Structure-Aware Network Representation Learning on Graphs

A dissertation submitted in partial fulfillment of the requirements
for the award of the degree of

Doctor of Philosophy

Submitted by

Anasua Mitra

Under the supervision of

Dr. Sanasam Ranbir Singh, Prof. Diganta Goswami, Prof. Balaraman Ravindran

Department of Computer Science and Engineering
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
Guwahati 781039, Assam, India

October 2022

“Our greatest glory is not in never falling, but in rising every time we fall.”

Confucius

Dedicated to my parents, my fiancée, dear sister and her family for their unconditional love and support.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to Dr. Sanasam Ranbir Singh, Prof. Diganta Goswami, and Prof. Balaraman Ravindran, my Ph.D. supervisors. They are excellent mentors and have always supported me in all my research endeavors giving me opportunities to work on research problems that excite me the most. At the same time, their guidance helped in shaping ideas to concrete objectives while aiming at the ultimate goal. I will be ever indebted to my past supervisors, Dr. Amit Awekar and Dr. Deepanjan Kesh, with whom I embarked upon my Ph.D. journey. Dr. Amit Awekar has been an enthusiastic mentor, and working with him has always been very exciting. I express my heartfelt gratitude to my collaborators Prof. Srinivasan Parthasarathy and Priyesh Vijayan — their continued encouragement and guidance have been instrumental in my growth as a researcher and as a person. I am incredibly grateful and privileged to have had this opportunity to work with them.

I am also thankful to my present and past doctoral committee members – Dr. V. Vijaya Saradhi, Dr. Ashish Anand, Dr. Rashmi Dutta Baruah, Prof. Amit Sethi, and Prof. Sukumar Nandi for providing valuable feedback during the research. I hereby express my gratitude towards Prof. Hemangee K. Kapoor, Dr. John Jose, Prof. Pradip Kr. Das, Dr. R. Inkulu, for all their moral support. I also sincerely thank all the faculty members for their direct and indirect support.

I am grateful to Prof. Balaraman Ravindran, RISE Lab, and IIT Madras for accommodating me with excellent internship opportunities, which have enriched me as a researcher. I sincerely thank my mentor Dr. Sreyash Kenkre from IBM Research Lab India, for hosting me for a research internship. Interactions with numerous brilliant minds – Sudarsun Santhiappan, Tarun Kumar, Deepak Maurya, Yash Chandak, Beethika Tripathi, Nikita Moghe, Shweta Bhardwaj, Preksha Nema, Aakash Srinivasan, Prithwish Basu Roy, Sapana Chaudhary, Ujjawal Soni, Gayathri Ravichandran, Anirban Laha, Disha Srivastava, Tamali Banerjee during the internships always pushed me to strive for excellence in my research endeavors.

Also, I thank MHRD, the Government of India, for providing financial assistance throughout the Ph.D. program. Thanks to various funding agencies and esteemed conferences that awarded travel grants and fellowships encouraging participation in various research venues. I am immensely thankful to various funding agencies such as BRNS (project no. 2013/13/8-BRNS/10026), Dept. of Biotechnology (project no. BT/COE/34/SP28408/2018) and Dept. of CSE, Indian Institute of Technology Guwahati, for providing necessary computing resources used during the work. A vote of thanks to all the computing resources provided by the Reconfigurable Intelligent Systems Engineering (RISE) Lab, and Robert Bosch Centre for Data Science and Artificial Intelligence (RBC-DSAI), Indian Institute of Technology Madras.

I would also like to acknowledge the technical staff, system admins at IIT Guwahati and at RBC-DSAI, the ever-supportive administrative staff in the Dept. of CSE, Computer & Communication Centre, Academic Affairs, and Student Affairs.

I am particularly thankful to my fellow lab members of OSINT and CLST at IIT Guwahati, special mentions of Akash, Gyanendro, Neelakshi, Bornali, Durgesh, Hemanta, Sujit, Roshan, Rajib Sir, and many more. I would like to thank all the amazing peers at IIT Guwahati – Abhishek, Sriniwas, Arunangshu, Subrata, Brijesh, Abhijit, Palash, Aparajita, Manjari, Madhurima, Divya, Debanjan, Shrestha, Sheel, Sathish, Swarup, Saroj, Akshay, Arijit, Pawan, Prasen, Dipika, Pallabi, Meenakshi, Vanshali, Deepankar, Parikshit, Satya, Sadu, Vijay and many more. A vote of thanks to the seniors and juniors – Shounak, Shuvendu, Shibaji, Sisir Sir, Pranav Sir, Tania, Shilpa, Sonia, Nayantara, Saptarshi, Manojit, and many more.

I am fortunate to have several wonderful friends within and outside IIT Guwahati who were instrumental in providing me with strong support during my Ph.D. endeavor. Life never has a dull moment with them. I especially would like to thank Somrita Rudra, Animesh Ghosh, Debopam Basu, Sharmishta Chatterjee, Puja Mukherjee, Rajasrima Hore, Koyel Ghosh Hajra, Shawley Dey, Dipankar Mondal, Prashant Kr. Srivastava, Amrita Bhattacharjee, Deepika Bishnoi, Avijit Dutta, Tanmoy Kundu, Jaydeep Das, Ayan Kumar Bhowmick, Anirban Santara, Suvradip Chakraborty, and Arghya Pal. Thanks to the rest of my IIT Guwahati friends (including wall-mates and hostel-friends) who made my stay at IIT Guwahati lively and fun-filled.

I would like to thank my favorite set of people, the pillars of my strength — my parents (Mrs. Prativa Roy Mitra and Mr. Surjya Kr. Mitra), my fiancée (Mr. Debanjan Roy

Chowdhury), my elder sister (Mrs. Arundhati Mitra Basu), my brother-in-law (Mr. Debjit Basu), and my sweet nephew (Mr. Ishan Basu).

Finally, heartfelt thanks to IIT Guwahati for providing a beautiful serene campus with high-quality facilities. Last but not the least, a vote of thanks to the medical staff, security personnel, mess-staff, and cleaning-staff of IIT Guwahati.

Declaration

I certify that,

- The work contained in this thesis is original and has been done by myself and under the general supervision of my supervisor(s).
- The work reported herein has not been submitted to any other Institute for any degree or diploma.
- Whenever I have used materials (concepts, ideas, text, expressions, data, graphs, diagrams, theoretical analysis, results, etc.) from other sources, I have given due credit by citing them in the text of the thesis and giving their details in the references. Elaborate sentences used verbatim from published work have been clearly identified and quoted.
- I also affirm that no part of this thesis can be considered plagiarism to the best of my knowledge and understanding and take complete responsibility if any complaint arises.
- I am fully aware that my thesis supervisor(s) are not in a position to check for any possible instance of plagiarism within this submitted work.

