On local and global graph structure mining

On Local and Global Graph Structure Mining

Yulong Pei

On Local and Global Graph Structure Mining by Yulong Pei.
Eindhoven: Technische Universiteit Eindhoven, 2020. Proefschrift.

A catalogue record is available from the Eindhoven University of Technology Library

ISBN 978-90-386-4975-7

SIKS Dissertation Series No. 2020-05
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

On Local and Global Graph Structure Mining

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus prof.dr.ir. F.P.T. Baaijens, voor een commissie aangewezen door het College voor Promoties, in het openbaar te verdedigen op woensdag 5 februari 2020 om 16:00 uur

door

Yulong Pei

geboren te Xinyang, China

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	prof.dr.ir. O.J. Boxma
1e promotor:	dr. G.H.L. Fletcher
2e promotor:	prof.dr. M. Pechenizkiy
leden:	prof.dr. T. De Bie (Universiteit Gent)
	dr. J. Gama (University of Porto)
	prof.dr. M.T. de Berg
adviseur(s):	dr. N. Yakovets
	dr. W. Duivesteijn

Het onderzoek of ontwerp dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

Summary

Graphs are widely used in representing complex relationships among entities, for instance, social networks, biological networks, and traffic networks. As graphs are seemingly ubiquitous in different domains, it is of great theoretical and practical value to study graphs. In order to analyze graphs, one of the important tasks is to grasp the structures of the target graphs. Graph structures can be analyzed locally or globally from different perspectives. Meanwhile in practice many graphs are dynamic with evolving structures and the dynamics may further lead to uncertainties on graphs. All these challenges lead to difficulties in graph structure mining. Therefore, it is of great demand to find effective and efficient solutions to mine the local and global structures on static and dynamic graphs.

In this thesis, we study the fundamental problem: *Can local and global structures of graphs be effectively mined in both static and dynamic scenarios?* Firstly, we study the problem of local structure mining on dynamic graphs. We propose two novel approaches including

- dFGM (dynamic Factor Graph Model): a factor graph model for node classification from the local perspective in dynamic networks. dFGM models three types of factors, node factor, correlation factor and dynamic factor, based on node features, node correlations and temporal correlations, respectively.

- DNGE (dynamic network embedding method with Gaussian Embedding): a network embedding framework to mine the local structures in dynamic networks. DNGE integrates Gaussian embedding techniques and temporal

regularization to map nodes to Gaussian distributions. Thus, it can capture temporal information and model uncertainties in dynamic graphs.

These two approaches are validated on the tasks of community detection and node classification to demonstrate their effectiveness in mining local graph structures.

Secondly, we explore the problem of global structure mining on static and dynamic graphs. We design a series of new approaches including:

- *struc2gauss*: a flexible global structure preserving network embedding framework in static networks. struc2gauss learns node representations in the space of Gaussian distributions by modeling the global network structures. It is capable of preserving structural roles and modeling uncertainties.
- IMM (Infinite Motif Stochastic Blockmodel): a nonparametric Bayesian model to generate higher-order motif information in static networks. IMM is a high-order model which takes advantage of the motifs in the generative process and it is a nonparametric Bayesian model which can automatically infer the number of roles from the data.
- DyNMF (dynamic nonnegative matrix factorization): a unified model to discover role and role transition simultaneously in dynamic networks. DyNMF simultaneously obtains the role matrix of snapshot $t+1$ and the role transition from snapshot t to $t+1$ by using information in snapshot t+1 and the role matrix of snapshot t.

We employ a role discovery experiment to validate the advantages of these models in mining global graph structures.

Finally, we study the problem of jointly mining local and global structures on static graphs. We propose a novel joint approach REACT to detect roles and communities simultaneously:

- REACT combines role discovery and community detection using two nonnegative matrix tri-factorization components and integrates the diversity relation between roles and communities using the $L_{2,1}$ norm; and
- REACT is capable of automatically determining the number of roles and communities.

We conclude this thesis with discussion on how effective the local community detection and global role discovery approaches work on static and dynamic

graphs. The main findings and general lessons we have in the investigations are of value to the community as a basis for mining and understanding local and global structures of graphs.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A graph is a set of nodes, pairs of which might be connected by edges. Real-world data from different domains can be cast into this form directly or indirectly. For example, Twitter network consists of users and their relationships (e.g., following) and interconnections (e.g., reply or repost). Web graph contains webpages as nodes and hyperlinks between them as the edges. Academic graphs connect researchers using different types of relationships such as paper co-authoring and citing. An example of a research graph is shown in Figure 1.1[1]. As graphs are ubiquitous in different domains, it is of great theoretical and practical value to study graphs. Analyzing graphs can prove valuable for a broad spectrum of research areas and applications. From the perspective of research, analyzing graph structures attracts considerable attention from both computer science and social science, e.g., social network analysis. From the perspective of practice, analyzing graphs can shed light on a variety of applications such as friend recommendation.

In order to analyze graphs, one of the important tasks is to grasp the structures of the target graphs or networks[2]. As edges exist between nodes that disobey the *i.i.d* assumption, it is non-trivial to apply traditional machine learning and data mining techniques in graphs directly. Meanwhile in practice many graphs are dynamic with evolving structures and the dynamics may further lead to uncertainties on graphs. All these challenges lead to difficulties in graph structural pattern mining. Therefore, it is of great demand to find effective and

[1] `https://research.tue.nl/en/persons/yulong-pei/network/`

[2] In this thesis, these two terms *graph* and *network* are used interchangeably.

efficient approaches to mine graph structures by taking local and global structures as well as uncertainties into consideration.

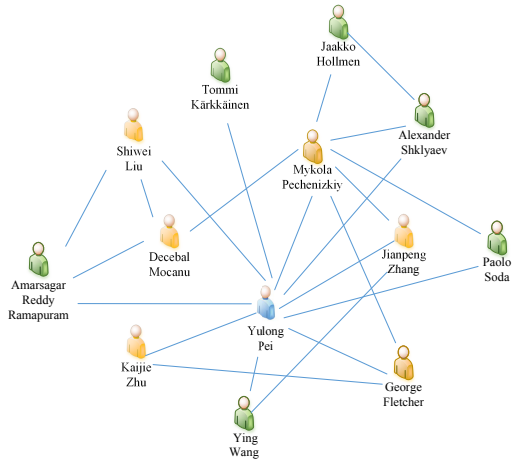


Figure 1.1: A research graph which consists of authors and units as nodes and collaboration as edges.

1.1 What is Graph Structure Mining?

Graph structure mining research aims to find solutions to analyze the structural properties of graphs and predict how these structural properties of a given graph might affect some applications. Graph structures can be viewed from different perspectives. For instance, the edges between two nodes, the density of a subgraph, the community structure of a subset of nodes, and the role that a node plays. In this thesis we distinguish two types of graph structures:

- *Local graph structure*: it captures the topological properties of graphs by observing a *bounded* part of the input graph, e.g., edges and common neighbors between two given nodes, community structures forming by a subset of nodes, etc. Different graphs may have different local structural patterns. For example, in different graphs, the density of edges and the number of nodes inside each community could be different.

- *Global graph structure*: it reflects the topological properties of graphs through the *unbounded* observation of the input graph as an entirety, e.g., roles and positions, high-order motifs, graph centrality measures, etc. Global structural properties could be ubiquitous in different graphs. For instance, the structural roles of core and periphery may exist in many different graphs.

An example to illustrate local and global structures of graphs is shown in Figure 1.2. These ten nodes belonging to

- three groups (different colors indicate different groups) based on global structural information, i.e., *roles*. For example, the yellow nodes are bridges which connect two subgraphs and the blue nodes are central which are linked by many other nodes (belong to different roles) inside each subgraph.
- two groups (groups are shown by the ellipses) based on local structural information, i.e., *communities*. This is because nodes have denser internal connections to each other inside each community than external connections outside of communities.

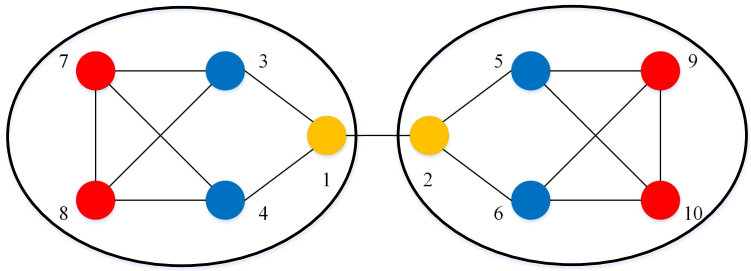


Figure 1.2: Illustration of local and global structures of graphs using the Borgatti-Everett graph [BE92] as an example.

The local and global structures of graphs can be further explained from three perspectives:

- *The spatial perspective*. In Figure 1.3, it is assumed that there are many nodes between the left and right communities. To detect each community, what we need to know is the local structural information and it is not

required to know the rest part of this graph. For instance, detecting the left community does not require the information of the right community. But for role discovery, we need to have a global view of this graph. Otherwise, it is impossible to assign node 1 and 2 to the same role.

- *The node perturbation perspective*. In Figure 1.4, we swap node 4 and node 5. After the perturbation, role of node 4 and 5 does not change because their global structural information stays the same, i.e., they are still the central nodes. However, the communities of node 4 and 5 are changed, because their local structures are different from that in Figure 1.3. For example, the neighbors of node 4 are different now.

- *The node removal perspective*. In Figure 1.3, part of the graph has been removed. After removing, the community on the left side stays the same because there is no influence on its local structure. However, the roles may be changed because the global information has been changed (the community in the right side has been removed).

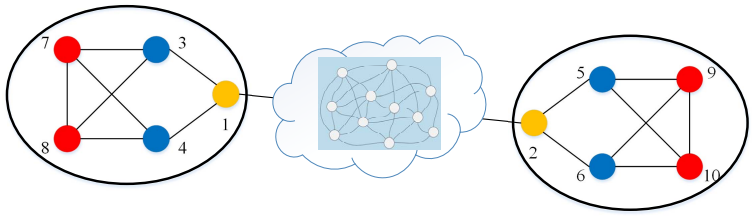


Figure 1.3: Illustration of local and global structures of graphs from the spatial perspective.

Graph structure mining, both locally and globally, significantly contributes to a wide range of applications and many of them have been widely studied in both academia and industry. Mining local graph structures play a crucial role in different tasks. To name a few,

- Community detection [For10] analyzes graphs from the local perspective. It aims to group together nodes that are highly connected to each other. For instance, an online social group consists of users having the same interests and interacting with each other frequently.

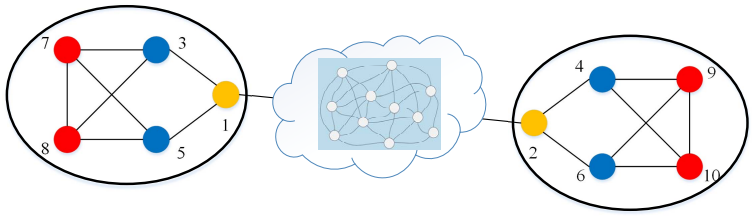


Figure 1.4: Illustration of local and global structures of graphs from the node perturbation perspective.

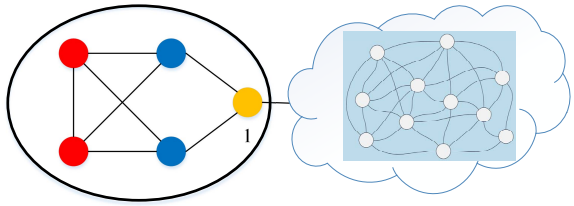


Figure 1.5: Illustration of local and global structures of graphs from the node removal perspective.

- Link prediction [LNK07] is to predict whether there will be links between two nodes based on the observed local structural information. For instance, friend recommendation on Facebook is mainly based on the common friends between a user and her potential friends.

On the other hand, mining global structures is instrumental in applications, e.g.,

- Role discovery [RA+15]. It clusters nodes into groups with different structural patterns and each group indicates a specific role. For example, bridge nodes in a social network play an important role in information diffusion.
- Information diffusion [GHFZ13] studies the how and why the information spreads on graphs like social networks including identifying the influential spreaders and modeling the information spread patterns.

Among these applications based on local and global structures of graphs, we focus on *community detection* and *role discovery* in this thesis because communities and roles can represent local and global structures of graphs respectively

and discovering them on graphs are representative applications in mining graph structures.

1.2 Research Questions

Classical machine learning and data mining techniques are mostly designed for data which is independent and identically distributed. However, graph data contains complex structures between instances which are reflected by edges or motifs. It makes graph structure mining a more challenging problem compared to traditional data mining problems. Meanwhile, most real-world graphs are dynamic with evolving nodes and structures. Therefore, mining graph structures by incorporating the dynamic information is important. Beside, with evolving structures and noisy information, how to capture the uncertainties when mining graph structures is another emerging practical issue. To address these problems, the ultimate question to answer is:

> Q0: Which local and global structures of graphs can be effectively mined in both static and dynamic scenarios?

In this thesis, we attempt to answer this question in three different aspects: local structure mining, global structure mining and joint mining of local and global structures. In each aspect, we discuss the motivation and introduce our solutions by proposing a series of graph structure mining approaches on static and dynamic graphs. Specifically, in this thesis we would like to answer the following questions.

Local structure mining on graphs. The first research question is how to effectively mine the local structures of dynamic graphs. Earlier studies focused on static graph structure mining [For10] and ignored the evolving structures of dynamic graphs. Recent dynamic structure mining approaches, although modeled the dynamics, failed to capture the uncertainties in modeling the dynamics. However, uncertainties are inevitable due to the noise contained in the data or incomplete information during the data collection. Therefore, the first question raises resulting from the limitations in previous studies:

> Q1: How can we effectively mine the local structures of graphs in the dynamic scenario?

To answer this general question, we divide it into two concrete subquestions using community detection and node classification as the applications to discuss the problem of local structure mining:

> Q1.1: How can we effectively detect local communities by capturing dynamics and uncertainties on dynamic graphs?
>
> Q1.2: How can we make good use of local structures and temporal information to improve the performance of node classification on dynamic graphs?

Global structure mining on graphs. Global structure mining has been widely studied in the literature. However, most of existing methods focus on static graphs without considering the evolving structures of dynamic graphs [RA+15]. Furthermore, the uncertainties resulting from noisy or incomplete information has also been neglected. Another issue in previous studies is they are incapable of determining the number of roles automatically. Thus, we have the second research question:

> Q2: How can we effectively mine the global structures of graphs in static and dynamic scenarios?

Similarly, we propose the solution to this question by dividing it into three concrete subquestions using role discovery as the application:

> Q2.1: How can we effectively discover roles on static graphs by capturing the global structures and uncertainties?
>
> Q2.2: Is is feasible to determine the number of roles automatically given a static graph?
>
> Q2.3: How can we effectively discover roles on dynamic graphs by capturing the global structures and dynamics?

Joint Mining of local and global structures on graphs. Local structures and global structures are two complementary views to describe graphs, so role discovery and community detection can enhance each other. However, studies on these two topics have been performed independently [RP14]. Hence, simultaneous detection of communities and roles can make better use of local and global graph structures and lead to better performance for each task. Naturally, the third question is:

> Q3: How can we jointly mine the local and global structures of graphs by modeling the relationship between global roles and local communities?

By answering Q1 to Q3, we aim to partly answer the open-ended question Q0. In brief, our research procedures are as following: Firstly, we propose to mine the local structures of graphs in dynamic scenario. The proposed approach is applied to the tasks of community detection and node classification. Then, methods for global structure mining of graphs in both static and dynamic scenarios are explored. The methods are validated on the task of role discovery. Finally, we address the problem to jointly detect roles and communities by explicitly modeling the relation between global roles and local communities.

1.3 Main Contributions

In this thesis, we aim to propose a series of novel approaches to mine the graph structures from static and dynamic perspectives. To summarize, this thesis contains the following main contributions:

- We propose graph mining approaches to mine graphs from the local perspective including:
  - dFGM (dynamic Factor Graph Model): a factor graph model[3] for node classification from the local perspective in dynamic networks. dFGM models three types of factors, i.e., node factor, correlation factor and dynamic factor, based on node features, node correlations and temporal correlations, respectively.
  - DNGE (dynamic network embedding method with Gaussian Embedding): a network embedding[4] framework to mine the local structures in dynamic networks. DNGE integrates Gaussian embedding techniques and temporal regularization to map nodes to Gaussian distributions. Thus, it can capture temporal information and model uncertainties in dynamic networks.
- We propose graph mining approaches to mine graphs from the global perspective including:

[3] A factor graph is a bipartite graph that expresses how a global function of many variables into a product of local functions [KFL+01]. Factor graph model is a probabilistic graphical model and can learn the graph structures using Bayesian statistics.

[4] Network embedding aims to assign nodes in a network to low-dimensional representations and effectively preserves the network structure [CWPZ18].

  - *struc2gauss*: a flexible global structure preserving network embedding framework in static networks. struc2gauss learns node representations in the space of Gaussian distributions by modeling the global network structures. It is capable of preserving structural roles and modeling uncertainties.
  - IMM (Infinite Motif Stochastic Blockmodel): a nonparametric Bayesian model to generate higher-order motif information in static networks. IMM is a high-order model which takes advantage of the motifs in the generative process and it is a nonparametric Bayesian model which can automatically infer the number of roles from the data.
  - DyNMF (dynamic nonnegative matrix factorization): a unified model to discover role and role transition simultaneously in dynamic networks. DyNMF simultaneously obtains the role matrix of snapshot t+1 and the role transition from snapshot t to t+1 by using information in snapshot t+1 and the role matrix of snapshot t.

- We propose a joint approach Role and Community Detection (REACT) to mine graph structure from local and global views simultaneously by modeling the role-community diversity relation.
  - REACT combines role discovery and community detection using two nonnegative matrix tri-factorization components and integrates the diversity relation between roles and communities using the L2,1 norm; and
  - REACT is capable of automatically determining the number of roles and communities.

1.4 Thesis Organization

This thesis contributes to the research in graph structure mining through a series of theoretical and empirical studies. These studies that comprise different chapters of this thesis have appeared (or are under review) in peer-reviewed conferences and journals.

The main body of this thesis is structured in four parts. This first part presents a survey on local and global structure mining. The second part investigates the local structural pattern mining in dynamic graphs. The third part studies the global structural pattern mining in static and dynamic graphs. The

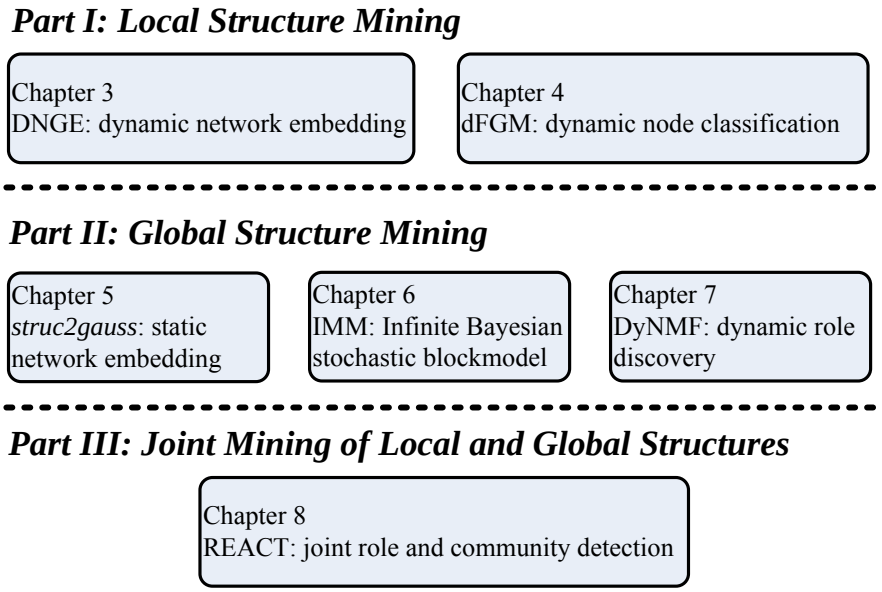


Figure 1.6: Structure of this thesis.

last part exploits the solution to simultaneously detect communities and discovery roles from a joint view of both local and global structures. The structure of this thesis is depicted in Figure 1.6. More concretely, the thesis is organized as follows:

Chapter 2 We study taxonomies of methods for both local and structure mining on graphs. For each method type, several representative approaches are discussed in details.

Part I studies the problem of local structural pattern mining on graphs. This part is an extension of papers [PDZ+19] and [PZFP16]:

> Yulong Pei, Xin Du, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. Dynamic Network Representation Learning via Gaussian Embedding. *Under review*, 2019.
>
> Yulong Pei, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. Node classification in dynamic social networks. In *Pro-*

ceedings of AALTD 2016: 2nd ECMLPKDD International Workshop on Advanced Analytics and Learning on Temporal Data, pages 1–8, Riva del Garda, Italy, 2016.

Chapter 3 Network embedding achieves promising results in mining local structures of graphs. However, two major challenges exist in previous network embedding studies: *dynamics modeling* and *uncertainty modeling*. We propose *DNGE*, a novel dynamic network embedding framework using Gaussian embedding, *DNGE*, to tackle these limitations. *DNGE* learns node representations by explicitly modeling temporal information as regularization using two different smoothness strategies. Furthermore, *DNGE* utilizes Gaussian embedding to represent each node as a Gaussian distribution where its mean indicates the position of this node in the embedding space and its covariance represents its uncertainty. Our experimental results demonstrate that *DNGE* effectively preserves community structures and captures dynamic information, achieves comparable results to state-of-the-art methods in link prediction and provides more information on uncertainties of node representations.

Chapter 4 Nodes and edges of graphs evolve in time which makes node classification difficult in practice. Thus, we propose a dynamic factor graph model, named *dFGM*, to classify nodes in dynamic graphs. To capture the temporal information, graph factors based on node attributes, node correlations and dynamic information are integrated in the *dFGM*. To overcome the limitation in graph feature extraction, we also utilize an unsupervised graph feature extraction method, i.e., DeepWalk, to extract features from the networks. The experiments have been conducted on a real-world data set and the experimental results demonstrate the effectiveness of the *dFGM*. We also analyze the influence of feature dimension and size of training data. Two different graph feature extraction methods have also been compared in the experiments.

Part II studies the problem of global structural pattern mining on graphs. This part is an extension of papers [PDZ+18], [PZFP19] and [PZFP18]:

Yulong Pei, Xin Du, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. *struc2gauss*: Structure Preserving Network Embedding via Gaussian Embedding. arXiv preprint arXiv:1805.10043, 2018.

Yulong Pei, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. Infinite Motif Stochastic Blockmodel for Role Discovery

in Networks. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Vancouver, Canada, 2019.

Yulong Pei, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. DyNMF: Role Analytics in Dynamic Social Networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 3818–3824, Stockholm, Sweden, 2018.

Chapter 5 Two major limitations exist in previous network embedding studies: i.e., the lack of *global structure preservation* and the lack of *uncertainty modeling*. Firstly, random-walk based embedding methods fail in capturing global structural information. Secondly, representing a node into a point vector are not capable of modeling the uncertainties of node representations. We propose a flexible role structure preserving network embedding framework, *struc2gauss*, to tackle these limitations. On the one hand, *struc2gauss* learns node representations based on structural similarity measures so that global structural information can be taken into consideration. On the other hand, *struc2gauss* utilizes Gaussian embedding to represent each node as a Gaussian distribution where the mean indicates the position of this node in the embedding space and the covariance represents its uncertainty. By conducting experiments from different perspectives, we demonstrated that *struc2gauss* excels in capturing global structural information, compared to state-of-the-art techniques. It also overcomes the limitation of uncertainty modeling and is capable of capturing different levels of uncertainties.

Chapter 6 High-order motif can effectively represent the global structures of graphs. Thus, we propose a novel generative model, infinite motif stochastic blockmodel (IMM), for role discovery. IMM is advantageous in two aspects: (1) it models higher-order motifs to infer the roles which can effectively capture the global structural information of networks, and (2) it is a nonparametric Bayesian model to infer the number of roles automatically which is more suitable in real-world network analytics. We evaluated IMM in role discovery compared to state-of-the-art blockmodels and the results indicate the effectiveness of IMM.

Chapter 7 Dynamic structures of graphs makes global structure mining, e.g., role discovery, a challenging problem. To solve this problem, we propose DyNMF,

a novel dynamic non-negative matrix factorization approach to discover roles and role transitions simultaneously in dynamic graphs. Current and historical views have been combined for the node-feature matrix factorization. The current view is based on structural information in the current snapshot and the historical view captures the correlation between previous roles and current roles using role transition matrices. We conduct comprehensive experiments on both synthetic and real-world data sets to validate the performance of DyNMF in role discovery and role transition learning. Experimental results from three aspects including role discovery, role transition, and role prediction indicate the effectiveness of our proposed method for the challenging problem of dynamic role analytics.

Part III studies the joint problem of local and global structural pattern mining on graphs. This part is an extension of paper [PFP19]:

> Yulong Pei, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. Joint Role and Community Detection in Networks via L2,1 Norm Regularized Nonnegative Matrix Tri-Factorization. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Vancouver, Canada, 2019.

Chapter 8 We explore the problem of joint role and community detection because role and community are correlated in reflecting the global and local structures of graphs respectively. We propose a novel approach named REACT which consists of three components: role discovery, community detection and community-role relation. The first two components are based on nonnegative matrix tri-factorization (NMTF) and the last component is a regularization term to capture the diversity relation between roles and communities which is based on $L_{2,1}$ norm. REACT is evaluated in both role discovery and community detection compared to state-of-the-art methods. The results indicate the effectiveness of our proposed method in both tasks. We also investigate the effect of the trade-off parameter for community-role relation on both tasks.

Chapter 9 We discuss the limitations in our proposed graph structure mining approaches. Then we conclude the thesis with directions of future work based on the obtained results and discussed limitations.

Chapter 2

A Survey on Graph Structure Mining

In this chapter, we introduce the necessary concepts used in the remainder of this thesis and then present a comprehensive survey on different methods for local and global structure mining. In Section 2.1, we introduce the basic concepts of graphs. Then, Section 2.2 introduces the definition of event logs. Section 2.3 discusses a taxonomy of methods for global structure learning including equivalence based, similarity based, blockmodel based, feature based and embedding based methods.

2.1 Definitions and Notation

Definition 2.1. (**Static Graph**) A static graph G is defined as $G = (V, E)$ where V is a finite set of nodes and $E \subseteq V \times V$ is a set of edges, such that $n = |V|$ is the number of nodes and $m = |E|$ is the number of edges. Each edge in E is represented as $< u, v >$ where u and v are the two incident nodes of the edge where $u \neq v$.

Note that Definition 2.1 can be extended to weighted graphs where each edges is represented as $< u, v, w >$, i.e., a positive weight associated to the edges. For simplicity, we study unweighted undirected graph in this thesis.

Definition 2.2. (**Dynamic Graph**) Let V be a finite set of nodes, a dynamic graph $\mathcal{G} = \{G^t | t = 1, ..., T\}$ consists of a series of graph snapshots $G^t = \{V, E^t\}$, where each snapshot G^t is a static graph with node set V and edge set E^t.

Structural role is a concept from social science and is used to describe nodes in a network from a global perspective.

Definition 2.3. Structural role. In a network, s set of nodes have the same role if they share similar structural properties (such as degree, clustering coefficient, and betweenness) and structural roles can often be associated with various functions in a network.

For example, hub nodes with high degree in a social network are more likely to opinion leaders, whereas bridge nodes with high betweenness are gatekeepers to connect different groups. Structural roles can reflect the global structural information because two nodes which have the same role could be far from each other and have no direct links or shared neighbors. In contrast to roles, community structures focus on local connections between nodes.

Definition 2.4. Community structure. In a network, a community is a set of nodes where nodes in this set are densely connected internally and sparsely connected to nodes outside of this community.

It can be seen that the focus of community structure is the internal and local connections so it aims to capture the local structural information of networks

2.2 Local Structure Mining

As introduced in Section 1, local structures of graphs capture the topological properties of graphs from the local perspective. Local structures can be represented by edges, common neighbors, subgraphs or community structures. Therefore, local structure mining are studied in several different tasks, e.g., link prediction, community detection, subgroup discovery, etc., since they are related to these local topological properties. In this section, we focus on the problem of community detection as the fundamental task in local structure mining. For other tasks, we refer the interested reader to survey papers such as link prediction [LZ11] and subgraph discovery [LRJA10].

In the literature, community detection approaches can be categorized into different types with different taxonomies. We follow [For10] to categorize these methods into seven families:

- traditional methods based on clustering algorithms;
- divisive algorithms based on hierarchical clustering approaches;
- modularity based algorithms aiming to optimize modularity;
- spectral algorithms based on the eigenvectors of Laplacian matrices;
- dynamic algorithms employing processes running on the graph;
- statistical inference based model such as generative models; and
- other miscellaneous methods.

Note that these categories are not disjoint and there are overlapping methods can belong to multiple families. For example, [YCH+16] combines modularity and nonnegative matrix factorization model. Newman-Girvan modularity can be optimized by using the eigenvectors of the modularity matrix [For10]. Therefore, in this section we introduce several representative methods.

2.2.1 Nonnegative Matrix Factorization Method

Community detection can be viewed as a clustering problem essentially so earlier studies employed different traditional clustering algorithms to detect communities on graphs, e.g., k-means. Among these clustering methods, nonnegative matrix factorization (NMF) achieves promising performance in detecting communities [WLW+11, YL13, PCS15].

NMF [LS01] is a popular model in multivariate analysis and linear algebra where a matrix is factorized into two matrices, with the property that both matrices have no negative elements. There are several advantages in NMF including ease of implementing inference and ease of interpreting results. Hence, this model is widely used in clustering tasks. An extension of NMF is nonnegative matrix tri-factorization (NMTF) [DLPP06] which factorizes the input matrix into three matrices. Given a adjacency matrix $G_{n\times n}$, where n is the number of nodes, the idea of NMF is to generate a rank r approximation $XSX^T \approx G$ where r is the number of communities, matrix $X_{n\times r}$ denotes the nodes' membership in communities and matrix $S_{r\times r}$ represents the interaction between different communities. Thus, the problem of NMF based community detection is to seek two low rank matrices X and S to satisfy:

$$\min_{X,S} \|G - XSX^T\|^2, \quad s.t.\ X \geq 0, S \geq 0 \tag{2.1}$$

where $\|\cdot\|$ is the Frobenius norm. The non-negativity constraint in Eq. 2.1 makes the representation of the original data easier to interpret and more semantically meaningful compared with other factorization methods, e.g., SVD and PCA [LS01]. Using multiplicative update rules, the solution for Eq. 2.1 is shown as follows:

$$X_{ik} \leftarrow X_{ik} \circ \Big(\frac{(G^T XS + GXS^T)_{ik}}{(XSX^T XS^T + XS^T X^T XS)_{ik}} \Big)^{\frac{1}{4}}, \qquad (2.2)$$
$$S_{kl} \leftarrow S_{kl} \circ \frac{(X^T GX)_{kl}}{(X^T XSX^T X)_{kl}},$$

where $\circ$ denotes the element-wise product. Note that in an undirected graph where G is a symmetric matrix, the lower-rank matrix S will be symmetric as well. Otherwise S will be asymmetric. By extending the original NMF method, it can be used in overlapping community detection [YL13] and in attributed graphs [PCS15].

2.2.2 Girvan Newman Method

Girvan Newman method is a divisive and hierarchical method for community detection [GN02]. It identifies edges in a network that lie between communities and then removes them, leaving behind just the communities themselves. The identification is performed by employing the graph-theoretic measure betweenness centrality.

Betweenness centrality [Fre77] is based on shortest paths to measure the centrality of nodes. The betweenness centrality for each node is the number of these shortest paths that pass through the node. Formally, the betweenness centrality of node v is defined as:

$$bc(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \qquad (2.3)$$

where σ_{st} is the total number of shortest paths from node s to t and $\sigma_{st}(v)$ is the number of those paths that pass through v. Although the original definition of betweenness is designed for nodes, it can be extended to edges easily. Formally, the betweenness centrality of edge e is defined as:

$$bc(e) = \sum_{s \neq v} \frac{\sigma_{st}(e)}{\sigma_{st}} \qquad (2.4)$$

where $\sigma_{st}(e)$ is the total number of shortest paths from node s to node t that pass through edge e, and σ_{st} is the total number of shortest paths from node s to node t.

The intuition of Girvan Newman method is shown in Figure 2.1. Edge $e_i j$ (green dashed line) is the edge with highest betweenness value. After removing this edge, nodes in the left side and the right side can form two communities.

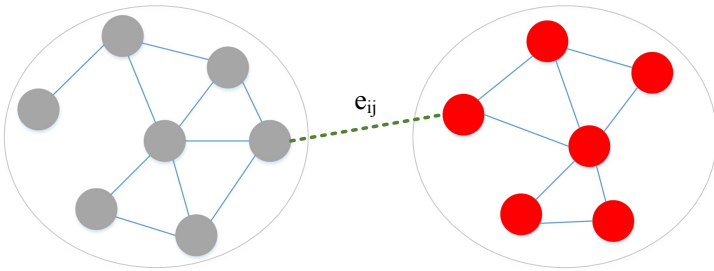


Figure 2.1: The intuition of Girvan Newman method.

Girvan Newman method for community detection consists of four steps:

- The betweenness of all existing edges in the network is calculated first.
- The edge with the highest betweenness is removed.
- The betweenness of all edges affected by the removal is recalculated.
- Steps 2 and 3 are repeated until no edges remain.

2.2.3 Modularity based Methods

Modularity is one measure of the structure of graphs. It was designed to measure the strength of division of a graph into communities. A graph with high modularity has dense connections between the nodes within communities but sparse connections between nodes in different communities. Modularity is often used in optimization methods for detecting community structure in graphs. Thus, larger modularity indicates denser connections within communities and sparser connections in different communities, i.e., better community structure representations.

Based on the definition of modularity, it is intuitive to partition a graph to achieve higher modularity. Modularity maximization [New06] is such a method

following this intuition. There are different ways to calculate modularity, e.g., definition-based and spectral optimization. In this section we introduce the basic method in [New06]. Modularity Q is defined as the fraction of edges that fall within community 1 or 2, minus the expected number of edges within community 1 and 2 for a random graph with the same node degree distribution as the given graph. Formally,

$$Q = \frac{1}{2m} \sum_{v,w} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \frac{s_v s_w + 1}{2}, \tag{2.5}$$

where A is the adjacency matrix, m is the number of edges, k_v is the degree of node v, and s_v is the community indicator which is defined as:

$$s_v = \begin{cases} 1 = & \text{if node } v \text{ belongs to community 1} \\ -1 = & \text{if node } v \text{ belongs to community 2} \end{cases} \tag{2.6}$$

Note that this method only works for parititioning a graph into two communities. To extend it to identify more communities, we can (1) use hierarchical partitioning strategy: first partitioning a graph into 2 communities, then each community can be further partitioned into two smaller communities using the same idea (i.e., maximizing modularity Q within this community); or (2) Generalizing the objective function (i.e., maximizing Q) for partitioning a graph into multiple communities.

2.2.4 Embedding based Methods

With the rapid development of deep learning techniques [GBC16], network embedding approaches which are based on deep learning have attracted enormous attention from machine learning and graph mining communities recently. Existing network embedding methods have reported promising results in mining local structures of graphs, e.g., link prediction [GL16], community detection [WCW+17] and node classification [PARS14].

Community Embedding (ComE) [CZC+17] is used as the example to introduce the embedding based method for community detection and there exists a closed loop among community detection, community embedding and node embedding, as shown in Figure 2.2. Similar to node embedding, community embedding aims to learn a latent representation for each community. The intuition in ComE is that node embedding can help improve community detection (i.e., Step 1), which outputs good communities for fitting meaningful community embedding (i.e., Step 2). On the other hand, community embedding can be used to optimize node embedding (i.e., Step 3).

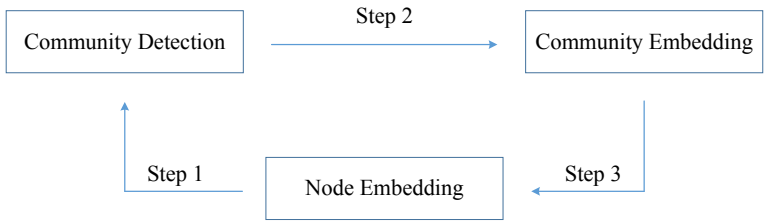


Figure 2.2: The closed loop in ComE [CZC+17].

Modularized Nonnegative Matrix Factorization (M-NMF) model [WCW+17] is another embedding based method for community detection. M-NMF exploits the consensus relationship between the representations of nodes and community structure, and then jointly optimizes NMF based representation learning model and modularity based community detection model in a unified framework. To capture the local structures, it combines the first- and second-order proximity, i.e., combining adjacency matrix and cosine similarity between two nodes.

2.3 Global Structure Mining

As introduced in Section 1, global structures of graphs capture the topological properties of graphs from the global perspective. Global structures can be represented by roles and positions. Therefore, global structure mining are studied in several different tasks, e.g., role discovery [RA+15], graph transfer learning [HGER+12], etc., since these tasks consider the global topological properties and are graph-independent. In this section, we focus on the problem of role discovery as the fundamental task in global structure mining.

There are limited number of surveys on role discovery compared to community detection problem. In [RA+15], role discovery methods have been categorized into graph-based, feature-based and hybrid methods. But this categorization is coarse and some representative studies have been ignored. In this section, we categorize these methods into seven families:

- equivalence based methods which are based on the defined equivalence relation;
- similarity based methods which require the pairwise similarity calculation;

- blockmodel based methods which are Bayesian statistical methods;
- feature based methods which first extract structural features explicitly; and
- embedding based methods which learn latent representations of nodes as features.

In this section we will introduce several most representative methods in each category .

2.3.1 Equivalence based Methods

Roles have been studied in social science and one of the first methods proposed by social scientists is equivalence based method. Two nodes that have the same role are in an equivalence relation. Formally, an equivalence relation E is any relation that satisfies these three conditions:

- *Transitivity*: $(a,b),\ (b,c) \in E \Rightarrow (a,c) \in E$;
- *Symmetry*: $(a,b) \in E \Leftrightarrow (b,a) \in E$;
- *Reflexivity*: $(a,a) \in E$.

Different types of equivalence relations can be defined to meet the above conditions. Among them, four types of equivalences are the most well-known and have been widely used in social science and computer science research: structural, automorphic, regular and stochastic equivalence. The taxonomy of these relations are shown in Figure 2.3. Structural, automorphic, and regular equivalence are deterministic since nodes are partitioned into different roles according the corresponding equivalence definitions. Besides, the relation among these three deterministic equivalences is shown in Figure 2.4 where structural equivalence can be viewed as a special case of automorphic equivalence and automorphic equivalence is part of regular equivalence. Stochastic equivalence is probabilistic because it gives a probability of each node belongs to different roles. In this section, we will discuss each type of equivalence with examples and methods.

Structural Equivalence

Two nodes i and j are structurally equivalent if, for all nodes, $k = 1,2,...,g$ $(k \neq i,j)$, node i has an edge to k, if and only if j also has an edge to k, and i has an edge from k if and only if j also has an edge from k. Formally,

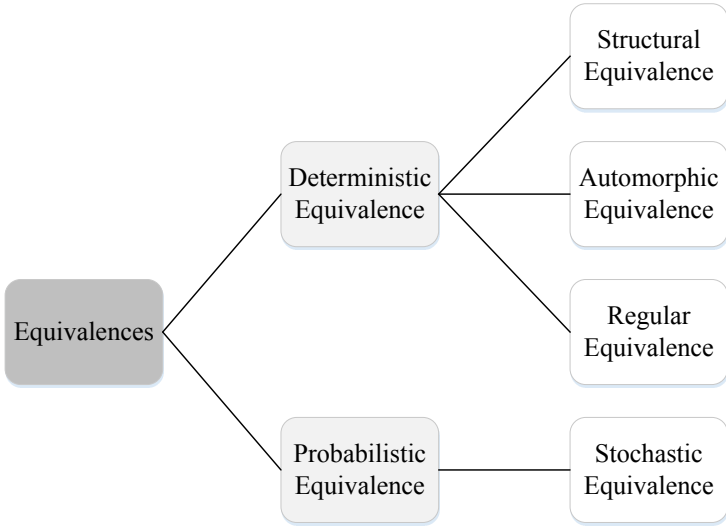


Figure 2.3: The taxonomy of structural, automorphic, regular and stochastic equivalence relations.

Definition 2.5. (**Structural Equivalence** [LW71,WF94]) Two nodes i and j are structurally equivalent if $i \longrightarrow k$ if and only if $j \longrightarrow k$, and $k \longrightarrow i$ if and only if $k \longrightarrow j$, for all nodes, $k = 1,2,...,g$ $(k \neq i, j)$.

Note that this is a general definition on arbitrary social network. We show an example of structural equivalence on the Borgatti and Everett network in Figure 2.5. In this example, it can be observed that nodes are partitioned into many different roles. In fact, this is not desired especially in real-world graphs. Structural equivalence may lead to a large number of roles and rarely appears in real-world networks.

Methods

CONCOR (Convergence of iterated correlations) [BBA75] is a hierarchical divisive method to discovery roles according to the definition of structural equivalence. The procedure of CONCOR consists of two steps:

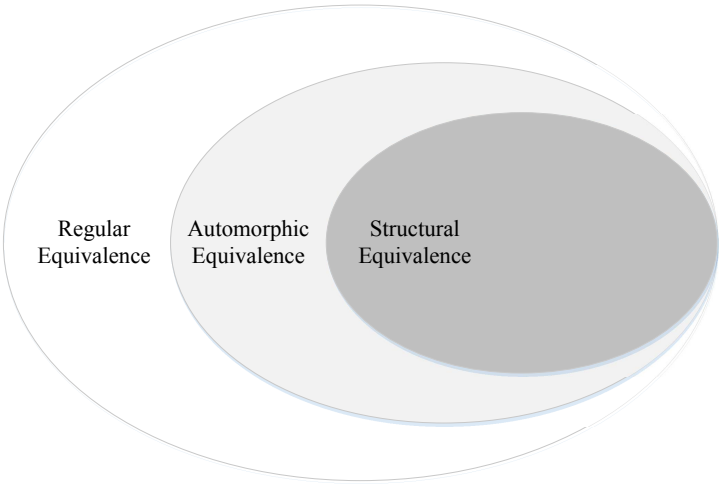


Figure 2.4: The relation of three deterministic equivalence relations.

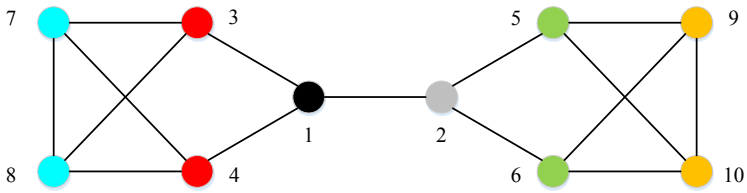


Figure 2.5: Structural equivalence on the Borgatti and Everett network.

Step 1 Calculate correlations, e.g., Pearson correlation, between rows (or columns) repeatedly on the adjacency matrix until the resultant correlation matrix consists of +1 and -1 entries;

Step 2 Split the last correlation matrix into two structurally equivalent submatrices (a.k.a. blocks): one with +1 entries, another with -1 entries.

Note that successive split can be applied to submatrices in order to produce a hierarchy (where every node has a unique position). Nodes in the same submatrix

belong to the same role, i.e., any pair of two nodes in the role are structurally equivalent.

STRUCTURE [Bur76] is a hierarchical agglomerative approach and has three steps:

Step 1 For each node i, create its feature vector by concatenating its row and column vectors from the adjacency matrix;

Step 2 For each pair of nodes (i, j), measure the square root of sum of squared differences between the corresponding entries in their feature vectors;

Step 3 Merge entries in hierarchical fashion until their difference is less than a predefined threshold.

Non-negative matrix tri-factorization (NMTF) based methods Recently, non-negative matrix tri-factorization (NMTF) based methods have been proposed by data mining researchers [BWT+17, BQD18] and they claimed that NMTF can effectively model the definition of structural equivalence. Formally, the objective function is defined as:

$$\min_{C,M} \|A - CMC^T\| \quad s.t.\ C^T C = I, \tag{2.7}$$

where A is the adjacency matrix, C is the role membership matrix and M indicates the interaction between roles.

Different variants extend the basic objective function by incorporating different components. For instance, FactorBlock [CLK+13] takes into account the noise and sparsity of network to discover more accurate blocks. Formally, it aims to minimize the following objective function:

$$\min_{C,M} \|(A - CMC^T) \circ U\| + \beta \|M_{ideal} - M\| \quad s.t.\ C^T C = I, \tag{2.8}$$

where the first component is similar to the basic NMTF based framework with an extra variable U to handle the sparsity of networks and the second constraint term tries to find an M that is as close as possible to the ideal for the particular equivalence required.

Non-negative symmetric matrix tri-factorization model with orthogonality constraint and spatial continuity regularization (ONMFtF-SCR) [BWT+17] integrates a graph/spatial regularization:

$$\min_{C,M} \|A - CMC^T\| + \beta tr(C^T \Theta C) \quad s.t.\ C^T C = I \tag{2.9}$$

where the regularization $tr(C^T \Theta C)$ guarantees each discovered node to be spatially continuous.

A semi-supervised framework based on NMTF has been proposed in [GCS+18] where *must-link* and *cannot-link* constraints have been incorporated into the objective function as the guided supervision. Formally,

$$\min_{C,M} \|A - CMC^T\| + \frac{1}{2}(1-C) \otimes (Q_{ML} \times C) + \frac{1}{2} C \otimes (Q_{CL} \times C) \tag{2.10}$$

where matrices Q_{ML} and Q_{CL} represent the cost coefficients for each pairwise constraint. In particular, Q_{ML} and Q_{CL} are $n \times n$ non-negative real valued matrices quantifying the cost of violating each of the must-link and cannot-link constraints respectively.

Automorphic Equivalence

Automorphic equivalence is based on the concept of automorphism. In graph theory, an automorphism of a graph is a form of symmetry in which the graph is mapped onto itself while preserving the edge–vertex connectivity. Formally,

Definition 2.6. (**Graph Automorphism**) An automorphism of a graph $G = (V, E)$ is a permutation σ of the node set V, such that the pair of nodes (u, v) form an edge if and only if the pair $(\sigma(u), \sigma(v))$ also form an edge, i.e., it is a graph isomorphism from G to itself.

Based on this definition, automorphic equivalence is formally defined as:

Definition 2.7. (**Automorphic Equivalence** [BE92]) Two nodes u and v of a labeled graph G are automorphically equivalent if there exists an automorphism σ with $\sigma(u) = v$ and $\sigma(v) = u$, i.e., two automorphically equivalent nodes share exactly the same label-independent properties.

Automophic equivalence generalizes the concept of the structural equivalence, i.e., if node i and j are structurally equivalent, they are also automorphically equivalent. Intuitively, actors are automorphically equivalent if we can permute the graph in such a way that exchanging the two actors has no effect on the distances among all actors in the graph. If we want to assess whether two actors are automorphically equivalent, we first imagine exchanging their positions in the network. Then, we look and see if, by changing some other actors as well, we can create a graph in which all of the actors are the same distance that they were from one another in the original graph.

Methods

Although the automophic equivalence is more generalized in modeling the equivalence relation compared to the structural equivalence, it attracts less attention in both social science and data mining communities. A representative method to cluster nodes based on automorphic equivalence has been proposed in [Spa93] which scales linearly to the number of edges. This method uses numerical signatures on degree sequences of neighborhoods to cluster nodes into different equivalent sets.

