, R.A.: Estimation of local subgraph counts. In: IEEE BigData. pp. 1–10 (2016)
4. Akaike, H.: A new look at the statistical model identification. TOAC 19(6) (1974)

5. Anderson, C., Wasserman, S., Faust, K.: Building stochastic blockmodels. Social Networks 14(1), 137–161 (1992)
6. Arabie, P., Boorman, S., Levitt, P.: Constructing blockmodels: How and why. Journal of Mathematical Psychology 17(1), 21–63 (1978)
7. Batagelj, V., Mrvar, A., Ferligoj, A., Doreian, P.: Generalized blockmodeling with pajek. Metodoloski zvezki 1, 455–467 (2004)
8. Bennett, W.R.: Spectra of quantized signals. Bell Sys. Tech. 27(3), 446–472 (1948)
9. Borgatti, S., Everett, M., Johnson, J.: Analyzing Social Networks. Sage Pub. (2013)
10. Bregman, L.M.: The relaxation method of finding the common point of convex sets. USSR Comp. Math. and Math. Physics 7(3), 200–217 (1967)
11. Doreian, P., Batagelj, V., Ferligoj, A.: Generalized Blockmodeling, vol. 25. Cambridge University Press (2005)
12. Getoor, L., Taskar, B. (eds.): Intro. to Stat. Relational Learning. MIT Press (2007)
13. Grünwald, P.D.: The minimum description length principle. MIT press (2007)
14. Henderson, K., et al.: Rolx: Structural role extraction & mining in large graphs. In: KDD. pp. 1231–1239 (2012)
15. HollandKathryn Blackmond, P., Leinhardt, S.: Stochastic blockmodels: First steps. Social Networks 5(2), 109–137 (1983)
16. Huffman, D.A., et al.: A method for the construction of minimum-redundancy codes. Proceedings of the IRE 40(9), 1098–1101 (1952)
17. Li, M., Vitányi, P.: An introduction to Kolmogorov complexity and its applications. Springer Science & Business Media (2009)
18. Lloyd, S.: Least squares quantization in pcm. TOIT 28(2), 129–137 (1982)
19. Lorrain, F., White, H.: Structural equivalence of individuals in social networksâĂă. Journal of Mathematical Sociology 1(1), 49–80 (1971)
20. Macskassy, S., Provost, F.: A simple relational classifier. In: KDD MRDM (2003)
21. Macskassy, S.A., Provost, F.: Classification in networked data: A toolkit and a univariate case study. JMLR 8, 935–983 (2007)
22. Max, J.: Quantizing for minimum distortion. TOIT 6(1), 7–12 (1960)
23. Nowicki, K., Snijders, T.: Estimation and prediction for stochastic blockstructures. Journal of the American Statistical Association 96(455), 1077–1087 (2001)
24. Oliver, B., Pierce, J., Shannon, C.E.: The philosophy of pcm. IRE 36(11) (1948)
25. Rahman, M., Hasan, M.A.: Link prediction in dynamic networks using graphlet. In: PKDD. pp. 394–409 (2016)
26. Rissanen, J.: Modeling by shortest data description. Automat. 14(5), 465–471 (1978)
27. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015), http://networkrepository.com
28. Rossi, R.A., Ahmed, N.K.: Role discovery in networks. TKDE 27(4), 1112 (2015)
29. Rossi, R.A., Gallagher, B., Neville, J., Henderson, K.: Role-Dynamics: Fast Mining of Large Dynamic Networks. In: WWW Companion. pp. 997–1006 (2012)
30. Rossi, R.A., Gallagher, B., Neville, J., Henderson, K.: Modeling dynamic behavior in large evolving graphs. In: WSDM. pp. 667–676 (2013)
31. Rossi, R.A., McDowell, L.K., Aha, D.W., Neville, J.: Transforming graph data for statistical relational learning. JAIR 45(1), 363–441 (2012)
32. Rossi, R.A., Zhou, R.: Parallel Collective Factorization for Modeling Large Heterogeneous Networks. In: Social Network Analysis and Mining. p. 30 (2016)
33. Schwarz, G., et al.: Estimating the dimension of a model. Ann. of stat. 6(2) (1978)
34. Shannon, C.E.: A mathematical theory of communication. Bell Sys. T. 27(1) (1948)
35. Van Leeuwen, J.: On the construction of huffman trees. In: ICALP. p. 382 (1976)
36. Vishwanathan, S.V.N., Schraudolph, N.N., Kondor, R., Borgwardt, K.M.: Graph kernels. JMLR 11, 1201–1242 (2010)