Anasua Mitra

October 2022

Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati - 781039 Assam India

Certificate

This is to certify that this thesis entitled "**Structure-Aware Network Representation Learning on Graphs**" submitted by **Anasua Mitra**, in partial fulfilment of the requirements for the award of the degree of Doctor of Philosophy, to the Indian Institute of Technology Guwahati, Assam, India, is a record of the bonafide research work carried out by him under my guidance and supervision at the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati, Assam, India. To the best of my knowledge, no part of the work reported in this thesis has been presented for the award of any degree at any other institution.

Date: October 2022
Place: Guwahati

Dr. Sanasam Ranbir Singh	Prof. Diganta Goswami	Prof. Balaraman Ravindran
(Main-supervisor)	(Co-supervisor)	(External-supervisor)
Associate Professor	Professor	Professor
Dept. of C.S.E	Dept. of C.S.E	Dept. of C.S.E
IIT Guwahati	IIT Guwahati	IIT Madras

Abstract

Graphs provide a powerful way to model real-world scenarios. Entities in real-world data can be modeled as nodes in a graph. And, various kinds of interactions among the entities are modeled as edges in such a graph. It is a ubiquitous data structure to represent linked data from diverse domains — social, technological, biological, financial, transports, cellular networks, recommender systems, and many more. Several classes of graphs such as homogeneous, heterogeneous, multiplex graphs exist that cater to the need for modeling the specific nature of node interactions in real-world data. Nevertheless, modeling real-world data as graphs often suffers from challenges such as noise, incomplete and unobserved information, sparsity, heterogeneity, structural diversities, lack of annotations, the existence of imbalanced graph components, inter-dependencies of graph components, etc. Mining useful information and obtaining inference from the graphs are of utmost importance and have immense applicabilities. Traditional Machine Learning algorithms require useful features from the graphs as input to obtain insightful inferences. Earlier approaches to feature engineering often require user-defined heuristics, the intervention of domain experts, and critical resources. Recently, Graph Embedding, aka Network Representation Learning (NRL), has become a very popular means of automatically extracting useful latent features from graphs, especially for large graphs. The NRL methods aim to learn a mapping function to project high-dimensional non-Euclidean graph data (for representing nodes, edges, paths, subgraphs, or the entire graph) into a low dimensional latent embedding space optimally without compromising the underlying structural properties of the original graph.

In a trivial setup, NRL aims at incorporating local neighborhood contexts, aka microscopic views surrounding graph elements of interest. The microscopic views are often inadequate in learning discriminative features for various downstream tasks. As observed in recent studies, learning higher-order macroscopic views (global structure) can improve the discriminative capacity of the features for various applications. However, based on the complexities and scale of the underlying network, capturing the macroscopic view is a non-trivial task. This dissertation carefully examines existing research gaps while incorporating macroscopic views

and proposes three novel network embedding methods for homogeneous, heterogeneous, and multiplex graphs.

Past research efforts in learning structures in homogeneous graphs primarily focus on either capturing k-hop local neighborhood contexts of nodes or jointly learning communities to enrich node embeddings. All the community enforcing models use unsupervised clustering criteria based on either network-only node proximities or embedding-based node proximities. To address this, the thesis first investigates incorporating supervised non-network node proximity measures to group nodes in homogeneous graphs. The framework unifies ways to include supervision knowledge for enriched node embedding learning. Robust node classification and clustering performance are obtained even in challenging experiment setups, with varying ratios of class labels and different node-sampling strategies.

Next, this thesis considers improving the InfoMax based learning strategy as a useful means to incorporate global graph structures into node embeddings for multiplex graphs. InfoMax based learning provides a scalable way to incorporate both local and global node representations via maximizing Mutual Information (MI) between them. Nevertheless, in a typical setup, it uses a common global graph summary for all the local node embeddings, thereby encoding a lot of noisy and trivial information. The thesis proposes a novel way to contextualize global graph summaries for each node to encode non-trivial personalized graph summaries in node embeddings. The effectiveness of the proposed framework is verified with several downstream tasks, such as node classification, clustering, and similarity-search.

Finally, this thesis considers incorporating various structural contexts at multiple granularities between the nodes for improving link prediction performance in heterogeneous graphs. Very few research efforts have been made to understand various structural cues that exist for link prediction in heterogeneous graphs. Also, no NRL study has investigated the roles that communities of the end nodes play in predicting links between the nodes. To address this, this dissertation proposes a novel, first-of-its-kind community view of the edges in a graph. The proposed framework considers relational paths between the nodes and their communities apart from the popularly used common subgraph contexts. It also proposes a fine-grained attention mechanism to combine all the candidate contexts judiciously for link prediction. The framework outperforms the most recent benchmark heterogeneous link prediction method by a huge margin. Visualizing attention weights of candidate structural contexts establishes their usefulness and complementarity in aiding link prediction at various challenging scenarios. This dissertation shows that learning structure-aware network representations facilitates learning of enriched target graph-component embeddings that can benefit various downstream ML tasks.

Table of Contents

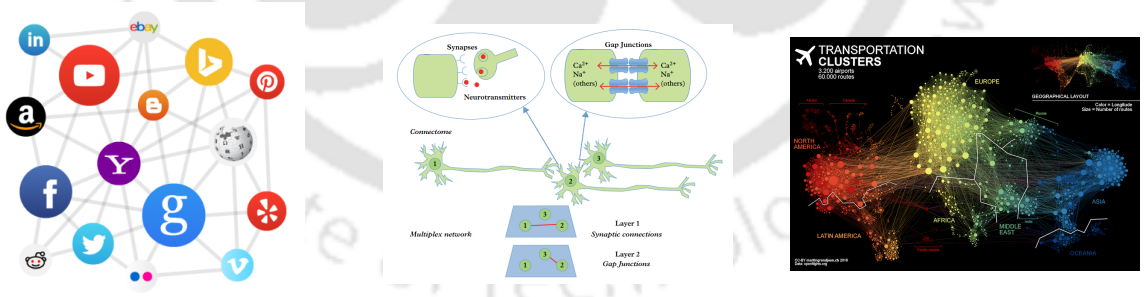
List of Figures

List of Tables

Chapter 1

Introduction

Graph is a *powerful* way to represent many real-world scenarios. It is a *ubiquitous data structure* to represent linked data. Entities in many real-world scenarios are related to each other in multiple ways. Such relations are often modeled as graph-structured data where nodes represent entities, and edges between a pair of nodes represent the interactions between the entities. From social networks to biological interactions, financial systems, recommender systems, transport networks, bibliographic domains, data from multiple modalities such as text, images, videos, signals – data from many more diverse fields can be modeled using graphs. Figure 1.1 shows examples of graph data from social, biological and transport domains.