Regular Equivalence

Two nodes are said to play the same role (i.e., are regularly equivalent) if they have edges to the same roles. For example, two PhD students will have similar types of relations (as each other) to professors, secretary, other PhD students, and so on. Similarly, two professors tend to have similar relations as each other with students and other professors. We show an example of regular equivalence on the Borgatti and Everett network in Figure 2.6. In this example, we can observe that the partitioned results are more practically meaningful compared to other equivalence relations.

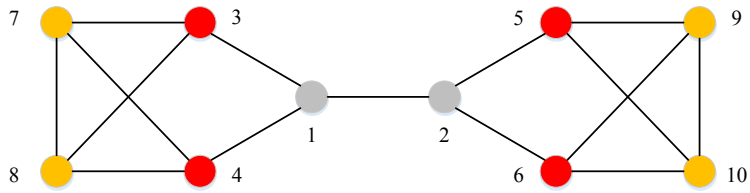


Figure 2.6: Regular equivalence on the Borgatti and Everett network.

Definition 2.8. (**Regular Equivalence** [EB94, Dor17]) An equivalence relation $\equiv$ over a network $G = (V, E)$ is regular if and only if for all $v_i, v_j, v_k \in V$, $v_i \equiv v_j$ implies:

- if $(v_i, v_k) \in E$, then there exists a $v_l \in V$ such that $(v_j, v_l) \in E$ and $v_l \equiv v_k$; and
- if $(v_k, v_j) \in E$, then there exists a $v_l \in V$ such that $(v_l, v_j) \in E$ and $v_l \equiv v_k$.

Methods

REGE REGE is an iterative algorithm which produces a measure M_{ij} of the extent of equivalence of all pairs of node i and j. It starts by setting $M_{ij} = 1$ for all pairs of actors. In each iteration, REGE re-computes M_{ij} for all pairs based on the degree to which i alters correspond to alters of j. In the first iteration, M_{ij} is calculated by numbering the extent to which i's edges to her alters correspond to j's edges to

The REGE algorithm computes a matrix of similarity scores based on how well a particular edge between two actors in different equivalence classes can be mapped to edges that span the same two classes. It works as follows [WF94] [Dor17]:

- Define a measure matrix M^t which quantifies the degree to which nodes v_i and v_j are regularly equivalent at t^{th} iteration (where $M^t_{ij} = 1$ if they are perfectly equivalent, $M^t_{ij} = 0$ if they are perfectly inequivalent). Initialize $M^0_{ij} = 1$ for all i and j and $t = 0$.
- Select a maximum number of iterations x.
- for $t = 0, 1, ..., x$ compute:

$$M^{t+1}_{ij} = \frac{\sum_{k=1}^{g} \max_{m=1}^{g} M^t_{km}({}_{ij}\mathcal{M}_{km} + {}_{ji}\mathcal{M}_{km})}{\sum_{k=1}^{g} \max_{m}^{*}({}_{ij}Max_{km} + {}_{ji}Max_{km})} \tag{2.11}$$

- Return M^x.

where ${}_{ij}Max_{km} = \max(x_{ik}, x_{jm}) + \max(x_{ki}, x_{mj})$ and ${}_{ij}\mathcal{M}_{km} = \min(x_{ik}, x_{jm}) + \min(x_{ki}, x_{mk})$ scores how well edges from i to a specific node k are matched by ties from node j to some other node m.

It is worth noting that regular equivalence from social science is equal to **bisimulation** from computer science [MM03]. Bisimulation captures the behavior of a node in terms of its outgoing edges and the behavior of the target nodes of those edges [LFH+13, vHFP16]. To be more generalized, we consider k-bisimulation where k is a flexible parameter to control the depth of this behavior. Formally, given a graph $G = \{V, E\}$ and non-negative integer k, nodes $u, v \in V$ are k-bisimulation (denoted as $u \approx^k v$) if and only if the following three conditions hold:

- if $k > 0$ then $\forall u' \in V : u \to u' \Rightarrow \exists v' \in V : v \to v' \bigwedge u' \approx^{k-1} v'$, and

- if $k > 0$ then $\forall v' \in V : v \to v' \Rightarrow \exists u' \in V : u \to u' \bigwedge v' \approx^{k-1} u'$.

Same to regular equivalence, the k-bisimulation relation is an equivalence relation, and hence we can partition the set of nodes V into equivalence classes and each class denotes a role.

2.3.2 Similarity based Methods

In the literature, a variety of structural similarity measures have been proposed to calculate node similarity based on the structures of networks, e.g., SimRank [JW02], MatchSim [LLK09] and RoleSim [JLH11, JLL14]. However, not all these measures can capture the global structural role information because some requirement of Axiomatic Role Similarity Properties for modeling the equivalence [JLH11]. Therefore, in this section we only introduce RoleSim as the similarity based method. The higher similarity between nodes, the more likely they belong to the same role. RoleSim also generalizes Jaccard coefficient and corresponds linearly to the maximal weighted matching. RoleSim similarity between two nodes u and v is defined as:

$$RoleSim(u,v) = (1-\beta) \max_{M(u,v)} \frac{\sum_{(x,y)\in M(u,v)} RoleSim(x,y)}{|N(u)| + |N(v)| - |M(u,v)|} + \beta \tag{2.12}$$

where $|N(u)|$ and $|N(v)|$ are the numbers of neighbors of node u and v, respectively. $M(u,v)$ is a matching between $N(u)$ and $N(v)$, i.e., $M(u,v) \subseteq N(u) \times N(v)$ is a bijection between $N(u)$ and $N(v)$. The parameter β is a decay factor where $0 < \beta < 1$. The intuition of RoleSim is that two nodes are structurally similar if their corresponding neighbors are also structurally similar. This intuition is consistent with the notion of automorphic and regular equivalence [WF94].

In practice, RoleSim values can be computed iteratively and are guaranteed to converge. The procedure of computing RoleSim consists of three steps:

- Step 1: Initialize matrix of RoleSim scores R^0;
- Step 2: Compute the k^{th} iteration R^k scores for the $(k-1)^{th}$ iteration's values, R^{k-1} using:

$$R^k(u,v) = (1-\beta) \max_{M(u,v)} \frac{\sum_{(x,y)\in M(u,v)} R^{k-1}(x,y)}{|N(u)| + |N(v)| - |M(u,v)|} + \beta \tag{2.13}$$

- Step 3: Repeat Step 2 until R values converge for each pair of nodes.

2.3.3 Feature based Methods

Feature based methods for role discovery are typically data mining techniques. These methods consist of two steps: feature extraction and role assignment [RA+15].

Cascade based Social Roles

The problem of dynamic inference of social roles in information cascades has been studied in [CRPS14]. Three types of features are used in this study including degree-based, neighbor-based and global features. These features correspond to the first, second and higher order proximity which have been widely used in graph mining tasks. In specific, these features are:

- *degree-based features*: the normalized node degree, the standard deviation of degree and the normalized average degree;
- *neighbor-based features*:the locality index, the common neighbors and the clustering coefficient;
- *global features*: the eigenvector centrality.

After extracting these features, an dynamic unsupervised methodology based on ensemble clustering is used to group users into their social roles in a network. In order to study the dynamic role, the clustering method exploits not only the current topological positions, but also considering the history over time.

Social Roles and Statues Model

Social Roles and Statues model (SRS) [ZWY+13], although aim at inferring social roles, employs a series of global structural features. By exploiting the correlations with the network structures of users and their corresponding social roles/statues, five social principles and concepts are used as the features, i.e., homophily, triadic closure, reachability, embeddedness and structural holes. These features can be grouped into:

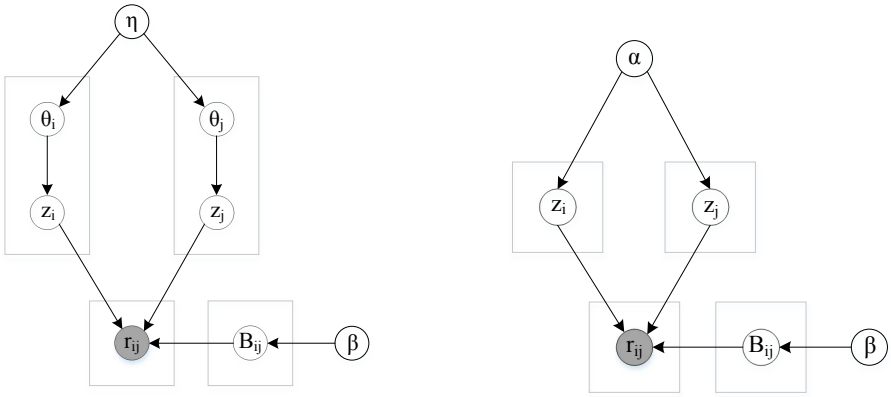
- *first-order features*: homophily which is based on the edges between nodes;
- *second-order features*: reachability which consists of degree centrality, average neighbor degrees and median neighbor degrees and embeddedness which is based on common neighbors;
- *first-order features*: triadic closure and structural holes which are based on the global properties.

Role Extraction Model

Role eXtraction (RolX) [HGER+12] uses a different feature extraction method. Instead of using predefining features based on social science and graph theory, RolX defines a recursive feature extraction method: it first extracts local and egonet features based on counts of links adjacent to a node and within and adjacent to the egonet of the same node, and then aggregates egonet-based features in a recursive fashion until no informative feature can be added. After obtaining the feature matrix, nonnegative matrix factorization [LS01] has been used to assign role by factorizing the feature matrix into two lower rank matrices, a role membership matrix and a feature associated matrix.

2.3.4 Blockmodel based Methods

Previous work on role discovery can be categorized into two types: graph-based methods and feature-based methods [RA+15]. Different from feature-based methods, graph-based methods take the graph as input directly and then learn the role assignment where each block indicates a role. The most representative graph-based method is stochastic blockmodel [HLL83, ABFX08, KTG+06]. We will introduce several selective blockmodel based methods.



(a) Graphical representations of MMSB.

(b) Graphical representations of IRM.

Figure 2.7: Graphical representations of different stochastic blockmodels.

Mixed Membership Stochastic Blockmodel (MMSB)

Stochastic blockmodel (SBM) [HLL83] is the original generative model to detect blocks in networks. SBM partitions nodes in hard clustering and is not flexible to incorporate prior knowledge. To solve these problems, mixed membership stochastic blockmodel (MMSB) [ABFX08] has been proposed where a role distribution for each node will be inferred and the graphical representation of MMSB is shown in Figure 2.7a. Formally, the generative process of MMSB is:

$$\begin{aligned} \theta|\alpha &\sim Dirichlet(\eta) \\ z|\theta &\sim Multinomial(\theta) \\ B_{ij}|\beta^1,\beta^2 &\sim Beta(\beta^1,\beta^2) \\ r_{ij}|z,B &\sim Bernoulli(B(z_i,z_j)) \end{aligned} \tag{2.14}$$

where $a|b \sim Distribution(b)$ means that sampling the variable a from the distribution with parameter b.

Infinite Relational Blockmodel (IRM)

However, MMSB requires the number of roles/blocks as the input which may be difficult to obtain in advance in practice. Infinite relational model (IRM) [KTG+06] has been proposed using the Chinese restaurant process (CRP) [P+02] as the prior. As a discrete-time stochastic process, CRP can generate an infinite number of clusters so IRM can infer the right number of roles based on the observed edges. The graphical representation of IRM is shown in Figure 2.7b and the formal generative process is:

$$\begin{aligned} z|\alpha &\sim CRP(\alpha) \\ B_{ij}|\beta^1,\beta^2 &\sim Beta(\beta^1,\beta^2) \\ r_{ij}|z,B &\sim Bernoulli(B(z_i,z_j)) \end{aligned} \tag{2.15}$$

Note that in both MMSB and IRM, the essence is to infer the latent variables, i.e., roles, based on the observed edges between nodes.

2.3.5 Embedding based Methods

Network embedding aims to map nodes in a network into a low-dimensional space according to their structural information in the network. It has been

reported that using embedded node representations can achieve promising performance on many network analysis tasks including community detection, role discovery, link prediction.

Role-based Graph Embedding

Role2Vec [ARL+18] is a role-based graph embedding framework. *Role2Vec* framework uses the flexible notion of attributed random walks, and generalizes existing network embedding methods such as DeepWalk, node2vec, and many others that leverage random walks. *Role2Vec* first utilizes a function $\Phi: x \rightarrow \omega$ to map each node attribute vector to a type, such that two nodes belong to the same type if they are structurally similar. In specific, there are two types of mapping functions proposed in [ARL+18]: (1) binary function, which is defined as:

$$\Phi(x) = x_1 \circ x_2 \circ ... \circ x_K, \tag{2.16}$$

where $x = [x_1, x_2, .., x_K]$ is an attribute vector and $\circ$ is a binary operator such as concatenation, sum, among others. (2) factorization based function, which is learned by solving an objective function.

Then it extends the traditional random walks by incorporating the mapped node types, named attributed random walks. Let x_i be a K-dimensional vector for node v_i. An attributed walk of length L is defined as a sequence of adjacent node-type (defined in the first step), i.e.,

$$\Phi(x_{v_0}), ..., \Phi(x_{v_t}), \Phi(x_{v_{t+1}}), ..., \Phi(x_{v_{L-1}}), \tag{2.17}$$

induced by a randomly chosen sequence of indices generated by a random walk of length L starting at v_0, and a function $\Phi: x \rightarrow \omega$ that maps an input vector x to a node type $\Phi(x)$.

Finally, it learns the node embedding by maximizing the conditional probability of context given the observed node. Formally,

$$\mathbb{P}\big[\Phi(x_{c_i})|\Phi(x_i)\big] = \prod_{j \in c_i} \mathbb{P}(\Phi(x_j)|\Phi(x_i)). \tag{2.18}$$

In this way, the learned embedding structure can be shared among the nodes with the same type.

Deep Recursive Network Embedding

Previous network embedding approaches learn representations based on the first-order, i.e., edges, or second-order, i.e., edges, proximity. However, in practice nodes have similar roles or occupy similar positions may not be linked or

have common neighbor. To solve this problem, Deep Recursive Network Embedding (DRNE) has been proposed to learn network embeddings with regular equivalence introduced in Section 2.3.1. The essence of DRNE is that the embedding of a target node can be approximated by the aggregation of its neighbors' embeddings. Formally,

$$\mathscr{L} = \sum_{v \in V} \|X_v - Agg(\{X_u | u \in N(v)\})\|_F^2, \tag{2.19}$$

where $N(v)$ is the neighbor set of node v and $Agg(\cdot)$ is the aggregating function.

The framework of DRNE is shown in Figure 2.8 which consists of four components:

- (a) Sampling neighborhoods. The degrees of networks in practice follow the power-law distribution, so DRNE first samples the neighbors for each node to make the neighbor distribution more balanced.
- (b) Sorting neighborhoods. Due to the missing of orders in the neighbors of a node, DRNE uses the degree of nodes as the criterion to sort neighbors into an ordered sequence. Degree is a simple and efficient measure for neighbor ordering and degree often plays an important role in many graph-theoretic measures.
- (c) Layer-normalized LSTM. LSTM is known to be effective for modeling sequences. Thus, Layer-normalized LSTM is applied to learn the representations with regular equivalence and the layer normalization results in more stable dynamics.

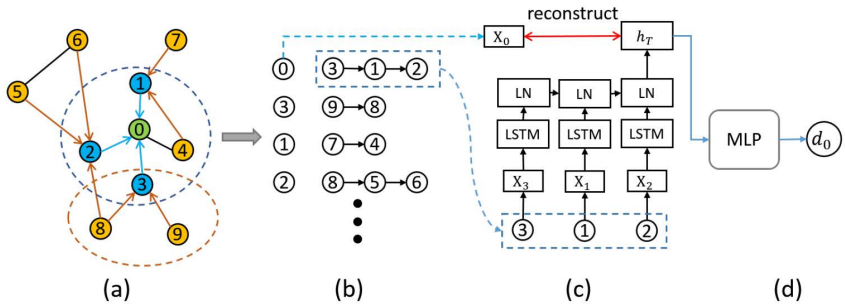


Figure 2.8: Framework of DRNE [TCW+18].

- (d) A weakly guided regularizer. Node degree is utilized as the weakly guided information and impose a constraint that the learned embedding of a node should be able to approximate the degree of the node.

GraphWave

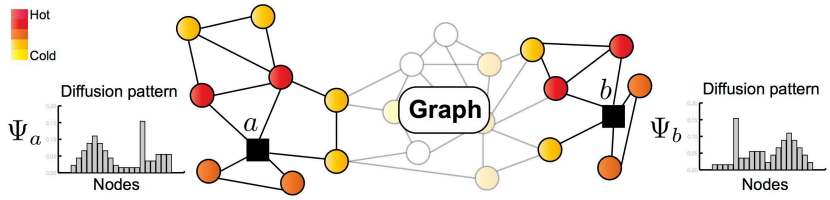


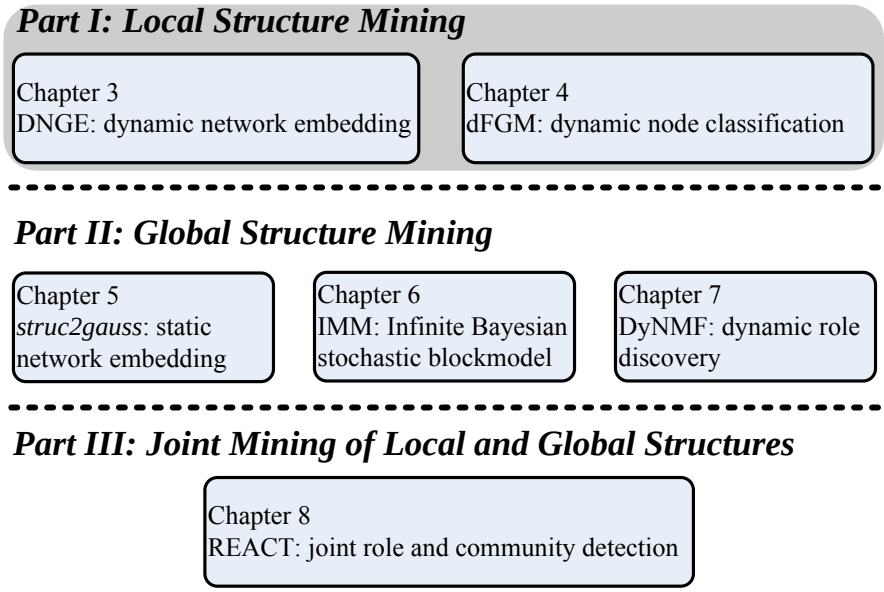
Figure 2.9: Illustration of GraphWave approach [DZHL18].

GraphWave [DZHL18] represents network neighborhood of each node via a low-dimensional embedding based on the diffusion of a spectral graph wavelet centered at the node. Graph wavelet borrows the technique from graph signal processing, i.e., heat wavelet diffusion patterns. GraphWave is based the intuition that each node propagates a unit of energy over the graph and characterizes its neighboring topology based on the response of the network to this probe. The graphical illustration of GraphWave approach is shown in Figure 2.9 and the procedure of GraphWave includes:

- Using spectral graph wavelets to obtain a diffusion pattern for every node.
- Embedding spectral graph wavelet coefficient distributions into $2d$ space.
- Obtaining the embedding of node a by sampling the $2d$-dimensional parametric function at d evenly spaced points and concatenating the values.

Part I

Local Structure Mining



Chapter 3 presents the proposed model dFGM (dynamic Factor Graph Model): a factor graph model for node classification from the local perspective in dynamic networks. dFGM models three types of factors, i.e., node factor, correlation factor and dynamic factor, based on node features, node correlations and temporal correlations, respectively.

Chapter 4 presents DNGE (dynamic network embedding method with Gaussian Embedding): a network embedding framework to mine the local structures in dynamic networks. DNGE integrates Gaussian embedding techniques and temporal regularization to map nodes to Gaussian distributions. Thus, it can capture temporal information and model uncertainties in dynamic networks.

Chapter 3

Community Detection and Link Prediction on Dynamic Graphs

3.1 Introduction

In this chapter, we aim to answer the research question Q1.1 introduced in Chapter 1:

> Q1.1: How can we effectively detect local communities by capturing dynamics and uncertainties on dynamic graphs?

In order to detect local communities on dynamic graphs, different approaches have been proposed in the literature [GDC10] [LCZ+08] [LDH+17]. Earlier methods detected communities by exploiting local structural features either from predefined features based external knowledge or from heavy computation such as centrality based features. With the rapid development of deep learning techniques [GBC16], network embedding approaches based on deep learning become the most advanced method to extract structural features of graphs. Network embedding maps nodes in a graph into a low-dimensional space according to their structural information. Many attempts showed the effectiveness of network embedding techniques in representing the local structural information of graphs and improving the performance of different graph structure mining tasks [CLX15, GL16, PARS14, TQW+15]. Therefore, in this chapter, we propose a network embedding approach to mine the local graph structures and detect local communities on dynamic graphs.

To learn effective representations for graphs, a vast number of models and theories have been developed in recent years but there are some limitations. Early network embedding methods mainly focus on static networks [PARS14, TQW+15, GL16]. However, many real-world networks are dynamic with evolving structures, e.g., friends added or deleted in a social network. The existence of dynamics makes network analysis a more challenging problem [LCZ+08, RGNH13, PZFP18]. Although current network embedding methods can be extended to dynamic scenario by learning node representations separately on each network snapshot, this leads to an alignment problem to represent nodes in a single coordinate space [YSD+17]. To solve this problem, recently some studies have been proposed to take into account the temporal information to learn embeddings, either in dynamic networks [TFBZ19, DWS+18, ZYR+18] or in streaming networks [BBK+18, NLR+18, LHD+19]. Nonetheless, real-world networks, especially dynamic and temporal networks, may be noisy and incomplete and in most cases these uncertainties are inevitable because of anomaly or missing information, e.g., spammers in social networks. Thus, another limitation still exists: how to capture the uncertainties of embeddings. Previous static and dynamic embedding methods represent a node into a point vector in the learned embedding space. Point vector representations are deterministic [DSPG16] and are not capable of modeling the uncertainties of node representations.

Motivated by these observations, we tackle the two major challenges in learning network representations in this chapter:

- **Dynamics modeling**: Dynamics is ubiquitous in real-world networks with evolving structures. How to effectively capture the dynamic and temporal information in learning dynamic representation in real-world networks?

- **Uncertainty modeling**: Uncertainties exist in real-world networks or appear during the collection of data. How to effectively model the uncertainties of learned embeddings in real-world networks?

To overcome these limitations, we propose a dynamic network embedding method with Gaussian Embedding, *DNGE*. *DNGE* learns node representations for dynamic networks in the space of Gaussian distributions and models dynamic information by integrating temporal smoothness as the regularization. We propose two different strategies, i.e., smoothness in means and smoothness in distributions, to model the dynamic information. Smoothness in means guarantees the learned node representations are consistent and smoothness in distributions allows that both embeddings and uncertainties can be shared between two consecutive network snapshots. To evaluate the performance of the

proposed *DNGE*, we conduct extensive experiments on both synthetic and real-world networks from different domains. For synthetic networks, community detection is employed and for real-world networks, we test the performance of link prediction.

Our contributions can be summarized as follows:

- We propose a dynamic network embedding framework *DNGE* to learn node representations. *DNGE* integrates Gaussian embedding techniques and temporal regularization to map nodes to Gaussian distributions. Thus, it can capture temporal information and model uncertainties in dynamic networks.
- We explore two different strategies to model dynamic information, i.e., smoothness in means and smoothness in distributions. These strategies allow us to learn dynamic embedding by sharing information between two consecutive network snapshots.
- We evaluate the effectiveness of *DNGE* in both synthetic and real-world networks. The experimental results demonstrate that our method is capable of preserving network structures, capturing dynamic information and modeling uncertainties in dynamic networks.

The rest of this chapter is organized as follows. Section 3.2 provides an overview of related work. We present the problem statement in Section 3.3. Section 3.4 introduces the general Gaussian embedding method. Section 3.5 explains the technical details of *DNGE*. In Section 3.6 we then discuss our experimental study. Finally, in Section 3.7 we draw conclusions and outline directions for future work.

3.2 Related Work

In this section, we only focus on the related research in the areas of network embedding and dynamic network analysis. For more work on local structural mining and community detection, please refer to Chapter 2.

3.2.1 Network Embedding

Previous methods for network embedding mainly focused on matrix factorization and eigendecomposition [SJ09, TDSL00] to reduce the dimension of net-

work data. With increasing attention attracted by neural network research, unsupervised neural network techniques have opened up a new world for embedding. DeepWalk [PARS14] introduces such embedding mechanism to networks by treating nodes as words and random walks as sentences. Afterwards, a sequence of studies has been conducted to improve DeepWalk either by extending the definition of neighborhood to higher-order proximity. LINE [TQW+15] captures both 1-step and 2-step local relational information of networks and then learns non-linear transformations from the data. Motivated by LINE, GraRep [CLX15] integrates global structural information by capturing k-step relational information of networks with $k > 2$. *node2vec* [GL16] proposes a new search strategy by combining BFS and DFS for modeling both homophily and structural equivalence. Other directions of network embedding includes: (1) incorporating extra information for node representations, such as attributes [LDH+17, WATL17] and heterogeneity [CHT+15, TQM15], (2) exploiting different types of networks, for instance, signed networks [WTA+17, KPLK18] and noisy networks [YCA+18, LJSP18], and (3) preserving specific structural properties, for example, community preserving embedding [WCW+17, ZLZ18] and role preserving embedding [TCW+18, PDZ+18]. The vast majority of existing studies focus on static networks and it is non-trivial to extend these methods to dynamic networks. Some exceptions include DANE [LDH+17], Dynamic Triad [ZYR+18] and DNE [DWS+18]. DANE models the dynamic network embedding problem as an online matrix factorization task. Dynamic Triad captures dynamics by modeling triadic closure process. DNE extends skip-gram by incorporating the influence of dynamic network changes. However, uncertainties are ignored in these studies.

However, real-world networks, especially dynamic and temporal networks, may be noisy and incomplete and in most cases these uncertainties are inevitable because of anomaly or missing information. Previous point based embedding methods are incapable of capturing the uncertainties. To capture the uncertainties of node representations, Gaussian embedding [VM14, ZCWZ18] have been used. In Gaussian embedding each node is represented as a Gaussian distribution instead of a deterministic vector where the mean indicates the position of this node in the embedding space and the covariance represents its uncertainty. Gaussian embedding for networks have been used in knowledge modeling [HLJZ15], graph classification [DSPG16], link prediction [BG17] and role discovery [PDZ+18].

An overview of state-of-the-art methods is shown in Table 3.1. We refer readers for a more detailed survey paper [CWPZ18]. In our work, we focus for the first time jointly on the aspects of **dynamics** and **uncertainty** modeling.

Table 3.1: A brief comparison of network embedding methods.

Methods	static vs. dynamic	uncertainty	streaming
DeepWalk [PARS14]	static	-	-
LINE [TQW+15]	static	-	-
node2vec [GL16]	static	-	-
GraphSAGE [HYL17]	static	-	-
Gauss Emb [VM14]	static	-	-
DVNE [ZCWZ18]	static	√	-
DyRep [TFBZ19]	dynamic	-	-
Dyn Triad [ZYR+18]	dynamic	-	-
DNE [DWS+18]	dynamic	-	-
DDNE [LZP+18]	dynamic	-	-
SGE [LHD+19]	-	-	√
CTDNE [NLR+18]	-	-	√
TO-GVAE [BBK+18]	-	-	√
DNGE (ours)	dynamic	√	-

3.2.2 Dynamic Network Analysis

Many real-world networks are dynamic with evolving structures, e.g., new friendship in Facebook. Hence, dynamic network analysis (DNA) is of huge demand in practice and theories. However, the existence of dynamics makes DNA a very challenging problem.

In the literature, several DNA problems have been explored. Facetnet [LCZ+08] discovers communities and analyze the evolution of communities in dynamic networks. ctRBM [LDL+14] predicts links in dynamic networks by proposing a new deep learning framework. [RSK+15] compares anomaly detection methods in dynamic networks. DBMM [RGNH13] and DyNMF [PZFP18] analyzes the global structures of evolving networks, i.e., role discovery. BCGD [ZGY+16] exploits the temporal latent space for dynamic networks to model the temporal link probability of node pairs. Although a range of DNA problems have been studied, they are designed to learn dedicated features for specific tasks, e.g., lower-rank representations of adjacency matrix for community detection and global structural patterns for role discovery. However, general dynamic network embedding, i.e., how to learn node representations in dynamic scenario for general purpose, remains a challenging problem.

3.3 Problem Statement

In this section, we first introduce the notation used in this study and then present the formal statement of the problem of dynamic network embedding using Gaussian embedding. A dynamic network is defined as $\mathcal{G} = \{G^t | t = 1, ..., T\}$ which consists of a series of graph snapshots $G^t = \{V, E^t\}$. Note that we assume edges to be undirected for simplicity and we also assume that the nodes are fixed in all snapshots and the edges change over time.

Problem 3.1. Given a dynamic network $\mathcal{G} = \{G^t | t = 1, ..., T\}$, the problem of **Dynamic Network Gaussian Embedding** aims to represent each node $v_i \in V$ in each snapshot t into a Gaussian distribution $P_i^{(t)}$ with mean $\mu_i^{(t)}$ and covariance $\Sigma_i^{(t)}$ in a low-dimensional space $\mathbb{R}^d$, i.e., learning a function $f : V \to \mathcal{N}(x; \mu, \Sigma)$, where $\mu \in \mathbb{R}^d$ is the mean, $\Sigma \in \mathbb{R}^{d \times d}$ is the covariance and $d \ll |V|$. In the space $\mathbb{R}^d$, the temporal information of nodes is preserved using explicit temporal regularization and the uncertainty of node representations is captured by the covariance Σ.

Note that by solving this problem, the ultimate target is to learn representations of nodes in a dynamic network which can well preserve the local structure of networks. Thus, the learned representations should perform well in community detection and link prediction.

3.4 Gaussian Embedding

Most previous embedding methods map each node to a fixed point vector in a low-dimension space so that the uncertainties of learned embeddings are ignored. Gaussian embedding aims to solve this problem by learning density-based distributed embeddings in the space of Gaussian distributions [VM14]. Gaussian embedding has been utilized in different graph mining tasks including triplet classification on knowledge graphs [HLJZ15], multi-label classification on heterogeneous graphs [DSPG16], link prediction and node classification on attributed graphs [BG17] and role discovery on graphs [PDZ+18].

The original Gaussian embedding [VM14] trains with a ranking-based loss based on the ranks of positive and negative samples. Positive samples consist of pairs of nodes that we want them to be close to each other in the embedding space while negative samples are the pairs of nodes that we aim to make them far from each other in the embedding space. The strategy to obtain these samples in our method will be introduced in Section 3.5.2. It can push scores of

positive pairs above negatives by a margin defined as:

$$\mathscr{L} = \sum_{(v,u)\in\Gamma_+} \sum_{(v',u')\in\Gamma_-} \max(0, m - \mathscr{E}(P_v, P_u) + \mathscr{E}(P_{v'}, P_{u'})), \tag{3.1}$$

where Γ_+ and Γ_- are the positive and negative pairs, respectively. $\mathscr{E}(\cdot,\cdot)$ is the energy function, P_v and P_u are the learned Gaussian distributions for nodes v and u, and m is the margin separating positive and negative pairs. Different energy functions can be used to measure the similarity of two distributions, e.g., expected likelihood based energy and KL divergence based energy [VM14]. Note that to reduce the computational complexity, for each Gaussian distribution $P \sim \mathcal{N}(x; \mu, \Sigma)$, we only consider the diagonal covariance for Σ. For simplicity, we use a similar but simpler objective function by removing the margin m in this work:

$$\mathscr{L}_{Gaussian} = -\sum_{(v,u)\in\Gamma_+} \mathscr{E}(P_v, P_u) + \sum_{(v',u')\in\Gamma_-} \mathscr{E}(P_{v'}, P_{u'}). \tag{3.2}$$

To avoid the means to grow to large and the covariances to be positive definite as well as reasonably sized, we regularize the means and covariances to learn the embedding. Due to the different geometric characteristics, two different hard constraint strategies have been used for means μ_i and covariances Σ_i, respectively. In particular, we have

$$\|\mu_i\| \leq C, \ \forall i, \qquad c_{min} I \prec \Sigma_i \prec c_{max} I, \ \forall i, \tag{3.3}$$

where C, c_{min} and c_{max} are the constraint parameters and in experiments they are set to 1.0, 0.5, and 2.0 respectively (these values can be set empirically and we leave the tuning of these parameters for future work). The constraint on means guarantees them to be sufficiently small and constraint on covariances ensures that they are positive definite and of appropriate size.

3.5 DNGE

3.5.1 Overview

As mentioned in Section 3.1, two major limitations exist in previous network embedding studies: *dynamics modeling* and *uncertainty modeling*. To overcome these limitations, we propose a dynamic network embedding framework with

Gaussian embedding (DNGE). *DNGE* consists of two components: Gaussian embedding component which learns node representations and models uncertainties, and dynamics modeling component which captures temporal information and smooths learned node representations. Formally, the objective function is defined as:

$$\mathscr{L}_{DNGE} = \min\Big(\sum_{t=1}^{T} \mathscr{L}_{Gaussian}^{(t)} + \lambda \sum_{t=1}^{T-1} \mathscr{L}_{Dynamic}^{(t)}\Big), \tag{3.4}$$

where $\mathscr{L}_{Gaussian}^{(t)}$ is the Gaussian embedding component in t^{th} snapshot and $\mathscr{L}_{Dynamic}^{(t)}$ is the dynamics modeling component between snapshot t and $t+1$. λ controls the importance of dynamics and in an extreme case, *DNGE* is equal to the static Gaussian embedding when $\lambda = 0$. The graphical representation of *DNGE* is shown in Fig. 3.1. We will introduce each component in details in following subsections.

3.5.2 Gaussian Embedding Component

The Gaussian embedding component can map each node i in the graph into a Gaussian distribution P_i with mean μ_i and covariance Σ_i. The formal objective

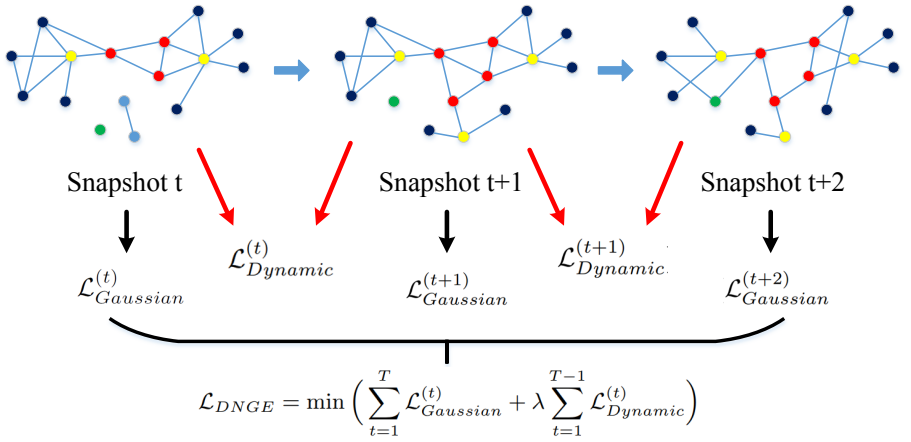


Figure 3.1: Graphical representation of *DNGE* framework which consists of Gaussian embedding component and dynamics modeling component.

function of Gaussian embedding is shown in Eq. (3.2). The key idea of network embedding is that two nodes connected in the network should have similar representations. In Gaussian embedding, we use Gaussian distributions to denote the representations so two connected nodes should have similar Gaussian distributions. Therefore, we should select an energy function to measure the similarity between two Gaussian distributions. As it has been reported that KL divergence based energy function can achieve better performance in word embedding [VM14] and network modeling [HLJZ15], we use KL divergence based energy function in this study.

KL divergence is a straightforward way to measure the similarity between two distributions. In this study, we define the KL divergence based energy function $\mathscr{E}_{KL}(P_i, P_j)$ as:

$$\begin{aligned} &\mathscr{E}_{KL}(P_i, P_j) = D_{KL}(P_i, P_j) \\ &= \int_{x\in\mathbb{R}} \mathcal{N}(x;\mu_i,\Sigma_i) \log \frac{\mathcal{N}(x;\mu_j,\Sigma_j)}{\mathcal{N}(x;\mu_i,\Sigma_i)} dx \\ &= \frac{1}{2}\Big\{ tr(\Sigma_i^{-1}\Sigma_j) + \mu^T \Sigma_i^{-1}\mu - \log\frac{\det(\Sigma_j)}{\det(\Sigma_i)} - d \Big\}, \end{aligned} \tag{3.5}$$

where $\mu = \mu_i - \mu_j$, d is the number of dimensions, P_i and P_j are the learned Gaussian embeddings for node i and j respectively. $tr(\cdot)$ is the trace function, $\det(\cdot)$ denotes the determinant of a matrix. In this work, we apply the one-order proximity hypothesis same to [PARS14, DSPG16], i.e., two connected nodes should have similar representations. Thus, the positive pair set Γ_+ (in Eq. (3.2)) are the edge set $E = \{(i,j)|i,j \in V\}$ and the negative pair set Γ_- are generated from random sampling [MCCD13]. For simplicity, we make the negative set contain the same number of pairs with the positive set.

3.5.3 Dynamics Modeling Component

The dynamics modeling component can capture the temporal information and smooth the learned node representations. To achieve this goal, we propose two smoothness strategies as a regularization term, smoothness in *means* and smoothness in *distributions*. Note that Gaussian embedding learns representations from pairs of nodes, so these dynamic regularization terms should be added on both nodes of a pair.

Smoothness in means

Smoothness in means guarantees the learned node representations, i.e., means of Gaussian distributions, are consistent. The most straightforward way to achieve it is to constrain the change of means in two consecutive network snapshots. Therefore, the formal definition of smoothness in *means* is defined as the square of Frobenius norm of means as follows:

$$SM_i^{(t)} = \|\mu_i^{(t)} - \mu_i^{(t-1)}\|_F^2. \tag{3.6}$$

Smoothness in distributions

In smoothness in *means* strategy, only representations have been taken into account but the uncertainties are ignored. Thus we propose another strategy which can integrate both means and covariances, smoothness in *distributions*. Smoothness in distributions allow that both embeddings and uncertainties can be shared between two consecutive snapshots. Formally, it is defined as:

$$SD_i^{(t)} = \mathscr{E}(P_i^{(t)}, P_i^{(t-1)}). \tag{3.7}$$

where $\mathscr{E}(\cdot,\cdot)$ is the method to measure the similarity between two distributions. To keep consistent, we continue to use KL divergence shown in Eq. (5.7).

3.5.4 Learning

To combine the Gaussian embedding component (Eq. (3.2)) and dynamics modeling component (Eq. (5.9) and Eq. (3.7)), we have the objective function for DNGE. For smoothness in means, in t^{th} snapshot we define the objective function $\mathscr{L}_{DNGE}^{(t)}$ as:

$$\min \sum_{\Gamma_+^{(t)}} \mathscr{E}_{KL}(P_i^{(t)}, P_j^{(t)}) - \sum_{\Gamma_-^{(t)}} \mathscr{E}_{KL}(P_{i'}^{(t)}, P_{j'}^{(t)}) + \lambda \sum_{\Gamma^{(t)}} (SM_i^{(t)} + SM_j^{(t)}), \tag{3.8}$$

where $\Gamma^{(t)}$ is obtained by concatenating positive pairs $\Gamma_+^{(t)}$ and negative pairs $\Gamma_-^{(t)}$ in snapshot t. We can compute the gradients of this objective function with

respect to the means $\mu^{(t)}$ and covariances $\Sigma^{(t)}$:

$$\frac{\partial \mathscr{L}^{(t)}}{\partial \mu_i^{(t)}} = -\Delta_{ij}^{(t)} + \lambda(2\mu_i^{(t)} - \mu_i^{(t-1)} - \mu_i^{(t+1)}) \tag{3.9}$$
$$\frac{\partial \mathscr{L}^{(t)}}{\partial \mu_j^{(t)}} = \Delta_{ij}^{(t)} + \lambda(2\mu_j^{(t)} - \mu_j^{(t-1)} - \mu_j^{(t+1)})$$
$$\frac{\partial \mathscr{L}^{(t)}}{\partial \Sigma_i^{(t)}} = \frac{1}{2}\Big((\Sigma_i^{(t)})^{-1}\Sigma_j^{(t)}(\Sigma_i^{(t)})^{-1} + \Delta_{ij}^{(t)}\Delta_{ij}^{(t)T} - (\Sigma_i^{(t)})^{-1}\Big)$$
$$\frac{\partial \mathscr{L}^{(t)}}{\partial \Sigma_j^{(t)}} = \frac{1}{2}\Big((\Sigma_j^{(t)})^{-1} - (\Sigma_i^{(t)})^{-1}\Big)$$

where $\Delta_{ij}^{(t)} = (\Sigma_i^{(t)})^{-1}(\mu_i^{(t)} - \mu_j^{(t)})$. Note that for boundary conditions when $t = 1$ and $t = T$, some terms in the gradient will vary slightly.

Similarly, for smoothness in distributions, in t^{th} snapshot we define the objective function $\mathscr{L}_{DNGE}^{(t)}$ as:

$$\min \sum_{\Gamma_+^{(t)}} \mathscr{E}_{KL}(P_i^{(t)}, P_j^{(t)}) - \sum_{\Gamma_-^{(t)}} \mathscr{E}_{KL}(P_{i'}^{(t)}, P_{j'}^{(t)}) + \lambda \sum_{\Gamma^{(t)}} (SD_i^{(t)} + SD_j^{(t)}), \tag{3.10}$$

The gradients for means $\mu^{(t)}$ and covariances $\Sigma^{(t)}$ are:

$$\frac{\partial \mathscr{L}^{(t)}}{\partial \mu_i^{(t)}} = -\Delta_{ij}^{(t)} - \lambda(\tilde{\Delta}_i^{(t)} + \tilde{\Delta}_i^{(t+1)}) \tag{3.11}$$
$$\frac{\partial \mathscr{L}^{(t)}}{\partial \mu_j^{(t)}} = \Delta_{ij}^{(t)} - \lambda(\tilde{\Delta}_j^{(t)} + \tilde{\Delta}_j^{(t+1)})$$
$$\frac{\partial \mathscr{L}^{(t)}}{\partial \Sigma_i^{(t)}} = \frac{1}{2}\Big\{\Big((\Sigma_i^{(t)})^{-1}\Sigma_j^{(t)}(\Sigma_i^{(t)})^{-1} + \Delta_{ij}^{(t)}\Delta_{ij}^{(t)T} - (\Sigma_i^{(t)})^{-1}\Big)$$
$$+ \Big((\Sigma_i^{(t)})^{-1} - (\Sigma_i^{(t+1)})^{-1}\Big) + \Big((\Sigma_i^{(t)})^{-1}(\Sigma_i^{(t-1)})(\Sigma_i^{(t)})^{-1} + \tilde{\Delta}_i^{(t)}\tilde{\Delta}_i^{(t)T} - (\Sigma_i^{(t)})^{-1}\Big)\Big\}$$
$$\frac{\partial \mathscr{L}^{(t)}}{\partial \Sigma_j^{(t)}} = \frac{1}{2}\Big\{\Big((\Sigma_j^{(t)})^{-1} - (\Sigma_i^{(t)})^{-1}\Big) + \Big((\Sigma_j^{(t)})^{-1} - (\Sigma_j^{(t+1)})^{-1}\Big)$$
$$+ \Big((\Sigma_j^{(t)})^{-1}(\Sigma_j^{(t-1)})(\Sigma_j^{(t)})^{-1} + \tilde{\Delta}_j^{(t)}\tilde{\Delta}_j^{(t)T} - (\Sigma_j^{(t)})^{-1}\Big)\Big\}$$

where $\Delta_{ij}^{(t)} = (\Sigma_i^{(t)})^{-1}(\mu_i^{(t)} - \mu_j^{(t)})$, $\tilde{\Delta}_i^{(t)} = (\Sigma_i^{(t)})^{-1}(\mu_i^{(t)} - \mu_i^{(t-1)})$ and $\tilde{\Delta}_j^{(t)} = (\Sigma_j^{(t)})^{-1}(\mu_j^{(t)} - \mu_j^{(t-1)})$.

We use AdaGrad [DHS11] to optimize the objective function 8.8. The learning procedure is described in Algorithm 1. Initialization phase is from line 1 to 5, context generation is shown from line 6 to 8, and Gaussian embeddings are learned from line 9 to 12.

Algorithm 1 The Learning Algorithm of *DNGE*

Require: A dynamic graph $\mathcal{G} = \{G^t | t = 1, ..., T\}$, embedding dimension d, constraint values c_{max} and c_{min} for covariance, learning rate α, and max number of iterations n.

Ensure: Gaussian embeddings (mean vector $\mu_v^{(t)}$ and covariance matrix $\Sigma_v^{(t)}$) for nodes $v \in V$ in each snapshot t

1: **for all** $v \in V$ in each snapshot t **do**
2: Initialize mean $\mu_v^{(t)}$ for v in snapshot t randomly
3: Initialize covariance $\Sigma_v^{(t)}$ for v in snapshot t randomly
4: Regularize $\mu_v^{(t)}$ and $\Sigma_v^{(t)}$ with constraint in Eq. (3.3)
5: **end for**
6: **for all** snapshot t **do**
7: Generate positive and negative sets $\Gamma_+^{(t)}$ and $\Gamma_-^{(t)}$ for each node
8: **end for**
9: **while** not reach the maximum iteration n **do**
10: Update means and covariances based on Eq. (3.9) or (3.11)
11: Regularize μ and Σ with constraint in Eq. (3.3)
12: **end while**

3.6 Experimental Analysis

To validate the effectiveness of *DNGE* from different perspectives, we conduct two types of experiments on several real-world networks from different domains, i.e., community detection and link prediction. For experiments on community detection, we validate the ability of preserving network structures in *DNGE*. For experiments on link prediction, we evaluate the performance of dynamic information modeling in *DNGE*. Additionally, we analyze the uncertainty modeling results and explore the parameter sensitivity of *DNGE*.

Data Sets. We conduct experiments on five real-world networks from different domains. The first data set, i.e., Epinions, is used for community detection[1].

[1] `https://www.cse.msu.edu/~tangjili/trust.html`

Table 3.2: A brief statistics of real-world networks.

Data set	# nodes	# edges	# snapshots
Epinions	8518	300548	10
Enron	147	1666	9
Messages	1889	59835	5
Reality	6809	9467	10
Facebook	44416	196414	12

The remaining data sets are used for link prediction[2]. A brief overview of these networks is shown in Table 3.2.

3.6.1 Community Detection

To validate *DNGE*'s effectiveness in preserving network structures, we first conduct community detection experiments on Epinions network. Note that we only use means of learned Gaussian embeddings as the representations of nodes in community detection task. Since some baselines are specific for static networks, we compare the average performance over all snapshots. Epinions is a product review site in which users can share their reviews and opinions about products. Also a network can be built based on user' trust relationship. Following [LDH+17], we take the major categories of reviews by users as the ground truth of class labels. As we have ground-truth community labels, Normalized Mutual Information (NMI) and Purity are employed to verify the results. Formally, Purity measures the extent to which each cluster contained data points from primarily one class. The purity of a clustering is obtained by the weighted sum of individual cluster purity values which defined as:

$$Purity = \frac{1}{N}\sum_{i=1}^{k} max_j |c_i \cap t_j|, \tag{3.12}$$

where N is number of objects, k is number of clusters, c_i is a cluster in C, and t_j is the classification which has the max count for cluster c_i. NMI evaluates the clustering quality based on information theory, and is defined by normalization on the mutual information between the cluster assignments and the pre-existing

[2] http://networkrepository.com/dynamic.php

input labeling of the classes:

$$NMI(C,D) = \frac{2 * I(C,D)}{H(C) + H(D)}, \tag{3.13}$$

where obtained cluster C and ground-truth cluster D. The mutual information $I(C,D)$ is defined as $I(C,D) = H(C) - H(C|D)$ and $H(C) = -\sum_{c \in C} p(c) \log p(c)$ is the entropy.