(a) Social networks

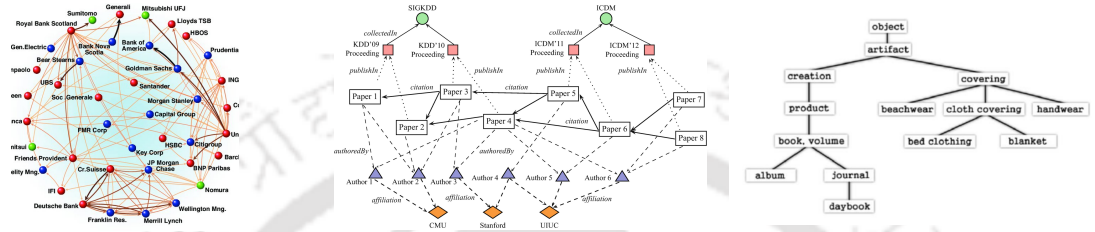
(b) Brain interactions

(c) Web of Transport

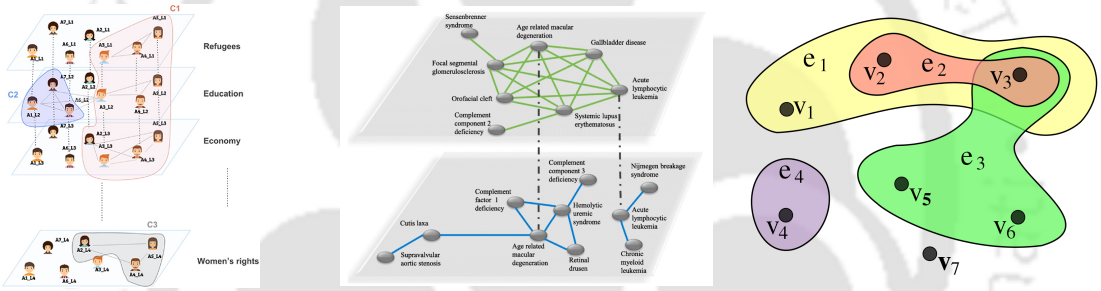
Fig. 1.1 Graphs from diverse domains [Image Source: Internet]

1.1 Representing real-world data as graphs

To represent the nature of interactions among the entities in networked data, various classes of graphs such as, *homogeneous, heterogeneous, tree, multilayer, multiplex, hypergraphs* etc. exist. Figure 1.2 elucidates more on representing relational data coming from various domains for different purposes. A *homogeneous network* is a simple graph structure whose node



(a) Homogeneous network (b) Heterogeneous network (c) Hierarchical network



(d) Multilayer network (e) Multiplex network (f) Hypergraph

Fig. 1.2 Various representations of networked data [Image Source: Internet]

interactions are modeled using a single adjacency matrix without any loss of information, i.e., the adjacency matrix representation is complete. Figure 1.2a shows a financial graph involving financial organizations, and their dependencies are modeled using directed edges. *Heterogeneous Information Networks (HINs)* is a kind of graph in which nodes and edges are associated with unique types, and nodes connect to each other via various relation types, conforming to the interaction rules of the underlying network schema. This is useful for modeling various relations exhibited among the nodes. Figure 1.2b shows an example bibliographic heterogeneous graph that includes paper, author, venue, institute entities in a linked manner. Often, interactions among a set of entities are hierarchical in nature. Ontologies, facts or knowledge, and biological components are examples of linked data that contain such hierarchical interactions. Figure 1.2c shows how a *tree (connected, undirected,*

acyclic graph) [1] can effectively model an *wordnet* [2]. Sometimes, graph analytics tasks require modeling relation-wise interactions of different types of entities in a specific manner and combining them. *Multilayer networks* are useful in representing such layer-wise interactions among the nodes. Figure 1.2d provides us with an example of how latent topics associated with user interactions are spread across and within relational layers in Twitter. Further, *multiplex networks* are a special case of multilayer networks, in which the same set of entities of the identical type are connected to each other via multiple types of relations, each relation representing a distinct layer. Figure 1.2e shows a disease-disease two-layer multiplex graph based on genotype [3] and phenotype [4] interactions. Whereas, *hypergraphs* [5] in Figure 1.2f provide us excellent ways to model higher-arity node-wise interactions instead of binary interactions between a node-pair. Thus, we can see that various complex graph structures exist, and they enable powerful ways of representing real-world networked data.

1.2 Machine Learning on graphs

Machine Learning (ML) on such graphs is one task of utmost importance as it has applicability in a diverse range of graph analytics tasks. Mining useful information from networked data such as predicting the role of an author in a collaboration network, annotating protein nodes with unknown biological functions in a protein-protein interactions graph, recommending new friends to a user in social network, finding unknown side effects of a combination of drugs in precision medicine, designing smart-city infrastructure from the web of multi-modal transportation, modeling disease progression using graphs generated from regions of the brain, discovering novel molecules with desired therapeutic properties, anomaly detection in transaction graphs — uncountable critical use-cases exist to justify why it is worthwhile to model real-world data as graphs and apply Machine Learning algorithm to obtain insightful inferences. Traditional ML algorithms require useful features from graphs as input to the inference algorithm. The input features can be based on — various node proximities including graph connectivity [8], graph summary statistics (degree, centrality, clustering coefficients etc.) [9], graph kernels [10], user-defined heuristics [11], handcrafted features curated by the domain experts [12] etc. This curation of useful features from graph data is a resource-

[1] In Mathematics, a connected acyclic undirected graph is known as a Tree. [5]

[2] WordNet is a lexical database of semantic relations between words in various languages. WordNet links words into semantic relations including synonyms, hyponyms, and meronyms. [6]

[3] https://en.wikipedia.org/wiki/Genotype

[4] https://en.wikipedia.org/wiki/Phenotype

[5] In mathematics, a hypergraph is a generalization of a graph in which an edge can join any number of vertices. In contrast, in an ordinary graph, an edge connects exactly two vertices. [7]

consuming, tedious process and often requires human intervention. Also, the large scale of information networks often makes network analytic tasks computationally expensive or intractable. Again, based on the target task, one might require to curate useful features for optimally representing any graph component of interests such as nodes, edges, subgraphs, whole graph, etc. However, there exists no straightforward way to best curate information from the high-dimensional non-Euclidean graph data into a feature vector.

1.3 Network Representation Learning: an introduction

Recently, Graph Embedding, aka Network Representation Learning (NRL) [13–15, 1], has become a very popular and scalable means of automatically extracting useful, complexly correlated features from graphs, especially of very large size. NRL has emerged as a new learning paradigm that embeds various graph components via a mapping function into low-dimensional vector space while preserving the underlying prominent network properties. Representation learning models aim to optimize learning of the mapping function so that the geometric relationship of the learned graph component embeddings optimally captures the structural features of the underlying graph. This facilitates the original graph of any scale to be easily handled in the new vector space for further analysis. Instead of considering the embedding generation as a pre-processing step and using the generated features for a target ML task, NRL methods employs an end-to-end framework to obtain useful graph features in the form of embeddings in a data-driven way and further uses the obtained features from the projected vector space for various ML downstream tasks. Figure 1.3 gives a toy example for

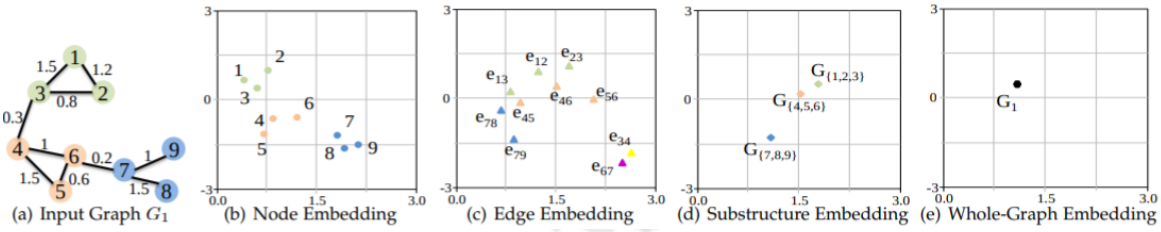


Fig. 1.3 A toy example of embedding a graph G_1 into 2D space with different granularities. $G_{\{1,2,3\}}$ denotes subgraph of G_1 containing nodes v_1, v_2, v_3 (likewise for other subgraphs). Image Source: NRL Survey [1].

an input graph G_1. In SubFigures a, b, c, d – the nodes, edges, subgraphs, and the entire graph are projected to a latent $2D$ embedding space, respectively. One of the most important design goals for proposing any NRL framework is to ensure globally relevant, useful, non-redundant graph features are obtained via any encoder mapping function.

1.4 Local versus global contexts in graphs for learning network representations

Most of the recent research efforts [16–21] in NRL is directed towards learning local neighborhood contexts surrounding any graph component of interest for a target downstream task. Few research works capture global contexts that have dependencies on the underlying network connectivities. Few examples include — i) incorporating neighborhood contexts surrounding a node [22, 18, 23, 20] for learning node embedding to classify the nodes, ii) incorporating common subgraphs surrounding a (source, target) node-pair for learning to embed edges for relation prediction [24–26], iii) considering only a certain length of the window from sampled random-walks for learning contextualized node embedding [16, 17, 12, 27], iv) considering network-only proximity to group nodes of a graph for node clustering task [10, 28–31]. i), ii), and iii) provide a microscopic view of the network, and the learned graph-component embeddings lack in discriminative capacity. Due to several challenges that frequently exhibit in real-world graphs of different types (see Sections 3.2, 4.2, 5.2, for details), these narrow views of graphs may not prove to be useful for a downstream task. Figure 1.4 elucidates

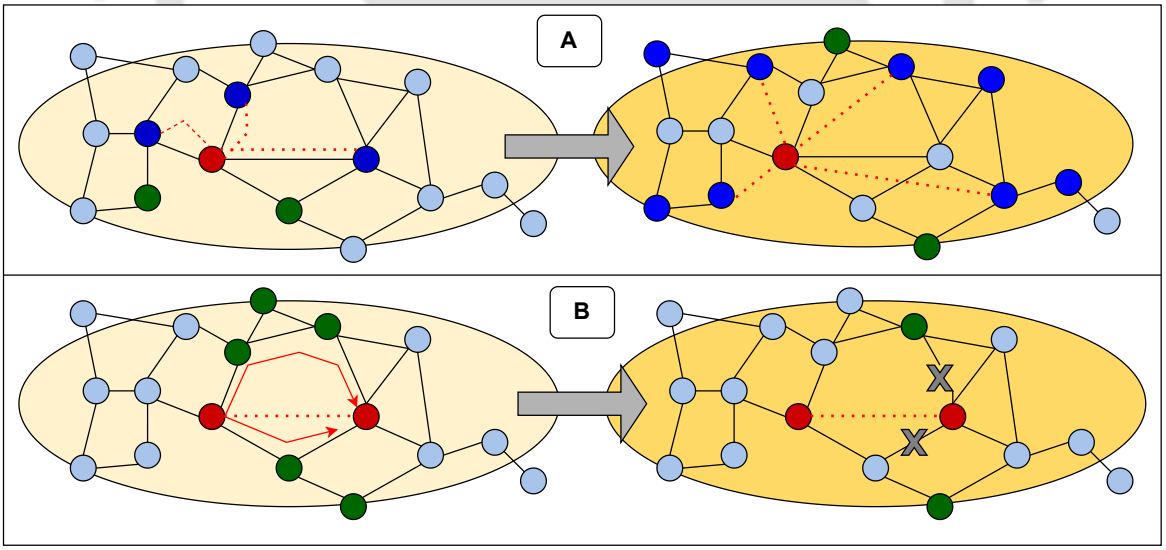


Fig. 1.4 A toy example of showing usefulness of community information for A) node classification, and, B) link prediction. Communities can see beyond the local neighborhoods. They are especially useful to resolve sparsity issues in graphs.

this point for more clarity. In SubFigure A of Figure 1.4, relational features for classifying node colored in Red is considered. Based on the Red node's functional classes (colored in Green and Blue) of one-hop neighbors, it is easy for a prediction model to annotate the

Red node with Blue functional class since Blue class labels surrounding the target node are abundant. However, in a more challenging scenario depicted on the right-hand side, the functional class annotations are missing for the Red node's one-hop neighbors. Due to this, the relational feature-based prediction algorithm finds it hard to classify the Red node. But, given some more ground-truth information about the Red node that it is part of the community underneath (highlighted in Yellow) and the majority of the member nodes of that community are annotated with Blue functional classes — gives some graph structure based cues to the prediction model that directly aids in predicting classes for the Red node. In SubFigure B of Figure 1.4, we consider learning embeddings based on common two-hop neighborhood structures (includes nodes, relations, and paths within) to predict the existence of a link between two Red nodes. Common neighbors up to two-hop are highlighted in Green. On the left-hand side, we observe that given the local contexts surrounding the node pair, it is easy to justify the formation of a link between the target nodes. However, in a more challenging scenario depicted on the right-hand side, we see that the network suffers from link sparsity issues. Due to this, a very less number of common neighbors are available to the link inference algorithm. The lack of local context makes it hard for the inference algorithm to predict links for the target nodes confidently. But, given some more higher-order structural cues that the Red nodes are part of the same community where other members, including the Red nodes' neighbors, are densely connected to the rest of the community — provides more evidence to the inference algorithm in support of the existence of a link between the Red nodes. *Therefore, the macroscopic network view, aka the higher-order structural features of the underlying network, is important and plays a critical role in the learning mechanism.*

In a graph, there are prominent higher-order structures as well, which can enrich the graph-component representations for a downstream task. Figure 1.5 shows various higher-order graph structures at multiple-scales (finer to coarser). To give a few examples, i) incorporating communities exhibited in a graph for node and/or edge embedding, ii) incorporating a set of higher-order relational paths for learning edge representations, iii) considering motifs, subgraphs, network-schemas for node embedding, iv) considering node proximities other than network-connectivity to group nodes — are some of the ways to incorporate the graph structures into graph component embeddings. These structural intuitions are not automatically incorporated i.e., local context based network representation learning models do not explicitly learn from them.