Baselines. We select three different types of methods for community detection: adjacency-based methods, static embedding methods and dynamic embedding methods.

- **Adjacency-based methods**: Spectral clustering and Girvan-Newman (GN) algorithm have been compared since they are simple and widely used in the task of community detection.
- **Static network embedding methods**: State-of-the-art network embedding approaches are utilized as baselines, i.e., DeepWalk [PARS14], LINE [TQW+15], *node2vec* [GL16], and Gaussian embedding (Gauss Emb) [VM14] (applying Gaussian word embedding method to network data). The learned representations of nodes are treated as features for clustering.
- **Dynamic network embedding methods**: we compare Dynamic Triad (Dyn Triad) [ZYR+18], DynGEM [GKHL18] and DANE [LDH+17]. Dyn Triad models how a closed triad, which consists of three vertices connected with each other, develops from an open triad that has two of three vertices not connected with each other. DynGEM utilizes autoencoders for dynamic graph embedding. DANE combines structural information and attributes for a consensus embedding and then leverages matrix perturbation theory to learn the dynamic embeddings.

For our method *DNGE*, we test both dynamics modeling strategies, i.e., $DNGE_{Mean}$ and $DNGE_{Dist}$. For parameter settings, the latent dimension is 100 for all embedding methods. For DeepWalk and *node2vec*, the number of walks is 10, walk length is 80 and window size is 10. For *node2vec*, $p = 1$ and $q = 0.5$. For Dynamic Triad and DANE, we use the default settings.

Results. The average NMI and Purity for community detection on the Epinions network are shown in Table 3.3. From these results, some conclusions can be drawn:

- *DNGE* with smoothness in distributions outperforms other baselines on Purity and performs the second best on NMI. The results show that *DNGE*

Table 3.3: Clustering performance on Epinions network.

Method		Purity	NMI
adjacency-based methods	Spectral	0.1161	0.1019
	GN	0.1258	0.1067
static embedding methods	DeepWalk	0.1842	0.1456
	LINE	0.1322	0.1239
	node2vec	0.1713	0.1360
	Gauss Emb	0.1865	0.1503
dynamic embedding methods	Dyn Triad	0.1840	**0.1629**
	DynGEM	0.1594	0.1441
	DANE	0.1425	0.1216
DNGE	$DNGE_{Mean}$	0.1873	0.1526
	$DNGE_{Dist}$	**0.1942**	0.1615

can effectively preserve network structures. Smoothness in distributions is better than that in mean in community detection. This is because smoothness in distributions can learn more consistent representations.

- Network embedding is an effective technique in capturing network structures because the performance of all embedding methods are of high quality in community detection and they all perform better than traditional clustering or community detection methods, e.g., Spectral and GN algorithm.

3.6.2 Link Prediction

To evaluate the effectiveness of *DNGE* in modeling dynamic information, we conduct link prediction experiment to test our method, following [GL16,WCZ16]. Since our aim is to validate dynamic information modeling, we learn node embeddings on the first $T-1$ network snapshots and predict the links on the last snapshot. Similar to [GL16], we regard link prediction as a classification task. We randomly sample an equal number of node pairs from each snapshot which have no edge connecting them as the negative examples. Note that embeddings for the last snapshot are learned on the graph after removing these random samples. Without loss of generality, learned node embeddings are used as the features and logistic regression is used as the classifier. We use AUC as the evaluation metric to compare the results.

Table 3.4: Traditional link prediction methods and definitions where $N(u)$ and $N(v)$ denote the neighbor sets of node u and v respectively.

Method	Definition
Jaccard Coefficient (JC)	$\frac{\lvert N(u)\cap N(v)\rvert}{\lvert N(u)\cup N(v)\rvert}$
Adamic-Adar (AA)	$\sum_{t\in N(u)\cap N(v)} \frac{1}{\log\lvert N(t)\rvert}$
Preferential Attachment (PA)	$\lvert N(u)\rvert\cdot\lvert N(v)\rvert$

Baselines. We compare four different types of methods for link prediction experiments: traditional link prediction methods, point embedding methods, dynamic point embedding methods and Gaussian embedding methods.

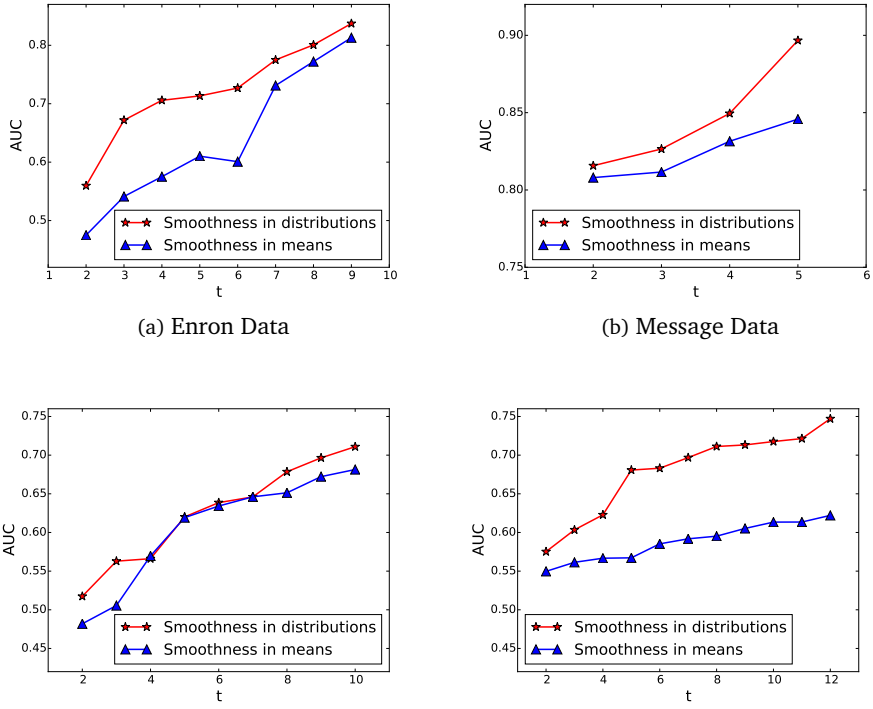
- **Traditional link prediction methods**: Traditional methods use heuristic scores to predict links given a pair of nodes. Following [GL16], we choose three widely used methods and their definitions are shown in Table 3.4.
- **Point embedding methods**: Point embedding methods map nodes to deterministic vectors, and state-of-the-art approaches are utilized as baselines, i.e., DeepWalk [PARS14], LINE [TQW+15] and *node2vec* [GL16].
- **Dynamic point embedding methods**: For dynamic point embedding methods, Dynamic Triad (Dyn Triad) [ZYR+18], DynGEM [GKHL18] and DANE [LDH+17] are compared same to community detection experiment.
- **Gaussian embedding methods**: For Gaussian embedding methods, we compare Gauss Emb and our proposed *DNGE* using two dynamics modeling strategies, i.e., $DNGE_{Mean}$ and $DNGE_{Dist}$.

For all embedding methods, the latent dimension of Enron, Message, Reality and Facebook are set to be 32, 64, 100 and 100, respectively. Other settings are same to Section 3.6.1.

Results. The AUC scores of different methods on the real-world networks are shown in Table 3.5. Note that the reported values are average AUC by conducting the experiment five times to avoid randomness. Note that for traditional link prediction methods, they are designed for static networks, so we report the performance of two strategies using these methods: aggregate and previous strategies. In aggregate strategy, we use data from all past snapshots to predict links in snapshot T while in previous strategy, we only use snapshot $T-1$ to predict links. From these results, we can draw the following conclusions:

- *DNGE* outperforms other baselines on all the networks except Reality. This is because Reality is extremely sparse compared to other networks (in Table 3.2). The strategy of smoothness in distributions performs better than smoothness in means because it models both embeddings and uncertainties as the dynamic regularization. This result is same to that in community detection.

- In general, Gaussian embedding methods achieve better performance than point embedding methods. It demonstrates that Gaussian embedding is a better method for representation learning in network data. In fact, Gaussian embedding can provide better interpretations of embeddings from



(a) Enron Data (b) Message Data

(c) Reality Data (d) Facebook Data

Figure 3.2: Link prediction vs. time t on different data sets.

the perspective of probability and Bayesian statistics.

- All network embedding methods beat traditional heuristic link prediction methods. It indicates the effectiveness of embedding methods in representation learning as well as in downstream network analysis applications such as link prediction.

We further study the link prediction performance over time t and the results on different datasets are shown in Figure 3.2. It can be observed that with more historical information being modeled, *DNGE* can perform better in link prediction. This trend can be observed in all datasets and it demonstrates that *DNGE* can effectively capture the dynamic information so that it leads to better link prediction performance with more historical information being modeled. We can also observe that the original Gaussian embedding method is more sensitive to noise. It may result from the traditional random walk based sampling would be influenced more by noisy edges comparing to *node2vec*.

3.6.3 Uncertainty Modeling

Mapping a node in a network into a distribution rather than a point vector allows us to model the uncertainty of the learned representation which is another advantage of *DNGE*. It is intuitive that the more noisy edges a node has, the less discriminative information it contains, thus making its embedding more uncertain.

To study the uncertainty modeling, we add random edges to Enron network to introduce uncertainties. To introduce different levels of uncertainties, we increase the number of random edges in the networks from 100 to 500. Since we select diagonal covariance in our experiments, we compute the traces of covariance matrices to denote the uncertainties under different conditions. The result is shown in Figure 5.8 where it can be observed that with more random edges being added to the network, we have larger traces of convariance matrices. This demonstrates that our proposed *DNGE* can capture the uncertainties of learned node representations. It also indicates that smoothness on distributions achieves smaller trace because it learns more consistent representations.

We further synthesize a dynamic network consisting of 400 nodes in 4 snapshots and these nodes belong to 4 communities. To introduce noise, we add different numbers of random edges to each snapshots from 1000 to 5000. The results of different embedding methods are shown in Table 3.6. Note that we select some representative methods from all baselines. From these results, it can

be observed that our *DNGE* can effectively learn embeddings in noisy networks. *DNGE* with distribution smoothness is better than mean smoothness.

3.6.4 Parameter Sensitivity

We consider four major parameters in *DNGE*, i.e., *latent dimensions*, *dynamic regularization parameter* λ, constraint c_{max} on covariance Σ and constraint C on mean μ. In order to evaluate how changes to these parameters affect performance, we conducted the same link prediction experiment on the Enron network introduced in Section 3.6.2 with different parameter values.

In the interest of brevity, we vary one target parameter by fixing other parameters to study the parameter sensitivity. Specifically, the number of latent dimensions varies from 2^3 to 2^7, the dynamic regularization parameter λ varies from 0.1 to 0.5, and constraint c_{max} and C vary from 1 to 5 respectively. The results of parameter sensitivity are shown in Figure 3.4. From these results, it can be observed that:

- As shown in Figure 5.3a, the latent dimension should be not too large or too small. Larger embedding dimension will degrade the performance and it may result from that larger dimension could contain noisy and redundant information. A better strategy to select optimal latent dimension is to use validation data.
- For dynamic regularization parameter λ, 0.3 and 0.4 perform best on link prediction task. With larger λ, the performance will decrease due to more

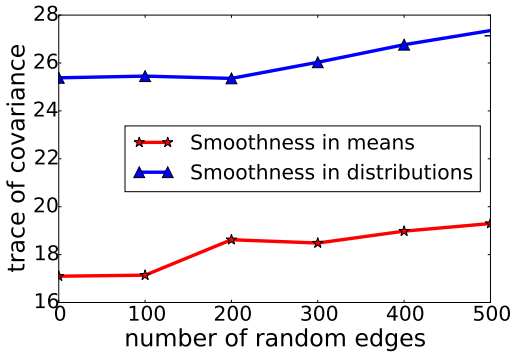
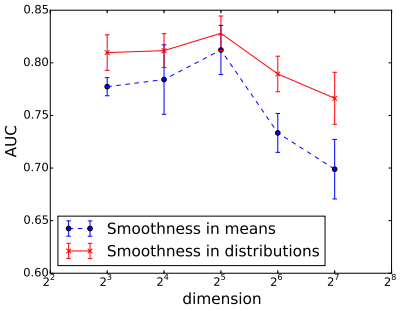


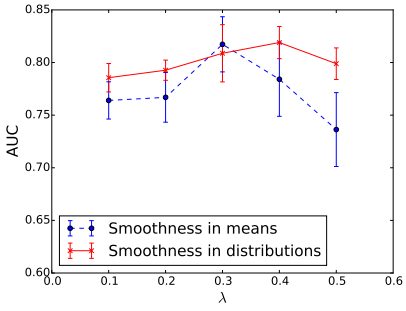
Figure 3.3: Traces of covariance matrices on Enron network.

weights are put on previous snapshots. Therefore, in general, medium size of latent dimensions and relatively small dynamic regularization parameter can achieve better performance.

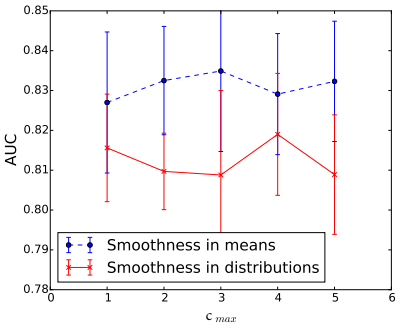
- From Figure 5.3c and 5.3d, we can observe that both constraints on mean and covariance have little impact on the performance. It may result from that these constraints work on all values so the learned representations and covariances will be scaled up and down simultaneously and overall performance will not be influenced.



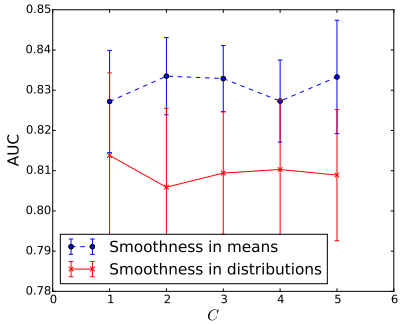
(a) AUC vs. dimension



(b) AUC vs. λ



(c) AUC vs. covar constraint c_{max}



(d) AUC vs. mean constraint C

Figure 3.4: Parameter sensitivity results on Enron.

3.7 Concluding Remarks

Back to the research question in Section 3.1: *How can we effectively detect local communities by capturing dynamics and uncertainties on dynamic graphs?*, to answer this question, we proposed *DNGE*, a novel dynamic network embedding framework using Gaussian embedding, *DNGE*, to tackle the two major challenges exist in previous network embedding studies: *dynamics modeling* and *uncertainty modeling*. *DNGE* learns node representations by explicitly modeling temporal information as regularization using two different smoothness strategies. Furthermore, *DNGE* utilizes Gaussian embedding to represent each node as a Gaussian distribution where the mean indicates the position of this node in the embedding space and the covariance represents its uncertainty.

Our experimental study demonstrated that *DNGE* effectively preserves community structures and captures dynamic information, achieves comparable results to state-of-the-art methods in link prediction and provides more information on uncertainties of node representations.

On the basis of DNGE, several new research lines can be pursued. For example, it is interesting to learn representations of dynamic networks where nodes can be added or deleted over time, or nodes may have attributes. We leave these extensions for future work.

Table 3.5: Link prediction results of different methods. Note that for traditional methods, we employ two strategies: aggregate and previous (format in *aggregate/previous* for the first three rows). To predict links in snapshot T, aggregate strategy combines all past $T-1$ snapshots and previous strategy uses only the snapshot $T-1$.

Data set		Enron	Messages	Reality	Facebook
Methods		AUC			
traditional link prediction	JC	0.6044/0.5960	0.5420/0.5424	0.5018/0.4864	0.4313/0.4209
	AA	0.6055/0.5957	0.5420/0.5420	0.5007/0.5014	0.4513/0.4221
	PA	0.4852/0.4921	0.6074/0.5877	0.4941/0.4972	0.4143/0.4023
static point embedding	DeepWalk	0.7564	0.6007	0.5874	0.5457
	LINE	0.7535	0.5213	0.7018	0.5564
	node2vec	0.7819	0.6687	0.5910	0.5384
dynamic point embedding	Dyn Triad	0.8355	0.8504	**0.7204**	0.7255
	DynGEM	0.8129	0.8297	0.6776	0.7004
	DANE	0.8023	0.7912	0.6743	0.7122
Gaussian embedding	Gauss Emb	0.7883	0.8116	0.6317	0.6213
	$DNGE_{Mean}$	0.8130	0.8458	0.6814	0.6221
	$DNGE_{Dist}$	**0.8373**	**0.8967**	0.7109	**0.7422**

Table 3.6: Clustering performance on Epinions network with noisy edges.

Methods	Number of noisy edges					
	0	1000	2000	3000	4000	5000
node2vec	0.9090	0.8693	0.8612	0.8441	0.8412	0.8223
Gauss Emb	0.8825	0.8254	0.8104	0.8002	0.8124	0.7991
Dyn Triad	0.9472	0.8743	0.8722	0.8589	0.8603	0.8502
$DNGE_{Mean}$	0.9287	0.8655	0.8743	0.8552	0.8522	0.8475
$DNGE_{Dist}$	0.9533	0.9101	0.9032	0.8699	0.8653	0.8701

Chapter 4

Node Classification on Dynamic Graphs

4.1 Introduction

In this chapter, we aim to answer the research question Q1.2 introduced in Section 1:

> Q1.2: How can we make good use of local structures and temporal information to improve the performance of node classification on dynamic graphs?

Assigning labels to unlabeled nodes in the graph is the node classification problem. In fact, there are different ways to define labels on graphs such as local labels based on homophily [MSLC01] and global labels based on positions and roles [RA+15]. In this chapter, we only focus on the local labels, i.e., node classification based on the local structures of graphs.

The increasing number of the network applications and the complicated relationships between graph nodes have made the labels of the graph data expensive and/or difficult to obtain. Therefore, the problem of node classification in networks has attracted extensive attention recently. Different from traditional classification tasks, the independent and identically distributed (*i.i.d.*) assumption does not hold for node classification in networks and methods should take the structure dependency into account. There have been a number of studies

on node classification in networks in recent years [AL11,XYWL13,ZWY+13] and these methods can be categorized into two types [BCM11]: (1) methods based on iterative application of traditional classifiers using structural properties as features and (2) methods propagate the labels via random walks. However, there are two major limitations in existing studies. First, most of these studies focus on static networks. In fact, many real-world networks are dynamic and nodes/edges in the networks may change during time. For instance, a user in the social network may add more friends and an author in the bibliographic network may collaborate with new authors. In such dynamic scenario, temporal information can also play an important role in classifying nodes. Second, it is difficult to extract features from network structures. Zhao et al. [ZWY+13] selected five types of features, i.e., homophily, triadic closure, reach, embeddedness and structural holes, for node classification. [CRPS14] calculated several predefined network properties as the feature representation including the normalized node degree, the clustering coefficient, the common neighbors, etc. However, these methods for feature extraction require data engineering and external knowledge and may also be network and task specific.

Aiming to breakthrough these limitations, in this section we propose the *dynamic Factor Graph Model* (dFGM) for node classification in dynamic social networks. In detail, the dynamic graph data is organized in the format of a series of graph snapshots and to model the graph snapshots, three types of factors, i.e., node factor, correlation factor and dynamic factor, are designed in the *dFGM* based on node features, node correlations and temporal correlations, respectively. Node factor and correlation factor can capture the global and local properties of the graph structures while the dynamic factor can make use of the temporal information. To address the problem of feature extraction in graphs, we utilize an unsupervised feature extraction method, i.e., DeepWalk [PARS14], to extract features from the networks and in this feature extraction process, no complicated feature engineering or external knowledge is required. To validate the effectiveness of the *dFGM*, a real-world data set, i.e., DBLP, is used for the experiments.

The main contributions in this chapter can be summarized as follows:

- We propose the *dynamic Factor Graph Model* (dFGM) for node classification in dynamic social networks and this model can capture node attributes, correlations and temporal information.

- We evaluate the proposed *dFGM* on a real-world data set and the experimental results demonstrate the effectiveness of our model compared with

other methods on two evaluation metrics, i.e., *accuracy* and *error* in probability.

- We compare different feature extraction methods for the problem of node classification in networks.

The rest of this chapter is organized as follows. Section 4.2 introduces the related work and Section 4.3 formally defines the problem. Section 4 briefly presents the feature extraction method. Section 4.5 explains the proposed *dynamic Factor Graph Model* for node classification. And then in Section 4.6 we discuss the experiments and analysis. Finally, in Section 4.7 we draw the conclusions and outline future work.

4.2 Related Work

The problem of node classification in graphs has attracted extensive attention recently. Nodes in social networks can be associated with labels and these labels come in many forms, e.g., demographic labels, labels which reflect political or religious beliefs; labels that represent interests, hobbies, and affiliations. Labeling nodes in social networks is beneficial for many practical applications such as expert search, recommendation systems, advertising systems.

Based on the specific problem, node classification can be defined in different ways, for instance, role classification [ZWY+13], community classification [JSD+10], and function classification [LGG+15]. Due to its importance, a variety of methods have been proposed for this task. Most of previous papers focus only on the static networks. Methods proposed in [XYWL13] and [ZWY+13] are similar to our method in this section and they are also based on the Factor Graph Model. Both models capture the node attributes and social relations. [XYWL13] extracts features using topic model [Hof99] while [ZWY+13] extracts features from the perspective of sociology. However, these methods on static networks cannot be extended to the dynamic scenario easily.

There are some methods have been proposed methods to classify nodes in dynamic networks [LGD+13, YH14] . Model in [LGD+13] can learn the latent feature representation and capture the dynamic patterns. However, this method requires data from all the historical snapshots to classify nodes in next snapshot while in practice some labels in previous data may be missing or incorrect. [YH14] uses SVM to classify nodes in each snapshot and combines the support vector from last snapshot and current training data for classification. However, this operation depends heavily on the performance of SVM and only

using support vector from previous snapshot may also loss useful dynamic information.

Most existing studies on node classification in networks exploit feature representation using specific social semantics or user-specific attributes. In [XYWL13], user generated content, i.e., the words in paper titles and the messages published by users, has been used for feature representation. Tang et al., [TZT11] extracted features based on specific social relations, such as co-advisor and co-advisee. There is a limitation in these studies since (1) predefined features based on specific relations require external knowledge about the network and it would be difficult to satisfy in practice; (2) in an arbitrary network, it may be difficult to obtain user-specific information, e.g., users may not make their profiles public. There are also some papers extract features from the topological structures of the networks. In [HGL+11], the features consists of local and egonet properties based on counts of links and the egonet-based properties generated in a recursive fashion. This method requires complicated data mining or machine learning techniques and heavy feature engineering work. In [ZWY+13], five types of network properties have been used as the features, i.e., homophily, triadic closure, reach, embeddedness and structural holes. Similarly, [CRPS14] calculates the normalized node degree and average degree, the clustering coefficient, the locality index, etc. as the node features. These feature extraction strategies require external knowledge about graph theory or sociology. Therefore, they are difficult to generalize in different node classification tasks.

4.3 Problem Definition

Based on the basic definitions introduced in Section 2, several necessary definitions are introduced and then the formal definition of the node classification problem is presented.

Definition 4.1. Partially labeled graph. Given a fixed finite non-empty label set R, a partially labeled graph is $G_L = \{V_L, V_U, E, X, r\}$ where

- V_L is a set of labeled nodes and V_U is a set of unlabeled nodes with $V_L \cup V_U = V$ and $V_L \cap V_U = \emptyset$;
- E is the set of edges, i.e., $E \subseteq V \times V$;
- X is an attribute matrix associated with nodes in V where each row corresponds to a node v, each column an attribute, and an element x_{ij} denotes the value of the j^{th} attribute of node v_i.

- r is a mapping function which maps each labeled node to a label, i.e., $r : V_L \rightarrow R$.

In this section, we assume edges to be undirected for simplicity. We also assume that the nodes are fixed and the edges may change over time. In our settings, the nodes are partially labeled in different snapshots and this partially labeled dynamic graph is defined the input of our problem. More precisely,

Definition 4.2. Partially labeled dynamic graph. Let V be a finite set of nodes, a partially labeled dynamic graph $\mathcal{G} = \{G^t | t = 1, ..., T\}$ consists of a series of graph snapshots $G^t = \{V_L^t, V_U^t, E^t, X^t, r^t\}$, where each snapshot G^t is a partially labeled graph as defined in Definition 4.1, and $V_L \cup V_U = V$.

Based on the definitions introduced above, we can define the problem of node classification in dynamic graphs. Given a partially labeled dynamic graph $\mathcal{G}$, our goal is to infer the labels of all the unlabeled nodes in the graph. Formally,

Problem 4.1. Node classification in dynamic graphs. Given a partially labeled dynamic graph $\mathcal{G} = \{G^t | t = 1, ..., T\}$, the goal of the node classification in dynamic graphs is to learn a predictive function $f : \mathcal{G} \rightarrow R$ such that $\forall G = \{V_L, V_U, E, X, r\} \in \mathcal{G}$, we have

$$\begin{cases} f(v|E, X) = r(v|E, X), & \forall v \in V_L \\ f(v|E, X) \in R \text{ is the correct label assignment}, & \forall v \notin V_L \end{cases}$$

4.4 Feature Extraction

Since we only consider the network structures in this study, the features should be extracted only from the topology. Different methods have been proposed in previous studies for feature extraction from network and these methods can be categorized into two types. One type is to use data mining or machine learning techniques to mine features from graphs. For instance, [HGL+11] extracts local and egonet features based on counts of links and also aggregates egonet-based features in a recursive fashion to generate new features. In [PZZY13], a minimum redundancy subgraph feature selection algorithm which is based on depth first search (DFS) code, is proposed for feature selection. Although there is no explicit feature extraction, the kernel methods have been used in similarity measure in [YH14, YH15].

Another type of feature extraction is based on the knowledge of graph theory or sociology. For example, [ZWY+13] extracts features from the perspective of sociology, and five types of features are selected, i.e., homophily, triadic closure, reach, embeddedness and structural holes. [CRPS14] calculates several predefined network properties as the node feature representations including the normalized node degree, the clustering coefficient, the common neighbors, etc. However, these methods either require heavy feature engineering work or require external knowledge which make them difficult to generalize. To tackle this problem, in this section, we apply an unsupervised method to extract the features in which no complicated feature engineering or external knowledge is required.

In this section we explore an unsupervised method, namely DeepWalk [PARS14], to extract the features from networks for node classification in dynamic social networks. Using deep learning techniques, DeepWalk learns latent representations of nodes in a continuous vector space. Similar to the *Word2vec* [MCCD13] which represents features of current term using the context information within a slide window of n-grams, the feature representation of each nodes can be learned using the context information from these random walk paths in DeepWalk. In detail, firstly, the truncated random walk paths are obtained from the graph and they are treated as sentences in NLP. Then these paths are used as the input of a neural network and the output of this neural network will be the features of these nodes.

4.5 Method

Probabilistic graphical models (PGM) [KF09] have been widely used in modeling dependency between entities that can be organized in a graph structure. Thus, it is intuitive to use PGM for graph mining. Factor graph models (FGM), as one type of PGM, are flexible in integrating with task-specific functions and previous studies have indicated that FGM can perform well in a variety of graph modeling tasks including social influence analysis [TWS13], action tracking [TTS+10], social tie inferring [TZT11], community analysis [YTLY10]. In this section, we present the proposed *dynamic Factor Graph Model* (dFGM), which is an extension of the traditional factor graph models, for node classification in dynamic social networks.

4.5.1 Intuitions

Intuitively the class/label of a node in a dynamic social network will be determined by three factors including

- Node attributes, i.e., the node's global and local characteristics at current time step. This factor corresponds to the node features extracted from the network, e.g., the number of outlinks of a node, or nodes' intrinsic characteristics, e.g., the profile of a user. For instance, a Twitter account who has lots of followers may be celebrity and an author published papers using words such as Bayesian, likelihood, and perplexity, may be a machine learning researcher.
- Node correlations, i.e., the relationships or interactions between nodes in a snapshot. This factor corresponds to the social relations of nodes in the networks. For example, in Twitter, a user followed Lionel Messi and Cristiano Ronaldo may be labeled as a soccer fan and an author collaborate with Jiawei Han and Christos Faloutsos has a higher probability of belonging to the data mining community.
- Previous labels. We assume that the label of a node will not change abruptly. This factor can capture the dynamic information in the network. A researcher published papers in KDD may continue to publish papers in data mining area.

4.5.2 Dynamic Factor Graph Model

Based on these intuitions introduced above, we propose the *dFGM* which consists of three factors, named node factor, correlation factor and dynamic factor, and they correspond to the node attributes, node correlations and previous labels, respective. In detail, these three factors are defined as follows.

- Node factor $f(r_i, \boldsymbol{x}_i)$. This factor represents the posterior probability of the label r_i give the feature $\mathbf{x}_i$ of node v_i.
- Correlation factor $c(r_i, N(r_i))$. This factor reflects the correlation between nodes, where $N(r_i)$ is the set of correlated labels to r_i. There are multiple ways to define the set of correlated labels and in this study, $N(r_i)$ denotes the labels of neighbors of node v_i.
- Dynamic factor $d(r_i^t, r_i^{t-1})$. This factor denotes the correlation between the labels of one node in two consecutive snapshots.

An example of the *dFGM* with three factors is shown in Fig 4.1. In this example, there are two labels: 1 and 2. For the labeled nodes, the probability of the ground label is 1.0 and the probability of another label is 0.0. For the unlabeled nodes, the probability values will be real numbers, e.g., P_1 and P_2 for node r_1^{t1} in snapshot t_1.

Based on all the factors introduced above, the joint distribution of labels R given the graph $\mathcal{G}$ can be defined as

$$P(R|\mathcal{G}) = \prod_t \prod_i f(r_i, \boldsymbol{x}_i) c(r_i, N(r_i)) d(r_i^t, r_i^{t-1}) \tag{4.1}$$

These factors can be instantiated in different ways, such as the exponential-linear function [TZT11], the discrete function [ZWY+13], the quadratic function [TTS+10], etc. In this section, to simplify the model learning, we choose the exponential-linear function to instantiate the factors. In detail, the node factor is defined as

$$f(r_i, \boldsymbol{x}_i) = \frac{1}{Z_1} \exp\{\boldsymbol{\alpha}^T \boldsymbol{\phi}(r_i, \boldsymbol{x}_i)\} \tag{4.2}$$

where Z_1 is the normalizing factor, $\boldsymbol{\alpha}$ is the weighting vector, and $\boldsymbol{\phi}$ is a vector of feature function. Similarly, the edge factor is defined as

$$c(r_i, N(r_i)) = \frac{1}{Z_2} \exp\{\sum_{r_j \in N(r_i)} \boldsymbol{\beta}^T \mathbb{I}_{corr}(r_i, r_j)\} \tag{4.3}$$

where Z_2 is the normalizing factor, $\boldsymbol{\beta}$ is the weighting vector, and $\mathbb{I}_{corr}$ is the indicator function for node correlations and defined as

$$\mathbb{I}_{corr}(r_i, r_j) = \begin{cases} 0, & \text{if } e_{ij} \notin E \\ 1, & \text{if } e_{ij} \in E \end{cases} \tag{4.4}$$

Then temporal factor is defined in the same way

$$d(r_i^t, r_i^{t-1}) = \frac{1}{Z_3} \exp\{\gamma^T \mathbb{I}_{dyn}(r_i^t, r_i^{t-1})\} \tag{4.5}$$

where Z_2 is the normalizing factor and γ is the weighting vector. $\mathbb{I}_{dyn}(r_i^t, r_i^{t-1})$ is the indicator function for the dynamic information and defined as

$$\mathbb{I}_{dyn}(r_i^t, r_i^{t-1}) = \begin{cases} 0, & \text{if } r_i^t \neq r_i^{t-1} \\ 1, & \text{if } r_i^t = r_i^{t-1} \end{cases} \tag{4.6}$$

4.5.3 Model Learning

For simplicity, we write the joint probability defined in Eq (1) as

$$P(R|\mathcal{G}) = \frac{1}{Z}\prod_t\prod_i \exp\{\theta^T s_i^t\} = \frac{1}{Z}\exp\{\theta^T \sum_t\sum_i s_i^t\} = \frac{1}{Z}\exp\{\theta^T \mathbf{S}\} \tag{4.7}$$

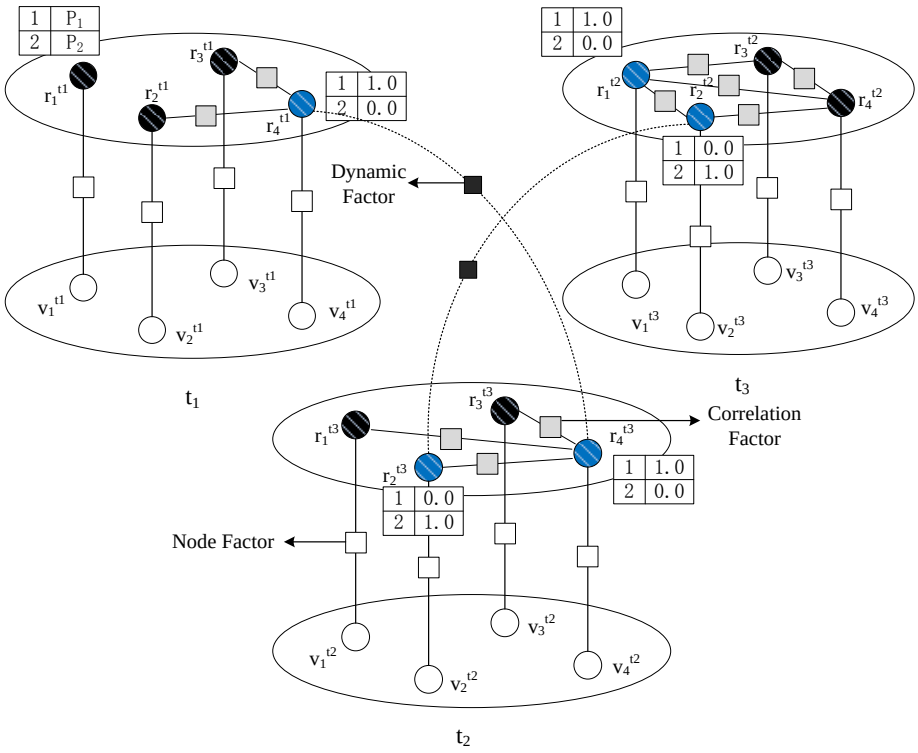


Figure 4.1: A four-node example in three snapshots of the *dFGM*. The white circles in the lower layers denote the nodes, the colored circles in the upper layers denote the labels and the squares are the factors. For these colored circles in the upper layer, the blue circles mean that these nodes are labeled and the black ones are unlabeled. The squares denote the factors. As shown in the figure, the white square denotes the node factor, the grey square denotes the correlation factor and the black square denotes the dynamic factor.

where $Z = Z_1 Z_2 Z_3$ is the normalizing factor, θ is the parameter configuration, i.e., $\theta = (\alpha, \beta, \gamma)$ and $\mathbf{S}$ is the concatenation of the factor functions, i.e., $\mathbf{S} = (\phi(r_i, \boldsymbol{x}_i)^T, \mathbb{I}_{corr}(r_i, r_j)^T, \mathbb{I}_{dyn}(r_i^t, r_i^{t-1})^T)^T$. Thus, model learning is to estimate the parameter configuration θ. However, it is intractable to calculate the normalizing factor Z, since it is the summation of the likelihood of the possible states for all the nodes but some nodes in the *partially labeled dynamic network* (defined in **Definition 2.**) are unlabeled. To solve this problem, we use the labeled data to infer the unknown labels. In specific, we use $R|R^L$ to denote the predicted labels inferred from the known labels and define the log-likelihood objective function $\mathcal{O}(\theta)$ as

$$\begin{aligned} \mathcal{O}(\theta) &= \log p(R^L|\mathcal{G}) = \log \sum_{R|R^L} \frac{1}{Z} \exp\{\theta^T \mathbf{S}\} \\ &= \log \sum_{R|R^L} \exp\{\theta^T \mathbf{S}\} - \log Z \\ &= \log \sum_{R|R^L} \exp\{\theta^T \mathbf{S}\} - \log \sum_{R} \exp\{\theta^T \mathbf{S}\} \end{aligned} \tag{4.8}$$

To solve this optimal problem without constraints, the gradient descent method is the simplest choice. In order to use the gradient descent method, the gradient of $\mathcal{O}(\theta)$ for the parameter θ is calculated

$$\begin{aligned} \frac{\partial \mathcal{O}(\theta)}{\partial \theta} &= \frac{\partial (\log \sum_{R|R^L} \exp\{\theta^T \mathbf{S}\} - \log \sum_R \exp\{\theta^T \mathbf{S}\})}{\partial \theta} \\ &= \frac{\sum_{R|R^L} \exp\{\theta^T \mathbf{S}\} \cdot \mathbf{S}}{\sum_{R|R^L} \exp\{\theta^T \mathbf{S}\}} - \frac{\sum_R \exp\{\theta^T \mathbf{S}\} \cdot \mathbf{S}}{\sum_R \exp\{\theta^T \mathbf{S}\}} \\ &= \mathbb{E}_{p_\theta(R|R^L, \mathcal{G})} \mathbf{S} - \mathbb{E}_{p_\theta(R, \mathcal{G})} \mathbf{S} \end{aligned} \tag{4.9}$$

Then the parameters θ can be updated using gradient descent method:

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \tag{4.10}$$

where η is the learning rate and $\nabla_\theta = \frac{\partial \mathcal{O}(\theta)}{\partial \theta}$ is the gradient.

Another challenge here is that the graphical structure in *dFGM* can be arbitrary and may contain cycles, which makes it intractable to directly calculate the expectation. Different approximate algorithms can be applied for learning *dFGM*, e.g., Loopy Belief Propagation (LBP) [MWJ99] and Mean-field [WJ08]. We use LBP for the model learning in this section similar to [TZT11, XYWL13]. The learning algorithm is summarized in Algorithm 1.

Algorithm 2 Model learning for dFGM

Require: learning rate η
Ensure: learned parameters
  while not converge **do**
    Calculate $\mathbb{E}_{p_\theta(R|R^L,\mathcal{G})}\mathbf{S}$ using LBP
    Calculate $\mathbb{E}_{p_\theta(R,\mathcal{G})}\mathbf{S}$ using LBP
    Calculate the gradient ∇_θ of θ according to Eq. (9)
    Update parameters θ with the learning rate η according to Eq. (10)
  end while

4.6 Experiments

In this section, we present our experiments on a real-world data set. To validate the performance of the *dFGM*, two evaluation metrics are applied, i.e., *accuracy* and *error* in probability. We also exploit the influence of parameters in this model including the feature dimension and the size of training data. To exploit the performance of different feature extraction methods, comparison between DeepWalk and a widely used recursive graph feature extraction method, i.e., ReFeX [HGL+11], is made.

4.6.1 Data Sets

We conduct the experiments to validate the performance of *dFGM* on the DBLP[1] data set. We extract a subset of the DBLP dataset for the experiments. Conferences from six research communities, i.e., artificial intelligence and machine learning (**AI & ML**), algorithm and theory (**AL & TH**), database (**DB**), data mining (**DM**), computer vision (**CV**), information retrieval (**IR**) have been extracted. In specific, we extract the co-author relations in these conferences from 2001 to 2010 and data in each year is organized in a graph snapshot. The details of conferences for each community is shown in Table 8.2 and a brief statistics of this data including numbers of authors and relations is shown in Table 4.2.

4.6.2 Experimental Settings

In the DBLP data set, each author represents a node in the network and if two authors collaborated on a paper, there will be an edge between these two nodes.

[1] `http://dblp.uni-trier.de/xml/`

Table 4.1: Communities and conferences in DBLP data set.

Community	Conferences
AI & ML	IJCAI, AAAI, ICML, UAI, AISTATS
AL & TH	FOCS, STOC, SODA, COLT
CV	CVPR, ICCV, ECCV, BMVC
DB	EDBT, ICDE, PODS, VLDB
DM	KDD, SDM. ICDM, PAKDD
IR	SIGIR, ECIR

Table 4.2: Statistics of DBLP data set.

Year	number of authors	number of relations
2001	3074	5743
2002	2557	5343
2003	3836	7700
2004	3464	7132
2005	5189	11171
2006	4494	9392
2007	7294	15708
2008	5780	12398
2009	6405	14321
2010	5757	12738

To compare our proposed *dFGM* with existing methods, three types of baseline methods have been used:

- Feature-based classification
  We use the Logistic Regression (LR) and Support Vector Machine (SVM) as the baseline in the feature-based classification, and both methods use features extracted using the method described in Section 4. In the experiments, we use the implementation of LR and SVM in scikit-learn[2].

- Link-based classification
  Two methods have been employed in the link-based classification type. The first method is majority voting method with dynamic information (MV+dynamic). In detail, if a node is labeled in previous snapshot, the

[2] `http://scikit-learn.org/stable/`

predicted label is copied from previous one. Otherwise, the node is labeled by majority voting from neighbors in current snapshot. The second one is the collective classification (CC) [SNB+08] which combines features and relations using Iterative Classification Algorithm (ICA) [NJ00].

- Factor graph models (FGM)
  To validate the effectiveness of the temporal information, we also compare *dFGM* with FGM using only the features (FGM_feat) and FGM using both features and correlations (FGM_corr). The features used in these methods are same to the feature-based classification methods and the correlations used here are same to the link-based classification methods.

4.6.3 Evaluation Metrics

Two types of evaluation metrics have been used in the experiments, i.e., *accuracy* and *error* in probability. The *accuracy* is defined as

$$accuracy = \frac{n}{N} \tag{4.11}$$

where n is the number of instances correctly classified, and N is the total number of instances in the test data. It is worth noting that in the DBLP data set, one author may publish equal number of papers in multiple research communities and the output of prediction is only one community for one author. In this case, if the predicted community label belongs to the set of multiple communities, it will be counted as a correctly classified instance.

To better evaluate the performance, especially for the case that an author belongs to more than one communities, we also use another evaluation metric, namely the *error* in probability. This metric is beneficial in two aspects: (1) it can match the output of *dFGM* which is the probability of labels; (2) it can evaluate the prediction of multiple labels, e.g., the overlapping communities. The *error* in probability is defined as

$$error = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{c}|\hat{p}_i^j - p_i^j| \tag{4.12}$$

where N is the number of instances in the test data and c is the number of labels. $\hat{p}_i^j$ and p_i^j are the predicted probability and ground probability of label j for user i, respectively. The ground probability of label j for user i is the ratio of the number of papers published by user i in community j to the total number of papers published by user i.

Table 4.3: Comparison of node classification performance in DBLP data set.

Methods		*accuracy*	*error*
Feature-based classification	LR	0.3157±0.0031	———
	SVM	0.4983±0.0029	1.2651±0.0027
Link-based classification	MV+dynamic	0.5049+0.0011	0.9068+0.0009
	CC	0.7935+0.0033	0.8642±0.0101
Factor Graph Models	FGM_feat	0.2684±0.0024	1.4917±0.0003
	FGM_corr	0.8360±0.0328	0.7865±0.0062
	dFGM	**0.8410±0.0058**	**0.7360±0.0073**

4.6.4 Results

The performance of *dFGM* and different baseline methods are shown in Table 4.3 and in this section, we use 70% data as the training set and 30% as the test set. From the results, some conclusions can be drawn:

- the *dFGM* outperforms other methods in both evaluation metrics which shows the effectiveness of our proposed model and the importance of the dynamic information;
- since FGM is used to model correlations in graphs here, if the correlation information is removed, i.e., in FGM_Feat, the performance will be extremely poor even compared with transitional classification methods, e.g., LR and SVM;
- link-based classification methods (MV+dynamic and CC) perform better than feature-based methods (LR and SVM), and it demonstrates the importance of correlations in graph classification problem;
- methods using only node features perform poor compared with methods capturing both node features and correlations and it indicates that graph feature extraction is still a challenging task.

It is worth noting that the improvement on *accuracy* is very small. This is because in *accuracy* calculation, the predicted labels are the labels with maximum probability. For example, assume **CV** is the correct label and the probability of label **CV** is 0.8 predicated by model A and 0.95 predicted by model B, although model B performs better (it gives a more precise prediction), A and B have the same predicted label for *accuracy* metric.

Table 4.4: Comparison between ReFeX features and DeepWalk features.

Method		*accuracy*	*error*
ReFeX	FGM_Feat	0.2622	1.4832
	FGM_Corr	0.8210	0.8375
	dFGM	0.8239	0.7869
DeepWalk	FGM_Feat	0.2684	1.4917
	FGM_Corr	0.8360	0.7865
	dFGM	**0.8410**	**0.7360**

4.6.5 Influence of Parameters

In this experiment, we analyze the influence of parameters in the *dFGM*, i.e., the influence of feature dimension and the influence of size of training data.

Influence of Feature Dimension. To analyze the influence of the feature dimension for node classification, we conduct the experiment by setting the dimension of features generated by DeepWalk from 30 to 100 with interval 10 and the results are shown in Figure 4.2. From Figure 4.2, we can observe that

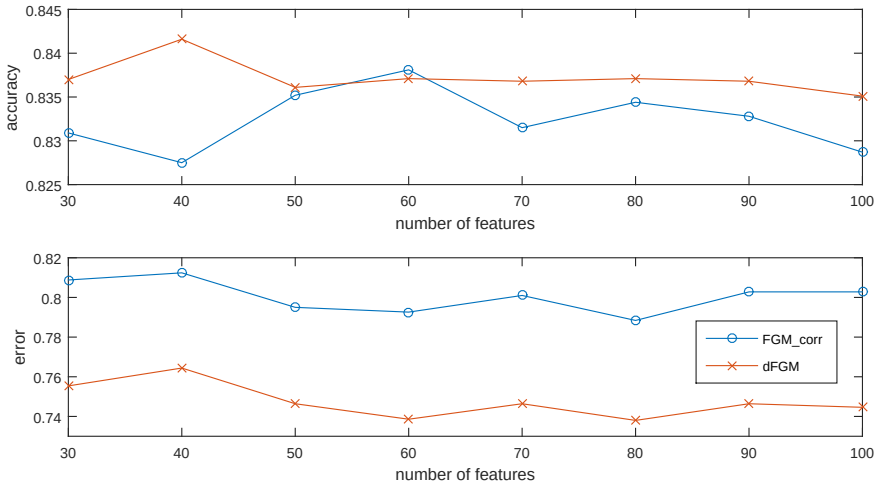


Figure 4.2: *Accuracy* and *error* vs. number of features

dFGM performs best on *accuracy* when the feature dimension is 40 and on *error* when the feature dimension is 80. While the optimal dimension for FGM_corr is 60 and 80 on *accuracy* and *error*, respectively. Considering both evaluation metrics, 80 would be relatively good choice for this data set.

Influence of Size of Training Data. We also analyze the influence of training data size. The size of training data is set from 10% to 90% and the results are shown in Fig 4.3 and Fig 4.4. Overall, better results can obtained when more training data is given. The results also demonstrate the robustness of the *dFGM* with different sizes of training data. Moreover, note that when the size of training data is relatively small (e.g., less than 50% in Fig 4.3 and less than 30% in Fig 4.4), the performance of *dFGM* is not good because the correlation and dynamic information will also be less given less training data which will influence the performance of *dFGM*.