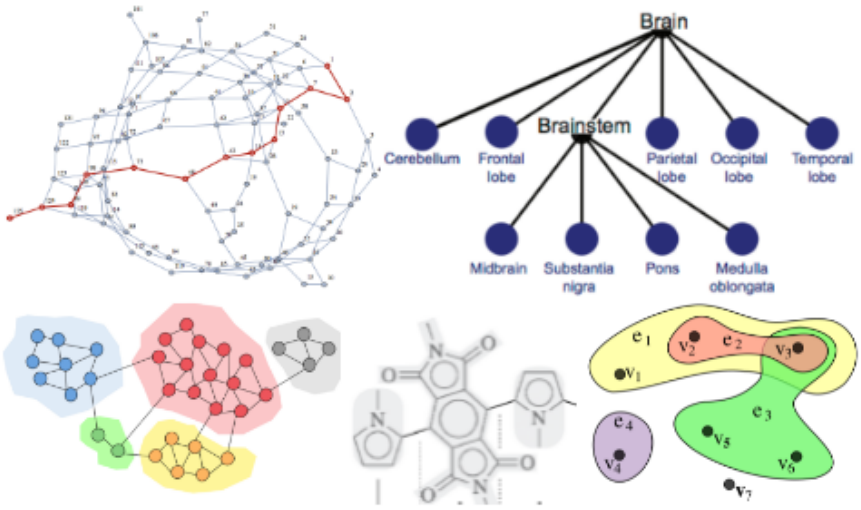


Fig. 1.5 Higher-order structures at multi-scale for graphs. Examples include: walks, hierarchies, hyperedges, motifs, communities (in clock-wise manner). Image Source: partially from the Internet.

1.5 Broad Challenges and Motivations

In this dissertation, we aim to incorporate structural cues from the underlying graph for network representation learning. We consider three types of graphs — simple homogeneous, heterogeneous, and multiplex networks as our subjects of study. As downstream ML tasks, we consider the problem of node classification for homogeneous and multiplex graphs, and the problem of link prediction for heterogeneous graphs. A thorough literature review (refer to Sections 3.6, 4.6, 5.6) of the NRL methods on these subject graphs reveal that there exist critical research gaps and immense scopes of research to address the central research theme of this dissertation — *How to incorporate higher-order multi-scale graph structure based intuitions to enrich target graph component embeddings for chosen downstream ML tasks?* Some of the critical research gaps for our chosen subject graphs are elucidated next, which essentially motivates us to design structure-aware NRL frameworks in the respective graph domains as the primary contributions.

Learning structures in homogeneous graphs primarily focus on, either capturing k-hop local contexts of nodes using i) random-walk-based methods [16, 17], and ii) Graph Neural Networks (GNNs) [18]. Or, encoding explicit clustering criteria in i) matrix factorization methods [20, 29] and ii) auto-encoders [32], iii) GNNs [33–35, 30]. However, random-walks, GNNs, auto-encoders, proximity-based methods are limited to only capturing k-hop local contexts for nodes. All the community enforcing models use unsupervised clustering

criteria based on either network-only node proximities or embedding-based node proximities. Whereas the classic assumptions of Semi-Supervised Learning (SSL) [36], advocate for the *clusterability* of nodes with similar target functions in dense regions. The *Cluster Assumption* of SSL [36] considers supervision information available in the graph data in association with the underlying linked data distribution as an indicator for community membership for the nodes. However, none of the existing SSL methods incorporate supervision knowledge-based global neighborhoods of nodes to enrich their local embeddings. *This calls for considering various sources of supervision knowledge exhibited in graphs to form logical groupings of nodes and see what advantages it offers in contrast to the unsupervised communities that are limited by ground-truth network connectivity patterns.* To address this, we design a Non-Negative Matrix Factorization based joint node and cluster embedding learning framework which explicitly incorporates all necessary priors of SSL, especially the *cluster assumption*.

For multiplex networks, among various learning paradigms such as graph convolutions [37–39], random walks [40, 41] or matrix factorization [21], only a few recent works encode global structures [38, 39] in multiplex graphs via explicit clustering using Graph Neural Networks (GNNs) [38] and InfoMax[6] based learning respectively. InfoMax based learning [2, 43, 39, 44] uses GNNs and provides a scalable way to incorporate both local and global node representations via maximizing Mutual Information (MI) between them. Nevertheless, in a typical setup, it considers a unique shared global graph summary obtained from a trivial aggregation of all the learned node representations and uses the same summary to optimize MI with all the local node representations. *This trivial graph summary indeed encodes a lot of noisy information in the learned embeddings. This calls for a unique contextualized global graph representation for each node.* To address this, we propose *Cluster-Aware InfoMax* learning objective and design a novel GNN framework based on the objective to jointly model nodes and clusters for learning enriched node representations.

State-of-the-art (SoTA) NRL link prediction methods [26, 19, 45] for heterogeneous graphs primarily consider graph neural networks for learning local neighborhood contexts such as enclosing subgraph context surrounding the (source, target) node-pair for learning edge representations. Very few attempts have been made so far to use different perspectives such as relational paths [46–48], network-schemas [49, 50] other than the surrounding context for predicting links. A number of link prediction models [24, 25] look to designing structure-aware GNNs by introducing a node-labeling scheme for distinguishing structural roles of the nodes in the common subgraph surrounding a node-pair under consideration. However, they

[6]InfoMax is an optimization principle that prescribes that a function that maps a set of input values I to a set of output values O should be chosen or learned so as to maximize the average Shannon mutual information between I and O, subject to a set of specified constraints and/or noise processes. [42]

do not propose any explicit learning objectives to incorporate structural cues other than via local subgraph contexts. Also, *there exists no study to investigate the roles that communities of the end nodes play in predicting links between the nodes. Even, there is no systematic study to understand the effectiveness and complementarity of each kind of structural cue present at multiple scales (from finer to coarser) in the heterogeneous graphs and how to effectively aggregate them for predicting links.* To address this, we propose a multi-view learning framework that advocates for incorporating two additional non-trivial views based on metapath and community contexts between a (source, target) node-pair, alongside the popularly used neighborhood context, for predicting the plausibility of a link between the nodes. Moreover, we formulate a fine-grained aggregation mechanism to combine these candidate views effectively for better link prediction.

1.6 Research Objectives

Recent research efforts in the direction of structure-aware NRL conclude that learning higher-order graph structures are helpful and improve the discriminative capacity of the learned graph-component embeddings in various experiment scenarios. However, based on the complexities and scale of the network under consideration, capturing the macroscopic view of the network can be a challenging task. *Therefore, it is necessary to study the structural intricacies of a subject graph and explore/ innovate various modeling choices to seamlessly incorporate the global graph properties into graph-component representations learned from local contexts.* This dissertation i) considers subject graphs of varying complexities — homogeneous graphs, heterogeneous graphs, and multiplex graphs, ii) carefully examines existing research gaps to incorporate global graph features in these subject networks, and iii) explores innovative ways to incorporate non-trivial higher-order structures into respective NRL frameworks facilitating learning of useful network features. Below are the important contributions of this thesis in the chosen subject graphs, followed by how this dissertation aims to address them.