4.6.6 Feature Comparison

We also compare the performance of different feature extraction methods. We compare features extracted by DeepWalk and ReFeX which is a widely used graph feature extraction method. The results are shown in Table 4.4. From the results, it can be observed that the effectiveness of the features extracted by DeepWalk and this result indicates the potential of unsupervised graph feature extraction method for node classification since it does not require heavy feature

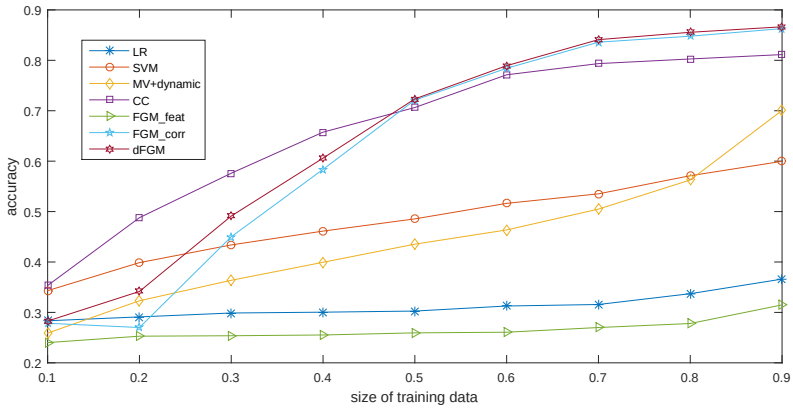


Figure 4.3: *Accuracy* vs. size of training data.

engineering or external knowledge about graph theory and sociology.

4.7 Conclusions

In this chapter, we aim to answer the research question *How can we make good use local structures and temporal information to improve the performance of node classification on dynamic graphs?*. We proposed a dynamic factor graph model, named *dFGM*, to classify nodes on dynamic graphs. To capture the temporal information, graph factors based on node attributes, node correlations and dynamic information are integrated in the *dFGM*. To breakthrough the limitation in graph feature extraction, we also utilized an unsupervised graph feature extraction method, i.e., DeepWalk, to extract features from the networks. The experiments have been conducted on a real-world data set and the experimental results demonstrate the effectiveness of the *dFGM*. We also analyze the influence of feature dimension and size of training data. Two different graph feature extraction methods also have been compared in the experiments.

As future work, we will study how to extract better features from networks. Since user generated content (UGC) is easy to access, taking the UGC information into consideration for node classification in the dynamic scenario will also be attractive. In addition, with rapid increase of network size, it will interesting to study more effective and efficient method for larger scale networks.

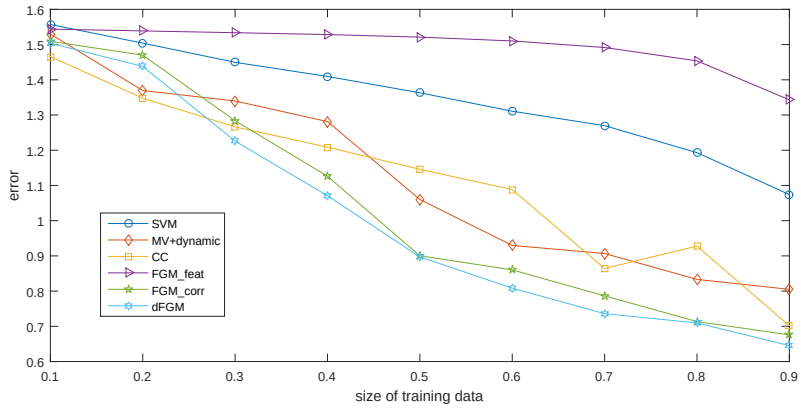
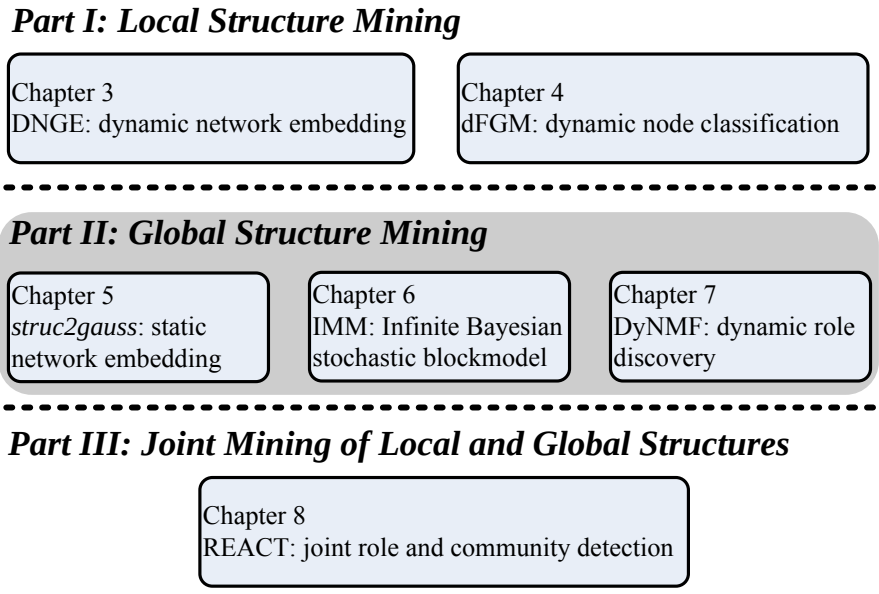


Figure 4.4: *Error* vs. size of training data.

Part II

Global Structure Mining



Chapter 5 We present *struc2gauss*: a flexible global structure preserving network embedding framework in static networks. struc2gauss learns node representations in the space of Gaussian distributions by modeling the global network structures. It is capable of preserving structural roles and modeling uncertainties.

Chapter 6 We present IMM (Infinite Motif Stochastic Blockmodel): a nonparametric Bayesian model to generate higher-order motif information in static networks. IMM is a high-order model which takes advantage of the motifs in the generative process and it is a nonparametric Bayesian model which can automatically infer the number of roles from the data.

Chapter 7 We present DyNMF (dynamic nonnegative matrix factorization): a unified model to discover role and role transition simultaneously in dynamic networks. DyNMF simultaneously obtains the role matrix of snapshot $t+1$ and the role transition from snapshot t to $t+1$ by using information in snapshot t+1 and the role matrix of snapshot t.

Chapter 5

Role Discovery on Static Graphs

5.1 Introduction

In this chapter, we aim to answer the research question Q2.1 introduced in Section 1:

> Q2.1: How can we effectively discover roles on static graphs by capturing the global structures and uncertainties?

As introduced in Chapter 3, network embedding fills the gap between traditional data mining and machine learning techniques and graph data by mapping nodes in a network into a low-dimensional space according to their structural information in the network. It has been reported that using embedded node representations can achieve promising performance on many network analysis tasks [PARS14, GL16, CLX15, RSF17]. Thus, similar to Chapter 3, we attempt to answer the research question Q2.1 by proposing a novel network embedding approach.

Previous network embedding techniques mainly relied on eigendecomposition [SJ09, TDSL00], but the high computational complexity of eigendecomposition makes it difficult to apply in real-world networks. With the fast development of neural network techniques, unsupervised embedding algorithms have been widely used in natural language processing (NLP) where words or phrases from the vocabulary are mapped to vectors in the learned embedding space, e.g., *word2vec* [MCCD13, MSC+13] and GloVe [PSM14]. By drawing

an analogy between paths consists of several nodes on networks and word sequences in text, DeepWalk [PARS14] learns node representations based on random walks using the same mechanism of *word2vec*. Afterwards, a sequence of studies have been conducted to improve DeepWalk either by extending the definition of neighborhood to higher-order proximity [CLX15, TQW+15, GL16, PKS16] or incorporating more information for node representations such as attributes [LDH+17, WATL17] and heterogeneity [CHT+15, TQM15].

Although a variety of network embedding methods have been proposed, two major limitations exist in previous studies: the lack of *global structure preservation* and the lack of *uncertainty modeling*. Previous methods focused only on one of these two limitations and neglected the other one. In particular, for *global structure preservation*, most studies applied random walk to learn representations. However, random walk based embedding strategies and their higher-order extensions can only capture local structural information, i.e., first-order and higher-order proximity within the neighborhood of the target node [LZZ17]. The local structural information is reflected in community structures of networks. But these methods may fail in capturing the global structural information, i.e., structural roles [RA+15, PZFP18]. The global structural information represents roles of nodes in networks and two nodes have the same role if they are structurally similar from a global perspective. In Figure 5.1, we use the same Borgatti-Everett graph [BE92] shown in Chapter 1 to illustrate global structural information (roles) and local structural informa-

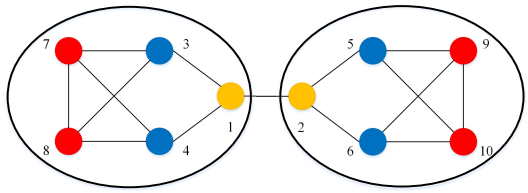


Figure 5.1: Borgatti-Everett graph of ten nodes belonging to (1) three groups (different colors indicate different groups) based on global structural information, i.e., the structural roles and (2) two groups (groups are shown by the ellipses) based on local structural information, i.e., the communities. For example, nodes 1, 3, 4, 7 and 8 belong to the same group Community 1 based on local structural perspective because they have more internal connections. Node 8 and 10 are far from each other, but they are in the same group based on global structural perspective.

tion (communities). In summary, nodes belong to the same community require dense local connections while nodes have the same role may have no common neighbors at all [TCW+18]. Empirical evidence based on this example for illustrating this limitation will be shown in Section 5.5.2. For *uncertainty modeling*, most of previous methods represented a node into a point vector in the learned embedding space. However, real-world networks may be noisy and imbalanced. For example, node degree distributions in real-world networks are often skewed where some low-degree nodes may contain less discriminative information [TCW+18]. Point vector representations learned by these methods are deterministic [DSPG16] and are not capable of modeling the uncertainties of node representations.

There are limited studies trying to address these limitations in the literature. For instance, *struc2vec* [RSF17] builds a hierarchy to measure similarity at different scales, and constructs a multilayer graph to encode the structural similarities. *SNS* [LZZ17] discovers graphlets as a pre-processing step to obtain the structural similar nodes. DRNE [TCW+18] learns network embedding by modeling regular equivalence [WF94]. However, these studies aim only to solve the problem of **role preservation** to some extent. Thus the limitation of *uncertainty modeling* remains a challenge. [DSPG16] and [BG17] put effort in improving classification tasks by embedding nodes into Gaussian distributions but both methods only capture the neighborhood information based on random walk techniques. DVNE [ZCWZ18] learns Gaussian embedding for nodes in the Wasserstein space as the latent representations to capture uncertainties of nodes, but they also focus only on first- and second-order proximity of networks. Therefore, the problem of *global structure preservation* has not been solved in these studies.

In this chapter, we propose *struc2gauss*, a new structural role preserving network embedding framework. *struc2gauss* learns node representations in the space of Gaussian distributions and performs network embedding based on global structural information so that it can address both limitations simultaneously. On the one hand, *struc2gauss* generates node context based on a globally structural similarity measure to learn node representations so that global structural information can be taken into consideration. On the other hand, *struc2gauss* learns node representations via Gaussian embedding and each node is represented as a Gaussian distribution where the mean indicates the position of this node in the embedding space and the covariance represents its uncertainty. Furthermore, we analyze and compare two different energy functions for Gaussian embedding to calculating the closeness of two embedded Gaussian distributions, i.e., expected likelihood and KL divergence. To investigate the

influence of structural information, we also compare *struc2gauss* to two other structural similarity measures for networks, i.e., MatchSim and SimRank.

We summarize the contributions of this chapter as follows:

- We propose a flexible structure preserving network embedding framework, *struc2gauss*, which learns node representations in the space of Gaussian distributions. *struc2gauss* is capable of preserving structural roles and modeling uncertainties.
- We investigate the influence of different energy functions in Gaussian embedding and compare to different structural similarity measures in preserving global structural information of networks.
- We conduct extensive experiments in node clustering and classification tasks which demonstrate the effectiveness of *struc2gauss* in capturing the global structural role information of networks and modeling the uncertainty of learned node representations.

The rest of the chatper is organized as follows. Section 5.2 provides an overview of the related work. We present the problem statement in Section 5.3. Section 5.4 explains the technical details of *struc2gauss*. In Section 5.5 we then discuss our experimental study. Finally, in Section 5.6 we draw conclusions and outline directions for future work.

5.2 Related Work

In this section, we focus more on the relate work of structural similarity. For network embedding studies, more details have been discussed in Chapter 3 so in this section we only brief discuss the local and global structure preservation of selected embedding approaches.

5.2.1 Network Embedding

Most network embedding methods only concern the local structural information represented by paths consists of linked nodes, i.e., the community structures of networks. But they fail to capture global structural information, i.e., structural roles. SNS [LZZ17], *struc2vec* [RSF17] and DRNE [TCW+18] are exceptions which take global structural information into consideration. SNS uses graphlet information for structural similarity calculation as a pre-propcessing

Table 5.1: A brief summary of different network embedding methods. Note that (1) we only list methods for homogeneous networks without attributes, and (2) *node2vec* [GL16] aims to capture both local and global structure information but walk-based sampling strategy is not good at capturing global structure information shown in our experiments in Section 5.5.

Method	community (local)	role (global)	uncertainty
DeepWalk [PARS14]	√		
LINE [TQW+15]	√		
GraRep [CLX15]	√		
PTE [TQM15]	√		
Walklets [PKS16]	√		
node2vec [GL16]	√		
struc2vec [RSF17]		√	
DRNE [TCW+18]		√	
graph2gauss [BG17]	√		√
DVNE [ZCWZ18]	√		√
SNS [LZZ17]		√	
our method		√	√

step. *struc2vec* applies the dynamic time warping to measure similarity between two nodes' degree sequences and builds a new multilayer graph based on the similarity. Then similar mechanism used in DeepWalk has been used to learn node representations. DRNE explicitly models regular equivalence, which is one way to define the structural role, and leverages the layer normalized LSTM [BKH16] to learn the representations for nodes. A brief summary of these network embedding methods is list in Table 5.1.

5.2.2 Structural Similarity

Structure based network analysis tasks can be categorized into two types: structural similarity calculation and network clustering .

Calculating structural similarities between nodes is a hot topic in recent years and different methods have been proposed. SimRank [JW02] is one of the most representative notions to calculate structural similarity. It implements a recursive definition of node similarity based on the assumption that two objects are similar if they relate to similar objects. SimRank++ [AMC08] adds an evidence weight which partially compensates for the neighbor matching car-

dinality problem. P-Rank [ZHS09] extends SimRank by jointly encoding both in- and out-link relationships into structural similarity computation. MatchSim [LLK09] uses maximal matching of neighbors to calculate the structural similarity. RoleSim [JLH11] is the only similarity measure which can satisfy the automorphic equivalence properties.

Network clusters can be based on either global or local structural information. Graph clustering based on global structural information is the problem of role discovery [RA+15]. In social science research, roles are represented as concepts of equivalence [WF94]. Graph-based methods and feature-based methods have been proposed for this task. Graph-based methods take nodes and edges as input and directly partition nodes into groups based on their structural patterns. For example, Mixed Membership Stochastic Blockmodel [ABFX08] infers the role distribution of each node using the Bayesian generative model. Feature-based methods first transfer the original network into feature vectors and then use clustering methods to group nodes. For example, RolX [HGER+12] employs ReFeX [HGL+11] to extract features of networks and then uses non-negative matrix factorization to cluster nodes. Local structural information based clustering corresponds to the problem of community detection [For10]. A community is a group of nodes that interact with each other more frequently than with those outside the group. Thus, it captures only local connections between nodes.

5.3 Problem Statement

We illustrated local community structure and global role structure in Section 5.1 using the example in Figure 5.1. In this section, we formally define the problem of structural role preserving network embedding. The formal definitions of structural role 2.3 and community structure 2.4 can be found in Section 2.1 of Chapter 2.

Problem 5.1. Structural Role Preserving Network Embedding. Given a network $G = (V, E)$, where V is a set of nodes and E is a set of edges between the nodes, the problem of **Structural Preserving Network Embedding** aims to represent each node $v \in V$ into a Gaussian distribution with mean μ and covariance Σ in a low-dimensional space $\mathbb{R}^d$, i.e., learning a function

$$f : V \to \mathcal{N}(x; \mu, \Sigma),$$

where $\mu \in \mathbb{R}^d$ is the mean, $\Sigma \in \mathbb{R}^{d \times d}$ is the covariance and $d \ll |V|$. In the space $\mathbb{R}^d$, the global structural information of nodes can be preserved and the uncertainty of node representations can be captured.

Note that by solving this problem, the ultimate target is to learn representations of nodes in a dynamic network which can well preserve the global structure and model uncertainties of networks. Thus, the learned representations should perform well in discovering role and capturing the uncertainties of embeddings.

5.4 *struc2gauss*

An overview of our proposed *struc2gauss* framework is shown in Figure 5.2. Given a network, a similarity measure is employed to calculate the similarity matrix, then the training set which consists of positive and negative pairs are sampled based on the similarity matrix. Finally, Gaussian embedding techniques are applied on the training set and generate the embedded Gaussian distributions as the node representations and uncertainties of the representations. Besides, we the computational complexity and the flexibility of our *struc2gauss* framework.

5.4.1 Structural Similarity Calculation

It has been theoretically proved that random walk sampling based embedding methods are not capable of capturing structural equivalence [LZZ17] which is one way to model the structural roles in networks [WF94]. Thus, to capture the

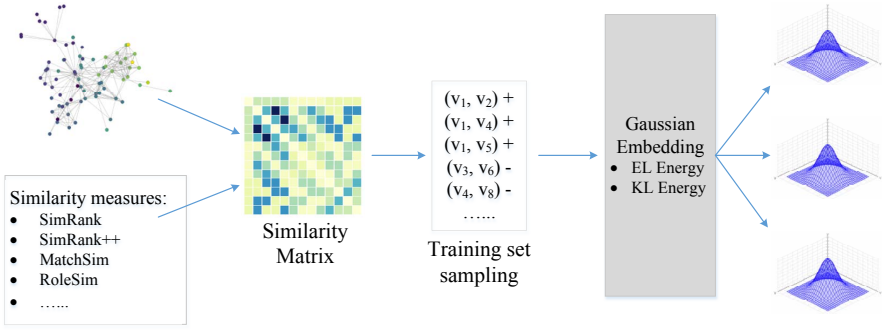


Figure 5.2: Overview of the *struc2gauss* framework. *struc2gauss* consists of three components: similarity calculation, training set sampling and Gaussian embedding.

global structural information, we calculate the pairwise structural similarity as a pre-processing step similar to [LZZ17, RSF17].

In the literature, a variety of structural similarity measures have been proposed to calculate node similarity based on the structures of networks, e.g., SimRank [JW02], MatchSim [LLK09] and RoleSim [JLH11, JLL14]. However, not all these measures can capture the global structural role information which we will show the empirical evidence in the experiments in Section 5.5. Therefore, in this section we leverage RoleSim for the structural similarity since it satisfies all the requirements of Axiomatic Role Similarity Properties for modeling the equivalence [JLH11], i.e., the structural roles. RoleSim also generalizes Jaccard coefficient and corresponds linearly to the maximal weighted matching. RoleSim similarity between two nodes u and v is defined as:

$$RoleSim(u,v) = (1-\beta) \max_{M(u,v)} \frac{\sum_{(x,y)\in M(u,v)} RoleSim(x,y)}{|N(u)| + |N(v)| - |M(u,v)|} + \beta \tag{5.1}$$

where $|N(u)|$ and $|N(v)|$ are the numbers of neighbors of node u and v, respectively. $M(u,v)$ is a matching between $N(u)$ and $N(v)$, i.e., $M(u,v) \subseteq N(u) \times N(v)$ is a bijection between $N(u)$ and $N(v)$. The parameter β is a decay factor where $0 < \beta < 1$. The intuition of RoleSim is that two nodes are structurally similar if their corresponding neighbors are also structurally similar. This intuition is consistent with the notion of automorphic and regular equivalence [WF94]. The computation procedure of RoleSim in practice has been introduced in Section 2.3.2 of Chapter 2.

Note that there other strategies can be used to capture the global structural role information except structural similarity, and these possible strategies will be discussed in Section 5.4.6. The advantage of RoleSim in capturing structural roles to other structural measures will also be discussed empirically in Section 5.5.6.

5.4.2 Training Set Sampling

The target of structural role preserving network embedding is to map nodes in the network to a latent space where the learned latent representations of two nodes are (1) more similar if these two nodes are structurally similar, and (2) more dissimilar if these two nodes are not structurally similar. Hence, we need to generate structurally similar and dissimilar node pairs as the training set based on the similarity we learned in Section 5.4.1. We name the structurally similar pairs of nodes the positive set and the structurally dissimilar pairs the negative set.

In detail, for node v, we rank its similarity values towards other nodes and then select top-k most similar nodes $u_i, i = 1, ..., k$ to form its positive set $\Gamma_+ = \{(v, u_i) | i = 1, ..., k\}$. For the negative set, we randomly select the same number of nodes $\{u'_i, i = 1, ..., k\}$ same to [VM14] and other random walk sampling based methods [PARS14, TQW+15, GL16], i.e., $\Gamma_- = \{(v, u'_i) | i = 1, ..., k\}$. Therefore, k is a parameter indicating the *number of positive/negative nodes per node*. We will generate r positive and negative sets for each node where r is a parameter indicating the *number of samples per node*. The influence of these parameters will be analyzed empirically in Section 5.5.7.

5.4.3 Gaussian Embedding

Overview

Recently language modeling techniques such as *word2vec* have been extensively used to learn word representations in and almost all network embedding studies are based on these word embedding techniques. However, these studies map each entity to a fixed point vector in a low-dimension space so that the uncertainties of learned embeddings are ignored. Gaussian embedding aims to solve this problem by learning density-based distributed embeddings in the space of Gaussian distributions [VM14]. Gaussian embedding has been utilized in different graph mining tasks including triplet classification on knowledge graphs [HLJZ15], multi-label classification on heterogeneous graphs [DSPG16] and link prediction and node classification on attributed graphs [BG17].

Gaussian embedding trains with a ranking-based loss based on the ranks of positive and negative samples. Following [VM14], we choose the max-margin ranking objective which can push scores of positive pairs above negatives by a margin defined as:

$$\mathcal{L} = \sum_{(v,u)\in\Gamma_+} \sum_{(v',u')\in\Gamma_-} \max(0, m - \mathcal{E}(z_v, z_u) + \mathcal{E}(z_{v'}, z_{u'})) \tag{5.2}$$

where Γ_+ and Γ_- are the positive and negative pairs, respectively. $\mathcal{E}(\cdot, \cdot)$ is the energy function which is used to measure the similarity of two distributions, z_v and z_u are the learned Gaussian distributions for nodes v and u, and m is the margin separating positive and negative pairs. In this section, we present two different energy functions to measure the similarity of two distributions for node representation learning, i.e., expected likelihood and KL divergence based energy functions. For the learned Gaussian distribution $z_i \sim \mathcal{N}(0; \mu_i, \Sigma_i)$

for node i, to reduce the computational complexity, we restrict the covariance matrix Σ_i to be diagonal and spherical in this work.

Expected Likelihood based Energy

Although both dot product and inner product can be used to measure similarity between two distributions, dot product only considers means and does not incorporate covariances. Thus, we use inner product to measure the similarity. Formally, the integral of inner product between two Gaussian distributions z_i and z_j (learned Gaussian embeddings for node i and j respectively), a.k.a., expected likelihood, is defined as:

$$E(z_i, z_j) = \int_{x \in \mathbb{R}} \mathcal{N}(x; \mu_i, \Sigma_i) \mathcal{N}(x; \mu_j, \Sigma_j) dx = \mathcal{N}(0; \mu_i - \mu_j, \Sigma_i + \Sigma_j). \tag{5.3}$$

For simplicity in computation and comparison, we use the logarithm of Eq. (5.3) as the final energy function:

$$\begin{aligned} &\mathscr{E}_{EL}(z_i, z_j) = \log E(z_i, z_j) = \log \mathcal{N}(0; \mu_i - \mu_j, \Sigma_i + \Sigma_j) \\ =&\frac{1}{2}\Big\{(\mu_i - \mu_j)^T (\Sigma_i + \Sigma_j)^{-1} (\mu_i - \mu_j) + \log\det(\Sigma_i + \Sigma_j) + d\log(2\pi)\Big\} \end{aligned} \tag{5.4}$$

where d is the number of dimensions. The gradient of this energy function with respect to the means μ and covariances Σ can be calculated in a closed form as:

$$\begin{aligned} \frac{\partial \mathscr{E}_{EL}(z_i, z_j)}{\partial \mu_i} &= -\frac{\partial \mathscr{E}(z_i.z_j)_{EL}}{\partial \mu_j} = -\Delta_{ij} \\ \frac{\partial \mathscr{E}_{EL}(z_i, z_j)}{\partial \Sigma_i} &= \frac{\partial \mathscr{E}(z_i.z_j)_{EL}}{\partial \Sigma_j} = \frac{1}{2}(\Delta_{ij}\Delta_{ij}^T - (\Sigma_i + \Sigma_j)^{-1}), \end{aligned} \tag{5.5}$$

where $\Delta_{ij} = (\Sigma_i + \Sigma_j)^{-1}(\mu_i - \mu_j)$ [HLJZ15, VM14]. Note that expected likelihood is a symmetric similarity measure, i.e., $\mathscr{E}_{EL}(z_i, z_j) = \mathscr{E}_{EL}(z_j, z_i)$.

KL Divergence based Energy

KL divergence is another straightforward way to measure the similarity between two distributions so we utilize the energy function $\mathscr{E}_{KL}(z_i, z_j)$ based on the KL divergence to measure the similarity between Gaussian distributions z_i and z_j

(learned Gaussian embeddings for node i and j respectively):

$$\begin{aligned} &\mathscr{E}_{KL}(z_i, z_j) = D_{KL}(z_i, z_j) \\ &= \int_{x\in\mathbb{R}} \mathcal{N}(x;\mu_i,\Sigma_i)\log\frac{\mathcal{N}(x;\mu_j,\sigma_j)}{\mathcal{N}(x;\mu_i,\Sigma_i)}dx \\ &= \frac{1}{2}\Big\{tr(\Sigma_i^{-1}\Sigma_j) + (\mu_i-\mu_j)^T\Sigma_i^{-1}(\mu_i-\mu_j) - \log\frac{\det(\Sigma_j)}{\det(\Sigma_i)} - d\Big\} \end{aligned} \tag{5.6}$$

where d is the number of dimensions. Similarly, we can compute the gradients of this energy function with respect to the means μ and covariances Σ:

$$\begin{aligned} \frac{\partial\mathscr{E}_{KL}(z_i,z_j)}{\partial\mu_i} &= -\frac{\partial\mathscr{E}_{KL}(z_i.z_j)}{\partial\mu_j} = -\Delta'_{ij} \\ \frac{\partial\mathscr{E}_{KL}(z_i,z_j)}{\partial\Sigma_i} &= \frac{1}{2}(\Sigma_i^{-1}\Sigma_j\Sigma_i^{-1} + \Delta'_{ij}\Delta'^{T}_{ij} - \Sigma_i^{-1}) \\ \frac{\partial\mathscr{E}_{KL}(z_i,z_j)}{\partial\Sigma_j} &= \frac{1}{2}(\Sigma_j^{-1} - \Sigma_i^{-1}) \end{aligned} \tag{5.7}$$

where $\Delta'_{ij} = \Sigma_i^{-1}(\mu_i - \mu_j)$.

Note that KL divergence based energy is asymmetric but we can easily extend to a symmetric similarity measure as follows:

$$\mathscr{E}(z_i, z_j) = \frac{1}{2}(D_{KL}(z_i, z_j) + D_{KL}(z_j, z_i)). \tag{5.8}$$

5.4.4 Learning

To avoid the means to grow to large and the covariances to be positive definite as well as reasonably sized, we regularize the means and covariances to learn the embedding [VM14]. Due to the different geometric characteristics, two different hard constraint strategies have been used for means and covariances, respectively. Note that we only consider diagonal and spherical covariances. In particular, we have

$$\|\mu_i\| \leq C,\ \forall i, \qquad c_{min}I \prec \Sigma_i \prec c_{max}I,\ \forall i. \tag{5.9}$$

The constraint on means guarantees them to be sufficiently small and constraint on covariances ensures that they are positive definite and of appropriate size. For example, $\Sigma_{ii} \leftarrow \max(c_{min}, \min(c_{max}, \Sigma_{ii}))$ can be used to regularize diagonal covariances.

We use AdaGrad [DHS11] to optimize the parameters. The learning procedure is described in Algorithm 3. Initialization phase is from line 1 to 4, context generation is shown in line 7, and Gaussian embeddings are learned from line 8 to 14.

Algorithm 3 The Learning Algorithm of *struc2gauss*

Input: An energy function $\mathscr{E}(z_i, z_j)$, a graph $G = (V, E)$, embedding dimension d, constraint values C for mean and c_{max} and c_{min} for covariance, learning rate α, and maximum epochs n.

Output: Gaussian embeddings (mean vector μ and covariance matrix Σ) for nodes $v \in V$

1: **for all** $v \in V$ **do**
2: Initialize mean μ for v randomly
3: Initialize covariance Σ for v randomly
4: Regularize μ and Σ with constraint in Eq. (5.9)
5: **end for**
6: **while** not reach the maximum epochs n **do**
7: Generate positive and negative sets Γ_+ and Γ_- for each node
8: **if** use expected likelihood based energy **then**
9: Update means and covariances based on Eq. (5.5)
10: **end if**
11: **if** use KL divergence based energy **then**
12: Update means and covariances based on Eq. (5.7)
13: **end if**
14: Regularize μ and Σ with constraint in Eq. (5.9)
15: **end while**

5.4.5 Computational Complexity

The complexity of different components of *struc2gauss* are analyzed as follows:

1 For structural similarity calculation using RoleSim, the computational complexity is $O(kn^2d)$, where n is the number of nodes, k is the number of iterations and d is the average of $y \log y$ over all node-pair bipartite graph in G [JLH11].

2 To generate the training set based on similarity matrix, we need to sample from the most similar nodes for each node, i.e., to select k largest numbers

from an unsorted array. Using heap, the complexity is $O(n \log k)$.

3 For Gaussian embedding, the operations include matrix addition, multiplication and inversion. In practice, as stated above, we only consider two types of covariance matrices, i.e., diagonal and spherical, so all these operations have the complexity of $O(n)$.

Overall, the component of similarity calculation is the bottleneck of the framework. One possible and effective way to optimize this part is to set the similarity to be 0 if two nodes have a large difference in degrees. The reason is: (1) we generate the context only based on most similar nodes; and (2) two nodes are less likely to be structural similar if their degrees are very different.

5.4.6 Discussion

The proposed *struc2gauss* is a flexible framework for node representations. As shown in Figure 5.2, different similarity measures can be incorporated into this framework and empirical studies will be presented in Section 5.5.6. Furthermore, other types of methods which model structural information can be utilized in *struc2gauss* as well.

To illustrate the potential to incorporate different methods, we categorize different methods for capturing structural information into three types:

- **Similarity-based methods**. Similarity-based methods calculate pairwise similarity based on the structural information of a given network. Related work has been reviewed in Section 5.2.2.

- **Ranking-based methods**. PageRank [PBMW99] and HITS [Kle99] are two most representative ranking-based methods which learns the structural information. PageRank has been used for network embedding in [MWR+17].

- **Partition-based methods**. This type of methods, e.g., role discovery, aims to partition nodes into disjoint or overlapping groups, e.g., REGE [BE93] and RolX [HGER+12].

In this section, we focus on **similarity-based methods**. For **ranking-based methods**, we can use a fixed sliding window on the ranking list, then given a node the nodes within the window can be viewed as the context. In fact, this mechanism is similar to DeepWalk. For **partition-based methods**, we can consider the nodes in the same group as the context for each other.

Table 5.2: A brief introduction to data sets.

Type	Dataset	# nodes	# edges	# groups
with labels	Brazilian-air	131	1038	4
	European-air	399	5995	4
	USA-air	1190	13599	4
without labels	Arxiv GR-QC	5242	28980	8
	Advogato	6551	51332	11
	Hamsterster	2426	16630	10

5.5 Experiments

We evaluate *struc2gauss* in different tasks in order to understand its effectiveness in capturing structural information, capability in modeling uncertainties of embeddings and stability of the model towards parameters. We also study the influence of different similarity measures empirically.

5.5.1 Experimental Setup

Datasets

We conduct experiments on two types of network datasets: networks with and without ground-truth labels where these labels can represent the global structural role information of nodes in the networks. For networks with labels, to compare to state-of-the-art, we use air-traffic networks from [RSF17] where the networks are undirected, nodes are airports, edges indicate the existence of commercial flights and labels correspond to their levels of activities. For networks without labels, we select three real-world networks in different domains from Network Repository[1]. A brief introduction to these datasets is shown in Table 5.2. The numbers of groups for networks without labels are determined by Minimum Description Length (MDL) [HGER+12]. Note that we only use means of learned Gaussian distributions as the node embeddings in role clustering and classification tasks. The covariances are left for uncertainty modeling.

[1] `http://networkrepository.com/index.php`

Baselines

We compare *struc2gauss* with several state-of-the-art network embedding methods.

- DeepWalk [PARS14]: DeepWalk [PARS14] learns node representations based on random walks using the same mechanism of *word2vec* by drawing an analogy between paths consists of several nodes on networks and word sequences in text. The structural information is captured by the paths of nodes generated by random walks.
- *node2vec* [GL16]: It extends DeepWalk to learn latent representations from the node paths generated by biased random walk. Two hyper-parameters p and q are used to control the random walk to be breadth-first or depth-first. In this way, *node2vec* can capture the structural information in networks. Note that when $p = q = 1$, *node2vec* degrades to DeepWalk.
- LINE [TQW+15]: It learns node embeddings via preserving both the local and global network structures. By extending DeepWalk, LINE aims to capture both the first-order, i.e., the neighbors of nodes, and second-order proximities, i.e., the shared neighborhood structures of nodes.
- *struc2vec* [RSF17]: It learns latent representations for the structural identity of nodes. Due to its high computational complexity, we use the combination of all optimizations proposed in the paper for large networks.
- *graph2gauss* [VM14]: It maps each node into a Gaussian distribution where the mean indicates the position of a node in the embedded space and the covariance denotes the uncertainty of the learned representation. [BG17] and [DSPG16] extend the original Gaussian embedding method to network embedding task.
- DRNE [TCW+18]: It learns node representations based on the concept of regular equivalence. DRNE utilizes a layer normalized LSTM to represent each node by aggregating the representations of their neighborhoods in a recursive way so that the global structural information can be preserved.

For all baselines, we use the implementation released by the original authors. For our framework *struc2gauss*, we test four variants: *struc2gauss* with expected likelihood and diagonal covariance (*s2g*_el_d), expected likelihood and spherical covariance (*s2g*_el_s), KL divergence and diagonal covariance (*s2g*_kl_d), and KL divergence and spherical covariance (*s2g*_kl_s).

For other settings including parameters and evaluation metrics, different settings will be discussed in each task.

5.5.2 Case Study: Visualization in 2-D space

We use the toy example shown in Figure 5.1 to demonstrate the effectiveness of *struc2gauss* in capturing the global structural information and the failure of other state-of-the-art techniques in this task. The toy network consists of ten nodes and they can be clustered from two different perspectives:

- from the perspective of the global role structure, they belong to three groups, i.e., $\{0,1,2,3\}$ (yellow color), $\{4,5,6,7\}$ (blue color) and $\{8,9\}$ (red color) because different groups have different structural functions in this network;
- from the perspective of the local community structure, they belong to two groups, i.e., $\{0,1,4,5,6,8\}$ and $\{2,3,6,7,9\}$ because there are denser connections/more edges inside each community that outside the community.

Note that from the perspective of role discovery, these three groups of nodes can be explained to play the roles of *periphery*, *star* and *bridge*, respectively.

In this study, we aim to preserve the global structural information in network embedding. Figure 5.3 shows the learned node representations by different methods. For shared parameters in all methods, we use the same settings by default: representation dimension: 2, number of walks per node: 20, walk length: 80, skipgram window size: 5. For *node2vec*, we set $p = 1$ and $q = 2$. For *graph2gauss* and *struc2gauss*, the number of walks per node is 20 and the number of positive/negative nodes per node is 5. The constraint for means C is 2 and constraints for covariances c_{min} and c_{max} are 0.5 and 2, respectively. From the visualization results, it can be observed that:

- Our proposed *struc2gauss* outperforms all other methods. Both diagonal and spherical covariances can separate nodes based on global structural information and *struc2gauss* with spherical covariances performs better than diagonal covariances since it can recognize *star* and *bridge* nodes better.
- Methods aim to capture the global structural information performs better than random walk sampling based methods. For example, *struc2vec* can solve this problem to some extent. However, there is overlap between

Table 5.3: NMI for node clustering in air-traffic networks using different network embedding methods. In *struc2gauss*, el and kl mean expected likelihood and KL divergence, respectively. d and s mean diagonal and spherical covariances, respectively. The highest values are in bold.

	Brazil-air	Europe-air	USA-air
DeepWalk [PARS14]	0.1303	0.0458	0.0766
LINE [TQW+15]	0.2215	0.1563	0.1275
node2vec [GL16]	0.2516	0.1722	0.0945
graph2gauss [VM14]	0.1204	0.1109	0.0896
struc2vec [RSF17]	0.3758	0.2729	0.2486
DRNE [TCW+18]	0.5244	0.2766	0.2918
*s2g*_el_d	0.5615	0.3234	0.3188
*s2g*_el_s	0.5396	0.2974	0.2967
*s2g*_kl_d	0.5527	0.3145	0.3212
*s2g*_kl_s	**0.5675**	**0.3280**	**0.3217**

node 6 and 9. It has been stated that *node2vec* can capture the structural equivalence but the visualization shows that it still captures the local structural information similar to DeepWalk.

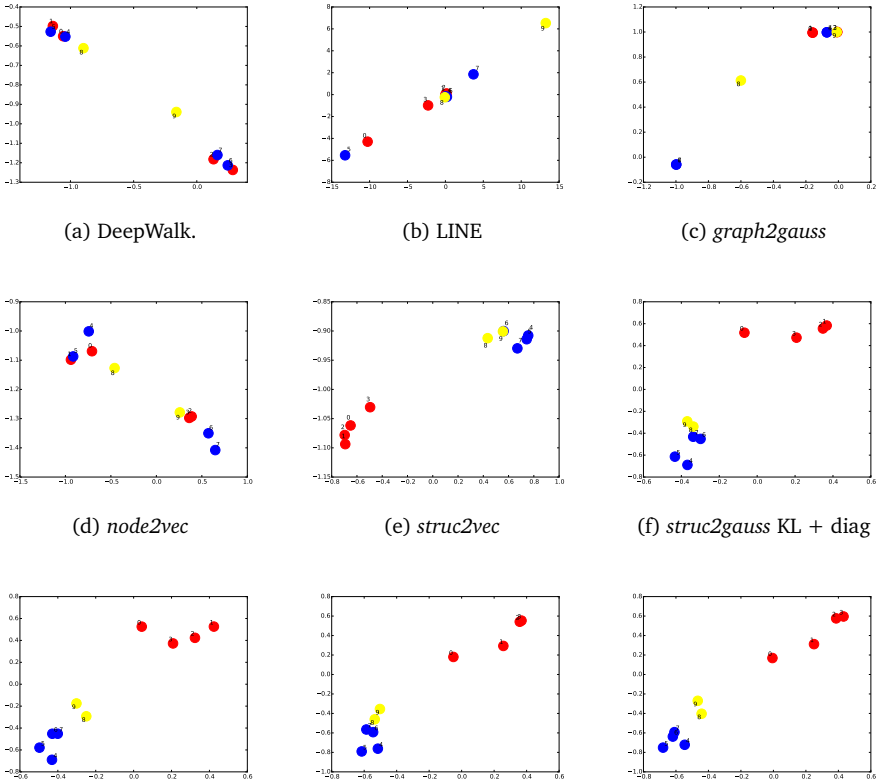
- DeepWalk, LINE and *graph2gauss* fail to capture the global structural information because these methods are based on random walk which only captures the local community structures. DeepWalk is capable to capture the local structural information since nodes are separated into two parts corresponding to the two communities shown in Figure 5.1.

5.5.3 Structural Role Discovery

The most common network mining application based on global structural information is the problem of *role discovery* and role discovery essentially is a clustering task. Thus, we consider this task to illustrate the potential of node representations learned by *struc2gauss*. We use the latent representations learned by different methods (in *struc2gauss*, we use means of learned Gaussian distribution) as features and K-means as the clustering algorithm to cluster nodes.

Parameters. For these baselines, we use the same settings in the literature: representation dimension: 128, number of walks per node: 20, walk length: 80, skipgram window size: 10. For *node2vec*, we set $p = 1$ and $q = 2$. For

graph2gauss and *struc2gauss*, we set the constraint for means C to be 2 and constraints for covariances c_{min} and c_{max} to be 0.5 and 2, respectively. The number of walks per node is 10, the number of positive/negative nodes per node is 120 and the representation dimension is also 128.



(a) DeepWalk. (b) LINE (c) *graph2gauss*

(d) *node2vec* (e) *struc2vec* (f) *struc2gauss* KL + diag

(g) *struc2gauss* KL + spher (h) *struc2gauss* EL + diag (i) *struc2gauss* EL + spher

Figure 5.3: Latent representations in $\mathbb{R}^2$ learned by (a) DeepWalk, (b) LINE, (c) GraRep, (d) *node2vec*, (e) *struc2vec*, (f) *struc2gauss* using KL divergence with diagonal covariance, (g) *struc2gauss* using KL divergence with spherical covariance, (g) *struc2gauss* using KL divergence with diagonal covariance, (h) *struc2gauss* using expected likelihood with diagonal covariance, and (i) *struc2gauss* using expected likelihood with spherical covariance.

Evaluation Metrics. To quantitatively evaluate clustering performance in labeled networks, we use *Normalized Mutual Information (NMI)* as the evaluation metric. NMI is obtained by dividing the mutual information by the arithmetic average of the entropy of obtained cluster and ground-truth cluster. For unlabeled networks, we use normalized *goodness-of-fit* as the evaluation metric. *goodness-of-fit* can measure how well the representation of roles and the relations among these roles fit a given network [WF94]. To make the evaluation metric value in the range of $[0,1]$, we normalize *goodness-of-fit* by dividing r^2 where r is number of groups/roles. For more details about *goodness-of-fit indices*, please refer to [WF94].

The NMI values for node clustering on networks with labels are shown in Table 5.3 and the normalized *goodness-of-fit* values for networks without labels are shown in Figure 5.4. From these results, some conclusions can be drawn:

- For both types of networks with and without clustering labels, *struc2gauss* outperforms all other methods in different evaluation metrics. It indicates the effectiveness of *struc2gauss* in capturing the global structural information.

- Comparing *struc2gauss* with diagonal and spherical covariances, it can be observed that spherical covariance can achieve better performance in

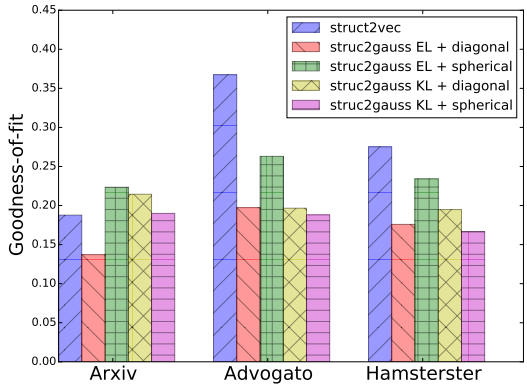


Figure 5.4: Goodness-of-fit of *struc2vec* and *struc2gauss* with different strategies and covariances on three real-world networks. Lower value means better performance.

node clustering. This finding is similar to the results of word embedding in [VM14].

- For baselines, *struc2vec* can capture the structural information to some extent since its performance is much better than DeepWalk and *node2vec* while both of them fail in capturing the global structural information for node clustering.

5.5.4 Structural Role Classification

Node classification is another widely used task for embedding evaluation. Different from previous studies which focused on community structures, our approach aims to preserve the global role structures. Thus, we evaluate the effectiveness of *struc2gauss* in role classification task. Same to the node clustering task in Section 5.5.3, we use the latent representations learned by different methods as features. Each dataset is separated into training set and test set (we will explore the classification performance with different percentages of training set). To focus on the learned representation, we use logistic regression as the classifier.

Structural role classification as a supervised task, the ground-truth labels are required. Thus we only use two air-traffic networks for evaluation. We compare our approach to the same state-of-the-art network embedding algorithms as baselines used in Section 5.5.3, i.e., DeepWalk, LINE, *node2vec*, *graph2gauss*, *struc2vec* and DRNE. Same to [TCW+18], we also compare to four centrality measures, i.e., closeness centrality, betweenness centrality, eigenvector centrality and k-core. Since the combination of these four measures perform best [TCW+18], we only compare the classification performance of the combination as features in this task. The parameters of baselines and *struc2gauss*, we use the same settings in Section 5.5.3.

The average accuracies for structural role classification in Europe-air and USA-air are shown in Figure 5.5 and 5.6. From the results, we can observe that:

- *struc2gauss* outperforms all other methods in both networks. It indicates the effectiveness of *struc2gauss* in modeling the structural role information. Although not the same combination of energy function and covariance form performs best in two networks, different variants of *struc2gauss* are always the best.

- Among the baselines, only DRNE can capture the global structural information so that it achieves better classification accuracy than other base-

lines. *struc2vec* is the second best baseline because it also aims to capture structural roles.

- Random walk sampling based network embedding methods only capture the local community structures so they perform worse than *struc2gauss* and DRNE. Node that methods considering first- and/or second-order proximity, e.g., DeepWalk and LINE, is not capable of modeling global structural information.

5.5.5 Uncertainty Modeling

Mapping a node in a network into a distribution rather than a point vector allows us to model the uncertainty of the learned representation which is another advantage of *struc2gauss*. It is intuitive that the more noisy edges a node has, the less discriminative information it contains, thus making its embedding more uncertain. To verify it, we conduct the following experiment using Brazil-air and Europe-air networks. We randomly insert certain number of edges to the network and then learn the latent representations and covariances. The average variance is used to measure the uncertainties. For Brazil-air network, we range the number of noisy edges from 50 to 300 and for Europe-air it ranges from 500 to 3000. The other parameter settings are same to Section 5.5.3.

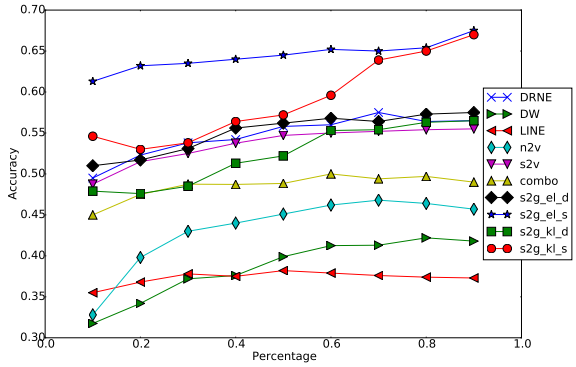


Figure 5.5: Average accuracy for structural role classification in Europe-air network.

The results are shown in Figure 5.8 where it can be observed that with more noisy edges being added to the networks, larger average variance values we have. *struc2gauss* with different energy functions and covariance forms have the same trend. This demonstrates that our proposed *struc2gauss* is able to model the uncertainties of learned node representations. It is interesting that *struc2gauss* with expected likelihood and diagonal covariance (s2g_el_d) always has the lowest average variance while *struc2gauss* with KL divergence and diagonal (s2g_kl_d) always has the largest value. This may result from the learning mechanism of different energy functions when measuring the distance between two distributions.