1.7 Thesis Overview and Contributions

In this dissertation, we propose three ways to incorporate higher-order graph structures into various NRL frameworks. We choose three kinds of graphs, namely, homogeneous, heterogeneous, and multiplex graphs to establish that incorporating diverse higher-order

graph structures other than mere local neighborhood contexts into network (node, edge) representations facilitates learning of more discriminative network component embeddings. To this end, we propose three ways — i) Semi-Supervised Cluster Invariance Property for homogeneous graphs, ii) Cluster-Aware Graph InfoMax for multiplex graphs, and iii) Metapath and Community Views for relations in heterogeneous graphs to learn structure-aware network representations. We empirically establish the usefulness of learning such representations via i) various end-tasks — node classification, node clustering, link predictions and, ii) case-studies — node similarity search, analyzing cluster similarity, metapath clustering, and interpretable t-SNE visualizations. Figure 6.1 summarises the contributions made towards designing structure-aware NRL frameworks.

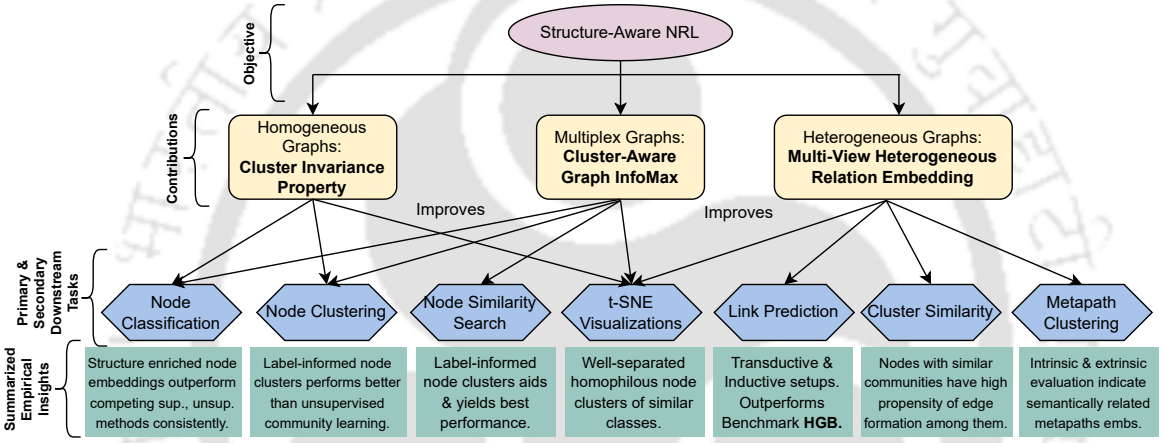


Fig. 1.6 Summary of contributions in the dissertation

This dissertation makes the following three primary contributions towards designing structure-aware NRL frameworks:

1. A *Semi-Supervised Cluster Invariance Property* for the nodes has been proposed with a view to building a unified transductive node representation learning framework called *Unified Semi-Supervised Non-Negative Matrix Factorization (USS-NMF)*. USS-NMF allows for explicitly encoding different priors necessary for facilitating Semi-Supervised Learning (SSL) and learning efficient node representations in a homogeneous graph. USS-NMF specializes in encoding the important yet largely ignored necessary prior for SSL, the *Cluster Assumption*.

2. Motivated by the need for contextualized global graph representations for nodes in a typical InfoMax based representation learning setup, a new learning scheme called *Cluster-Aware InfoMax* is proposed for simultaneously learning node and cluster embeddings in multiplex graphs. Learned clusters encode the node-specific global graph context with which the Mutual Information (MI) of a node's local representation

is maximized in a novel joint node-cluster representation learning framework called *Semi-Supervised Deep Clustered Multiplex (SSDCM)*.

3. A multi-view learning framework called *Multi-View Heterogeneous Relation Embedding (MV-HRE)* is proposed that advocates for incorporating two non-trivial views such as relational path view and community view beside the typical enclosing subgraph view between the (source, target) node-pairs for predicting plausibility of a link between them. Empirical reports and case studies suggest that the chosen candidate views are complementary and useful with regard to the target relation type to be predicted, thus boosting the link prediction performance by a large margin.

1.8 Thesis Organization

The dissertation is organized into the following chapters:

1. Introduction The introductory chapter discusses the main theme of this dissertation. It introduces the network representation learning paradigm and its importance in modeling real-world networked data. It also formally introduces the high-level research objective detailing the contributions made.

2. Background This chapter provides all the basic definitions, concepts, and notations related to the chosen subject graphs. It also explains in detail the network representation learning background necessary to propose the contributions in the following chapters.

3. Structure-Aware Network Representation Learning on Homogeneous Graphs A unified semi-supervised framework is proposed to encompass all the learning principles of SSL, especially the cluster assumption in homogeneous graphs.

4. Structure-Aware Network Representation Learning on Multiplex Graphs A novel cluster-aware graph InfoMax learning strategy is proposed to contextualize global graph summaries for the nodes in multiplex graphs.

5. Structure-Aware Network Representation Learning on Heterogeneous Graphs A novel community view is proposed to represent an edge in heterogeneous graphs. Metapath view and community views are incorporated along with the existing subgraph view to aid link prediction.

For each of the three chapters $(3,4,5)$ above, the specific subject graph and the challenges that exist for them in learning network representations are introduced. Important intuitions

and empirical insights that provide motivations to propose the solution are discussed. The chapter-specific research objectives and a detailed list of contributions are discussed. Next, various related research directions are reviewed to find critical research gaps for addressal.

Consequently, elaborate formulations of each of the proposed frameworks are given. Evaluation strategies are discussed to ensure reproducibility. At last, the empirical evidences are presented to establish the novelty of the proposed methods. Finally, the performance of each proposed framework is critically analyzed to gain more insights.

6. Conclusions and Future Work The final chapter draws important conclusions from the thesis work and discusses the future research directions to explore.

Chapter 2

Background: Graphs and Network Representation Learning

In this chapter, we discuss all the background necessary to understand the contributions made in this thesis. This thesis proposes three frameworks to learn graph structure-aware network representations for three kinds of target graphs – homogeneous, multiplex, and heterogeneous graphs. The preliminaries in the following sections elaborate more on the backgrounds specific to each graph and NRL concepts related to the contributions made.

2.1 Preliminaries: Homogeneous Graphs

The basic notations and definitions related to homogeneous graphs are explained in this section.

Homogeneous networks are graphs with no non-trivial network-connectivity representation, i.e., the node-to-node connection pattern is specified by only an adjacency matrix.

Let, $G = (V, A, Y)$ be a networked data with a set of N number of vertices, V; an (un)weighted and (un)directed adjacency matrix, $A \in \mathbb{R}^{N \times N}$; a set of class labels, Z and an one-hot vector representation of classes, $Y \in \{0,1\}^{q \times N}$, where $q = |Z|$. There are L number of labeled data, such that, $L = |\{(i, z_i) : i \in V, z_i \in Z\}|$ and, UL number of unlabeled data, such that, $L + UL = N$. In this work, we assume no features are associated with the nodes and edges in the graph. Figure 2.3 illustrates a few examples of real-world data such as

collaboration networks, social networks, brain regions, protein-protein interactions, financial dependencies among organizations, etc., that can be modeled as homogeneous graphs.