5.5.6 Influence of Similarity Measures

As we mentioned not all structural similarity measures can capture the global structural role information, to validate the rationale to select RoleSim as the similarity measure for structural role information, we investigate the influence of different similarity measures on learning node representations. In specific, we select two other widely used structural similarity measures, i.e., SimRank [JW02] and MatchSim [LLK09], and we incorporate these measures by replacing RoleSim in our framework. The datasets and evaluation metrics used in this experiment are the same to Section 5.5.3. For simplicity, we only show the results of *struc2gauss* using KL divergence with spherical covariance in this experiment

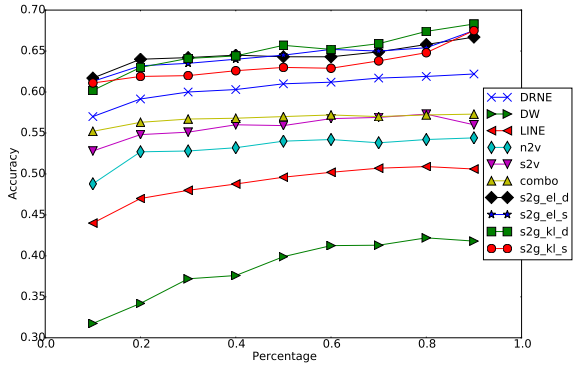


Figure 5.6: Average accuracy for structural role classification in USA-air network

Table 5.4: NMI for node clustering in air-traffic networks of Brazil, Europe and USA using *struc2gauss* with different similarity measures.

	Brazil-air	Europe-air	USA-air
SimRank	0.1695	0.0524	0.0887
MatchSim	0.3534	0.2389	0.0913
RoleSim	**0.5675**	**0.3280**	**0.3217**

because different variants perform similarly in previous experiments.

The NMI values for networks with labels are shown in Table 5.4 and the *goodness-of-fit* values are shown in Figure 5.7. We can come to the following conclusions:

- RoleSim outperforms other two similarity measures in both types of networks with and without clustering labels. It indicates RoleSim can better capture the global structural information. Performance of MatchSim varies on different networks and is similar to *struc2vec*. Thus, it can capture the global structural information to some extent.
- SimRank performs worse than other similarity measures as well as *struc2vec* (Table 5.3). Considering the basic assumption of SimRank that "two objects are similar if they relate to similar objects", it computes the similarity

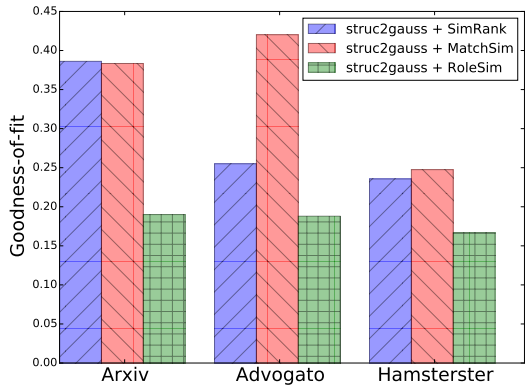


Figure 5.7: Goodness-of-fit of *struc2gauss* with different similarity measures. Lower values are better.

also via relations between nodes so that the mechanism is similar to random walk based methods which have been proved not being capable of capturing the global structural information [LZZ17].

5.5.7 Parameter Sensitivity

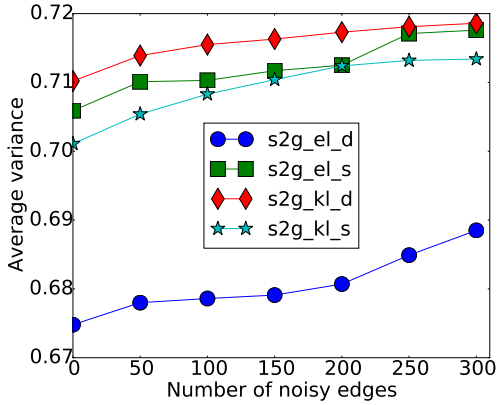
We consider two types of parameters in *struc2gauss*: (1) parameters also used in other network embedding methods including *latent dimensions*, *number of samples per node* and *number of positive/negative nodes per node*; and (2) parameters only used in Gaussian embedding including mean constraint C and covariance constraint c_{max} (note that we fix the minimal covariance c_{min} to be 0.5 for simplicity). In order to evaluate how changes to these parameters affect performance, we conducted the same node clustering experiment on the labeled USA-air network introduced in Section 5.5.3. In the interest of brevity, we tune one parameter by fixing all other parameters. In specific, the number of latent dimensions varies from 10 to 200, the number of samples varies from 5 to 15 and the number of positive/negative nodes varies from 40 to 190. Mean constraint C is from 1 to 10, and covariance constraint c_{max} ranges from 1 to 10.

The results of parameter sensitivity are shown in Figure 5.9 and Figure 5.10. It can be observed from Figure 5.9 (a) and 5.9 (b) that the trends are relatively stable, i.e., the performance is insensitive to the changes of representation dimensions and numbers of samples. The performance of clustering is improved with the increase of numbers of positive/negative nodes shown in Figure 5.9 (c). Therefore, we can conclude that *struc2guass* is more stable than other methods. It has been reported that other methods, e.g., DeepWalk [PARS14], LINE [TQW$^+$15] and *node2vec* [GL16], are sensitive to many parameters. In general, more dimensions, more walks and more context can achieve better performance. However, it is difficult to search for the best combination of parameters in practice and it may also lead to overfitting. For Gaussian embedding specific parameters C and c_{max}, both trends are stable, i.e., the selection of these contraints have little effect on the performance. Although with larger mean constraint C, the NMI decreases but the difference is not huge.

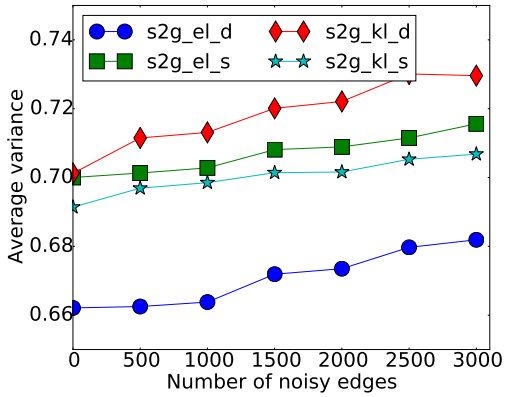
5.6 Conclusions

In Section 5.1, the research question, *How can we effectively discover roles on static graphs by capturing the global structures and uncertainties?*, has been raised.

To explore the answer, we first present the two major limitations exist in previous network embedding studies: i.e., **structure preservation** and **uncertainty modeling**. Random-walk based network embedding methods fail in capturing global structural information and representing a node into a point vector are



(a) Average variance with different numbers of noisy edges on Brazil-air.



(b) Average variance with different numbers of noisy edges on Europe-air.

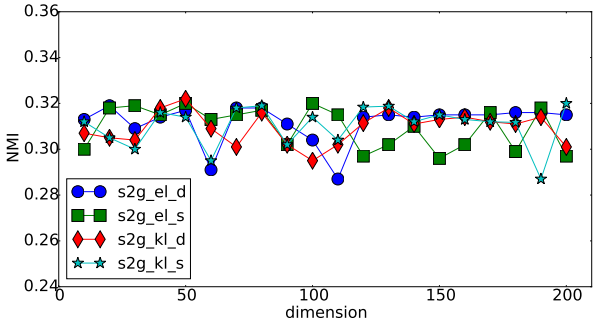
Figure 5.8: Uncertainties of embeddings with different levels of noise.

not capable of modeling the uncertainties of node representations.

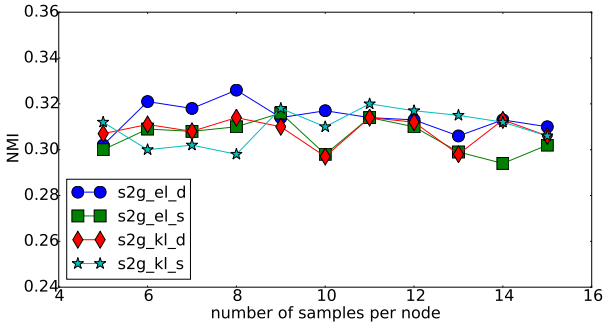
We proposed a flexible structure preserving network embedding framework, *struc2gauss*, to tackle these limitations. On the one hand, *struc2gauss* learns node representations based on structural similarity measures so that global structural information can be taken into consideration. On the other hand, *struc2gauss* utilizes Gaussian embedding to represent each node as a Gaussian distribution where the mean indicates the position of this node in the embedding space and the covariance represents its uncertainty.

We experimentally compared three different structural similarity measures for networks and two different energy functions for Gaussian embedding. By conducting experiments from different perspectives, we demonstrated that *struc2gauss* excels in capturing global structural information, compared to state-of-the-art embedding techniques such as DeepWalk, *node2vec* and *struc2vec*. It outperforms other competitor methods in role discovery task and structural role classification on several real-world networks. It also overcomes the limitation of uncertainty modeling and is capable of capturing different levels of uncertainties. Additionally, *struc2gauss* is less sensitive to different parameters which makes it more stable in practice without putting more effort in tuning parameters.

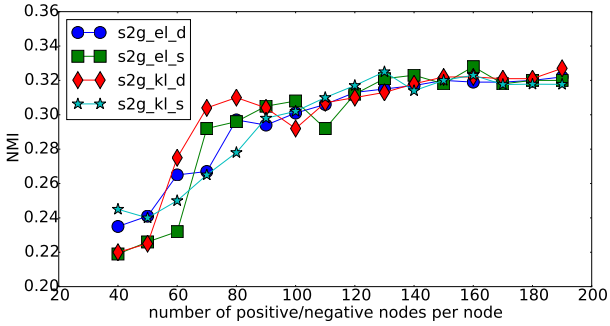
In the future, we will explore faster RoleSim measures for more scalable network embedding methods, for example, fast method to select k most similar nodes for a given node. Also, it is a promising research direction to investigate different strategies to model global structural information except structural similarity in network embedding tasks. Besides, our other future investigations in this area will seek to learn node representations in dynamic and temporal networks.



(a) Representation dimensions vs. NMI.

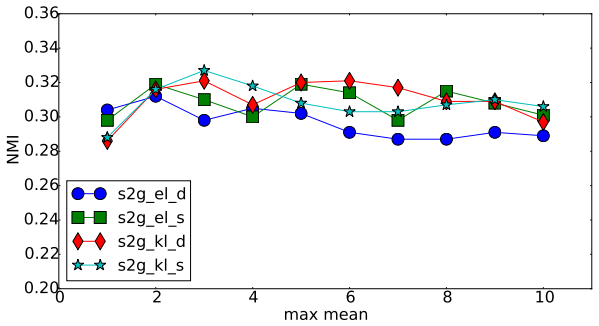


(b) Number of samples per node vs. NMI.

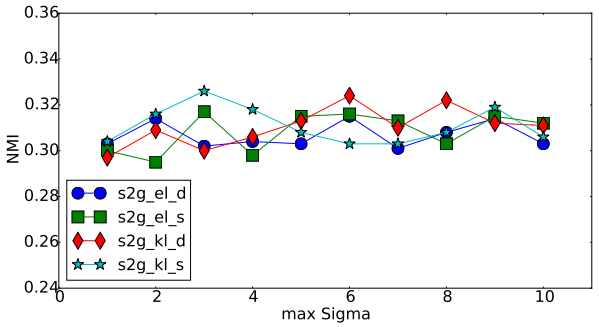


(c) Number of positive/negative nodes per node vs. NMI.

Figure 5.9: Parameter Sensitivity Study.



(a) mean constraint C vs. NMI.



(b) covariance constraint c_{max} vs. NMI.

Figure 5.10: Parameter sensitivity in Gaussian distributions.

Chapter 6

Infinite Motif Blockmodel on Static Graphs

6.1 Introduction

In this chapter, we aim to answer the research question Q2.2 introduced in Section 1:

> Q2.2: Is is feasible to determine the number of roles automatically given a static graph?

Role/block discovery plays an essential role in graph structure mining because roles are representative of the global structures of graphs as introduced in Chapter 1. In practice, the number of roles are unknown but most of previous role discovery studies require the number of roles as input. Therefore, it is meaningful to explore how to determine the number of roles automatically.

Previous work on role discovery can be categorized into two types: graph-based methods and feature-based methods [RA+15, PZFP18]. The most representative graph-based method is stochastic blockmodel [HLL83,ABFX08,KTG+06]. Feature-based methods consist of two steps: feature extraction and role assignment, e.g., RolX [HGER+12]. However, previous studies either relied on first or second-order structural information to group nodes but neglected the higher-order information or required the number of roles/blocks as the input but failed to infer it automatically. Therefore, two challenges remain in role discovery

Table 6.1: Numbers of different types of motifs on the Borgatti-Everett network.

Node ID	1	2	3	4	5	6	7	8	9	10
Type 1	9	9	6	6	6	6	3	3	3	3
Type 2	0	0	1	1	1	1	2	2	2	2

task. The first challenge is how to capture the high-order graph structures. Since role discovery analyzes networks from the global perspective, the first and second-order structural patterns, i.e., edges and shared neighbors, may fail in identifying roles. Another issue in using first and second-order structural patterns is they are incapable of dealing with the sparsity of networks. However, in real-world networks, both the direct edges and shared neighbors are often sparse. The second challenge is how to determine the number of roles/blocks. In the literature, there are two types of methods to select the right number of blocks/roles: (1) nonparametric models and (2) model selection methods. Nonparametric models, e.g., IRM [KTG+06], automatically choose an appropriate number of blocks using stochastic process as the prior that can generate an infinite number of clusters. Most widely used model selection methods include Bayesian information criterion (BIC) [S+78] and minimum description length (MDL) [Ris78]. For instance, MMSB [ABFX08] uses BIC to choose the number of blocks and RolX [HGER+12] selects the number of roles using MDL. To illustrate these challenges, we still use the Borgatti-Everett graph [BE92] in Chapter 1 as a motivating example shown in Figure 6.1.

A motivating example (Borgatti-Everett Network). Figure 6.1 shows the Borgatti-Everett network [BE92] which consists of ten nodes. For simplicity, we focus on undirected graphs and only consider two types of motifs shown in Figure 6.2 in this study. Based on the social theory in roles (a.k.a. posi-

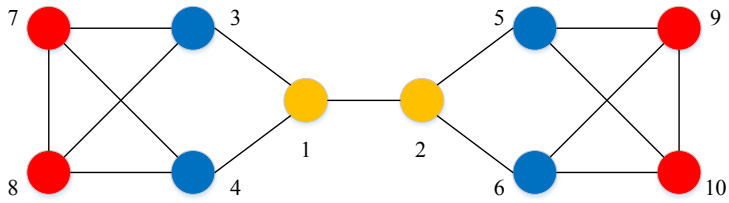


Figure 6.1: Borgatti-Everett graph. All nodes have the same degree and different colors denote different roles/blocks.

tions) [S+78], these nodes can be grouped into three roles (in different colors). These three roles can be interpreted as star (Node 1 and 2 in yellow), periphery (Node 3 - 6 in blue) and clique (Node 7 - 10 in red). Traditional methods based on first and second-order structures have problem clustering these nodes into right roles because: (1) all ten nodes have the same degree but they are in different roles, and (2) some nodes (e.g., Node 7 and 9) have no shared neighbors but they belong to the same role. However, motif-based method can solve this problem effectively. We count the numbers of two different types of motifs (shown in Figure 6.2) for nodes in the Borgatti-Everett network and the statistics is shown in Table 6.1. It is clear that these nodes can be clustered into three groups ({1, 2}, {3, 4, 5, 6} and {7, 8, 9, 10}) since they have exactly the same motif distributions inside each role group. Thus, the failure of traditional methods in role discovery encourages a new model which can take high-order motifs into consideration. Another issue is that in practice, we may not know the number of roles in advance. Thus, it is meaningful to determine the right number of roles automatically by the model itself.

To tackle these two challenges motivated by the above example, we propose a novel generative model named Infinite Motif Stochastic BlockModel (IMM). On the one hand, IMM is a high-order model which takes advantage of the motifs in the generative process. On the other hand, it is a nonparametric Bayesian model which can automatically infer the number of roles from the data. To validate the effectiveness of IMM, we conduct experiments in role discovery on both synthetic and real-world networks. We evaluate discovered roles quantitatively on synthetic networks and visualize the results on real-world networks as a case study.

The contributions of this chapter are summarized as follows:

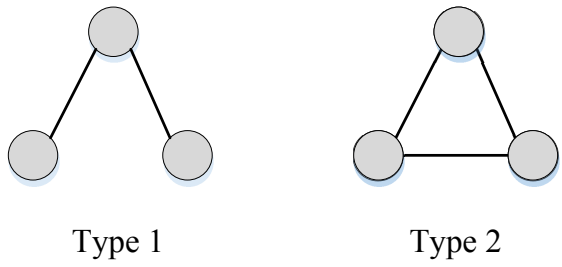


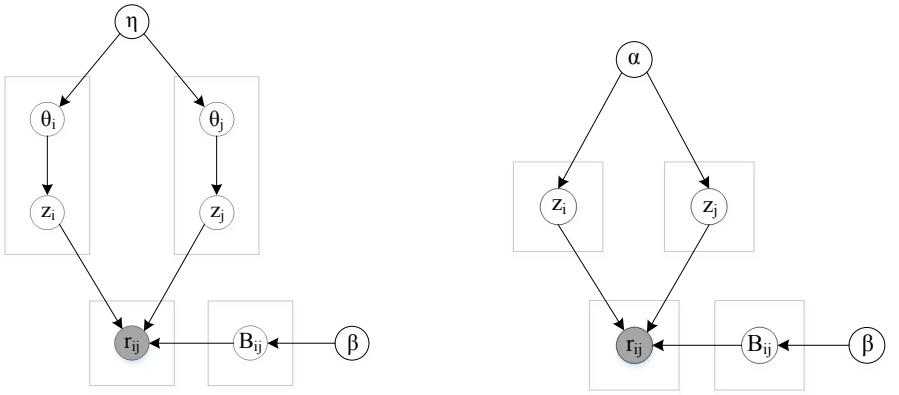
Figure 6.2: Two types of motifs used in this chapter.

- We propose a novel generative model, infinite motif stochastic blockmodel (IMM), to discover roles in networks. IMM is a nonparametric Bayesian model to generate higher-order motif information.
- We derive Gibbs sampling algorithm for model inference to learn the latent variables in IMM.
- The conducted experimental results on both synthetic and real-world networks demonstrate the effectiveness of IMM in role discovery.

The rest of this chapter is organized as follows. Notations and problem formulation are given in Section 6.2. Section 6.3 explains the proposed REACT model. In Section 6.4 we then discuss our experimental study. Finally, in Section 6.5 we draw conclusions and outline directions for future work.

6.2 Notations and Backgrounds

We first summarize the required notations in Table 6.2 and briefly review the related models, then present the problem statement of the motif-based nonparametric model for role discovery.



(a) Graphical representations of MMSB. (b) Graphical representations of IRM.

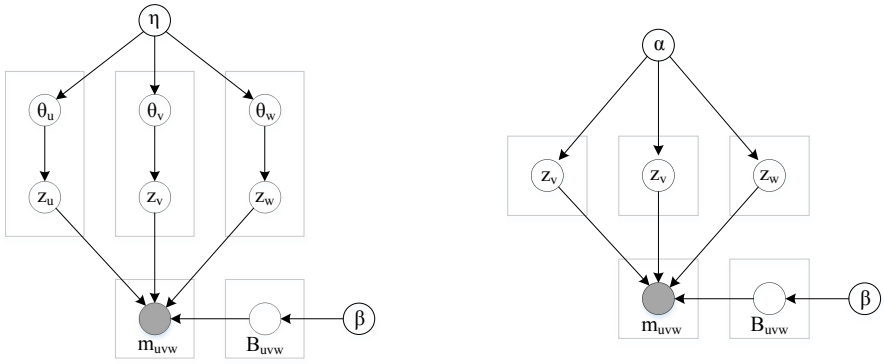
Figure 6.3: Graphical representations of edge-based models.

Stochastic blockmodel (SBM) [HLL83] is the original generative model to detect blocks in networks. SBM partitions nodes in hard clustering and is not flexible to incorporate prior knowledge. To solve these problems, mixed membership stochastic blockmodel (MMSB) [ABFX08] has been proposed where a role distribution for each node will be inferred and the graphical representation of MMSB is shown in Figure 6.3a. Formally, the generative process of MMSB is:

$$\begin{aligned} \theta|\alpha &\sim Dirichlet(\eta) \\ z|\theta &\sim Multinomial(\theta) \\ B_{ij}|\beta^1,\beta^2 &\sim Beta(\beta^1,\beta^2) \\ r_{ij}|z,B &\sim Bernoulli(B(z_i,z_j)) \end{aligned} \tag{6.1}$$

where $a|b \sim Distribution(b)$ means that sampling the variable a from the distribution with parameter b.

However, MMSB requires the number of roles/blocks as the input which may be difficult to obtain in advance in practice. Infinite relational model (IRM) [KTG+06] has been proposed using the Chinese restaurant process (CRP) [P+02] as the prior. As a discrete-time stochastic process, CRP can generate an infinite number of clusters so IRM can infer the right number of roles based on the observed edges. The graphical representation of IRM is shown in Figure 6.3b and



(a) Graphical representations of MMTM.

(b) Graphical representations of our proposed IMM

Figure 6.4: Graphical representations of motif-based models.

the formal generative process is:

$$\begin{aligned} z|\alpha &\sim CRP(\alpha) \\ B_{ij}|\beta^1,\beta^2 &\sim Beta(\beta^1,\beta^2) \\ r_{ij}|z,B &\sim Bernoulli(B(z_i,z_j)) \end{aligned} \tag{6.2}$$

Note that in both MMSB and IRM, the essence is to infer the latent variables, i.e., roles, based on the observed edges between nodes. These two models have been introduced in Chapter 2 and to better compare different Bayesian stochastic models we emphasize them again in this chapter.

Although IRM can infer the role number automatically, it only models the first-order patterns, i.e., edges, in networks. Mixed membership triangular model (MMTM) [YHX13] solved this problem by modeling higher-order motifs and MMTM aims to infer roles from the observed motifs in networks. The graphical representation of MMTM is shown in Figure 6.4a and the formal generative process is:

$$\begin{aligned} \theta|\alpha &\sim Dirichlet(\alpha) \\ z|\theta &\sim Multinomial(\theta) \\ B_{uvw}|\beta &\sim Dirichlet(\beta) \\ m_{uvw}|z,B &\sim Multinomial(B(z_u,z_v,z_w)) \end{aligned} \tag{6.3}$$

Problem Statement Given a network G represented by a motif sequence $M = \{(e_u, e_v, e_w, t)\}$ where each motif is a 4-tuple, $e_u, e_v, e_w \in E$, and $t \in R$ is the type of motif (in this work there are two motif types shown in Figure 6.2), the motif-based nonparametric model for role discovery aims to assign each node to a role and infer the number of roles simultaneously.

6.3 Infinite Motif Stochastic Blockmodel

Infinite Motif Stochastic Blockmodel (IMM) takes motifs as the input which can effectively capture the high-order structural patterns of networks. It also utilizes CRP as the prior which can infer the role numbers automatically. Now we introduce the generative process of IMM and the inference algorithm in detail.

Table 6.2: Summary of the notations.

Symbol	Description
N	Number of nodes.
$n_{i,k}$	Number of times that node i being assigned to role k
m_{uvw}	Motif type for node u, v and w
B	Role compatible matrix
z_i	Role assignment for node i
Z_{-i}	Role assignment for all nodes except node i
θ	Role interaction matrix
α	Hyperparameter of CRP
η	Hyperparameter of Dirichlet distribution
β	Hyperparameter of Beta distribution
Distribution	Description
$Dirichlet(\eta)$	Dirichlet distribution with parameter η
$Multinomial(\theta)$	Multinomial distribution with parameter θ
$Beta(\beta^1, \beta^2)$	Beta distribution with parameter β^1 and β^2
$Bernoulli(\omega)$	Bernoulli distribution with parameter ω
$CRP(\alpha)$	Chinese restaurant process with parameter α

6.3.1 The Generative Model

The graphical representation of IMM is shown in Figure 6.4b and the formal generative process is:

$$\begin{aligned} z|\alpha &\sim CRP(\alpha) \\ B_{ijk}|\beta &\sim Dirichlet(\beta) \\ m_{uvw}|z, B &\sim Multinomial(B(z_u, z_v, z_w)) \end{aligned} \tag{6.4}$$

In detail, in the first line of Eq. (6.4), IMM employs the CRP as the prior for the latent variables, i.e., roles of nodes, so it can generate an infinite number of roles based on the input networks. Similar to MMSB, in the second line the Dirichlet distribution is used as the prior to generate the role interaction B_{ijk} for three nodes which form a predefined motif. In the last line, the observed motif is generated from a multinomial distribution with the role interaction as parameter.

The posterior of Z can be computed as:

$$P(Z|M, \alpha, \beta) \propto P(M|Z)P(Z|\alpha), \tag{6.5}$$

Algorithm 4 Gibbs Sampling Algorithm for IMM

Input: A network G represented by a motif sequence $M = \{(e_u, e_v, e_w, type)\}$, hyperparameter for CRP prior α, and hyperparameter for Dirichlet prior β
Output: Role assignment z and role interaction pattern B
1: Initialize z for each node
2: **while** not converge **do**
3: Update role assignment statistics $n_{i,k}$
4: Sample role for each node by assigning it into existing roles or a new role according to Eq. (6.7)
5: **end while**

where the probability $P(M|Z)$ of generating M is:

$$P(M|Z) \propto \int P(M|Z,B,\beta)P(B|\beta)dB. \tag{6.6}$$

6.3.2 The Inference Algorithm

The aim of inference algorithm for IMM is to infer the latent variable z and B based on the observed motif sequence. In this work, we use Gibbs sampling algorithm to approximate the conditional probability of role assignment since it is a moderately efficient method for Bayesian models. Due to the page limit, we omit the detailed process but list the conditional distribution for sampling as follows:

$$P(z_i = k|M, Z_{-i}, \alpha, \beta) \propto \tag{6.7}$$
$$\begin{cases} \frac{\alpha}{N-1+\alpha} \prod_{u,v,w} \Gamma(m_{uvw} + \beta) & \text{for existing cluster} \\ \frac{n_{i,k}}{N-1+\alpha} \prod_{u,v,w} \Gamma(m_{uvw} + \beta) & \text{for new cluster} \end{cases}$$

where $\Gamma(n) = (n-1)!$ is the Gamma function. The Gibbs sampling algorithm for IMM inference is shown in Algorithm 4.

6.4 Experiments

6.4.1 Experimental Setup

We evaluate our model on role discovery using synthetic and real-world networks. For synthetic data, we generate two networks using SBM [HLL83] and

they are visualized in Figure 6.5. Each network consists of 100 nodes and these nodes belong to 4 roles. For real-world networks, we use Zachary karate and Les Misérables network. IMM is compared to some baseline models including MMSB, MMTM and IRM. We use the same (hyper)parameters in all models: CRP parameter $\alpha = 5$, Dirichlet parameter $\eta = 4$ and Beta parameter $\beta = 7$. We leave the hyperparameter selection as our future work.

6.4.2 Evaluation Metrics

To evaluate the experimental results, we use purity and normalized mutual information (NMI) as the evaluation metrics. These metrics are widely used in the evaluation of clustering methods with ground-truth labels. Formal definitions of both evaluation metrics have been introduced in Section 3.6.1 of Chapter 3.

6.4.3 Results

Synthetic networks

The results of role discovery on two synthetic networks are shown in Table 6.3. From these results, some conclusions can be drawn:

- IMM outperform all other models in detecting roles which indicates the effectiveness of our model.
- Motif-based models perform better than edge-based models which demonstrate that high-order motifs can better capture the global role information.
- Nonparametric models (IMM and IRM) can effectively detect the number of roles which is more meaningful in practice when the role number is unknown.

Real-world networks

The visualization of roles on two real-world networks are shown in Figure 6.6 and 6.7. These results, demonstrate the effectiveness of IMM in identifying roles. In detail,

- *Zachary network*: The two blue nodes are stars for the left community, yellow and light blue are stars for the right community[1], and red nodes

[1] We have prior knowledge on Zachary karate network that it consists of two communities.

Table 6.3: Experimental results on the synthetic networks.

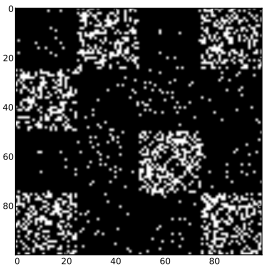
datasets	SYN 1		SYN 2	
metrics	Purity	NMI	Purity	NMI
MMSB [ABFX08]	0.45	0.2603	0.33	0.1681
IRM [KTG+06]	0.45	0.2452	0.44	0.1758
MMTM [YHX13]	0.60	0.3964	0.48	**0.2453**
IMM (our model)	**0.65**	**0.4204**	**0.51**	0.2242

are peripheries.

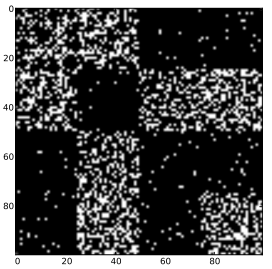
- *Les Misérables network*: The blue nodes are stars including dark and sky blue nodes, red nodes are cliques, orange nodes are peripheries, and yellow nodes are bridges to link stars and followers.

6.5 Concluding Remarks

In this chapter, we attempt to find the answer to the research question *Is is feasible to determine the number of roles automatically given a static graph?* raised in Section 6.1. To answer this question, we proposed a novel generative model, infinite motif stochastic blockmodel (IMM), for role discovery. IMM is advantageous in two aspects: (1) it models higher-order motifs to infer the roles which



(a) Diagonal block structures.



(b) Non-diagonal block structures.

Figure 6.5: Visualization of two synthetic networks.

can effectively capture the global structural information of networks, and (2) it is a nonparametric Bayesian model to infer the number of roles automatically which is more suitable in real-world network analytics. We evaluated IMM in

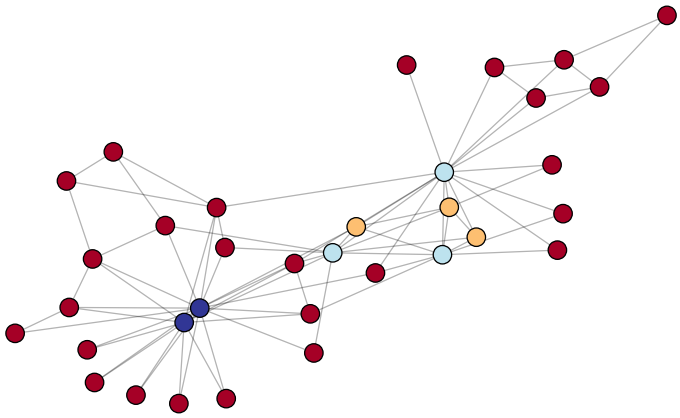


Figure 6.6: Roles on Zachary network.

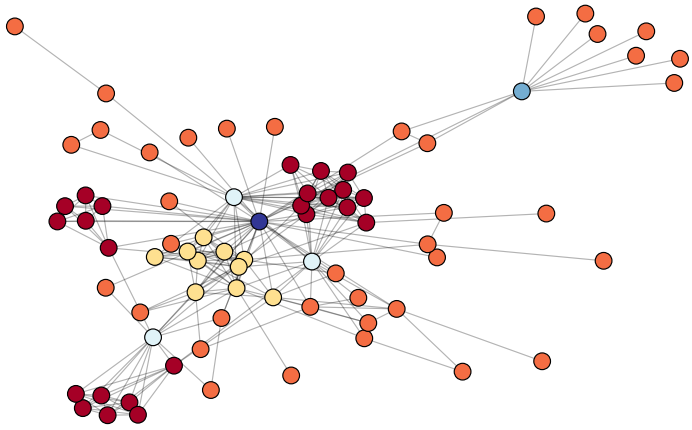


Figure 6.7: Roles on Les Misérables network.

role discovery compared to state-of-the-art blockmodels and the results indicate the effectiveness of IMM. In future work we will explore scalable inference algorithm for IMM, e.g., collapsed variational Bayesian inference method, and test our model on larger-scale networks. We will also extend our method to different types of networks, e.g., dynamic networks.

Chapter 7

Role Discovery on Dynamic Graphs

7.1 Introduction

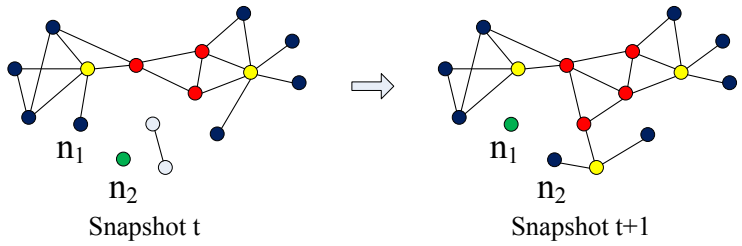
In this chapter, we aim to answer the research question Q2.3 introduced in Section 1:

> Q2.3: How can we effectively discover roles on dynamic graphs by capturing the global structures and dynamics?

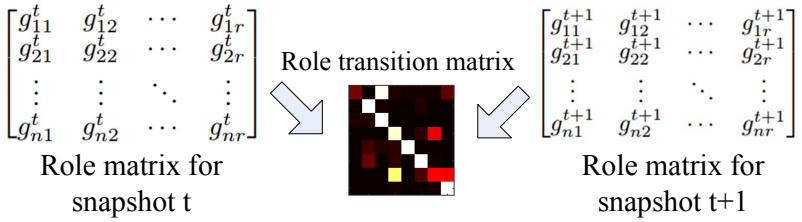
Existing role discovery methods focus predominantly on static SNs. For example, non-negative matrix factorization (NMF) based methods, such as RolX [HGER+12] and GLRD [GERD13], cluster a node-feature matrix to discover roles. Stochastic blockmodels employ Bayesian methods for role discovery [ABFX09, FSX09]. In the real world, however, *dynamic* SNs are ubiquitous and structures of the networks will change over time. State-of-the-art static methods are not easy to extend to dynamic SNs directly. Few attempts have been made to discover roles and analyze role transitions in dynamic SNs. These attempts either neglect role transition analysis or perform role discovery and role transition learning separately. Evolutionary role clustering method [CRPS15] integrates temporal information into a weighting function for user similarity and clustering. However, role transitions have not been analyzed in this chapter. DBMM [RGNH12, RGNH13] directly uses RolX to discover roles in each SN snapshot and then

analyzes the role transition based on obtained node-role matrices. Although role transitions are analyzed in this model, role discovery and role transition analytics are two separate steps, i.e., role transition information can be learned only after the role discovery process, as shown in Figure 7.1(b). As we show in this section, this strategy is inefficient and unstable in practice. These problems also remain in other studies for dynamic role discovery [LGD+13, ATRZ15]. A summary comparison of the state of the art in role discovery methods can be found in Table 7.1.

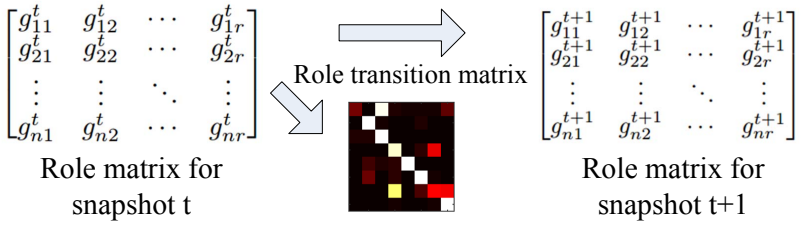
To sum up, there are two issues in previous work: (1) *lack of role transition*



(a) Example of roles and role transitions in SNs.



(b) Role detection and role transition analysis in previous studies.



(c) Role detection and role transition analysis in DyNMF.

Figure 7.1: Examples of roles and role analytics in previous methods and DyNMF.

analysis; and (2) *inefficiency in role transition analysis*. To tackle these issues, in this chapter we propose a new dynamic non-negative matrix factorization (DyNMF) approach. DyNMF is a unified model to discover role and role transition simultaneously in dynamic SNs. An illustration of DyNMF is shown in Figure 7.1(c) where we can simultaneously obtain the role matrix of snapshot $t+1$ and the role transition from snapshot t to $t+1$ by using information in snapshot $t+1$ and the role matrix of snapshot t. In particular, DyNMF can solve the two issues effectively and efficiently:

- For the issue of *lack of role transition analysis*, DyNMF explicitly introduces a *role transition matrix* for role transition, where roles and role transitions are modeled in a unified framework. Current and historical views are combined for role analytics. The current view follows RolX to discover roles in the current SN snapshot, while the historical view learns role transitions using past role information and the current SN snapshot.

- For the issue of *inefficiency in role transition analysis*, DyNMF, as a unified model, supports the simultaneous discovery of both roles and role transitions. In particular, it requires only one pass over the data to obtain roles and role transitions compared with previous studies.

DyNMF is also advantageous in a further aspect: by combining current and historical views, we can regularize the roles by capturing the temporal smoothness of roles and also reduce uncertainties and inconsistencies between snapshots. Thus, temporal information is better explored compared to [CRPS15] and the discovered roles are more stable compared to DBMM.

All of these advantages of DyNMF are validated through extensive experiments on both synthetic and real-world SNs. The results validate DyNMF is advantageous in discovering roles, capturing role transitions, and predicting roles.

7.2 Role Analytics Using DyNMF

We first briefly revisit the original RolX approach to static role discovery using NMF [HGER+12]. We then introduce DyNMF, our new approach for dynamic role analytics in dynamic SNs.

7.2.1 NMF based Role Discovery

Non-negative matrix factorization (NMF) [LS01] is a popular model in multivariate analysis and linear algebra where a matrix is factorized into two matrices, with the property that all three matrices have no negative elements. There are several advantages in NMF including ease of implementing inference and ease of interpreting results. Hence, this model is widely used in text mining.

RolX [HGER+12] is the first method to discover roles using NMF. Given a node-feature matrix $V_{n\times l}$, where n and l is the number of nodes and features respectively, the idea of RolX is to generate a rank r approximation $GF \approx V$ where r is the number of roles, matrix $G_{n\times r}$ denotes the nodes' membership and matrix $F_{r\times l}$ represents the association of roles and features. Thus, the problem of role discovery is to seek two low rank matrices G and F to satisfy:

$$\min_{G,F} \|V - GF\|^2, \quad s.t.\ G \geq 0, F \geq 0 \tag{7.1}$$

where $\|\cdot\|$ is the Frobenius norm. The non-negativity constraint in Eq. (1) makes the representation of the original data easier to interpret and more semantically meaningful compared with other factorization methods, e.g., SVD and PCA [LS01]. Using multiplicative update rules, the solution for Eq. (1) is shown as follows:

$$G \leftarrow G \circ \frac{VF^T}{GFF^T}, \quad F \leftarrow F \circ \frac{G^T V}{G^T GF} \tag{7.2}$$

where $\circ$ denotes the element-wise product. By introducing notions of sparsity, diversity and alternativeness, RolX was extended with more guidance used as constraints in the matrix factorization for role discovery [GERD13]. However, these methods focus only on static SNs and it is non-trivial to extend them to dynamic SNs directly.

7.2.2 DyNMF Approach

To solve problems in previous studies on dynamic role discovery, i.e., (1) requiring separate analysis for role transition after discovering roles [RGNH12, RGNH13] and (2) no role transition analysis [ATRZ15, CRPS15], we propose a novel dynamic non-negative matrix factorization (DyNMF) approach for role discovery and role transition analysis simultaneously.

DyNMF factorizes the current node-feature matrix from two different views: *current view* and *historical view*. Figure 7.2 is to illustrate the proposed DyNMF approach. The first view is derived from the current SN snapshot same to RolX

introduced in Section 7.2.1. Formally, the current view based factorization component for snapshot t is:

$$\min_{G^{(t)},F^{(t)}} \|V^{(t)} - G^{(t)}F^{(t)}\|^2 \tag{7.3}$$

where $V^{(t)}$, $G^{(t)}$ and $F^{(t)}$ are the node-feature, node-role, and role-feature matrix in snapshot t, respectively.

The second view captures historical information by looking at the roles in the previous SN snapshot. We explicitly introduce a role transition matrix M to capture the transitions between previous roles and current roles. Formally, the historical view based factorization component for snapshot t is:

$$\min_{F^{(t)},M^{(t)}} \|V^{(t)} - G^{(t-1)}M^{(t)}F^{(t)}\|^2 \tag{7.4}$$

where $V^{(t)}$ and $F^{(t)}$ have the same meaning in Eq. (7.3), $M^{(t)}$ denotes the role transition matrix from snapshot $t-1$ to t and $G^{(t-1)}$ denote the node-role matrix from snapshot $t-1$.

By combining these two views of factorization components, we can make good use of the temporal information to discover roles and learn role transitions

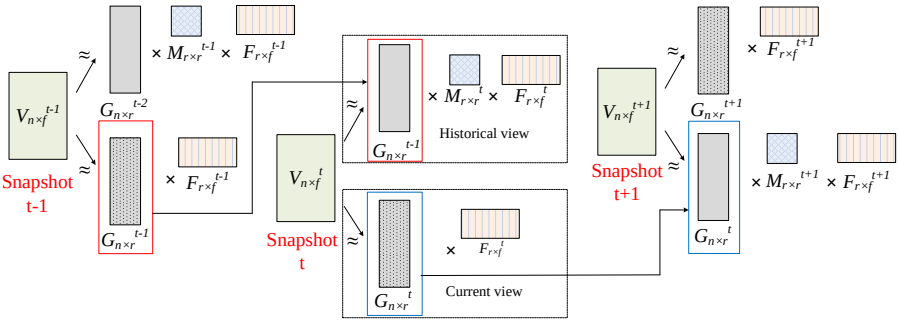


Figure 7.2: Graphical representation of DyNMF approach which consists of current view and historical view. To learn the node-role matrix $G^{(t)}$, role transition $M^{(t)}$ and associate matrix $F^{(t)}$ in snapshot t, we take node-feature matrix $V^{(t)}$ in snapshot t and node-role matrix $G^{(t-1)}$ from snapshot $t-1$ as the input.

simultaneously. We formulate the objective function of DyNMF as follows:

$$\min_{G^{(t)},F^{(t)},M^{(t)}} L = \min_{G^{(t)},F^{(t)},M^{(t)}} \|V^{(t)} - G^{(t)}F^{(t)}\|^2 + \|V^{(t)} - G^{(t-1)}M^{(t)}F^{(t)}\|^2 \quad (7.5)$$
$$s.t.\ G^{(t)} \geq 0, F^{(t)} \geq 0, M^{(t)} \geq 0.$$

Note that we use only the first-order assumption in Eq. (7.5). One can extend it to any higher-order version by setting $k > 1$, i.e., the roles depend on more than one previous snapshots:

$$\min_{G^{(t)},F^{(t)},M_i^{(t)},...,M_k^{(t)}} \|V^{(t)} - G^{(t)}F^{(t)}\|^2 \quad (7.6)$$
$$+ \|V^{(t)} - G^{(t-1)}M_1^{(t)}F^{(t)}\|^2 + ... + \|V^{(t)} - G^{(t-k)}M_k^{(t)}F^{(t)}\|^2$$
$$s.t.\ G^{(t)} \geq 0, F^{(t)} \geq 0, M_i^{(t)} \geq 0, 1 \leq i \leq k$$

where $M_k^{(t)}$ is the transition matrix to capture the role transition from snapshot $t-k$ to t. However, higher-order extension suffers from two limitations: (1) from the *computational* perspective it is more complex since there are more parameters to learn, i.e., $M_k^{(t)}$; (2) from the *empirical* perspective, we have observed that higher-order extensions have not improved performance compared to DyNMF (in Section 7.3.2). Therefore, unless explicitly stated otherwise, we only focus on the first-order version, i.e., Eq. (8.8), in following sections.

The objective function in Eq. (7.5) (or Eq. (7.6)) is not convex for all parameters $G^{(t)}$, $F^{(t)}$ and $M^{(t)}$ simultaneously. We use the multiplicative update rules to solve this optimization problem due to its good compromise between speed and ease of implementation [LS01]. The optimization is done by iterating the three following steps until the convergence (or the number of iteration exceeds a given threshold): (1) fix matrices $F^{(t)}$ and $M^{(t)}$ to update $G^{(t)}$ (2) fix matrices $G^{(t)}$ and $F^{(t)}$ to update $M^{(t)}$ and (3) fix matrices $G^{(t)}$ and $M^{(t)}$ to update $F^{(t)}$ (Algorithm 5). Formally, using the Karush-Kuhn-Tucker (KKT) conditions to solve the objective function in Eq. (7.5), we get the update rules:

$$G^{(t)} \leftarrow G^{(t)} \circ \frac{V^{(t)}F^{(t)T}}{G^{(t)}F^{(t)}F^{(t)T}} \quad (7.7)$$

$$M^{(t)} \leftarrow M^{(t)} \circ \frac{G^{(t-1)T}V^{(t)}F^{(t)T}}{G^{(t-1)T}G^{(t-1)}M^{(t)}F^{(t)}F^{(t)T}} \quad (7.8)$$

$$F^{(t)} \leftarrow F^{(t)} \circ \frac{G^{(t)T}V^{(t)} + M^{(t)T}G^{(t-1)T}V^{(t)}}{G^{(t)T}G^{(t)}F^{(t)} + M^{(t)T}G^{(t-1)T}G^{(t-1)}M^{(t)}F^{(t)}} \quad (7.9)$$

Algorithm 5 Optimization Algorithm

Input: previous node-role matrix $G^{(t-1)}$, node-feature matrix $V^{(t)}$
Output: node-role matrix $G^{(t)}$, transition matrix $M^{(t)}$
  Initialize $G^{(t)}$, $M^{(t)}$ and $F^{(t)}$ with random non-negative values
  while not converge **do**
    Update $G^{(t)}$ according to Eq. (7.7)
    Update $M^{(t)}$ according to Eq. (7.8)
    Update $F^{(t)}$ according to Eq. (7.9)
  end while

Complexity analysis of DyNMF. For simplicity, given two matrices $M_{n\times r}$ and $N_{r\times l}$, the computational complexity of the multiplication of M and N is $O(nrl)$. The complexity of Algorithm 1 is $O(nrl+nr^2+lr^2)$ and therefore by considering the number of iteration i the number of snapshots t the complexity is $O(ti(nrl+nr^2+lr^2))$. The complexity of feature extraction and model selection, discussed below, can be referred to [HGER+12].

7.2.3 Model Selection

In this section, we view role as a latent class and use a model selection method to determine the number of roles in the discovery process because this method has better generalization in SNs [HGER+12]. We adopt the model selection method proposed in [HGER+12]. It uses the Minimum Description Length (MDL) [Ris78] to decide on the number of roles. The appropriate number of roles r is the one that minimizes the description length $\mathscr{L}$, defined as sum of the coding cost $\mathscr{M}$ and the model description cost $\mathscr{E}$. $\mathscr{M}$ is defined as $br(n+l)$, where b is the bits used for each element. $\mathscr{E}$ is defined as sum of the KL divergence based errors from current view $\mathscr{E}_1$ and historical view $\mathscr{E}_2$:

$$\mathscr{E}_1 = \sum_t \sum_{i,j} \left(V_{i,j}^{(t)} \circ \log \frac{V_{i,j}^{(t)}}{U_{i,j}^{(t)}} - V_{i,j}^{(t)} + U_{i,j}^{(t)} \right) \tag{7.10}$$

$$\mathscr{E}_2 = \sum_t \sum_{i,j} (V_{i,j}^{(t)} \circ \log \frac{V_{i,j}^{(t)}}{W_{i,j}^{(t)}} - V_{i,j}^{(t)} + W_{i,j}^{(t)}) \tag{7.11}$$

where $U^{(t)} = G^{(t)}F^{(t)}$ and $W^{(t)} = G^{(t-1)}M^{(t)}F^{(t)}$. Note that to calculate $\mathscr{E}_1$ and $\mathscr{E}_2$, we sum the errors of all snapshots. We choose the optimal number of roles with

the minimal description length $\mathscr{L}$ empirically.