2.1.1 Graph Laplacian Matrix

We now discuss various forms of a graph *Laplacian* matrix that are used in this thesis.

Definition 2.1.1 (DEGREE MATRIX)**.** Given a graph G with vertex set $V : |V| = N$, the *Degree Matrix*, $D(A) \in \mathbb{R}^{N \times N}$ is a diagonal matrix defined as,

$$D(A)_{ij} = \begin{cases} deg(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

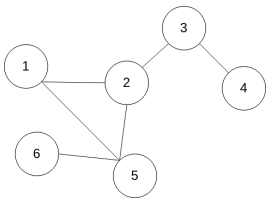
In an undirected graph, the non-zero diagonal elements refer to degree of the vertices. Whereas, for directed graphs, either the in-degree or out-degree of the vertices is considered.

Definition 2.1.2 (GRAPH LAPLACIAN MATRIX)**.** Given a simple graph G (no self-loops and parallel edges), with adjacency matrix A, its unnormalized *Laplacian Matrix*, $\Delta(A)_{N \times N}$ is defined in the matrix form as,

$$\Delta(A) = D(A) - A \tag{2.2}$$

In element-wise manner,

$$\Delta(A)_{ij} = \begin{cases} deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j,\ v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$



$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ -1 & -1 & 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

A : Adjacency Matrix of G D : Degree Matrix of G Δ : Laplacian Matrix of G

Fig. 2.1 An example graph G

Definition 2.1.3 (SYMMETRICALLY NORMALIZED LAPLACIAN)**.** The *symmetrically normalized Laplacian* operator on G is defined as,

$$\Delta^{sym} = D^{-\frac{1}{2}} \Delta D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \tag{2.3}$$

$$\begin{pmatrix} 1 & -0.408 & 0 & 0 & -0.408 & 0 \\ -0.408 & 1 & -0.408 & 0 & -0.333 & 0 \\ 0 & -0.408 & 1 & -0.707 & 0 & 0 \\ 0 & 0 & -0.707 & 1 & 0 & 0 \\ -0.408 & -0.333 & 0 & 0 & 1 & -0.577 \\ 0 & 0 & 0 & 0 & -0.577 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & -0.5 & 0 & 0 & -0.5 & 0 \\ -0.333 & 1 & -0.333 & 0 & -0.333 & 0 \\ 0 & -0.5 & 1 & -0.5 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ -0.333 & -0.333 & 0 & 0 & 1 & -0.333 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

Δ^{sym} : Symmetrically normalized Laplacian Δ^{rw} :Random walk normalized Laplacian

Fig. 2.2 Various Graph Laplacians

Where, I is an identity matrix of size $N \times N$.

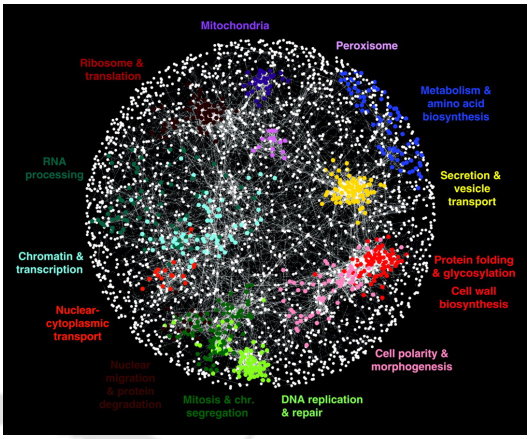
Definition 2.1.4 (RANDOM WALK NORMALIZED LAPLACIAN)**.** The *random walk normalized Laplacian* operator on G is defined as,

$$\Delta^{rw} = D^{-1}\Delta = I - D^{-1}A \tag{2.4}$$

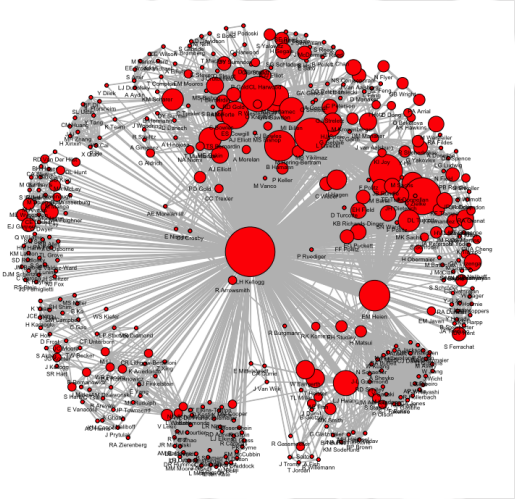
Elements in Δ^{rw} each row sums up to 0, since the *degree-normalized adjacency* $D^{-1}A$ matrix is *row-stochastic* in nature. That is, each row in $D^{-1}A$ denotes *transition probabilities* representing probability distribution (sums up to 1) of the transition states of a random-walker exploring the graph uniformly randomly, taking one step at a time, from one node to another.

The *Laplacian* can also be seen as an operator $f : V \mapsto \mathbb{R}$ which maps each vertex in G to real-values [51]. Figure 2.1 shows an example graph G, for which various unnormalized and normalized Laplacian matrices are shown. The Laplacian matrix Δ is a symmetric, positive-semidefinite matrix. It is related to many unique properties of the underlying graph. One can approximate the *sparsest cut* in a graph for partitioning the nodes. The number of connected components in a graph is obtained from the dimension of the *nullspace* of the Laplacian matrix. One notable property of the Laplacian matrix is that it has a block-diagonal matrix structure, when the underlying graph has multiple connected components. Each block represents the Laplacian matrix of the respective connected component. Since the Laplacian matrix is diagonally-dominant, in practice, we normalize it in various ways.

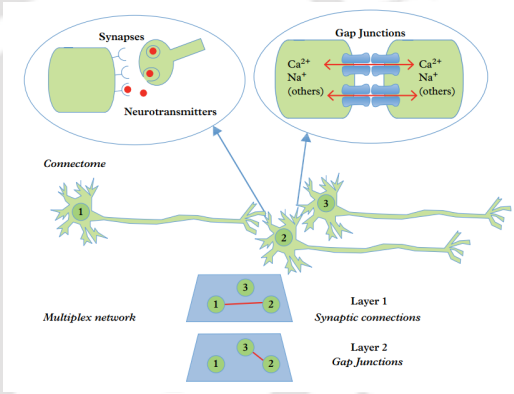
(a) Social networks



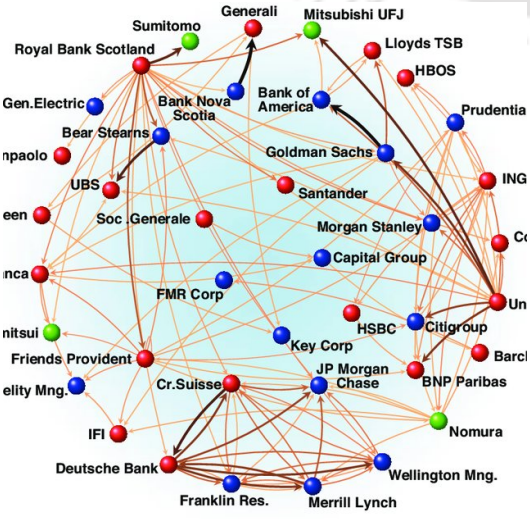
(b) Protein-protein interaction networks