7.2.4 Feature Extraction

In previous studies on role discovery, a variety of feature extraction methods have been employed. In [HGER+12], the features consist of local and egonet properties based on counts of links and the egonet-based properties generated in a recursive fashion [HGL+11]. In [ZWY+13], five types of network properties, i.e., homophily, triadic closure, reachability, embeddedness and structural holes, have been used as the features to infer the social roles in the SNs. The structural properties of nodes in the graph such as clustering coefficient and the locality index, have been considered as node features in [CRPS15]. In order to compare our proposed method with the original NMF-based role discovery method, we use the same feature extraction method as RolX [HGER+12] and DBMM [RGNH13] in this chapter.

7.3 Experimental Study

7.3.1 Settings

To validate the advantages of our proposed DyNMF for role discovery and role transition learning, we conduct experiments on one synthetic data set and four real-world data sets[1]. A summary of these data sets is shown in Table 7.2.

For the synthetic data set, we generate a series of graph snapshots with four roles, i.e., star-center nodes, star-edge nodes, bridges and cliques. In each snapshot, based on a fixed change rate, part of nodes will change to other roles randomly. As we have ground-truth labels of roles, we can use traditional clustering evaluation metrics, e.g., Normalized Mutual Information (NMI), to verify the experimental results. To further examine the sensitivity of DyNMF towards role changes in SNs, the change rate is set to vary from 0% to 25% and the performance will be compared. For the real-world data sets, there are no ground-truth labels of roles, so the numbers of roles shown in Table 7.2 are the “best" numbers of roles determined by the method introduced in Section 7.2.3. To evaluate the performance, *goodness-of-fit index* [WF94] is applied.

Our experimental study is aimed to analyze the performance of DyNMF on three analytics tasks:

[1] networkrepository.com/index.php

- *Role discovery analysis*. This task aims to analyze the performance of role discovery using DyNMF quantitatively. We first use *NMI* to evaluate DyNMF on the synthetic data set. Then we examine the sensitivity of DyNMF towards role changes. We also use *goodness-of-fit index* to measure DyNMF on real-world SNs. Furthermore we compare models with different orders.
- *Role transition analysis*. The goal of this task is to verify the effectiveness and stability of the role transition learned by DyNMF. In particular, we calculate the traces of the normalized role transition matrices learned by DyNMF and DBMM respectively for comparison.
- *Role prediction analysis*. The evaluation of the prediction ability is another way to validate the effectiveness of the role transition. In this experiment, we propose two strategies to predict nodes' roles using the transition matrices learned by DyNMF.

7.3.2 Role Discovery Analysis

NMI on Synthetic Data Set To quantitatively evaluate the performance of DyNMF, we use Normalized Mutual Information (NMI) as the measurement on the synthetic data set. NMI is obtained by dividing the mutual information by the arithmetic average of the entropy of obtained cluster $\mathscr{C}$ and ground-truth cluster $\mathscr{D}$. We compare three types of methods: feature-based methods in social science, graph-based methods and feature-based methods in data mining. Note that all the feature-based methods use the same features extracted by the method introduced in Section 7.2.4 and graph-based methods use the adjacency matrix from the graph as the input.

The NMI results of different methods are shown in Table 7.3. Note that these results are based on 5% role change rate. From the results, it can be observed that: (1) DyNMF outperforms all the other methods. It indicates the advantage of DyNMF in role discovery by explicitly leveraging the temporal smoothness in role transition. (2) Feature-based methods in data mining, i.e., RolX and DyNMF, perform better than other methods. This result demonstrate that latent models like NMF are better choices in role discovery.

Smoothness Experiment on Synthetic Data Set We investigate the influence of smoothness in role transition and sensitivity of DyNMF towards role changes in SNs. The change rate between two consecutive snapshots is set to vary from 0% to 25% and the NMIs of DyNMF and RolX are shown in Figure 7.3.

This result shows the limitation of our proposed DyNMF in the condition that the assumption of smoothness in role transition does not hold, i.e., the change rate becomes larger. However, this limitation does not influence the performance of DyNMF on real-world data sets for role discovery and role transition analysis. More results and discussion are shown in following experiments.

***Goodness-of-fit indices* on Real-world Data Sets** Due to lack of ground-truth roles in real-world data sets, evaluation of role discovery is challenging in practice. Based on structural equivalence [LW71], researchers in sociology have applied *goodness-of-fit indices* to measure how well the representation of roles and the relations among these roles fit a given SN [WF94]. In *goodness-of-fit indices*, it is assumed that the output of a role discovery method is an optimal model, and nodes belonging to the same role are predicted to be perfectly structurally equivalent. For more details about *goodness-of-fit indices*, please refer to [WF94].

To validate the effectiveness of DyNMF, we compare it with some baselines including K-means, Agglomerative clustering, Spectral clustering and RolX. The comparison of average *goodness-of-fit index* over all snapshots is shown in Table 7.4. From the results, it can be observed that DyNMF outperforms other methods except on Slashdot. This may result from the extreme sparsity of Slashdot.

Higher-order Comparison on Real-world Data Set We validate the rationality and effectiveness of the first-order assumption empirically introduced in Eq. (7.6). We compare the *goodness-of-fit index* of DyNMF on Enron and Reality with second-order and third-order versions. The second-order DyNMF looks two steps back and uses role information from snapshot $t-1$ and $t-2$ to detect roles in snapshot t, i.e., $k=2$ in Eq. (7.6). Similarly, in third-order version, $k=3$. The results are shown in Table 7.5. It can be observed that higher order models

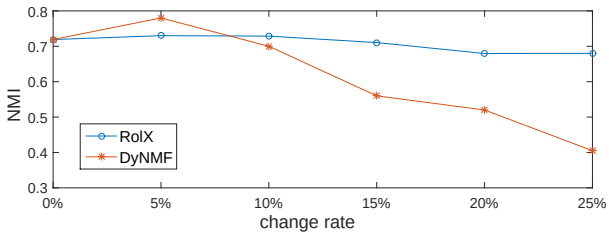


Figure 7.3: The influence of change rates on DyNMF and RolX.

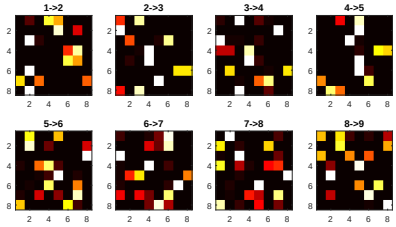
have similar performance with first-order model, i.e., DyNMF, but increase the computational complexity due to more parameters, i.e., more transition matrices, to learn.

7.3.3 Role Transition Analysis

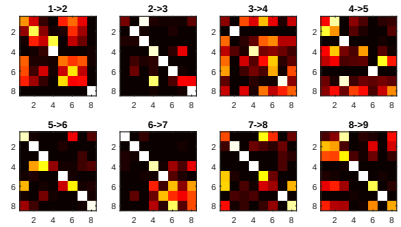
We use DBMM as a competitor approach to DyNMF in our analysis of the role transition stability. Since there is no constraint in the transition matrix M in DBMM, in order to compare DyNMF with DBMM, we also follow that work not adding constraint to M. One could easily add different types of constraint in M, e.g., orthogonal. As the values of elements in transition matrices between different snapshots may be in different ranges, we normalize each M by row to make it be a Markov matrix $\tilde{M}$, i.e., $\forall\ i,\ \sum_j \tilde{M}_{ij} = 1$.

We use DBMM as a competitor approach to DyNMF in our analysis of the role transition stability. Since there is no constraint in the transition matrix M in DBMM, in order to compare DyNMF with DBMM, we also follow that work not adding constraint to M. One could easily add different types of constraint in M, e.g., orthogonal, and we leave these variants to future work. As the values of elements in Ms between different snapshots may be in different ranges, we normalize each M by row to make them to be a Markov matrix $\tilde{M}$, i.e., $\forall\ i,\ \sum_j \tilde{M}_{ij} = 1$, for analysis. Two methods are employed for role transition analysis: visualization of transitions and trace of transitions.

Visualization of transitions. Based on normalized transition matrices, we visualize the role transitions in Enron data set generated by DBMM (Figure 7.4a) and DyNMF (Figure 7.4b). It can be observed that the transitions are erratic using DBMM while the diagonal values are larger using DyNMF. It indicates the



(a) Transition matrices using DBMM.



(b) Transition matrices using DyNMF.

Figure 7.4: Community and role interaction matrices in Email network.

roles detected by DyNMF are more stable than DBMM. This is consistent with the nature of social networks where the roles of nodes will not change abruptly.

Trace of transitions. We calculate the traces of the normalized role transition matrices, i.e., $Tr(\tilde{M}) = \sum_i \tilde{M}_{ii}$, to further validate this statement. Figure 7.5 shows the traces of role transition matrices using DyNMF and DBMM on the four data sets, respectively. Higher trace value means less change of roles. From the results, some conclusions can be drawn:

- DyNMF can obtain higher trace in almost all the snapshots of the SNs.

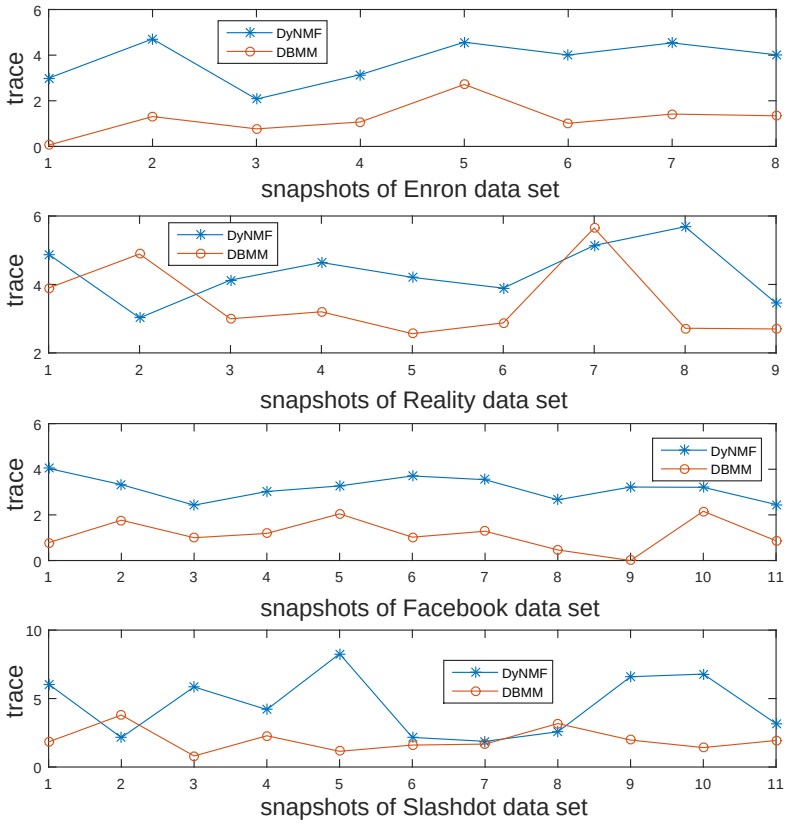


Figure 7.5: Traces of transition matrices using DBMM and DyNMF.

This result demonstrates the stability of the role transitions obtained by DyNMF.

- The stability of transition is more obvious in Enron and Facebook. The reason may be the characteristics of these SNs. Intuitively, the roles (positions) of employees in a company, e.g., Enron, rarely change abruptly and few users in a SN, e.g., Facebook, can become cyberstars suddenly. Therefore, the changes of roles are more smooth in these data sets. In addition, the stability also depends on the temporal granularity of snapshots.

- It is interesting that the stability is more obvious in Enron and Facebook data sets. The reason may be the characteristics of different data sets. It is intuitive that the roles (positions or titles) of employees in a company rarely change abruptly and there are few users in a SN can become cyberstars suddenly. Therefore, the changes of roles are more smooth in these data sets. In addition, the stability also depends on the temporal granularity of snapshots. Moreover, the finding of role stability is consistent with the discovery in [RDJ16] which claims there are six roles that are persistent in social networks: *popular*, *friendly*, *explorer*, *reciprocated*, *community member*, and *active-community member*.

7.3.4 Role Prediction Analysis

We formulate the goal to predict the roles $\hat{G}^{(t+1)}$ in $t+1$ snapshot using the role membership matrix $G^{(t)}$ in t snapshot and the transition matrices $M^{(i)}$ $(0 < i \leq t)$. Note that these transition matrices are also normalized. To analyze the prediction ability, we design two strategies:

- *average strategy (AvgDyNMF)*. This strategy is based on the *global consistency* and assumes that the role transitions are smooth in all the snapshots, i.e., $\hat{G}^{(t+1)} = G^{(t)} \sum_i^t M^{(i)} / t$.

- *previous strategy (PrevDyNMF)*. This strategy is based on the *local consistency* and assumes that the role transition is smooth between two consecutive snapshots, i.e., $\hat{G}^{(t+1)} = G^{(t)} M^{(t)}$.

We report the squared Frobenius norm of the prediction error, i.e., $\|G^{(t+1)} - \hat{G}^{(t+1)}\|_F^2$. Note that there is no ground-truth role indicator $G^{(t+1)}$ and therefore we use the node-role matrices detected by RolX in each snapshot as the golden standard which is similar to the experiments in [RGNH13]. The prediction

performance results are shown in Figure 7.6. To validate the effectiveness of DyNMF, we compare its performance with DBMM also using these two strategies.

In fact, DBMM learns the transition directly from the discovered roles, i.e., obtaining $M^{(t)}$ by minimizing $\|G^{(t+1)} - G^{(t)}M^{(t)}\|_F^2$ and intuitively calculating $G^{(t)}M^{(t)}$ to predict $\hat{G}^{(t+1)}$ will be a good approximation of $G^{(t+1)}$. Thus, this role prediction analysis does not aim to beat DBMM but to demonstrate that by learning role and transition simultaneously we can still obtain satisfactory results. Some conclusions can be drawn from these results:

- Our proposed DyNMF can effectively predict the roles based on the learned transition matrices since both the errors and the trends of curves for DyNMF are similar to those of DBMM on all the data sets.
- The *average strategy* performs better than the *previous strategy* on all the data sets. It indicates that the global consistency plays a more important role in the dynamics of the networks and all nodes exhibit the average behavior of networks. This conclusion is consistent with [RGNH13].
- The gaps between *average strategy* and *previous strategy*, e.g., snapshot 8 in Enron, snapshot 5 in Reality, snapshot 4 in Facebook and snapshot 10 in Slashdot, reflect that there are more role changes happened. DyNMF can effectively handle such changes because in *average strategy* DyNMF performs equally well as DBMM and in *previous strategy* DyNMF outperforms DBMM in all data sets.

7.4 Conclusions

The research question we studied in this chapter is *How can we effectively discover roles on dynamic graphs by capturing the global structures and dynamics?* To explore the answer to this question, we proposed DyNMF, a novel dynamic non-negative matrix factorization approach to discover roles and role transitions simultaneously in dynamic SNs. Current and historical views have been combined for the node-feature matrix factorization. The current view is based on structural information in the current snapshot and the historical view captures the correlation between previous roles and current roles using role transition matrices. We conducted comprehensive experiments on both synthetic and real-world data sets to validate the performance of DyNMF in role discovery

and role transition learning. We analyzed the experimental results from three aspects including role discovery, role transition, and role prediction. The results indicate the effectiveness of our proposed method for the challenging problem of dynamic role analytics.

For further study, first towards improving performance, it is interesting to study the impact of constraints on role indicator matrices. A second area for improvement is to utilize faster and more scalable matrix factorization methods.

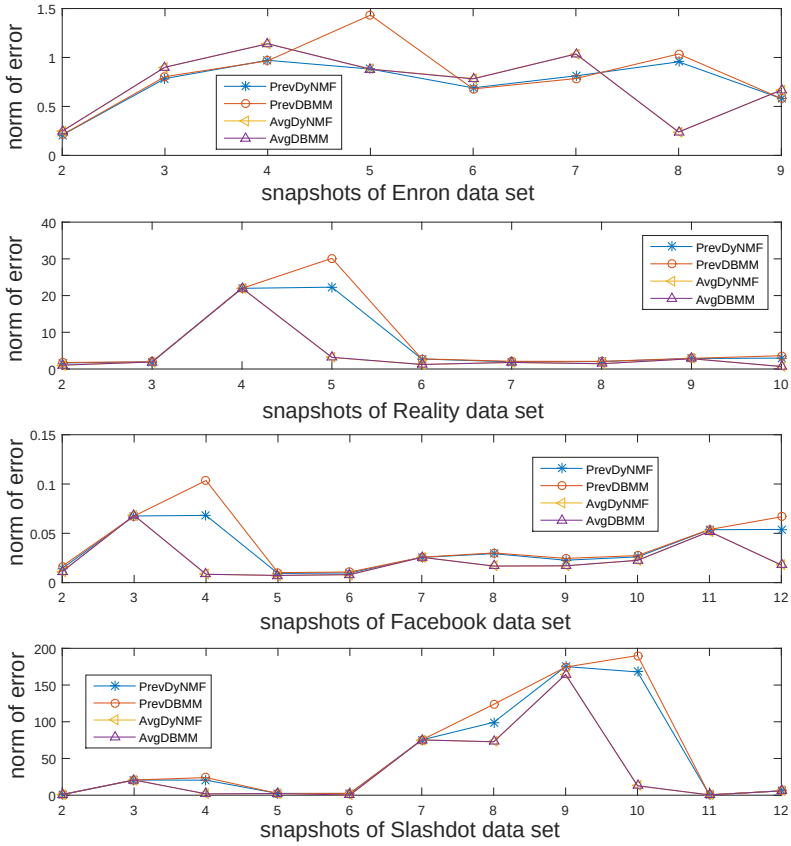


Figure 7.6: Role prediction using DBMM and DyNMF with average and previous strategies.

A third area for future exploration is to exploit other feature extraction methods which can capture temporal information.

Methods	Unsupervised	Dynamic	Transition	Role Definition
RolX [HGER+12]	√	-	-	role explanation after role discovery
GLRD [GERD13]	√	-	-	role explanation after role discovery
DBMM [RGNH13]	√	√	√	role explanation after role discovery
LAP [LGD+13]	-	√	-	predefined roles based on data sets
ERM [CRPS15]	√	√	-	predefined roles based on influence and blockage
SSRM [ATRZ15]	√	√	-	predefined roles (e.g., leader, mediator)
DyNMF (our method)	√	√	√	role explanation after role discovery

Table 7.1: Comparison of role discovery methods.

Data set	# Nodes	# Edges	# Roles	# Snapshots
Synthetic	100	1729	4	8
Enron	147	1666	8	9
Reality	6809	9467	11	10
Facebook	44416	196414	12	12
Slashdot	51068	130324	11	12

Table 7.2: Summary of data sets used in the experiments.

	Methods	NMI
Feature-based methods in social science	K-means	0.4831
	Agglomerative	0.5116
Graph-based methods	Spectral	0.6233
	MMSB	0.6382
NMF-based methods in data mining	RolX	0.7220
	DyNMF	**0.7816**

Table 7.3: Comparison of role discovery performance using *NMI*. The highest value is in bold.

Methods	Average *goodness-of-fit index*			
	Enron	Reality	Facebook	Slashdot
K-means	1.7722	1.5950	0.1802	0.0496
Agglomerative	0.8750	0.7875	0.0461	0.0888
Spectral	1.4235	0.9945	0.0864	0.1025
MMSB	1.0425	0.7596	0.1174	0.0950
RolX	1.1873	1.0685	0.1168	**0.0314**
DyNMF	**0.5788**	**0.6051**	**0.0393**	0.0345

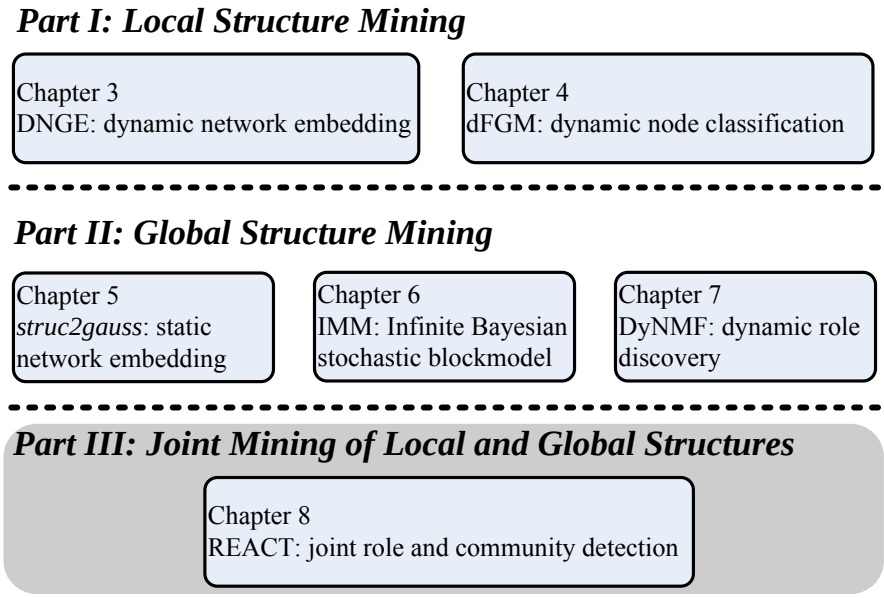
Table 7.4: Comparison of role discovery performance using *goodness-of-fit index*. The lowest error in each data set is in bold.

Method	*goodness-of-fit index*		Running time	
	Enron	Reality	Enron	Reality
first-order	0.5788	0.6051	1.6ms	307.1s
second-order	0.5758	0.6344	1.8ms	370.5s
third-order	0.5818	0.6129	2.3ms	414.5s

Table 7.5: *Goodness-of-fit index* and running time vs. orders.

Part III

Joint Mining of Local and Global Structures



Chapter 8 We present a novel joint approach REACT to detect roles and communities simultaneously. REACT combines role discovery and community detection using two nonnegative matrix tri-factorization components and integrates the diversity relation between roles and communities using the $L_{2,1}$ norm; and is capable of automatically determining the number of roles and communities.

Chapter 8

Joint Detection of Roles and Communities

8.1 Introduction

In this chapter, we aim to answer the research question Q3 introduced in Section 1:

> Q3: How can we jointly mine the local and global structures of graphs by modeling the relationship between global roles and local communities?

In order to answer this question, we have to explore the relation between roles and communities of graphs. As introduced in Chapter 1, roles and communities represent global and local structures of graphs. Previous studies viewed the task of role discovery and community detection orthogonally and solved them independently. The relation between them has been totally neglected in most previous studies: (1) most role discovery methods only exploit global structural features but ignore the community structures. For instance, RolX [HGER+12] extracts global features for role detection. (2) Most community detection methods only focus on the local structural patterns of networks without considering roles. For example, NMF-based community detection method [WLW+11] decomposes the adjacency matrix which only captures the local connections between nodes. However, it is intuitive that roles and communities in a network are correlated and complementary to each other. A good example is the

Borgatti-Everett graph and for convenience we repeat it in this chapter in Figure 8.1. It can be observed that roles and communities analyze networks from the global and local perspectives, respectively. There is a diversity relation between these two concepts, i.e., the role assignment inside each community are diverse. This information has been neglected in most previous studies.

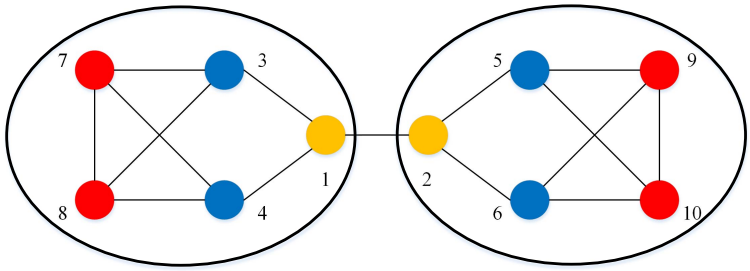


Figure 8.1: Illustration of local communities and global roles of Borgatti-Everett graph [BE92].

There are limited number of studies exploring the problem of joint detection of roles and communities, e.g., RC-Joint [RP14] and MMCR [CTS+16]. But there are some limitations in these studies. In RC-Joint, although the diversity relation between roles and communities has been integrated into the framework explicitly, it is incapable of learning the community interaction and role interaction patters. MMCR [CTS+16] extends MMSB by integrating the community-aware role assignment in a Bayesian framework. Thus, the community-aware role interactions can be learned same to MMSB but the interaction patterns between communities have not been studied.

In order to overcome these limitations in previous studies, we propose a novel model for simultaneous role and community detection (REACT). REACT combines role discovery and community detection using NMTF components respectively. The role discovery component factorize the pairwise RoleSim similarity matrix [JLH11] which models the global structural information and the community detection component decomposes the adjacency matrix which captures the local structural information of network. To model the diversity relation between roles and communities, we use the $L_{2,1}$ norm as a regularization term. REACT is advantageous over previous methods because it (1) detects roles and communities simultaneously in a unified model via $L_{2,1}$ norm to explicitly model the diversity relation between roles and communities, and (2) provides interac-

tion patterns for roles and communities respectively via the extra latent factor in the output of NMTF.

The contributions of this chapter are summarized as follows:

- We propose a novel model to discover roles and communities simultaneously (REACT) in networks. REACT combines role discovery and community detection using two NMTF components and integrates the diversity relation between roles and communities using the $L_{2,1}$ norm.
- We derive efficient updating rules to learn the parameters of REACT, and propose a selection model to automatically determine the number of roles and communities.
- The conducted experimental study on several real-world networks from different domains demonstrate the effectiveness of REACT for both role and community detection. It also provides extra information in interaction patterns for communities and roles.

The rest of this chapter is organized as follows. Notations and problem formulation are given in Section 8.2. Section 8.3 explains the proposed REACT model. In Section 8.4 we then discuss our experimental study. Finally, in Section 8.5 we draw conclusions and outline directions for future work.

8.2 Notations and Backgrounds

We first summarize some notations in Table 8.1 and then introduce the backgrounds of the techniques we will use in this chapter.

8.2.1 Non-negative Matrix Tri-factorization (NMTF)

Non-negative matrix factorization (NMF) [KP08] is a popular model in multivariate analysis and linear algebra where a matrix is factorized into two matrices, with the property that all three matrices have no negative elements. There are several advantages in NMF including ease of implementing inference and ease of interpreting results. Non-negative Matrix Tri-Factorization (NMTF) extends NMF by factorizing a matrix into three matrices so that it provides an extra latent matrix (the middle matrix) to denote the interaction patterns between clusters. Hence MNTF has been used in community detection

Table 8.1: Summary of the notations.

Notation	Description
n	Number of nodes.
r	Number of roles.
c	Number of communities.
$A_{n\times n}$	Adjacency matrix of the given network.
$S_{n\times n}$	RoleSim similarity matrix of the given network.
$C_{n\times c}$	Community membership matrix.
$R_{n\times r}$	Role membership matrix.
$B_{c\times c}$	Community interaction matrix.
$M_{r\times r}$	Role interaction matrix.
λ	Trade-off parameter for diversity relation.

recently [WLW+11, PCS15] to detect communities and learn community interactions. The objective function for NMTF is defined:

$$\min_{H,S} \|X - HSH^T\|_F^2, \quad s.t.\ H \geq 0, S \geq 0, H^T H = I \tag{8.1}$$

where $\|\cdot\|_F^2$ is the Frobenius norm. The non-negativity constraint in Eq. (1) makes the sparsity of the representation of the original data which is easier to interpret and more semantically meaningful compared with other factorization methods, e.g., SVD and PCA [LS01]. The orthogonal constraint can improve the interpretability of the clustering results. Using multiplicative update rules [DLPP06], the solution for Eq. (1) is shown as follows:

$$H_{jk} \leftarrow H_{jk} \circ \frac{(W^T HS)_{jk}}{(HH^T W^T HS)_{jk}}, \tag{8.2}$$
$$S_{ik} \leftarrow S_{ik} \circ \frac{(H^T WH)_{ik}}{(H^T HSH^T H)_{ik}}.$$

8.2.2 $L_{2,1}$ Norm

$L_{2,1}$ norm of a matrix is proposed as the rotational invariant of $L1$ norm [DZHZ06]. Because of its robustness to noise and outliers, It has been widely used in many machine learning tasks, e.g., clustering [KDH11], classification [CNHD11], and feature selection [NHCD10]. Given a matrix $M \in \mathbb{R}^{N\times K}$, $L_{2,1}$ norm is formally

defined as:

$$\|M\|_{2,1} = \sum_{i=1}^{N} \Big(\sum_{k=1}^{K} |M_{ik}|^2 \Big)^{1/2} = \sum_{i=1}^{n} \|m_i\|_2, \tag{8.3}$$

where $\|\cdot\|_2$ is the L2 norm. $L_{2,1}$ norm controls the capacity of M and also ensures M to be sparse in rows so it is robust to noise and outliers. However, the non-smoothness of this norm makes it difficult for optimization.

8.3 Our Proposed Model

8.3.1 REACT Model

In most previous studies on role discovery and community detection, one major problem is that they solved these problems separately and neglected the relation between roles and communities. To deal with this problem, we propose a novel model for joint role and community detection (REACT). REACT consists of three components: role discovery, community detection, and diversity relation component. The first and second components utilize NMTF to cluster nodes respectively and third component employs the $L_{2,1}$ norm to capture the diversity relation between roles and communities. A graphical representation of REACT is shown in Fig. 8.2.

Role discovery component

Role discovery clusters nodes according to the global structural information of the given network. Previous methods extract predefined features to capture the global structural information, For example, RolX [HGER+12] uses a recursive way to extract features, SRS [ZWY+13] extracts features based on social theories. However, this type of methods has the some limitations: (1) they require prior knowledge to define the extracted features; and (2) they are incapable of learning role interaction patterns. To overcome these limitations, we select NMTF to factorize the pairwise RoleSim similarity matrix. RoleSim can measure the global similarity between two nodes which is based on the structural information instead of prior knowledge on features and has proved to be effective in role discovery [PDZ+18]. NMTF factorizes a matrix into three matrices where the middle latent matrix/factor denote the interactions between clusters.

Formally, RoleSim similarity $S(u,v)$ between two nodes u and v is defined as:

$$S(u,v) = (1-\beta) \max_{M(u,v)} \frac{\sum_{(x,y)\in M(u,v)} S(x,y)}{|N(u)| + |N(v)| - |M(u,v)|} + \beta \tag{8.4}$$

where $N(u)$ and $N(v)$ are neighbors of node u and v, respectively. $M(u,v)$ is a matching between $N(u)$ and $N(v)$, i.e., $M(u,v) \subseteq N(u) \times N(v)$ is a bijection between $N(u)$ and $N(v)$. The parameter β is a decay factor where $0 < \beta < 1$. The intuition of RoleSim is that two nodes are structurally similar if their corresponding neighbors are also structurally similar. This intuition is consistent with the notion of automorphic and regular equivalence [WF94].

Then the objective function to decompose the RoleSim matrix for role discovery is defined as:

$$\min_{R,M} \|S - RMR^T\|_F^2, \tag{8.5}$$
$$s.t.\ R \geq 0,\ M \geq 0,\ R^T R = I.$$

Community detection component

Community detection aims to cluster nodes from the local perspective. Therefore, following [WLW+11], we directly factorize the adjacency matrix because

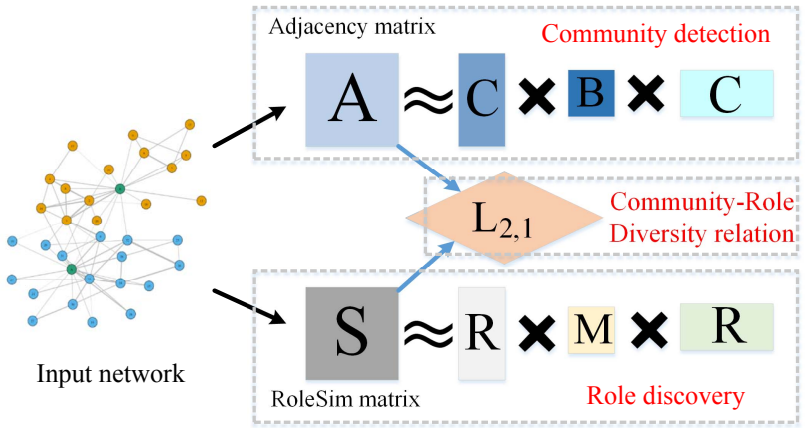


Figure 8.2: Our proposed REACT model.

the adjacency matrix captures the first-order local connections of nodes in the network. In order to obtain the community interaction patterns, we select NMTF to factorize the adjacency matrix. Formally, the objective function to factorize the adjacency matrix for community detection is defined as:

$$\min_{C,B} \|A - CBC^T\|_F^2, \tag{8.6}$$
$$s.t.\ C \geq 0,\ B \geq 0,\ C^T C = I.$$

The nonnegativity of NMTF ensures the explainability of node-community matrix and community interaction matrix. The latent factor B denotes the community interaction patterns.

Community-role relation component

As we mentioned in Section 8.1, roles and communities are correlated to play two complementary roles in network analysis. In order to jointly detect roles and communities in networks, we need to model the relation between roles and communities. In this work, we aim to explicitly model the diversity relation between them (shown in Fig. 8.1). In specific, the distribution of role assignment inside each community should be as diverse as possible so we aim to minimize the product of role assignment membership matrix and community membership matrix which is similar to [RP14]. To achieve this goal, different types of norms can be employed. Considering (1) the noise and outliers in networks and (2) the possible inaccurate assignment in role and community detection, we select $L_{2,1}$ norm as the regularization because it is well-known to be robust to noise [KDH11, SHP15]. Formally, we define the regularization as:

$$Reg = \|C^T R\|_{2,1}. \tag{8.7}$$

After defining two NMTF components for role discovery and community detection for networks, we propose to utilize both components jointly, and also integrate the diversity relation between roles and communities as the regularization. So we can discover roles and communities simultaneously:

$$L = \min_{C,B,R,M} \underbrace{\|S - RMR^T\|_F^2}_{role\ discovery} + \underbrace{\|A - CBC^T\|_F^2}_{community\ detection} \tag{8.8}$$
$$+ \underbrace{\lambda \|C^T R\|_{2,1}}_{diversity\ relation\ regularization}$$
$$s.t.\ C \geq 0, B \geq 0, R \geq 0, M \geq 0, C^T C = I, R^T R = I.$$

Optimization. The objective function in Eq. (8.8) is not convex for all parameters R, M C, and B simultaneously. We use the multiplicative update rules to solve this optimization problem due to its good compromise between speed and ease of implementation [LS01]. The optimization is done by iterating the three following steps until the convergence (or the number of iteration exceeds a given threshold): (1) fix matrices M, C and B to update R, (2) fix matrices R, C and B to update M, (3) fix matrices R, M and B to update C and (4) fix matrices R, M and C to update B (Algorithm 6). Note that we follow [NHCD10] to calculate the derivative of the objective function with $L_{2,1}$ norm.

Now we need to derive the update rules. First, we rewrite the objective function in Eq. (8.8) as follows:

$$\begin{aligned} L = Tr(&V^{(t)T}V^{(t)} - F^{(t)T}G^{(t)T}V^{(t)} - V^{(t)T}G^{(t)}F^{(t)} \\ &+ F^{(t)T}G^{(t)T}G^{(t)}F^{(t)}) + Tr(V^{(t)T}V^{(t)} - V^{(t)T}G^{(t-1)}M^{(t)}F^{(t)} \\ &- F^{(t)T}M^{(t)T}G^{(t-1)T}V^{(t)} + F^{(t)T}M^{(t)T}G^{(t-1)T}G^{(t-1)}M^{(t)}F^{(t)}) \end{aligned} \tag{8.9}$$

where $Tr(\cdot)$ is the trace function. Then the gradients of L with respect to $G^{(t)}$, $M^{(t)}$ and $F^{(t)}$ can be calculated as:

$$\nabla_{G^{(t)}} L = -2V^{(t)}F^{(t)T} + 2G^{(t)}F^{(t)}F^{(t)T} \tag{8.10}$$

$$\nabla_{M^{(t)}} L = -2G^{(t-1)T}V^{(t)}F^{(t)T} + 2G^{(t-1)T}G^{(t-1)}M^{(t)}F^{(t)}F^{(t)T} \tag{8.11}$$

$$\begin{aligned} \nabla_{F^{(t)}} L = &-2G^{(t)T}V^{(t)} + 2G^{(t)T}G^{(t)}F^{(t)} - 2M^{(t)T}G^{(t-1)T}V^{(t)} \\ &+ 2M^{(t)T}G^{(t-1)T}G^{(t-1)}M^{(t)}F^{(t)} \end{aligned} \tag{8.12}$$

Using the Karush-Kuhn-Tucker (KKT) conditions on this problem, we have

$$\begin{aligned} &G^{(t)} \geq 0, M^{(t)} \geq 0, F^{(t)} \geq 0 \\ &\nabla_{G^{(t)}} L \geq 0, \nabla_{M^{(t)}} L \geq 0, \nabla_{F^{(t)}} L \geq 0 \\ &G^{(t)} \circ \nabla_{G^{(t)}} L = 0, M^{(t)} \circ \nabla_{M^{(t)}} L = 0, F^{(t)} \circ \nabla_{F^{(t)}} L = 0 \end{aligned} \tag{8.13}$$

where $\circ$ denotes the element-wise product.

Using the Karush-Kuhn-Tucker (KKT) complementary condition, the update

rules of the objective function (8.8) are:

$$R_{jk} \leftarrow R_{jk} \circ \frac{(S^T RM)_{jk}}{(RR^T SRM + \lambda CDC^T R)_{jk}} \tag{8.14}$$

$$M_{jk} \leftarrow M_{jk} \circ \frac{(R^T SR)_{jk}}{(R^T RMR^T R)_{jk}} \tag{8.15}$$

$$C_{jk} \leftarrow C_{jk} \circ \frac{(A^T CB)_{jk}}{(CC^T ACB + \lambda RR^T CD)_{jk}} \tag{8.16}$$

$$B_{jk} \leftarrow B_{jk} \circ \frac{(C^T AC)_{jk}}{(C^T CBC^T C)_{jk}} \tag{8.17}$$

where D is the diagonal matrix with the j-th diagonal element which is defined as:

$$D_{jj} = \frac{1}{\|(C^T R)_j\|_2} \tag{8.18}$$

Algorithm 6 Optimization Algorithm

Input: Adjacency matrix A, number of roles r, number of communities c, trade-off parameter λ

Output: Role membership matrix R, community membership matrix C, role interaction matrix M, and community interaction matrix B

1: Calculate RoleSim similarity S according to Eq. (8.4)
2: Initialize R, M, C and B randomly
3: **while** not converge **do**
4: Update R according to Eq. (8.14)
5: Update M according to Eq. (8.15)
6: Update C according to Eq. (8.16)
7: Update B according to Eq. (8.17)
8: **end while**

Complexity analysis of REACT. For simplicity, given two matrices $M_{n\times r}$ and $N_{r\times f}$, the computational complexity of the multiplication of M and N is $O(nrf)$. The complexity of updating rules in Algorithm 6 (Line 4 - 7) is $O(n^2r + nr^2 + r^3 + n^2c + nc^2 + c^3)$. Since c and r can be viewed as the input constant and we also have $c \ll n$ and $r \ll n$, the complexity can be reduced to $O(n^2 + nr^2 + r^3 + nc^2 + c^3)$. By considering the number of iteration i the number of snapshots t the complexity

is $O(i(n^2 + nr^2 + r^3 + nc^2 + c^3))$. Besides, the complexity of RoleSim (Line 1) is $O(kn^2d)$, where n is the number of nodes, k is the number of iterations and d is the average of $y \log y$ over all node-pair bipartite graph in G [JLH11].

8.3.2 Model Selection

In practice, the number of roles and communities may be not available beforehand. Therefore, how to determine the suitable numbers is challenging. To tackle this problem, in this section we follow [HGER+12] to use a model selection method to determine the number of roles and communities in the discovery process because this method has better generalization in different networks.

We adopt the model selection method proposed in [HGER+12]. It uses the Minimum Description Length (MDL) [Ris78] to decide on the number of roles. Without loss of generality, we consider a general NMTF problem, i.e., $\|X - HSH^T\|_F^2$. The selected model, i.e., the appropriate number of clusters r, is the one that minimizes the description length $\mathscr{L}$, where $\mathscr{L}$ is sum of the model description cost $\mathscr{E}$ and the coding cost $\mathscr{M}$, i.e., $\mathscr{L} = \mathscr{M} + \mathscr{E}$. $\mathscr{M}$ is defined as $br(n+f)$, where b is the bits used for each element. $\mathscr{E}$ is defined as the KL divergence based error:

$$\mathscr{E} = \sum_{i,j} \left(X_{i,j} \log \frac{V_{i,j}}{(HSH^T)_{i,j}} - X_{i,j} + (HSH^T)_{i,j} \right) \tag{8.19}$$

In our case, we have two NMTF components so we select the suitable number of roles and communities separately.

8.4 Experimental Studies

8.4.1 Experimental Settings

To validate the effectiveness of our proposed REACT model for joint role and community detection, we conduct experiments on several real-world networks from different domains. We first evaluate REACT and state-of-the-art algorithms on the role discovery task (Section 8.4.3, and compare REACT with previous community detection methods (Section 8.4.4). We then investigate the influence of the trade-off parameter λ on the algorithm's performance (Section 8.4.5). We also empirically show that our method can provide extra community interaction and role interaction information using visualization (Section 8.4.6).

Table 8.2: Summary of data sets used in the experiments. NA means that the information is not available.

Type	Dataset	n	e	c	r
with community labels	Citeseer	3312	4732	6	NA
	Email	1005	25579	42	NA
with role labels	Brazil-air	131	1038	NA	4
	Europe-air	399	5995	NA	4
	USA-air	1190	13599	NA	4

Datasets

We conduct experiments on two types of network datasets: networks with ground-truth role labels[1] and community labels[2]. Community detection data consists of two networks and role discovery data consists of three networks. A brief summary of these datasets is shown in Table 8.2.

Baselines

As our model aims to jointly solve the problem of community detection and role discovery, we compare REACT with three types of state-of-the-art methods: role discovery methods, community detection methods, and joint community and role detection methods. For role discovery methods, we use the following baselines:

- NMTF [WLW+11]: We use the same NMTF model in [WLW+11] but to factorize the RoleSim matrix instead of the adjacency matrix. Note that this is the role discovery component in our proposed model.
- RolX [HGER+12]: It is also a NMF based method and it decomposes a feature matrix which is extracted from the network based on some predefined operations.
- MMSB [ABFX09]: It is a Bayesian model which treats the role of each node as a latent variable and then uses Bayesian statistics to infer these latent variables.

[1] The networks are from [RSF17].

[2] These datasets are from `http://snap.stanford.edu/data/index.html` and `https://linqs.soe.ucsc.edu/data`.

For community detection methods, we use

- NMTF [WLW+11]: It is the basic NMTF method for community detection which factorizes the adjacency matrix into three matrices. Note that this is the community detection component in our proposed model.
- BigClam [YL13]: BigClam models the affiliation strength of each node to each community and assigns each node-community pair a nonnegative latent factor as the degree of community membership. Note that it is designed for overlapping community detection.
- BNMF [PRES11]: It is the Bayesian version of NMF model to detect communities.

For joint methods:

- RC-Joint [RP14]: It aims to simultaneously identify community and structural role assignments in a network. In a nonparametric fashion, RC-Joint updates and improves community and role assignment iteratively.
- MMCR [CTS+16]: It detects the latent community and role of each node at the same time. By extending the role interaction probability to community-aware role interaction probabilities, MMCR can explicitly models the relation between communities and roles.

8.4.2 Evaluation Metrics

To evaluate the experimental results, we use purity and normalized mutual information (NMI) as the evaluation metrics. These metrics are widely used in the evaluation of clustering methods with ground-truth labels. Formal definitions of both evaluation metrics have been introduced in Section 3.6.1 of Chapter 3.

8.4.3 Role Discovery

The results for role discovery is shown in Table 8.3. For the baselines, we use the actual numbers of roles in each network as the input parameter. For our method, we report the results using different numbers of roles: (1) the actual numbers of roles as the input (REACT (actual)) and (2) using the method introduced in Section 8.3.2 to infer the role numbers (REACT (inferred)). Note that due to the network characteristics, in this experiment the inferred number of roles is the same to the actual role number, i.e., 4 roles in all three networks.

From these results, it can be observed that:

Table 8.3: Role Discovery results.

	Brazil		Europe		USA	
	NMI	Purity	NMI	Purity	NMI	Purity
NMTF [WLW+11]	0.5133	0.7328	0.2813	0.5037	0.2533	0.5066
RolX [HGER+12]	0.5032	0.6867	0.2642	0.4862	0.2767	0.4967
MMSB [ABFX09]	0.3796	0.4012	0.2851	0.5088	0.2356	0.4578
RC-Joint [RP14]	0.3625	0.3864	0.2565	0.4862	0.2056	0.4186
MMCR [CTS+16]	0.3856	0.4213	0.2732	0.4987	0.2461	0.4780
REACT (actual)	0.5302	0.7557	0.2976	0.5388	0.3180	0.5891
REACT (inferred)	**0.5302**	**0.7557**	**0.2976**	**0.5388**	**0.3180**	**0.5891**

- Our proposed method, REACT, outperforms other state-of-the-art methods in role discovery task which demonstrates the effectiveness of our method. REACT performs better than NMTF which is the role discovery component in our model. It indicates that by taking the diversity relation between roles and communities into consideration, we can achieve better performance.

- These two joint role and community detection methods do not perform well in role discovery task. This may result from that these methods perform better in optimizing different objective: (1) RC-Joint aims to maximize the likelihood of the network, and (2) MMCR aims to minimize the perplexity of the model.

- An interesting observation is that the inferred number of roles is exactly the same to the actual number of role. In fact, smaller number of roles can achieve better performance (we will present the empirically evidence below). This is because roles are defined based on the global structural patterns and the possible patterns are limited, i.e., they will not change with the increase/decrease of nodes/edges in a network.

Besides, we explore the effect on the performance with different numbers of roles and the results are shown in Fig. 8.3. As we mentioned before, smaller number of roles can achieve better performance. In these networks, the optimal number of roles is 4 which is the actual number of roles.

Table 8.4: Community detection results.

	Email		Citeseer	
	NMI	Purity	NMI	Purity
NMTF [WLW+11]	0.6059	0.5652	0.0790	0.2648
BNMF [PRES11]	0.5862	0.5085	0.0812	0.2385
BigClam [YL13]	0.5705	0.5883	0.0735	0.1543
RC-Joint [RP14]	0.3021	0.3644	0.0956	0.2178
MMCR [CGLH14]	0.5675	0.5330	0.1194	0.2967
REACT (actual)	0.6194	0.5920	0.1158	0.3077
REACT (inferred)	**0.6560**	**0.6547**	**0.1557**	**0.3373**

8.4.4 Community Detection

The results for community detection is shown in Table 8.4. Same to the role discovery task, for the baselines, we use the actual numbers of communities in each network as the input parameter. For our method, we report the results using the actual numbers of communities and using the method introduced in Section 8.3.2 to infer the community numbers. Note that the inferred number of communities for Email and Citesser network is 80 and 60, respectively.

From these results, it can be observed that:

- REACT performs better than other state-of-the-art methods in community detection task which demonstrates the effectiveness of our method. Simi-

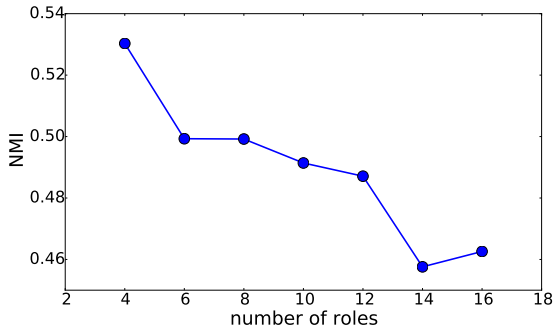


Figure 8.3: Effect of number of roles on Brazil-airport network.

lar, REACT performs better than NMTF which is the community detection component in our model. It indicates that by taking the diversity relation between roles and communities into consideration, we can achieve better performance.

- Same to the role discovery experiment, these two joint role and community detection methods do not perform well. The reason is similar to that in Section 8.4.3.
- Different from role discovery, larger community number is preferred in REACT for community detection task. This conclusion may stem from the definition of community: with the increase of network scale, the number of dense connected subgraphs may increase correspondingly. More empirical evidence will be introduced below.

8.4.5 Influence of the Trade-off Parameter

Furthermore, we investigate the influence of the trade-off parameter λ on the REACT's performance of role discovery (Fig. 8.5) and community detection (Fig. 8.6). We change the value of the trade-off parameter λ from 0.1 to 1.0 and compare the corresponding NMI values. From these results, a conclusion for optimal λ select can be drawn: The value of λ should not be too small or too large. In practice, $\lambda = 0.3$ performs best on both tasks in most networks. The only exception is community detection task on Citeseer network, and $\lambda = 0.4$ achieve the best performance.

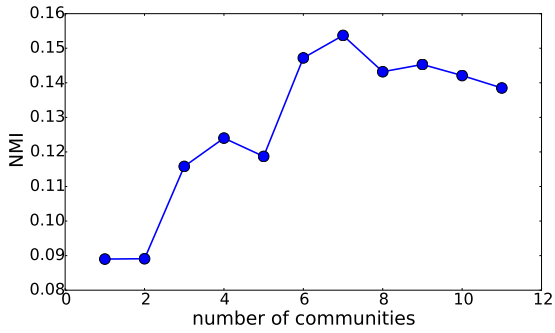
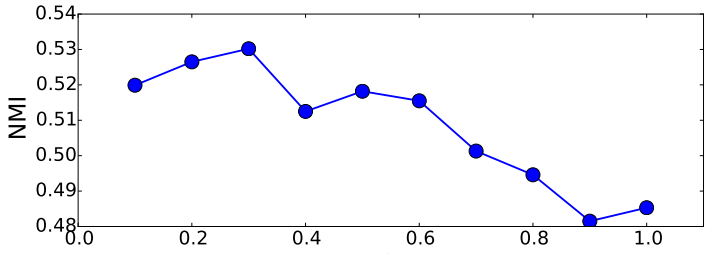


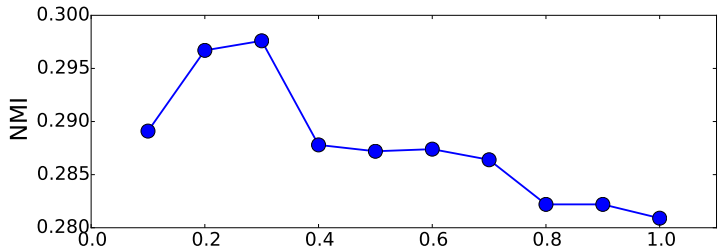
Figure 8.4: Effect of number of communities on Citeseer network.

8.4.6 Community and Role Interaction Patterns

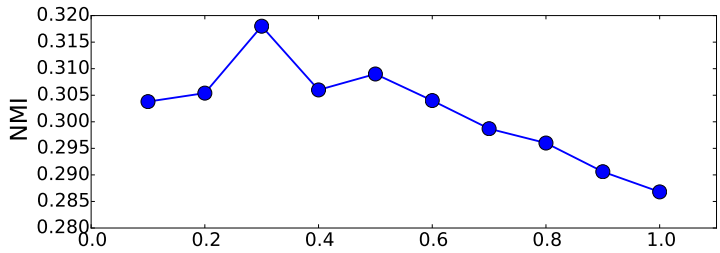
As we mentioned in the introduction, one advantage of our proposed method is that it can provide the information of community interaction patterns and



(a) NMI vs. λ on Brazil Data



(b) NMI vs. λ on Europe Data



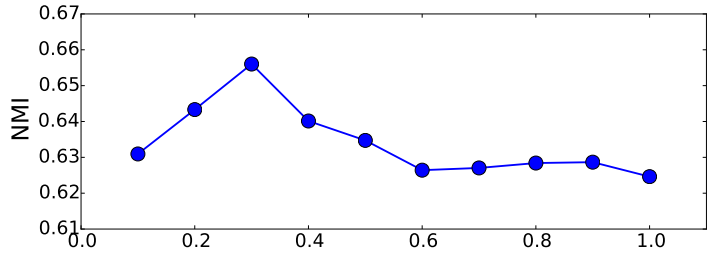
(c) NMI vs. λ on USA Data

Figure 8.5: Effect of different trade-off parameters on the role discovery task.

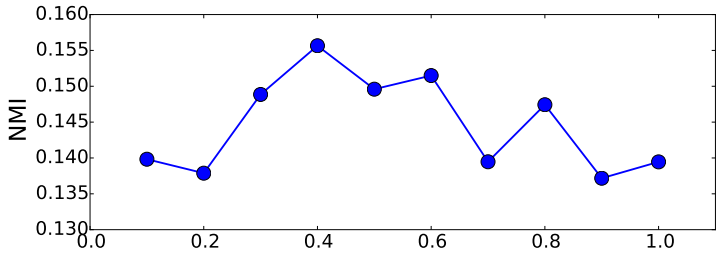
role interaction patters while other baselines fail to. In this experiment, we use the Email and USA-airport networks for case study to visualize these interaction patterns. The visualization results are shown in Fig. 8.7 and 8.8. From these results, we can observe that the community interaction matrix is approximately diagonal where more interactions happen inside each community. By contrast, the role interaction matrix reflects more complicated and global patterns that is consistent to the definitions of roles.

8.5 Concluding Remarks

Let's revisit the research question presented in Section 8.1: *How can we jointly mine the local and global structures of graphs by modeling the relationship between global roles and local communities?* To answer this question, in this chapter we proposed a novel joint role and community detection approach named REACT.



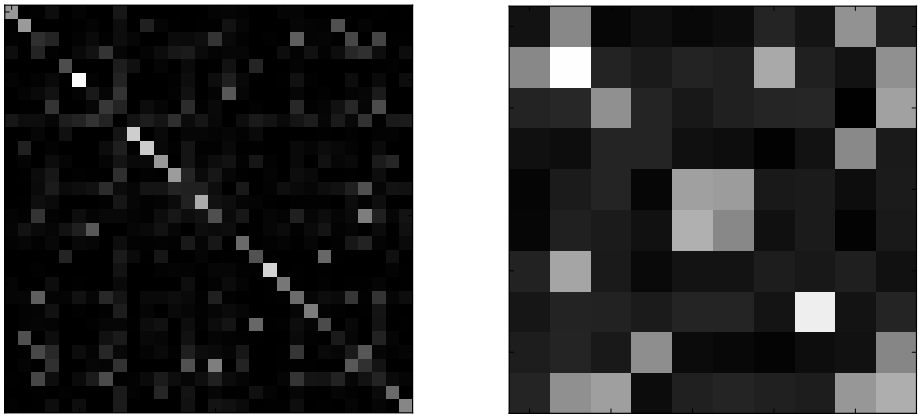
(a) NMI vs. λ on Email network.



(b) NMI vs. λ on Citeseer network.

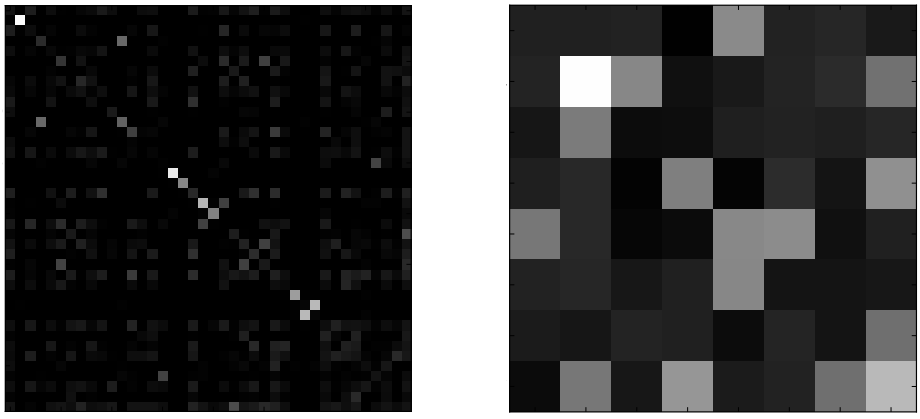
Figure 8.6: Effect of different trade-off parameters on the community detection task.

REACT consists of three components: role discovery, community detection and community-role relation. The first two components are based on nonnegative matrix tri-factorization (NMTF) and the last component is a regularization term to capture the diversity relation between roles and communities which is based



(a) Community Interaction (b) Role Interaction

Figure 8.7: Community and role interaction matrices in Email network.



(a) Community Interaction (b) Role Interaction

Figure 8.8: Community and role interaction matrices in USA-airport network.

on $L_{2,1}$ norm. We also extended MDL to determine the number of roles and communities automatically. We evaluated REACT in both role discovery and community detection compared to state-of-the-art methods. The results indicate the effectiveness of our proposed method in both tasks. We also investigated the effect of the trade-off parameter for community-role relation on both tasks. Besides, we empirically showed the interaction patters for roles and communities REACT can provide.

In future work we will exploit more relations between roles and communities in networks. We will also apply our method in different types of networks, e.g., dynamic and attributed. In the optimization, we will explore more advanced optimization methods for NMTF with regularization.

Chapter 9

Conclusions and Future Work

This chapter concludes this thesis by revisiting our research questions from Chapter 1 and summarizing our main findings (Section 9.1), discussing the limitations of our approaches (Section 9.2) and sketching directions for future research (Section 9.3). We focus on the main findings and general lessons, additional detailed findings are in the conclusion sections of the individual chapters.

9.1 Conclusions

Graph data is ubiquitous in daily life, for example, social networks, road networks, and Internet networks. Analyzing and mining graph structures is of both theoretical and practical values. Thus, a general goal is to develop a solution to the fundamental research question: *Which local and global structures of graphs can be effectively mined in both static and dynamic scenarios?* In this thesis, we have investigated a number of concrete research questions.

Local structure mining. In Part I, the local structure mining of dynamic graphs has been studied using community detection and node classification as applications (Q1).

- For community detection, we proposed *DNGE*, a novel dynamic network embedding framework using Gaussian embedding. *DNGE* learns node representations by explicitly modeling temporal information as regularization using two different smoothness strategies. Furthermore, *DNGE* utilizes Gaussian embedding to represent each node as a Gaussian distri-

bution where its mean indicates the position of this node in the embedding space and its covariance represents its uncertainty. Our experimental study demonstrated that *DNGE* effectively preserves community structures and captures dynamic information, achieves comparable results to state-of-the-art methods in link prediction and provides more information on uncertainties of node representations. Thus, our answer to Q1.1 is that it is feasible to detect local communities and capture uncertainties on dynamic graphs by modeling the dynamics as smoothness and embedding nodes into the Gaussian space.

- For node classification, we proposed a dynamic factor graph model, named *dFGM*, to classify nodes in dynamic social networks. To capture the temporal information, graph factors based on node attributes, node correlations and dynamic information are integrated in the *dFGM*. To overcome the limitation in graph feature extraction, we also utilized an unsupervised graph feature extraction method to extract features from the networks. Our experiments have been conducted on a real-world data set and the experimental results demonstrate the effectiveness of the *dFGM*. We also analyzed the influence of feature dimension and size of training data. Two different graph feature extraction methods also have been compared in the experiments. Hence, answer to Q1.2 is that local structures and temporal information can be jointly modeled in a factor graph model and this model can effectively classify nodes on dynamic graphs.

By answering the subquestions Q1.1 and Q1.2, our main finding for Q1 is that local structure on dynamic graphs can be effectively captured by modeling both structures and dynamics. Network embedding methods are promising in learning local structures and factor graph models are flexible to integrate different factors including structural properties and temporal information. Both methods can effectively detect local communities on graphs.

Global structure mining. In Part II, the global structure mining of static and dynamic graphs has been studied using role discovery as application (Q2).

- For static graphs, we proposed a flexible structure preserving network embedding framework, *struc2gauss*. On the one hand, *struc2gauss* learns node representations based on structural similarity measures so that global structural information can be taken into consideration. On the other hand, *struc2gauss* utilizes Gaussian embedding to represent each node as a Gaussian distribution where its mean indicates the position of this node in the embedding space and its covariance represents its uncertainty. By con-

ducting experiments from different perspectives, we demonstrated that *struc2gauss* excels in capturing global structural information, compared to state-of-the-art embedding techniques. It outperforms other competitor methods in role discovery task and structural role classification on several real-world networks. It also overcomes the limitation of uncertainty modeling and is capable of capturing different levels of uncertainties. Thus, our answer to Q2.1 is that global structure preserving network embedding method is beneficial to improve role discovery performance and Gaussian embedding is an effective way to capture the uncertainties in node representations.

- We also proposed a novel generative model, infinite motif stochastic blockmodel (IMM), for role discovery. IMM is advantageous in two aspects: (1) it models higher-order motifs to infer the roles which can effectively capture the global structural information of networks, and (2) it is a nonparametric Bayesian model to infer the number of roles automatically which is more suitable in real-world network analytics. We evaluated IMM in role discovery compared to state-of-the-art blockmodels and the results indicate the effectiveness of IMM. In summary, our answer to Q2.2 is that the nonparametric Bayesian stochastic model which models motifs is a promising method to determine the number of roles automatically.

- For dynamic graphs, we proposed DyNMF, a novel dynamic non-negative matrix factorization approach to discover roles and role transitions simultaneously in dynamic SNs. Current and historical views have been combined for the node-feature matrix factorization. The current view is based on structural information in the current snapshot and the historical view captures the correlation between previous roles and current roles using role transition matrices. We conducted comprehensive experiments on both synthetic and real-world data sets to validate the performance of DyNMF in role discovery and role transition learning. We analyzed the experimental results from three aspects including role discovery, role transition, and role prediction. The results indicate the effectiveness of our proposed method for the challenging problem of dynamic role analytics. Therefore, our answer to Q2.3 is that explicitly integrating role transition smoothness between graph snapshots is an effective way to discover roles and learn role transition on dynamic graphs.

By answering the subquestions Q2.1, Q2.2 and Q2.3, our main finding for Q2 is that global structure preserving network embedding method is beneficial to

improve role discovery performance and capture the uncertainties in node representations, nonparametric Bayesian stochastic model is a potential method to determine the number of roles automatically, and by integrating dynamic information, role and role transition can be effectively captured.

Joint mining local and global structure. In Part III, the jointly mining local and global structures of static graphs has been studied using community detection and role discovery as applications (Q3).

- We proposed a novel joint role and community detection approach named REACT. REACT consists of three components: role discovery, community detection and community-role relation. The first two components are based on nonnegative matrix tri-factorization (NMTF) and the last component is a regularization term to capture the diversity relation between roles and communities which is based on $L_{2,1}$ norm. We also extended MDL to determine the number of roles and communities automatically. We evaluated REACT in both role discovery and community detection compared to state-of-the-art methods. The results indicate the effectiveness of our proposed method in both tasks. Therefore, our answer to Q3 is that by explicitly modeling the relationship between local communities and global roles into a unified model, it is effective to jointly discover communities and roles on graphs.

To sum up, by exploring the answers to research questions Q1, Q2 and Q3, we can partly answer the fundamental question Q0. Network embedding approaches which preserve local and global structures can effectively mine the local and global structures of graphs to improve the performance of discovery of local communities and global roles. By integrating dynamic information, these approaches can be effectively extended to dynamic graphs. To improve the robustness and capture the uncertainties, learning node representations in the Gaussian space is a promising direction. Local and global structures are correlated and complementary to each other, so jointly mining local communities and global roles by explicitly modeling the relationship between them can further improve the performance.

9.2 Limitations

After summarizing the main contributions, we discuss the limitations of the presented techniques. We present the most important challenges and possible extensions for each part.

Local structure mining The proposed methods can effectively perform in community detection and node classification by mining the local structures of dynamic graphs. However, DNGE is only suitable for plain graphs where only the topological structure is available and dFGM works in plain and attributed graphs. How to model more complex graphs remains a question, e.g., heterogeneous networks [SHY+11] where multiple types of nodes and edges exist and signed networks [LHK10] where edges can be positive or negative. Furthermore, node removal is another issue in mining local structures on dynamic graphs.

Global structure mining For the proposed methods for global structure mining, similar to local structure mining, all methods are designed for plain graphs and extension to attributed, heterogeneous and signed networks is also an issue. Specific to each approach, *struc2gauss* has the computational complexity issue in calculating structural similarity. IMM has high computational complexity in Bayesian model inference. The bottleneck in learning DyNMF is the complexity of matrix factorization.

Joint mining of local and global structures The proposed method in jointly detecting roles and communities exploits the role-diversity relation between local and global structures. However, there exist other types of relations between these two types of structures which lead to some limitations in simultaneous mining local and global structures of graphs: 1) how to effectively model more types of relations between roles and communities remains unkonwn; 2) how to automatically learn the trade-off between local and global structures is a challenging problem; and 3) how to extend the proposed model to other types of graphs, e.g., dynamic graphs and heterogeneous graphs, is another limitation.

9.3 Future Work

In the above limitation discussions, we have touched on various extensions and possible future work directions. In this section, we summarize some of the most promising future directions.

Heterogeneous Graph Structures. In practice, graph data contains more complex structures. For instance, a co-author graph consists of different types of nodes including authors, papers, conferences and publishers. An opinion graph contains different types of edges because the edges can be positive or negative. Thus, how to capture such complex structures is a challenging problem. It is non-trivial to extend these methods which are proposed for plain graphs to more complex graphs. In the future, we plan to propose solutions to analyze

and mine such complicated structures of graphs.

Evaluation Framework. Study on evaluation measures for structure mining tasks, e.g., community detection and role discovery, is another interesting direction. If there exist ground-truth labels of communities or roles, evaluation on these tasks can be viewed as a standard clustering evaluation problem. Therefore, clustering evaluation measures with known labels can be utilized such as purity, normalized mutual information (NMI) and Adjusted Rand Index (ARI). However, if ground-truth labels are not available which is more common in practice, effective evaluation measures are required. In the future, we woud like to explore effective and interpretable evaluation measures for graph structure mining tasks.

Benchmark Dataset. Benchmark datasets are critical for developing, evaluating, and comparing different graph structure mining approaches. Current benchmark datasets for structure mining tasks are deficient in 1) most of them were collected for local structure mining tasks, i.e., community detection, and the benchmark datasets for global structure mining are missing; and 2) most of them are relatively small in terms of the number of nodes and edges compared to the increasing volume of graph data such as online social networks and extremely dynamic road networks. This requirement raises two promising directions: benchmark dataset collection and labeling systems and benchmark dataset generators.

Tool Implementation. There still exists a gap between the proposed algorithm and available tools with regard to the ease of use, scalability, and the intuitiveness of the user interface. Therefore, a meaningful direction for future work would be to implement tools for graph structure mining. In addition, it would also be beneficial to integrate these structure mining approaches into well-known distributed graph processing frameworks, e.g., Gelly [1].

[1] `https://flink.apache.org/news/2015/08/24/introducing-flink-gelly.html`

Bibliography

[ABFX08] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(Sep):1981–2014, 2008. (Cited on pages 31, 32, 92, 115, 116, 119, and 124.)

[ABFX09] Edo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. In *Advances in Neural Information Processing Systems*, pages 33–40, 2009. (Cited on pages 127, 161, and 163.)

[AL11] Charu C Aggarwal and Nan Li. On node classification in dynamic content-based networks. In *SDM*, pages 355–366. SIAM, 2011. (Cited on page 66.)

[AMC08] Ioannis Antonellis, Hector Garcia Molina, and Chi Chao Chang. Simrank++: query rewriting through link analysis of the click graph. *Proceedings of the VLDB Endowment*, 1(1):408–421, 2008. (Cited on page 91.)

[ARL+18] Nesreen K Ahmed, Ryan Rossi, John Boaz Lee, Theodore L Willke, Rong Zhou, Xiangnan Kong, and Hoda Eldardiry. Learning role-based graph embeddings. *arXiv preprint arXiv:1802.02896*, 2018. (Cited on page 33.)

[ATRZ15] Afra Abnar, Mansoureh Takaffoli, Reihaneh Rabbany, and Osmar R Zaïane. Ssrm: structural social role mining for dynamic social

networks. *Social Network Analysis and Mining*, 5(1):1–18, 2015. (Cited on pages 128, 130, and 143.)

[BBA75] Ronald L Breiger, Scott A Boorman, and Phipps Arabie. An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *Journal of mathematical psychology*, 12(3):328–383, 1975. (Cited on page 23.)

[BBK+18] Stephen Bonner, John Brennan, Ibad Kureshi, Georgios Theodoropoulos, Andrew Stephen McGough, and Boguslaw Obara. Temporal graph offset reconstruction: Towards temporally robust graph representation learning. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 3737–3746. IEEE, 2018. (Cited on pages 42 and 45.)

[BCM11] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. Node classification in social networks. In *Social network data analytics*, pages 115–148. Springer, 2011. (Cited on page 66.)

[BE92] Stephen P Borgatti and Martin G Everett. Notions of position in social network analysis. *Sociological methodology*, pages 1–35, 1992. (Cited on pages xv, xvii, 3, 26, 88, 116, and 152.)

[BE93] Stephen P Borgatti and Martin G Everett. Two algorithms for computing regular equivalence. *Social networks*, 15(4):361–376, 1993. (Cited on page 99.)

[BG17] Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of attributed graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*, 2017. (Cited on pages 44, 46, 89, 91, 95, and 101.)

[BKH16] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. (Cited on page 91.)

[BQD18] Zilong Bai, Buyue Qian, and Ian Davidson. Discovering models from structural and behavioral brain imaging data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1128–1137. ACM, 2018. (Cited on page 25.)

[Bur76] Ronald S Burt. Positions in networks. *Social forces*, 55(1):93–122, 1976. (Cited on page 25.)

[BWT+17] Zilong Bai, Peter Walker, Anna Tschiffely, Fei Wang, and Ian Davidson. Unsupervised network discovery for brain imaging data. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 55–64. ACM, 2017. (Cited on page 25.)

[CGLH14] Chengyao Chen, Dehong Gao, Wenjie Li, and Yuexian Hou. Inferring topic-dependent influence roles of twitter users. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 1203–1206. ACM, 2014. (Cited on page 164.)

[CHT+15] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 119–128. ACM, 2015. (Cited on pages 44 and 88.)

[CLK+13] Jeffrey Chan, Wei Liu, Andrey Kan, Christopher Leckie, James Bailey, and Kotagiri Ramamohanarao. Discovering latent blockmodels in sparse and noisy graphs using non-negative matrix factorisation. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 811–816. ACM, 2013. (Cited on page 25.)

[CLX15] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 891–900. ACM, 2015. (Cited on pages 41, 44, 87, 88, and 91.)

[CNHD11] Xiao Cai, Feiping Nie, Heng Huang, and Chris Ding. Multi-class l2, 1-norm support vector machine. In *2011 IEEE 11th International Conference on Data Mining*, pages 91–100. IEEE, 2011. (Cited on page 154.)

[CRPS14] Sarvenaz Choobdar, Pedro Ribeiro, Srinivasan Parthasarathy, and Fernando Silva. Dynamic inference of social roles in information

cascades. *Data Mining and Knowledge Discovery*, pages 1–26, 2014. (Cited on pages 30, 66, 68, and 70.)

[CRPS15] Sarvenaz Choobdar, Pedro Ribeiro, Srinivasan Parthasarathy, and Fernando Silva. Dynamic inference of social roles in information cascades. *Data Mining and Knowledge Discovery*, 29(5):1152–1177, 2015. (Cited on pages 127, 129, 130, 134, and 143.)

[CTS+16] Ting Chen, Lu-An Tang, Yizhou Sun, Zhengzhang Chen, Haifeng Chen, and Guofei Jiang. Integrating community and role detection in information networks. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 72–80. SIAM, 2016. (Cited on pages 152, 162, and 163.)

[CWPZ18] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2018. (Cited on pages 8 and 44.)

[CZC+17] Sandro Cavallari, Vincent W Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. Learning community embedding with community detection and node embedding on graphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 377–386. ACM, 2017. (Cited on pages xv, 20, and 21.)

[DHS11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011. (Cited on pages 52 and 98.)

[DLPP06] Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135. ACM, 2006. (Cited on pages 17 and 154.)

[Dor17] Derek Doran. Network role mining and analysis: An overview. In *Network Role Mining and Analysis*, pages 1–13. Springer, 2017. (Cited on pages 27 and 28.)

[DSPG16] Ludovic Dos Santos, Benjamin Piwowarski, and Patrick Gallinari. Multilabel classification on heterogeneous graphs with gaussian

embeddings. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 606–622. Springer, 2016. (Cited on pages 42, 44, 46, 49, 89, 95, and 101.)

[DWS+18] Lun Du, Yun Wang, Guojie Song, Zhicong Lu, and Junshan Wang. Dynamic network embedding: An extended approach for skip-gram based network embedding. In *IJCAI*, pages 2086–2092, 2018. (Cited on pages 42, 44, and 45.)

[DZHL18] Claire Donnat, Marinka Zitnik, David Hallac, and Jure Leskovec. Learning structural node embeddings via diffusion wavelets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1320–1329. ACM, 2018. (Cited on pages xv and 35.)

[DZHZ06] Chris Ding, Ding Zhou, Xiaofeng He, and Hongyuan Zha. R 1-pca: rotational invariant l 1-norm principal component analysis for robust subspace factorization. In *Proceedings of the 23rd international conference on Machine learning*, pages 281–288. ACM, 2006. (Cited on page 154.)

[EB94] Martin G Everett and Stephen P Borgatti. Regular equivalence: General theory. *Journal of mathematical sociology*, 19(1):29–52, 1994. (Cited on page 27.)

[For10] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010. (Cited on pages 4, 6, 16, 17, and 92.)

[Fre77] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977. (Cited on page 18.)

[FSX09] Wenjie Fu, Le Song, and Eric P Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of the 26th annual international conference on machine learning*, pages 329–336. ACM, 2009. (Cited on page 127.)

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016. (Cited on pages 20 and 41.)

[GCS+18] Mohadeseh Ganji, Jeffrey Chan, Peter J Stuckey, James Bailey, Christopher Leckie, Kotagiri Ramamohanarao, and Laurence Park. Semi-supervised blockmodelling with pairwise guidance. In *Joint*

European Conference on Machine Learning and Knowledge Discovery in Databases, pages 158–174. Springer, 2018. (Cited on page 26.)

[GDC10] Derek Greene, Donal Doyle, and Padraig Cunningham. Tracking the evolution of communities in dynamic social networks. In *2010 international conference on advances in social networks analysis and mining*, pages 176–183. IEEE, 2010. (Cited on page 41.)

[GERD13] Sean Gilpin, Tina Eliassi-Rad, and Ian Davidson. Guided learning for role discovery (glrd): framework, algorithms, and applications. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 113–121. ACM, 2013. (Cited on pages 127, 130, and 143.)

[GHFZ13] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A Zighed. Information diffusion in online social networks: A survey. *ACM Sigmod Record*, 42(2):17–28, 2013. (Cited on page 5.)

[GKHL18] Palash Goyal, Nitin Kamra, Xinran He, and Yan Liu. Dyngem: Deep embedding method for dynamic graphs. *arXiv preprint arXiv:1805.11273*, 2018. (Cited on pages 54 and 56.)

[GL16] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016. (Cited on pages xix, 20, 41, 42, 44, 45, 54, 55, 56, 87, 88, 91, 95, 101, 103, and 110.)

[GN02] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002. (Cited on page 18.)

[HGER+12] Keith Henderson, Brian Gallagher, Tina Eliassi-Rad, Hanghang Tong, Sugato Basu, Leman Akoglu, Danai Koutra, Christos Faloutsos, and Lei Li. Rolx: structural role extraction & mining in large graphs. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1231–1239. ACM, 2012. (Cited on pages 21, 31, 92, 99, 100, 115, 116, 127, 129, 130, 133, 134, 143, 151, 155, 160, 161, and 163.)

[HGL+11] Keith Henderson, Brian Gallagher, Lei Li, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, and Christos Faloutsos. It's who you

know: graph mining using recursive structural features. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 663–671. ACM, 2011. (Cited on pages 68, 69, 75, 92, and 134.)

[HLJZ15] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 623–632. ACM, 2015. (Cited on pages 44, 46, 49, 95, and 96.)

[HLL83] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983. (Cited on pages 31, 32, 115, 119, and 122.)

[Hof99] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999. (Cited on page 67.)

[HYL17] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017. (Cited on page 45.)

[JLH11] Ruoming Jin, Victor E Lee, and Hui Hong. Axiomatic ranking of network role similarity. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 922–930. ACM, 2011. (Cited on pages 29, 92, 94, 98, 152, and 160.)

[JLL14] Ruoming Jin, Victor E Lee, and Longjie Li. Scalable and axiomatic ranking of network role similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(1):3, 2014. (Cited on pages 29 and 94.)

[JSD+10] Ming Ji, Yizhou Sun, Marina Danilevsky, Jiawei Han, and Jing Gao. Graph regularized transductive classification on heterogeneous information networks. In *Machine Learning and Knowledge Discovery in Databases*, pages 570–586. Springer, 2010. (Cited on page 67.)

[JW02] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM, 2002. (Cited on pages 29, 91, 94, and 108.)

[KDH11] Deguang Kong, Chris Ding, and Heng Huang. Robust nonnegative matrix factorization using l21-norm. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 673–682. ACM, 2011. (Cited on pages 154 and 157.)

[KF09] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. (Cited on page 70.)

[KFL+01] Frank R Kschischang, Brendan J Frey, Hans-Andrea Loeliger, et al. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001. (Cited on page 8.)

[Kle99] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999. (Cited on page 99.)

[KP08] Hyunsoo Kim and Haesun Park. Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. *SIAM Journal on Matrix Analysis and Applications*, 30(2):713–730, 2008. (Cited on page 153.)

[KPLK18] Junghwan Kim, Haekyu Park, Ji-Eun Lee, and U Kang. Side: Representation learning in signed directed networks. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 509–518. International World Wide Web Conferences Steering Committee, 2018. (Cited on page 44.)

[KTG+06] Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, volume 3, page 5, 2006. (Cited on pages 31, 32, 115, 116, 119, and 124.)

[LCZ+08] Yu-Ru Lin, Yun Chi, Shenghuo Zhu, Hari Sundaram, and Belle L Tseng. Facetnet: a framework for analyzing communities and their

evolutions in dynamic networks. In *Proceedings of the 17th international conference on World Wide Web*, pages 685–694. ACM, 2008. (Cited on pages 41, 42, and 45.)

[LDH+17] Jundong Li, Harsh Dani, Xia Hu, Jiliang Tang, Yi Chang, and Huan Liu. Attributed network embedding for learning in a dynamic environment. *arXiv preprint arXiv:1706.01860*, 2017. (Cited on pages 41, 44, 53, 54, 56, and 88.)

[LDL+14] Xiaoyi Li, Nan Du, Hui Li, Kang Li, Jing Gao, and Aidong Zhang. A deep learning approach to link prediction in dynamic networks. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 289–297. SIAM, 2014. (Cited on page 45.)

[LFH+13] Yongming Luo, George HL Fletcher, Jan Hidders, Yuqing Wu, and Paul De Bra. External memory k-bisimulation reduction of big graphs. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 919–928. ACM, 2013. (Cited on page 28.)

[LGD+13] Kang Li, Suxin Guo, Nan Du, Jing Gao, and Aidong Zhang. Learning, analyzing and predicting object roles on dynamic networks. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 428–437. IEEE, 2013. (Cited on pages 67, 128, and 143.)

[LGG+15] Kang Li, Jing Gao, Suxin Guo, Nan Du, and Aidong Zhang. Functional node detection on linked data. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 1–9. SIAM, 2015. (Cited on page 67.)

[LHD+19] Xi Liu, Ping-Chun Hsieh, Nick Duffield, Rui Chen, Muhe Xie, and Xidao Wen. Real-time streaming graph embedding through local actions. 2019. (Cited on pages 42 and 45.)

[LHK10] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Signed networks in social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1361–1370. ACM, 2010. (Cited on page 175.)

[LJSP18] Jiongqian Liang, Peter Jacobs, Jiankai Sun, and Srinivasan Parthasarathy. Semi-supervised embedding in attributed networks

with outliers. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 153–161. SIAM, 2018. (Cited on page 44.)

[LLK09] Zhenjiang Lin, Michael R Lyu, and Irwin King. Matchsim: a novel neighbor-based similarity measure with maximum neighborhood matching. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1613–1616. ACM, 2009. (Cited on pages 29, 92, 94, and 108.)

[LNK07] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology*, 58(7):1019–1031, 2007. (Cited on page 5.)

[LRJA10] Victor E Lee, Ning Ruan, Ruoming Jin, and Charu Aggarwal. A survey of algorithms for dense subgraph discovery. In *Managing and Mining Graph Data*, pages 303–336. Springer, 2010. (Cited on page 16.)

[LS01] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001. (Cited on pages 17, 18, 31, 130, 132, 154, and 158.)

[LW71] Francois Lorrain and Harrison C White. Structural equivalence of individuals in social networks. *The Journal of mathematical sociology*, 1(1):49–80, 1971. (Cited on pages 23 and 136.)

[LZ11] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170, 2011. (Cited on page 16.)

[LZP+18] Taisong Li, Jiawei Zhang, S Yu Philip, Yan Zhang, and Yonghong Yan. Deep dynamic network embedding for link prediction. *IEEE Access*, 6:29219–29230, 2018. (Cited on page 45.)

[LZZ17] Tianshu Lyu, Yuan Zhang, and Yan Zhang. Enhancing the network embedding quality with structural similarity. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 147–156. ACM, 2017. (Cited on pages 88, 89, 90, 91, 93, 94, and 110.)

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. (Cited on pages 49, 70, and 87.)

[MM03] Maarten Marx and Michael Masuch. Regular equivalence and dynamic logic. *Social Networks*, 25(1):51–65, 2003. (Cited on page 28.)

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. (Cited on page 87.)

[MSLC01] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001. (Cited on page 65.)

[MWJ99] Kevin P Murphy, Yair Weiss, and Michael I Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 467–475. Morgan Kaufmann Publishers Inc., 1999. (Cited on page 74.)

[MWR+17] Yao Ma, Suhang Wang, ZhaoChun Ren, Dawei Yin, and Jiliang Tang. Preserving local and global information for network embedding. *arXiv preprint arXiv:1710.07266*, 2017. (Cited on page 99.)

[New06] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006. (Cited on pages 19 and 20.)

[NHCD10] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint l2, 1-norms minimization. In *Advances in neural information processing systems*, pages 1813–1821, 2010. (Cited on pages 154 and 158.)

[NJ00] Jennifer Neville and David Jensen. Iterative classification in relational data. In *Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data*, pages 13–20, 2000. (Cited on page 77.)

[NLR+18] Giang H Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim. Dynamic network embeddings: From random walks to temporal random walks. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1085–1092. IEEE, 2018. (Cited on pages 42 and 45.)

[P+02] Jim Pitman et al. Combinatorial stochastic processes. Technical report, Technical Report 621, Dept. Statistics, UC Berkeley, 2002. Lecture notes for . . . , 2002. (Cited on pages 32 and 119.)

[PARS14] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014. (Cited on pages 20, 41, 42, 44, 45, 49, 54, 56, 66, 70, 87, 88, 91, 95, 101, 103, and 110.)

[PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999. (Cited on page 99.)

[PCS15] Yulong Pei, Nilanjan Chakraborty, and Katia Sycara. Nonnegative matrix tri-factorization with graph regularization for community detection in social networks. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015. (Cited on pages 17, 18, and 154.)

[PDZ+18] Yulong Pei, Xin Du, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. struc2gauss: Structure preserving network embedding via gaussian embedding. *arXiv preprint arXiv:1805.10043*, 2018. (Cited on pages 11, 44, 46, and 155.)

[PDZ+19] Yulong Pei, Xin Du, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. Dynamic network representation learning via gaussian embedding. In *Under review*, 2019. (Cited on page 10.)

[PFP19] Yulong Pei, George Fletcher, and Mykola Pechenizkiy. Joint role and community detection in networks via l2,1 norm regularized nonnegative matrix tri-factorization. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE Press, 2019. (Cited on page 13.)

[PKS16] Bryan Perozzi, Vivek Kulkarni, and Steven Skiena. Walklets: Multiscale graph embeddings for interpretable network classification. *arXiv preprint arXiv:1605.02115*, 2016. (Cited on pages 88 and 91.)

[PRES11] Ioannis Psorakis, Stephen Roberts, Mark Ebden, and Ben Sheldon. Overlapping community detection using bayesian non-negative matrix factorization. *Physical Review E*, 83(6):066114, 2011. (Cited on pages 162 and 164.)

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. (Cited on page 87.)

[PZFP16] Y Pei, J Zhang, GHL Fletcher, and M Pechenizkiy. Node classification in dynamic social networks. In *Proceedings of AALTD 2016: 2nd ECMLPKDD International Workshop on Advanced Analytics and Learning on Temporal Data, 19 September 2016, Riva del Garda, Italy*, pages 1–8, 2016. (Cited on page 10.)

[PZFP18] Yulong Pei, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. Dynmf: role analytics in dynamic social networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 3818–3824. AAAI Press, 2018. (Cited on pages 11, 42, 45, 88, and 115.)

[PZFP19] Yulong Pei, Jianpeng Zhang, George Fletcher, and Mykola Pechenizkiy. Infinite motif stochastic blockmodel for role discovery in networks. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE Press, 2019. (Cited on page 11.)

[PZZY13] Shirui Pan, Xingquan Zhu, Chengqi Zhang, and Philip S Yu. Graph stream classification using labeled and unlabeled graphs. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 398–409. IEEE, 2013. (Cited on page 69.)

[RA+15] Ryan Rossi, Nesreen K Ahmed, et al. Role discovery in networks. *Knowledge and Data Engineering, IEEE Transactions on*, 27(4):1112–1131, 2015. (Cited on pages 5, 7, 21, 30, 31, 65, 88, 92, and 115.)

[RDJ16] Matt Revelle, Carlotta Domeniconi, and Aditya Johri. Persistent roles in online social networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 47–62. Springer, 2016. (Cited on page 139.)

[RGNH12] Ryan Rossi, Brian Gallagher, Jennifer Neville, and Keith Henderson. Role-dynamics: fast mining of large dynamic networks. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 997–1006. ACM, 2012. (Cited on pages 127 and 130.)

[RGNH13] Ryan A Rossi, Brian Gallagher, Jennifer Neville, and Keith Henderson. Modeling dynamic behavior in large evolving graphs. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 667–676. ACM, 2013. (Cited on pages 42, 45, 127, 130, 134, 139, 140, and 143.)

[Ris78] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978. (Cited on pages 116, 133, and 160.)

[RP14] Yiye Ruan and Srinivasan Parthasarathy. Simultaneous detection of communities and roles from large networks. In *Proceedings of the second edition of the ACM conference on Online social networks*, pages 203–214. ACM, 2014. (Cited on pages 7, 152, 157, 162, 163, and 164.)

[RSF17] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 385–394. ACM, 2017. (Cited on pages 87, 89, 90, 91, 94, 100, 101, 103, and 161.)

[RSK+15] Stephen Ranshous, Shitian Shen, Danai Koutra, Steve Harenberg, Christos Faloutsos, and Nagiza F Samatova. Anomaly detection in dynamic networks: a survey. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(3):223–247, 2015. (Cited on page 45.)

[S+78] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978. (Cited on pages 116 and 117.)

[SHP15] Weixiang Shao, Lifang He, and S Yu Philip. Multiple incomplete views clustering via weighted nonnegative matrix factorization with $l_{2,1}$ regularization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 318–334. Springer, 2015. (Cited on page 157.)

[SHY+11] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, 4(11):992–1003, 2011. (Cited on page 175.)

[SJ09] Blake Shaw and Tony Jebara. Structure preserving embedding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 937–944. ACM, 2009. (Cited on pages 43 and 87.)

[SNB+08] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93, 2008. (Cited on page 77.)

[Spa93] Malcolm K Sparrow. A linear algorithm for computing automorphic equivalence classes: the numerical signatures approach. *Social Networks*, 15(2):151–170, 1993. (Cited on page 27.)

[TCW+18] Ke Tu, Peng Cui, Xiao Wang, Philip S Yu, and Wenwu Zhu. Deep recursive network embedding with regular equivalence. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2357–2366. ACM, 2018. (Cited on pages xv, 34, 44, 89, 90, 91, 101, 103, and 106.)

[TDSL00] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000. (Cited on pages 43 and 87.)

[TFBZ19] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. 2019. (Cited on pages 42 and 45.)

[TQM15] Jian Tang, Meng Qu, and Qiaozhu Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1165–1174. ACM, 2015. (Cited on pages 44, 88, and 91.)

[TQW+15] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015. (Cited on pages 41, 42, 44, 45, 54, 56, 88, 91, 95, 101, 103, and 110.)

[TTS+10] Chenhao Tan, Jie Tang, Jimeng Sun, Quan Lin, and Fengjiao Wang. Social action tracking via noise tolerant time-varying factor graphs. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1049–1058. ACM, 2010. (Cited on pages 70 and 72.)

[TWS13] Jie Tang, Sen Wu, and Jimeng Sun. Confluence: Conformity influence in large social networks. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 347–355. ACM, 2013. (Cited on page 70.)

[TZT11] Wenbin Tang, Honglei Zhuang, and Jie Tang. Learning to infer social ties in large networks. In *Machine Learning and Knowledge Discovery in Databases*, pages 381–397. Springer, 2011. (Cited on pages 68, 70, 72, and 74.)

[vHFP16] Wouter van Heeswijk, George HL Fletcher, and Mykola Pechenizkiy. On structure preserving sampling and approximate partitioning of graphs. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 875–882. ACM, 2016. (Cited on page 28.)

[VM14] Luke Vilnis and Andrew McCallum. Word representations via gaussian embedding. *arXiv preprint arXiv:1412.6623*, 2014. (Cited on pages 44, 45, 46, 47, 49, 54, 95, 96, 97, 101, 103, and 106.)

[WATL17] Suhang Wang, Charu Aggarwal, Jiliang Tang, and Huan Liu. Attributed signed network embedding. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 137–146. ACM, 2017. (Cited on pages 44 and 88.)

[WCW+17] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. Community preserving network embedding. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. (Cited on pages 20, 21, and 44.)

[WCZ16] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1225–1234. ACM, 2016. (Cited on page 55.)

[WF94] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994. (Cited on pages 23, 28, 29, 89, 92, 93, 94, 105, 134, 136, and 156.)

[WJ08] Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008. (Cited on page 74.)

[WLW+11] Fei Wang, Tao Li, Xin Wang, Shenghuo Zhu, and Chris Ding. Community discovery using nonnegative matrix factorization. *Data Mining and Knowledge Discovery*, 22(3):493–521, 2011. (Cited on pages 17, 151, 154, 156, 161, 162, 163, and 164.)

[WTA+17] Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu. Signed network embedding in social media. In *Proceedings of the 2017 SIAM international conference on data mining*, pages 327–335. SIAM, 2017. (Cited on page 44.)

[XYWL13] Huan Xu, Yujiu Yang, Liangwei Wang, and Wenhuang Liu. Node classification in social network via a factor graph model. In *Advances in Knowledge Discovery and Data Mining*, pages 213–224. Springer, 2013. (Cited on pages 66, 67, 68, and 74.)

[YCA+18] Wenchao Yu, Wei Cheng, Charu C Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2672–2681. ACM, 2018. (Cited on page 44.)

[YCH+16] Liang Yang, Xiaochun Cao, Dongxiao He, Chuan Wang, Xiao Wang, and Weixiong Zhang. Modularity based community detection with deep learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2252–2258. AAAI Press, 2016. (Cited on page 17.)

[YH14] Yibo Yao and Lawrence Holder. Scalable svm-based classification in dynamic graphs. In *Data Mining (ICDM), 2014 IEEE International Conference on*, pages 650–659. IEEE, 2014. (Cited on pages 67 and 69.)

[YH15] Yibo Yao and Lawrence B Holder. Scalable classification for large dynamic networks. In *Big Data (Big Data), 2015 IEEE International Conference on*, pages 609–618. IEEE, 2015. (Cited on page 69.)

[YHX13] Junming Yin, Qirong Ho, and Eric P Xing. A scalable approach to probabilistic latent space inference of large-scale networks. In *NIPS*, pages 422–430, 2013. (Cited on pages 120 and 124.)

[YL13] Jaewon Yang and Jure Leskovec. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 587–596. ACM, 2013. (Cited on pages 17, 18, 162, and 164.)

[YSD+17] Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. Discovery of evolving semantics through dynamic word embedding learning. *arXiv preprint arXiv:1703.00607*, 2017. (Cited on page 42.)

[YTLY10] Zi Yang, Jie Tang, Juanzi Li, and Wenjun Yang. Social community analysis via a factor graph model. *IEEE Intelligent Systems*, (3):58–65, 2010. (Cited on page 70.)

[ZCWZ18] Dingyuan Zhu, Peng Cui, Daixin Wang, and Wenwu Zhu. Deep variational network embedding in wasserstein space. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2827–2836. ACM, 2018. (Cited on pages 44, 45, 89, and 91.)

[ZGY+16] Linhong Zhu, Dong Guo, Junming Yin, Greg Ver Steeg, and Aram Galstyan. Scalable temporal latent space inference for link prediction in dynamic social networks. *IEEE Transactions on Knowledge and Data Engineering*, 28(10):2765–2777, 2016. (Cited on page 45.)

[ZHS09] Peixiang Zhao, Jiawei Han, and Yizhou Sun. P-rank: a comprehensive structural similarity measure over information networks. In

Proceedings of the 18th ACM conference on Information and knowledge management, pages 553–562. ACM, 2009. (Cited on page 92.)

[ZLZ18] Yuan Zhang, Tianshu Lyu, and Yan Zhang. Cosine: community-preserving social network embedding from information diffusion cascades. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. (Cited on page 44.)

[ZWY+13] Yuchen Zhao, Guan Wang, Philip S Yu, Shaobo Liu, and Simon Zhang. Inferring social roles and statuses in social networks. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 695–703. ACM, 2013. (Cited on pages 30, 66, 67, 68, 70, 72, 134, and 155.)

[ZYR+18] Le-kui Zhou, Yang Yang, Xiang Ren, Fei Wu, and Yueting Zhuang. Dynamic network embedding by modeling triadic closure process. In *AAAI*, 2018. (Cited on pages 42, 44, 45, 54, and 56.)

Acknowledgments

The thesis in front of you would not have existed without the help and support of many people. I would like to take this opportunity to thank to them for helping and supporting me to arrive at the end of the 4-year PhD program.

I would like to thank my supervisors, dr. George Fletcher and prof.dr. Mykola Pechenizkiy. Thank you for your guidance, many interesting and insightful discussions, support and assistance during my PhD. Especially I want to thank you give me all the support and freedom for me to conduct research. I would like to thank George for your guidance in critical thinking and academic article wiring and thank Mykola for your help in improving my presentation skills.

I would like to take the opportunity to thank the committee members of this PhD defense, for reviewing this dissertation and for your insightful comments: prof.dr. Tijl De Bie from Ghent University, dr. João Gama from University of Porto, and prof.dr. Mark de Berg from Eindhoven University of Technology. I would also like to thank my advisors dr. Nikolay Yakovets and dr. Wouter Duivesteijn from Eindhoven University of Technology. Your insightful and detailed comments make my text more complete, and help me to understand the problems better.

During my graduate studies I was fortunate to do the internship at NEC Laboratories Europe. This experience contributed significantly to my academic and production development. Mathias was my mentors during my stay at NEC. I really enjoyed working and interacting with Mathias and the colleagues from the machine learning group at NEC Labs. In this moment, I want to express my gratitude to Mathias for guiding me during my stay at NEC Labs and my colleagues at NEC for interesting discussion and events. Ye, Cheng, Cheng (yes, another Cheng), Brandon, Timo, Daniel, Alberto, Xiao, Michio, Kenichi, Giampaolo, and all the ones not explicitly mentioned here, thank you so much.

In October 2019, I had an opportunity to shortly visit the Interesting Patterns

Research Team, Ghent University, led by prof. Tijl De Bie. I would like to thank Tijl for his host and supervision and thank Jefrey, Bo and Alexandru for our interesting discussion on the network embedding interpretation problem. I want to thank Maryam for her invitation of the Game Night and other members in the team for their warm hospitality while I was staying in Ghent.

I want to thank Jianpeng, Cong, and Guangming for introducing me to the group when I first came to TU/e. I also would like to thank them and other friends including Xin, Tianjin, Shiwei, Lu, Yuhao, Kaijie, Zhaohuan and Mengting for organizing the Movie Nights and Hotpot Parties.

A special thank you I would like to express to the secretaries in our group. José, Ine and Riet thank you for always take caring about all bureaucratic burdens and organizing wonderful events in our group. My gratitude also goes to prof. Paul De Bra for leading the Web Engineering group and providing us the pleasant, intellectual atmosphere in the group. Based on the Web Engineering group, we can have the Database and Data Mining groups.

During my time at TU/e, I shared the office with a group of amazing office mates: Wilco, Xuming, Negar, Alexandr, and Julia. It was really fun to discuss all the things related to the PhD life and all the other interesting things not related to the PhD life. I would also like to thank other colleagues and former colleagues in the Database and Data Mining group including Nikolay, Vlado, Decebal, Odysseas , Rafael, Alejandro, Anil, Simon, Tita, Ricky, Samaneh, Tita, Oren, Pieter, Loek, Bilge, Marijn, and all the ones not explicitly mentioned here, with whom I had the same great discussion and nice chats. It was really fun to discuss all the things related to the PhD life and all the other interesting things not related to the PhD life.

Special thanks to my dear parents who have been so supportive and understanding throughout my life. Your love and kindness are enormous even on such a long distance. It does not matter how far I am, you are always my first help. Last but not least, I would like to thank my beloved wife Shengzhe for always supporting me and being by my side.

Yulong Pei
Eindhoven, January 2020

Curriculum Vitae

Yulong Pei was born on November 17, 1988 in Xinyang, China. He received the bachelor degree in Computer Science and Technology from Jilin University in 2010. Then, he received his masters in Computer Science from Peking University in 2013 and in Artificial Intelligence from Carnegie Mellon University in 2015, respectively. During this period, he completed his thesis on the topics of natural language processing and information retrieval. In August 2015, he started working as a PhD student in the Department of Mathematics and Computer Science at Eindhoven University of Technology under the supervision of dr. George Fletcher and prof. dr. Mykola Pechenizkiy, of which the results are presented in this dissertation.

List of Publications

Yulong Pei has the following publications:

Conferences and Journals

- **Yulong Pei**, Xin Du, Jianpeng Zhang, George Fletcher, Mykola Pechenizkiy. struc2gauss: Structural Role Preserving Network Embedding via Gaussian Embedding*. In *Data Mining and Knowledge Discovery*, under review.
- **Yulong Pei**, Xin Du, George Fletcher, Mykola Pechenizkiy. Dynamic Network Representation Learning via Gaussian Embedding*. In *NeurIPS 2019 Workshop on Graph Representation Learning*, 2019.
- **Yulong Pei**, George Fletcher, Mykola Pechenizkiy. Joint Role and Community Detection in Networks via L2,1 Norm Regularized Nonnegative Matrix Tri-Factorization*. In *ASONAM*, 2019.
- **Yulong Pei**, Jianpeng Zhang, George Fletcher, Mykola Pechenizkiy. Infinite Motif Stochastic Blockmodel for Role Discovery in Networks*. In *ASONAM*, 2019.
- Jianpeng Zhang, **Yulong Pei**, George Fletcher, Mykola Pechenizkiy. Evaluation of the Sample Clustering Process on Graphs. In *IEEE Transactions on Knowledge and Data Engineering*, 2019.
- Negar Ahmadi, **Yulong Pei**, Mykola Pechenizkiy. Effect of linear mixing in EEG on synchronization and complex network measures studied using the Kuramoto model. In *Physica A: Statistical Mechanics and its Applications*, 2019.
- Jianpeng Zhang, Kaijie Zhu, **Yulong Pei**, George Fletcher, Mykola Pechenizkiy. Cluster-preserving sampling from fully-dynamic streaming graphs. In *Information Sciences*, 2019.
- **Yulong Pei**, Jianpeng Zhang, George Fletcher, Mykola Pechenizkiy. DyNMF: Role Analytics in Dynamic Social Networks*. In *IJCAI*, 2018.
- Wouter Ligtenberg, **Yulong Pei**, George Fletcher, Mykola Pechenizkiy. Tink: a temporal graph analytics library for Apache Flink. Poster in *WWW (Companion Volume)*, 2018.

- Rosa Sicilia, Stella Lo Giudice, **Yulong Pei**, Mykola Pechenizkiy, Paolo Soda. Twitter rumour detection in the health domain. In *Expert Systems with Applications*, 2018.
- Negar Ahmadi, **Yulong Pei**, Mykola Pechenizkiy. Detection of alcoholism based on EEG signals and functional brain network features extraction. In *IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*, 2017.
- Jianpeng Zhang, Kaijie Zhu, **Yulong Pei**, George Fletcher, Mykola Pechenizkiy. Clustering-structure representative sampling from graph streams. In *International Conference on Complex Networks and their Applications*, 2017.
- **Yulong Pei**, Jianpeng Zhang, George Fletcher, Mykola Pechenizkiy. Node Classification in Dynamic Social Networks*. In *AALTD'2016: 2nd ECML/PKDD International Workshop on Advanced Analytics and Learning on Temporal Data*, 2016.
- Jianpeng Zhang, **Yulong Pei**, George Fletcher, Mykola Pechenizkiy. Structural measures of clustering quality on graph samples. In *ASONAM*, 2016.

Preprint

- Shiwei Liu, Decebal Constantin Mocanu, Amarsagar Reddy Ramapuram Matavalam, **Yulong Pei**, Mykola Pechenizkiy. Sparse evolutionary Deep Learning with over one million artificial neurons on commodity hardware. *arXiv preprint arXiv:1901.09181*, 2019.
- Wouter Ligtenberg, **Yulong Pei**. Introduction to a temporal graph benchmark. *arXiv preprint arXiv:1703.02852*, 2017.

SIKS dissertations

2011

01 Botond Cseke (RUN), *Variational Algorithms for Bayesian Inference in Latent Gaussian Models.*
02 Nick Tinnemeier(UU), *Organizing Agent Organizations. Syntax and Operational Semantics of an Organization-Oriented Programming Language.*
03 Jan Martijn van der Werf (TUE), *Compositional Design and Verification of Component-Based Information Systems.*
04 Hado van Hasselt (UU), *Insights in Reinforcement Learning; Formal analysis and empirical evaluation of temporal-difference.*
05 Base van der Raadt (VU), *Enterprise Architecture Coming of Age - Increasing the Performance of an Emerging Discipline..*
06 Yiwen Wang (TUE), *Semantically-Enhanced Recommendations in Cultural Heritage.*
07 Yujia Cao (UT), *Multimodal Information Presentation for High Load Human Computer Interaction.*
08 Nieske Vergunst (UU), *BDI-based Generation of Robust Task-Oriented Dialogues.*
09 Tim de Jong (OU), *Contextualised Mobile Media for Learning.*
10 Bart Bogaert (UvT), *Cloud Content Contention.*
11 Dhaval Vyas (UT), *Designing for Awareness: An Experience-focused HCI Perspective.*
12 Carmen Bratosin (TUE), *Grid Architecture for Distributed Process Mining.*
13 Xiaoyu Mao (UvT), *Airport under Control. Multiagent Scheduling for Airport Ground Handling.*
14 Milan Lovric (EUR), *Behavioral Finance and Agent-Based Artificial Markets.*
15 Marijn Koolen (UvA), *The Meaning of Structure: the Value of Link Evidence for Information Retrieval.*
16 Maarten Schadd (UM), *Selective Search in Games of Different Complexity.*
17 Jiyin He (UVA), *Exploring Topic Structure: Coherence, Diversity and Relatedness.*
18 Mark Ponsen (UM), *Strategic Decision-Making in complex games.*
19 Ellen Rusman (OU), *The Mind ' s Eye on Personal Profiles.*
20 Qing Gu (VU), *Guiding service-oriented software engineering - A view-based approach.*

21 Linda Terlouw (TUD), *Modularization and Specification of Service-Oriented Systems.*
22 Junte Zhang (UVA), *System Evaluation of Archival Description and Access.*
23 Wouter Weerkamp (UVA), *Finding People and their Utterances in Social Media.*
24 Herwin van Welbergen (UT), *Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior.*
25 Syed Waqar ul Qounain Jaffry (VU)), *Analysis and Validation of Models for Trust Dynamics.*
26 Matthijs Aart Pontier (VU), *Virtual Agents for Human Communication - Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots.*
27 Aniel Bhulai (VU), *Dynamic website optimization through autonomous management of design patterns.*
28 Rianne Kaptein(UVA), *Effective Focused Retrieval by Exploiting Query Context and Document Structure.*
29 Faisal Kamiran (TUE), *Discrimination-aware Classification.*
30 Egon van den Broek (UT), *Affective Signal Processing (ASP): Unraveling the mystery of emotions.*
31 Ludo Waltman (EUR), *Computational and Game-Theoretic Approaches for Modeling Bounded Rationality.*
32 Nees-Jan van Eck (EUR), *Methodological Advances in Bibliometric Mapping of Science.*
33 Tom van der Weide (UU), *Arguing to Motivate Decisions.*
34 Paolo Turrini (UU), *Strategic Reasoning in Interdependence: Logical and Game-theoretical Investigations.*
35 Maaike Harbers (UU), *Explaining Agent Behavior in Virtual Training.*
36 Erik van der Spek (UU), *Experiments in serious game design: a cognitive approach.*
37 Adriana Burlutiu (RUN), *Machine Learning for Pairwise Data, Applications for Preference Learning and Supervised Network Inference.*
38 Nyree Lemmens (UM), *Bee-inspired Distributed Optimization.*
39 Joost Westra (UU), *Organizing Adaptation using Agents in Serious Games.*
40 Viktor Clerc (VU), *Architectural Knowledge Management in Global Software Development.*
41 Luan Ibraimi (UT), *Cryptographically Enforced Distributed Data Access Control.*
42 Michal Sindlar (UU), *Explaining Behavior through Mental State Attribution.*
43 Henk van der Schuur (UU), *Process Improvement through Software Operation Knowledge.*
44 Boris Reuderink (UT), *Robust Brain-Computer Interfaces.*
45 Herman Stehouwer (UvT), *Statistical Language Models for Alternative Sequence Selection.*
46 Beibei Hu (TUD), *Towards Contextualized Information Delivery: A Rule-based Architecture for the Domain of Mobile Police Work.*
47 Azizi Bin Ab Aziz(VU), *Exploring Computational Models for Intelligent Support of Persons with Depression.*

48 Mark Ter Maat (UT), *Response Selection and Turn-taking for a Sensitive Artificial Listening Agent.*
49 Andreea Niculescu (UT), *Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality.*

2012

01 Terry Kakeeto (UvT), *Relationship Marketing for SMEs in Uganda.*
02 Muhammad Umair(VU), *Adaptivity, emotion, and Rationality in Human and Ambient Agent Models.*
03 Adam Vanya (VU), *Supporting Architecture Evolution by Mining Software Repositories.*
04 Jurriaan Souer (UU), *Development of Content Management System-based Web Applications.*
05 Marijn Plomp (UU), *Maturing Interorganisational Information Systems.*
06 Wolfgang Reinhardt (OU), *Awareness Support for Knowledge Workers in Research Networks.*
07 Rianne van Lambalgen (VU), *When the Going Gets Tough: Exploring Agent-based Models of Human Performance under Demanding Conditions.*
08 Gerben de Vries (UVA), *Kernel Methods for Vessel Trajectories.*
09 Ricardo Neisse (UT), *Trust and Privacy Management Support for Context-Aware Service Platforms.*
10 David Smits (TUE), *Towards a Generic Distributed Adaptive Hypermedia Environment.*
11 J.C.B. Rantham Prabhakara (TUE), *Process Mining in the Large: Preprocessing, Discovery, and Diagnostics.*
12 Kees van der Sluijs (TUE), *Model Driven Design and Data Integration in Semantic Web Information Systems.*
13 Suleman Shahid (UvT), *Fun and Face: Exploring non-verbal expressions of emotion during playful interactions.*
14 Evgeny Knutov(TUE), *Generic Adaptation Framework for Unifying Adaptive Web-based Systems.*
15 Natalie van der Wal (VU), *Social Agents. Agent-Based Modelling of Integrated Internal and Social Dynamics of Cognitive and Affective Processes..*
16 Fiemke Both (VU), *Helping people by understanding them - Ambient Agents supporting task execution and depression treatment.*
17 Amal Elgammal (UvT), *Towards a Comprehensive Framework for Business Process Compliance.*
18 Eltjo Poort (VU), *Improving Solution Architecting Practices.*
19 Helen Schonenberg (TUE), *What's Next? Operational Support for Business Process Execution.*
20 Ali Bahramisharif (RUN), *Covert Visual Spatial Attention, a Robust Paradigm for Brain-*

Computer Interfacing.
21 Roberto Cornacchia (TUD), *Querying Sparse Matrices for Information Retrieval.*
22 Thijs Vis (UvT), *Intelligence, politie en veiligheidsdienst: verenigbare grootheden?.*
23 Christian Muehl (UT), *Toward Affective Brain-Computer Interfaces: Exploring the Neurophysiology of Affect during Human Media Interaction.*
24 Laurens van der Werff (UT), *Evaluation of Noisy Transcripts for Spoken Document Retrieval.*
25 Silja Eckartz (UT), *Managing the Business Case Development in Inter-Organizational IT Projects: A Methodology and its Application.*
26 Emile de Maat (UVA), *Making Sense of Legal Text.*
27 Hayrettin Gurkok (UT), *Mind the Sheep! User Experience Evaluation & Brain-Computer Interface Games.*
28 Nancy Pascall (UvT), *Engendering Technology Empowering Women.*
29 Almer Tigelaar (UT), *Peer-to-Peer Information Retrieval.*
30 Alina Pommeranz (TUD), *Designing Human-Centered Systems for Reflective Decision Making.*
31 Emily Bagarukayo (RUN), *A Learning by Construction Approach for Higher Order Cognitive Skills Improvement, Building Capacity and Infrastructure.*
32 Wietske Visser (TUD), *Qualitative multi-criteria preference representation and reasoning.*
33 Rory Sie (OUN), *Coalitions in Cooperation Networks (COCOON).*
34 Pavol Jancura (RUN), *Evolutionary analysis in PPI networks and applications.*
35 Evert Haasdijk (VU), *Never Too Old To Learn – On-line Evolution of Controllers in Swarm- and Modular Robotics.*
36 Denis Ssebugwawo (RUN), *Analysis and Evaluation of Collaborative Modeling Processes.*
37 Agnes Nakakawa (RUN), *A Collaboration Process for Enterprise Architecture Creation.*
38 Selmar Smit (VU), *Parameter Tuning and Scientific Testing in Evolutionary Algorithms.*
39 Hassan Fatemi (UT), *Risk-aware design of value and coordination networks.*
40 Agus Gunawan (UvT), *Information Access for SMEs in Indonesia.*
41 Sebastian Kelle (OU), *Game Design Patterns for Learning.*
42 Dominique Verpoorten (OU), *Reflection Amplifiers in self-regulated Learning.*
43 Withdrawn, .
44 Anna Tordai (VU), *On Combining Alignment Techniques.*
45 Benedikt Kratz (UvT), *A Model and Language for Business-aware Transactions.*
46 Simon Carter (UVA), *Exploration and Exploitation of Multilingual Data for Statistical Machine Translation.*
47 Manos Tsagkias (UVA), *Mining Social Media: Tracking Content and Predicting Behavior.*
48 Jorn Bakker (TUE), *Handling Abrupt Changes in Evolving Time-series Data.*
49 Michael Kaisers (UM), *Learning against Learning - Evolutionary dynamics of reinforcement learning algorithms in strategic interactions.*

50 Steven van Kervel (TUD), *Ontologogy driven Enterprise Information Systems Engineering.*
51 Jeroen de Jong (TUD), *Heuristics in Dynamic Sceduling; a practical framework with a case study in elevator dispatching.*

2013

01 Viorel Milea (EUR), *News Analytics for Financial Decision Support.*
02 Erietta Liarou (CWI), *MonetDB/DataCell: Leveraging the Column-store Database Technology for Efficient and Scalable Stream Processing.*
03 Szymon Klarman (VU), *Reasoning with Contexts in Description Logics.*
04 Chetan Yadati(TUD), *Coordinating autonomous planning and scheduling.*
05 Dulce Pumareja (UT), *Groupware Requirements Evolutions Patterns.*
06 Romulo Goncalves(CWI), *The Data Cyclotron: Juggling Data and Queries for a Data Warehouse Audience.*
07 Giel van Lankveld (UvT), *Quantifying Individual Player Differences.*
08 Robbert-Jan Merk(VU), *Making enemies: cognitive modeling for opponent agents in fighter pilot simulators.*
09 Fabio Gori (RUN), *Metagenomic Data Analysis: Computational Methods and Applications.*
10 Jeewanie Jayasinghe Arachchige(UvT), *A Unified Modeling Framework for Service Design..*
11 Evangelos Pournaras(TUD), *Multi-level Reconfigurable Self-organization in Overlay Services.*
12 Marian Razavian(VU), *Knowledge-driven Migration to Services.*
13 Mohammad Safiri(UT), *Service Tailoring: User-centric creation of integrated IT-based homecare services to support independent living of elderly.*
14 Jafar Tanha (UVA), *Ensemble Approaches to Semi-Supervised Learning Learning.*
15 Daniel Hennes (UM), *Multiagent Learning - Dynamic Games and Applications.*
16 Eric Kok (UU), *Exploring the practical benefits of argumentation in multi-agent deliberation.*
17 Koen Kok (VU), *The PowerMatcher: Smart Coordination for the Smart Electricity Grid.*
18 Jeroen Janssens (UvT), *Outlier Selection and One-Class Classification.*
19 Renze Steenhuizen (TUD), *Coordinated Multi-Agent Planning and Scheduling.*
20 Katja Hofmann (UvA), *Fast and Reliable Online Learning to Rank for Information Retrieval.*
21 Sander Wubben (UvT), *Text-to-text generation by monolingual machine translation.*
22 Tom Claassen (RUN), *Causal Discovery and Logic.*
23 Patricio de Alencar Silva(UvT), *Value Activity Monitoring.*
24 Haitham Bou Ammar (UM), *Automated Transfer in Reinforcement Learning.*

25 Agnieszka Anna Latoszek-Berendsen (UM), *Intention-based Decision Support. A new way of representing and implementing clinical guidelines in a Decision Support System.*
26 Alireza Zarghami (UT), *Architectural Support for Dynamic Homecare Service Provisioning.*
27 Mohammad Huq (UT), *Inference-based Framework Managing Data Provenance.*
28 Frans van der Sluis (UT), *When Complexity becomes Interesting: An Inquiry into the Information eXperience.*
29 Iwan de Kok (UT), *Listening Heads.*
30 Joyce Nakatumba (TUE), *Resource-Aware Business Process Management: Analysis and Support.*
31 Dinh Khoa Nguyen (UvT), *Blueprint Model and Language for Engineering Cloud Applications.*
32 Kamakshi Rajagopal (OUN), *Networking For Learning; The role of Networking in a Lifelong Learner's Professional Development.*
33 Qi Gao (TUD), *User Modeling and Personalization in the Microblogging Sphere.*
34 Kien Tjin-Kam-Jet (UT), *Distributed Deep Web Search.*
35 Abdallah El Ali (UvA), *Minimal Mobile Human Computer Interaction.*
36 Than Lam Hoang (TUe), *Pattern Mining in Data Streams.*
37 Dirk Börner (OUN), *Ambient Learning Displays.*
38 Eelco den Heijer (VU), *Autonomous Evolutionary Art.*
39 Joop de Jong (TUD), *A Method for Enterprise Ontology based Design of Enterprise Information Systems.*
40 Pim Nijssen (UM), *Monte-Carlo Tree Search for Multi-Player Games.*
41 Jochem Liem (UVA), *Supporting the Conceptual Modelling of Dynamic Systems: A Knowledge Engineering Perspective on Qualitative Reasoning.*
42 Léon Planken (TUD), *Algorithms for Simple Temporal Reasoning.*
43 Marc Bron (UVA), *Exploration and Contextualization through Interaction and Concepts.*

2014

01 Nicola Barile (UU), *Studies in Learning Monotone Models from Data.*
02 Fiona Tuliyano (RUN), *Combining System Dynamics with a Domain Modeling Method.*
03 Sergio Raul Duarte Torres (UT), *Information Retrieval for Children: Search Behavior and Solutions.*
04 Hanna Jochmann-Mannak (UT), *Websites for children: search strategies and interface design - Three studies on children's search performance and evaluation.*
05 Jurriaan van Reijsen (UU), *Knowledge Perspectives on Advancing Dynamic Capability.*
06 Damian Tamburri (VU), *Supporting Networked Software Development.*
07 Arya Adriansyah (TUE), *Aligning Observed and Modeled Behavior.*

08 Samur Araujo (TUD), *Data Integration over Distributed and Heterogeneous Data Endpoints.*
09 Philip Jackson (UvT), *Toward Human-Level Artificial Intelligence: Representation and Computation of Meaning in Natural Language.*
10 Ivan Salvador Razo Zapata (VU), *Service Value Networks.*
11 Janneke van der Zwaan (TUD), *An Empathic Virtual Buddy for Social Support.*
12 Willem van Willigen (VU), *Look Ma, No Hands: Aspects of Autonomous Vehicle Control.*
13 Arlette van Wissen (VU), *Agent-Based Support for Behavior Change: Models and Applications in Health and Safety Domains.*
14 Yangyang Shi (TUD), *Language Models With Meta-information.*
15 Natalya Mogles (VU), *Agent-Based Analysis and Support of Human Functioning in Complex Socio-Technical Systems: Applications in Safety and Healthcare.*
16 Krystyna Milian (VU), *Supporting trial recruitment and design by automatically interpreting eligibility criteria.*
17 Kathrin Dentler (VU), *Computing healthcare quality indicators automatically: Secondary Use of Patient Data and Semantic Interoperability.*
18 Mattijs Ghijsen (VU), *Methods and Models for the Design and Study of Dynamic Agent Organizations.*
19 Vincius Ramos (TUE), *Adaptive Hypermedia Courses: Qualitative and Quantitative Evaluation and Tool Support.*
20 Mena Habib (UT), *Named Entity Extraction and Disambiguation for Informal Text: The Missing Link.*
21 Kassidy Clark (TUD), *Negotiation and Monitoring in Open Environments.*
22 Marieke Peeters (UT), *Personalized Educational Games - Developing agent-supported scenario-based training.*
23 Eleftherios Sidirourgos (UvA/CWI), *Space Efficient Indexes for the Big Data Era.*
24 Davide Ceolin (VU), *Trusting Semi-structured Web Data.*
25 Martijn Lappenschaar (RUN), *New network models for the analysis of disease interaction.*
26 Tim Baarslag (TUD), *What to Bid and When to Stop.*
27 Rui Jorge Almeida (EUR), *Conditional Density Models Integrating Fuzzy and Probabilistic Representations of Uncertainty.*
28 Anna Chmielowiec (VU), *Decentralized k-Clique Matching.*
29 Jaap Kabbedijk (UU), *Variability in Multi-Tenant Enterprise Software.*
30 Peter de Kock Berenschot (UvT), *Anticipating Criminal Behaviour.*
31 Leo van Moergestel (UU), *Agent Technology in Agile Multiparallel Manufacturing and Product Support.*
32 Naser Ayat (UVA), *On Entity Resolution in Probabilistic Data.*
33 Tesfa Tegegne Asfaw (RUN), *Service Discovery in eHealth.*
34 Christina Manteli (VU), *The Effect of Governance in Global Software Development: Analyzing Transactive Memory Systems.*

35 Joost van Oijen (UU), *Cognitive Agents in Virtual Worlds: A Middleware Design Approach.*
36 Joos Buijs (TUE), *Flexible Evolutionary Algorithms for Mining Structured Process Models.*
37 Maral Dadvar (UT), *Experts and Machines United Against Cyberbullying.*
38 Danny Plass-Oude Bos (UT), *Making brain-computer interfaces better: improving usability through post-processing..*
39 Jasmina Maric (UvT), *Web Communities, Immigration, and Social Capital.*
40 Walter Omona (RUN), *A Framework for Knowledge Management Using ICT in Higher Education.*
41 Frederic Hogenboom (EUR), *Automated Detection of Financial Events in News Text.*
42 Carsten Eijckhof (CWI/TUD), *Contextual Multidimensional Relevance Models.*
43 Kevin Vlaanderen (UU), *Supporting Process Improvement using Method Increments.*
44 Paulien Meesters (UvT), *Intelligent Blauw. Met als ondertitel: Intelligence-gestuurde politiezorg in gebiedsgebonden eenheden..*
45 Birgit Schmitz (OUN), *Mobile Games for Learning: A Pattern-Based Approach.*
46 Ke Tao (TUD), *Social Web Data Analytics: Relevance, Redundancy, Diversity.*
47 Shangsong Liang (UVA), *Fusion and Diversification in Information Retrieval.*

2015

01 Niels Netten (UvA), *Machine Learning for Relevance of Information in Crisis Response.*
02 Faiza Bukhsh (UvT), *Smart auditing: Innovative Compliance Checking in Customs Controls.*
03 Twan van Laarhoven (RUN), *Machine learning for network data.*
04 Howard Spoelstra (OUN), *Collaborations in Open Learning Environments.*
05 Christoph Bösch(UT), *Cryptographically Enforced Search Pattern Hiding.*
06 Farideh Heidari (TUD), *Business Process Quality Computation - Computing Non-Functional Requirements to Improve Business Processes.*
07 Maria-Hendrike Peetz(UvA), *Time-Aware Online Reputation Analysis.*
08 Jie Jiang (TUD), *Organizational Compliance: An agent-based model for designing and evaluating organizational interactions.*
09 Randy Klaassen(UT), *HCI Perspectives on Behavior Change Support Systems.*
10 Henry Hermans (OUN), *OpenU: design of an integrated system to support lifelong learning.*
11 Yongming Luo(TUE), *Designing algorithms for big graph datasets: A study of computing bisimulation and joins.*
12 Julie M. Birkholz (VU), *Modi Operandi of Social Network Dynamics: The Effect of Context on Scientific Collaboration Networks.*
13 Giuseppe Procaccianti(VU), *Energy-Efficient Software.*

14 Bart van Straalen (UT), *A cognitive approach to modeling bad news conversations.*
15 Klaas Andries de Graaf (VU), *Ontology-based Software Architecture Documentation.*
16 Changyun Wei (UT), *Cognitive Coordination for Cooperative Multi-Robot Teamwork.*
17 André van Cleeff (UT), *Physical and Digital Security Mechanisms: Properties, Combinations and Trade-offs.*
18 Holger Pirk (CWI), *Waste Not, Want Not! - Managing Relational Data in Asymmetric Memories.*
19 Bernardo Tabuenca (OUN), *Ubiquitous Technology for Lifelong Learners.*
20 Loïs Vanhée(UU), *Using Culture and Values to Support Flexible Coordination.*
21 Sibren Fetter (OUN), *Using Peer-Support to Expand and Stabilize Online Learning.*
22 Zhemin Zhu(UT), *Co-occurrence Rate Networks.*
23 Luit Gazendam (VU), *Cataloguer Support in Cultural Heritage.*
24 Richard Berendsen (UVA), *Finding People, Papers, and Posts: Vertical Search Algorithms and Evaluation.*
25 Steven Woudenberg (UU), *Bayesian Tools for Early Disease Detection.*
26 Alexander Hogenboom (EUR), *Sentiment Analysis of Text Guided by Semantics and Structure.*
27 Sándor Héman (CWI), *Updating compressed colomn stores.*
28 Janet Bagorogoza(TiU), *KNOWLEDGE MANAGEMENT AND HIGH PERFORMANCE; The Uganda Financial Institutions Model for HPO.*
29 Hendrik Baier (UM), *Monte-Carlo Tree Search Enhancements for One-Player and Two-Player Domains.*
30 Kiavash Bahreini(OU), *Real-time Multimodal Emotion Recognition in E-Learning.*
31 Yakup Koç (TUD), *On the robustness of Power Grids.*
32 Jerome Gard(UL), *Corporate Venture Management in SMEs.*
33 Frederik Schadd (TUD), *Ontology Mapping with Auxiliary Resources.*
34 Victor de Graaf(UT), *Gesocial Recommender Systems.*
35 Jungxao Xu (TUD), *Affective Body Language of Humanoid Robots: Perception and Effects in Human Robot Interaction.*

2016

01 Syed Saiden Abbas (RUN), *Recognition of Shapes by Humans and Machines.*
02 Michiel Christiaan Meulendijk (UU), *Optimizing medication reviews through decision support: prescribing a better pill to swallow.*
03 Maya Sappelli (RUN), *Knowledge Work in Context: User Centered Knowledge Worker Support.*
04 Laurens Rietveld (VU), *Publishing and Consuming Linked Data.*
05 Evgeny Sherkhonov (UVA), *Expanded Acyclic Queries: Containment and an Application in Explaining Missing Answers.*

06 Michel Wilson (TUD), *Robust scheduling in an uncertain environment.*
07 Jeroen de Man (VU), *Measuring and modeling negative emotions for virtual training.*
08 Matje van de Camp (TiU), *A Link to the Past: Constructing Historical Social Networks from Unstructured Data.*
09 Archana Nottamkandath (VU), *Trusting Crowdsourced Information on Cultural Artefacts.*
10 George Karafotias (VUA), *Parameter Control for Evolutionary Algorithms.*
11 Anne Schuth (UVA), *Search Engines that Learn from Their Users.*
12 Max Knobbout (UU), *Logics for Modelling and Verifying Normative Multi-Agent Systems.*
13 Nana Baah Gyan (VU), *The Web, Speech Technologies and Rural Development in West Africa - An ICT4D Approach.*
14 Ravi Khadka (UU), *Revisiting Legacy Software System Modernization.*
15 Steffen Michels (RUN), *Hybrid Probabilistic Logics - Theoretical Aspects, Algorithms and Experiments.*
16 Guangliang Li (UVA), *Socially Intelligent Autonomous Agents that Learn from Human Reward.*
17 Berend Weel (VU), *Towards Embodied Evolution of Robot Organisms.*
18 Albert Meroño Peñuela (VU), *Refining Statistical Data on the Web.*
19 Julia Efremova (Tu/e), *Mining Social Structures from Genealogical Data.*
20 Daan Odijk (UVA), *Context & Semantics in News & Web Search.*
21 Alejandro Moreno Celleri (UT), *From Traditional to Interactive Playspaces: Automatic Analysis of Player Behavior in the Interactive Tag Playground.*
22 Grace Lewis (VU), *Software Architecture Strategies for Cyber-Foraging Systems.*
23 Fei Cai (UVA), *Query Auto Completion in Information Retrieval.*
24 Brend Wanders (UT), *Repurposing and Probabilistic Integration of Data; An Iterative and data model independent approach.*
25 Julia Kiseleva (TU/e), *Using Contextual Information to Understand Searching and Browsing Behavior.*
26 Dilhan Thilakarathne (VU), *In or Out of Control: Exploring Computational Models to Study the Role of Human Awareness and Control in Behavioural Choices, with Applications in Aviation and Energy Management Domains.*
27 Wen Li (TUD), *Understanding Geo-spatial Information on Social Media.*
28 Mingxin Zhang (TUD), *Large-scale Agent-based Social Simulation - A study on epidemic prediction and control.*
29 Nicolas Höning (TUD), *Peak reduction in decentralised electricity systems -Markets and prices for flexible planning.*
30 Ruud Mattheij (UvT), *The Eyes Have It.*
31 Mohammad Khelghati (UT), *Deep web content monitoring.*
32 Eelco Vriezekolk (UT), *Assessing Telecommunication Service Availability Risks for Crisis Organisations.*
33 Peter Bloem (UVA), *Single Sample Statistics, exercises in learning from just one exam-*

ple.
34 Dennis Schunselaar (TUE), *Configurable Process Trees: Elicitation, Analysis, and Enactment.*
35 Zhaochun Ren (UVA), *Monitoring Social Media: Summarization, Classification and Recommendation.*
36 Daphne Karreman (UT), *Beyond R2D2: The design of nonverbal interaction behavior optimized for robot-specific morphologies.*
37 Giovanni Sileno (UvA), *Aligning Law and Action - a conceptual and computational inquiry.*
38 Andrea Minuto (UT), *MATERIALS THAT MATTER - Smart Materials meet Art & Interaction Design.*
39 Merijn Bruijnes (UT), *Believable Suspect Agents; Response and Interpersonal Style Selection for an Artificial Suspect.*
40 Christian Detweiler (TUD), *Accounting for Values in Design.*
41 Thomas King (TUD), *Governing Governance: A Formal Framework for Analysing Institutional Design and Enactment Governance.*
42 Spyros Martzoukos (UVA), *Combinatorial and Compositional Aspects of Bilingual Aligned Corpora.*
43 Saskia Koldijk (RUN), *Context-Aware Support for Stress Self-Management: From Theory to Practice.*
44 Thibault Sellam (UVA), *Automatic Assistants for Database Exploration.*
45 Bram van de Laar (UT), *Experiencing Brain-Computer Interface Control.*
46 Jorge Gallego Perez (UT), *Robots to Make you Happy.*
47 Christina Weber (UL), *Real-time foresight - Preparedness for dynamic innovation networks.*
48 Tanja Buttler (TUD), *Collecting Lessons Learned.*
49 Gleb Polevoy (TUD), *Participation and Interaction in Projects. A Game-Theoretic Analysis.*
50 Yan Wang (UVT), *The Bridge of Dreams: Towards a Method for Operational Performance Alignment in IT-enabled Service Supply Chains.*

2017

01 Jan-Jaap Oerlemans (UL), *Investigating Cybercrime.*
02 Sjoerd Timmer (UU), *Designing and Understanding Forensic Bayesian Networks using Argumentation.*
03 Daniël Harold Telgen (UU), *Grid Manufacturing; A Cyber-Physical Approach with Autonomous Products and Reconfigurable Manufacturing Machines.*
04 Mrunal Gawade (CWI), *MULTI-CORE PARALLELISM IN A COLUMN-STORE.*
05 Mahdieh Shadi (UVA), *Collaboration Behavior.*

06 Damir Vandic (EUR), *Intelligent Information Systems for Web Product Search.*
07 Roel Bertens (UU), *Insight in Information: from Abstract to Anomaly.*
08 Rob Konijn (VU), *Detecting Interesting Differences:Data Mining in Health Insurance Data using Outlier Detection and Subgroup Discovery.*
09 Dong Nguyen (UT), *Text as Social and Cultural Data: A Computational Perspective on Variation in Text.*
10 Robby van Delden (UT), *(Steering) Interactive Play Behavior.*
11 Florian Kunneman (RUN), *Modelling patterns of time and emotion in Twitter #anticipointment.*
12 Sander Leemans (TUE), *Robust Process Mining with Guarantees.*
13 Gijs Huisman (UT), *Social Touch Technology - Extending the reach of social touch through haptic technology.*
14 Shoshannah Tekofsky (UvT), *You Are Who You Play You Are: Modelling Player Traits from Video Game Behavior.*
15 Peter Berck, Radboud University (RUN), *Memory-Based Text Correction.*
16 Aleksandr Chuklin (UVA), *Understanding and Modeling Users of Modern Search Engines.*
17 Daniel Dimov (UL), *Crowdsourced Online Dispute Resolution.*
18 Ridho Reinanda (UVA), *Entity Associations for Search.*
19 Jeroen Vuurens (TUD), *Proximity of Terms, Texts and Semantic Vectors in Information Retrieval.*
20 Mohammadbashir Sedighi (TUD), *Fostering Engagement in Knowledge Sharing: The Role of Perceived Benefits, Costs and Visibility.*
21 Jeroen Linssen (UT), *Meta Matters in Interactive Storytelling and Serious Gaming (A Play on Worlds).*
22 Sara Magliacane (VU), *Logics for causal inference under uncertainty.*
23 David Graus (UVA), *Entities of Interest— Discovery in Digital Traces.*
24 Chang Wang (TUD), *Use of Affordances for Efficient Robot Learning.*
25 Veruska Zamborlini (VU), *Knowledge Representation for Clinical Guidelines, with applications to Multimorbidity Analysis and Literature Search.*
26 Merel Jung (UT), *Socially intelligent robots that understand and respond to human touch.*
27 Michiel Joosse (UT), *Investigating Positioning and Gaze Behaviors of Social Robots: People's Preferences, Perceptions and Behaviors.*
28 John Klein (VU), *Architecture Practices for Complex Contexts.*
29 Adel Alhuraibi (UVT), *From IT-Business Strategic Alignment to Performance: A Moderated Mediation Model of Social Innovation, and Enterprise Governance of IT.*
30 Wilma Latuny (UVT), *The Power of Facial Expressions.*
31 Ben Ruijl (UL), *Advances in computational methods for QFT calculations.*
32 Thaer Samar (RUN), *Access to and Retrievability of Content in Web Archives.*
33 Brigit van Loggem (OU), *Towards a Design Rationale for Software Documentation: A Model of Computer-Mediated Activity.*

34 Maren Scheffel (OUN), *The Evaluation Framework for Learning Analytics.*
35 Martine de Vos (VU), *Interpreting natural science spreadsheets.*
36 Yuanhao Guo (UL), *Shape Analysis for Phenotype Characterisation from High-throughput Imaging.*
37 Alejandro Montes García (TUE), *WiBAF: A Within Browser Adaptation Framework that Enables Control over Privacy.*
38 Alex Kayal (TUD), *Normative Social Applications.*
39 Sara Ahmadi (RUN), *Exploiting properties of the human auditory system and compressive sensing methods to increase noise robustness in ASR.*
40 Altaf Hussain Abro (VUA), *Steer your Mind: Computational Exploration of Human Control in Relation to Emotions, Desires and Social Support For applications in human-aware support systems.*
41 Adnan Manzoor (VUA), *Minding a Healthy Lifestyle: An Exploration of Mental Processes and a Smart Environment to Provide Support for a Healthy Lifestyle.*
42 Elena Sokolova (RUN), *Causal discovery from mixed and missing data with applications on ADHD datasets.*
43 Maaike de Boer (RUN), *Semantic Mapping in Video Retrieval.*
44 Garm Lucassen (UU), *Understanding User Stories - Computational Linguistics in Agile Requirements Engineering.*
45 Bas Testerink (UU), *Decentralized Runtime Norm Enforcement.*
46 Jan Schneider (OU), *Sensor-based Learning Support.*
47 Yie Yang (TUD), *Crowd Knowledge Creation Acceleration.*
48 Angel Suarez (OU), *Colloborative inquiry-based learning.*

2018

01 Han van der Aa (VUA), *Comparing and Aligning Process Representations.*
02 Felix Mannhardt (TUE), *Multi-perspective Process Mining.*
03 Steven Bosems (UT), *Causal Models For Well-Being: Knowledge Modeling, Model-Driven Development of Context-Aware Applications, and Behavior Prediction.*
04 Jordan Janeiro (TUD), *Flexible Coordination Support for Diagnosis Teams in Data-Centric Engineering Tasks.*
05 Hugo Huurdeman (UVA), *Supporting the Complex Dynamics of the Information Seeking Process.*
06 Dan Ionita (UT), *Model-Driven Information Security Risk Assessment of Socio-Technical Systems.*
07 Jieting Luo (UU), *A formal account of opportunism in multi-agent systems.*
08 Rick Smetsers (RUN), *Advances in Model Learning for Software Systems.*
09 Xu Xie (TUD), *Data Assimilation in Discrete Event Simulations.*
10 Julia S. Mollee (VU), *Moving forward: supporting physical activity behavior change*

through intelligent technology.
11 Mahdi Sargolzaei (UVA), *Enabling Framework for Service-oriented Collaborative Networks.*
12 Xixi Lu (TUE), *Using Behavioral Context in Process Mining.*
13 Seyed Amin Tabatabaei (VUA), *Computing a Sustainable Future: Exploring the added value of computational models for increasing the use of renewable energy in the residential sector.*
14 Bart Joosten (UVT), *Detecting Social Signals with Spatiotemporal Gabor Filters.*
15 Naser Davarzani (UM), *Biomarker discovery in heart failure.*
16 Jaebok Kim (UT), *Automatic recognition of engagement and emotion in a group of children.*
17 Jianpeng Zhang (TU/e), *On Graph Sample Clustering.*
18 Henriette Nakad (UL), *De Notaris en Private Rechtspraak.*
19 Minh Duc Pham (VUA), *Emergent relational schemas for RDF.*
20 Manxia Liu (RUN), *Time and Bayesian Networks.*
21 Aad Slootmaker (OUN), *EMERGO: a generic platform for authoring and playing scenario-based serious games.*
22 Eric Fernandes de Mello Araujo (VUA), *Contagious: Modeling the Spread of Behaviours, Perceptions and Emotions in Social Networks.*
23 Kim Schouten (EUR), *Semantics-driven Aspect-Based Sentiment Analysis.*
24 Jered Vroon (UT), *Responsive Social Positioning Behaviour for Semi-Autonomous Telepresence Robots.*
25 Riste Gligorov (VUA), *Serious Games in Audio-Visual Collections.*
26 Roelof Anne Jelle de Vries (UT), *Theory-Based and Tailor-Made: Motivational Messages for Behavior Change Technology.*
27 Maikel Leemans (TUE), *Hierarchical Process Mining for Scalable Software Analysis.*
28 Christian Willemse (UT), *Social Touch Technologies: How they feel and how they make you feel.*
29 Yu Gu (UVT), *Emotion Recognition from Mandarin Speech.*
30 Wouter Beek (VU), *The "K" in "semantic web" stands for "knowledge": scaling semantics to the web.*

2019

01 Rob van Eijk (UL), *Comparing and Aligning Process Representations.*
02 Emmanuelle Beauxis Aussalet (CWI, UU), *Statistics and Visualizations for Assessing Class Size Uncertainty.*
03 Eduardo Gonzalez Lopez de Murillas (TUE), *Process Mining on Databases: Extracting Event Data from Real Life Data Sources.*
04 Ridho Rahmadi (RUN), *Finding stable causal structures from clinical data.*

05 Sebastiaan van Zelst (TUE), *Process Mining with Streaming Data.*
06 Chris Dijkshoorn (VU), *Nichesourcing for Improving Access to Linked Cultural Heritage Datasets.*
07 Soude Fazeli (TUD), *Recommender Systems in Social Learning Platforms.*
08 Frits de Nijs (TUD), *Resource-constrained Multi-agent Markov Decision Processes.*
09 Fahimeh Alizadeh Moghaddam (UVA), *Self-adaptation for energy efficiency in software systems.*
10 Qing Chuan Ye (EUR), *Multi-objective Optimization Methods for Allocation and Prediction.*
11 Yue Zhao (TUD), *Learning Analytics Technology to Understand Learner Behavioral Engagement in MOOCs.*
12 Jacqueline Heinerman (VU), *Better Together.*
13 Guanliang Chen (TUD), *MOOC Analytics: Learner Modeling and Content Generation.*
14 Daniel Davis (TUD), *Large-Scale Learning Analytics: Modeling Learner Behavior & Improving Learning Outcomes in Massive Open Online Courses.*
15 Erwin Walraven (TUD), *Planning under Uncertainty in Constrained and Partially.*
16 Guangming Li (TUE), *Process Mining based on Object-Centric Behavioral Constraint (OCBC) Models.*
17 Ali Hurriyetoglu (RUN), *Extracting actionable information from microtexts.*
18 Gerard Wagenaar (UU), *Artefacts in Agile Team Communication.*
19 Vincent Koeman (TUD), *Tools for Developing Cognitive Agents.*
20 Chide Groenouwe (UU), *Fostering technically augmented human collective intelligence.*
21 Cong Liu (TUE), *Software Data Analytics: Architectural Model Discovery & Design Pattern Detection.*
22 Martin van den Berg (VU), *Improving IT Decisions with Enterprise Architecture.*
23 Qin Liu (TUD), *Intelligent Control Systems: Learning, Interpreting, Verification.*
24 Anca Dumitrache (VU), *Truth in Disagreement - Crowdsourcing Labeled Data for Natural Language Processing.*
25 Emiel van Miltenburg (VU), *Pragmatic factors in (automatic) image description.*
26 Prince Singh (UT), *An Integration Platform for Synchromodal Transport.*
27 Alessandra Antonaci (OUN), *The Gamification Design Process applied to (Massive) Open Online Courses.*
28 Esther Kuindersma (UL), *Cleared for take-off: Game-based learning to prepare airline pilots for critical situations.*
29 Daniel Formolo (VU), *Using virtual agents for simulation and training of social skills in safety-critical circumstances.*
30 Vahid Yazdanpanah (UT), *Multiagent Industrial Symbiosis Systems.*
31 Milan Jelisavcic (VU), *Alive and Kicking: Baby Steps in Robotics.*
32 Chiara Sironi (UM), *Monte-Carlo Tree Search for Artificial General Intelligence in Games.*
33 Anil Yaman (TUE), *Evolution of Biologically Inspired Learning in Artificial Neural Networks.*
34 Negar Ahmadi (TUE), *EEG Microstate and Functional Brain Network Features for Clas-*

sification of Epilepsy and PNES.
35 Lisa Facey-Shaw (OUN), *Gamification with digital badges in learning programming.*
36 Kevin Ackermans (OUN), *Designing Video-Enhanced Rubrics to Master Complex Skills.*
37 Jian Fang (TUD), *Database Acceleration on FPGAs.*
38 Ákos Kádár (TiU), *Learning visually grounded and multilingual representations.*

2020

01 Armon Toubman (UL), *Calculated Moves: Generating Air Combat Behaviour.*
02 Macos de Paula Bueno (UL), *Unraveling Temporal Processes using Probabilistic Graphical Models.*
03 Mostafa Deghani (UVA), *Learning with Imperfect Supervision for Language Understanding.*
04 Maarten van Gompel (RUN), *Context as Linguistic Bridges.*
05 Yulong Pei (TUE), *On Local and Global Graph Structure Mining.*