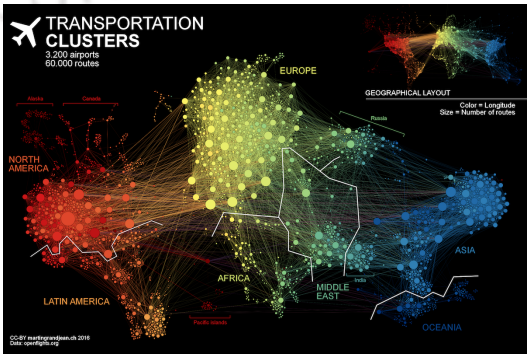
(c) Collaboration networks



(d) Brain networks



(e) Financial networks



(f) Traffic networks

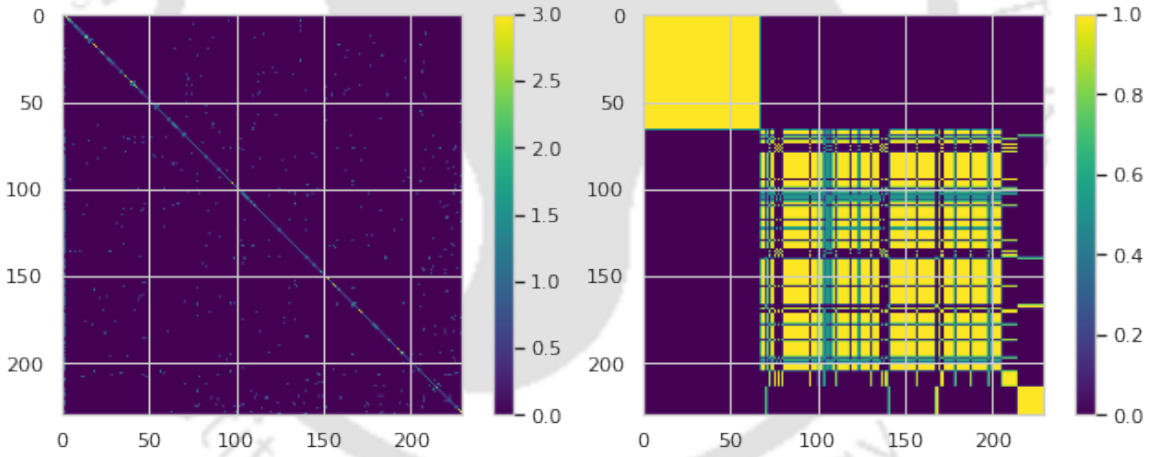
Fig. 2.3 Homogeneous Networks: Examples [Image Source: Internet]

Definition 2.1.5 (LABEL SIMILARITY NETWORK)**.** A *label similarity network* $E \in \mathbb{R}^{N \times N}$ over G is defined as the node-to-node similarity-association based on common labels, given the one-hot class labels for the nodes $Y \in \mathbb{R}^{q \times N} : q = |Z|, N = |V|$. This label similarity network is defined based on labeled train instances for a semi-supervised network representation learning setup. Therefore, a weight penalty matrix $W \in \mathbb{R}^{q \times N}$ is introduced which eliminates the labels of test instances upon element-wise Hadamard multiplication operation with Y.

$$E_G = Y^T Y \tag{2.5}$$

$$E_{G_{\text{TRAIN}}} = (W \odot Y)^T (W \odot Y) \tag{2.6}$$

This matrix is of importance in case of link and label sparsity exhibited in the underlying graph (see Section 3.2 for details). For graph SSL-based learning setup, the matrix defines a higher-order neighborhood between two nodes, even though they can be far apart in the network or may not even be connected. We adopt this notion of label-informed higher-order neighborhood very often in this thesis work to learn logical groupings of nodes. In Figure 2.4



(a) Graph Adjacency (b) Label-similarity Adjacency

Fig. 2.4 Visualizing node associations on Washington

we visualize the node association heatmaps based on simple network connectivity and the connections induced by our *label similarity network*. We clearly see the block-diagonal structures in Figure 2.4b. This figure also shows how the link-sparsity issues can be alleviated in a graph based on limited supervision knowledge available.

2.1.2 Non-Negative Matrix Factorization (NMF)

Non-Negative Matrix Factorization [52–54] is a learning paradigm to discover low-dimensional representations of a data. It decomposes a data matrix into latent basis matrices in such a way that the data can be reconstructed from the additive linear combination of vectors from the product matrices – which essentially gives a *parts-based representation* of the *whole* data. The graph adjacency A or any node-to-node similarity matrix derived from A is factored into two non-negative matrices $U, M \geq 0$, such as, basis matrix $U \in \mathbb{R}^{N \times m}$ and coefficient matrix $M \in \mathbb{R}^{N \times m}$. The objective is to seek a decent low-rank approximation of the data matrix $A \approx U^T M$ in terms of the low-dimensional product matrices (U, M). The resultant matrices give low-dimensional (m) representation of the high-dimensional (N) networked data where $m \ll N$. The decomposition preserves the local-neighborhood contexts of each node in latent space. The learning objective seeks to optimize the approximation error as follows,

$$\mathcal{O}_{network} = ||A - UM^T||_F^2 = \sum_{i,j} \left(A_{ij} - \sum_{k=1}^{m} U_{ik} M_{jk} \right)^2 \; : \; U, M \geq 0 \tag{2.7}$$

The *Matrix Factorization* methods optimize the same objective as above except the fact that it does not constrain the product matrices to be non-negative.

2.1.3 Pointwise Mutual Information Matrix from Graph Adjacency

Random walk-based methods [16, 17] and matrix factorization-based methods [20] are two important paradigms for learning node representations. In this part, we design a unified NMF framework for learning network representations from homogeneous graphs. Here, we discuss various matrices that are derived from the graph adjacency matrix A for factorization. We discuss the interpretability of those matrices. We also discuss the relatedness of the above-mentioned learning paradigms.

Deepwalk [16] is a *Skip-Gram* [55] based optimization framework that maximizes the average log probability of all vertex-context pairs in a window of t extracted from the set of generated random walks of vertex sequences $\mathcal{W} = \{v_1, v_2, ..., v_{|\mathcal{W}|}\}$.

$$\frac{1}{|\mathcal{W}|} \sum_{i=1}^{\mathcal{W}} \sum_{-t \leq j \leq t, j \neq 0} log \; p(v_{i+j} | v_i) \tag{2.8}$$

Where, $p(v_{i+j}|v_i)$ is given by *softmax* function as,

$$p(v_{i+j}|v_i) = \frac{exp(h_{v_{i+j}}^T h_{v_i})}{\sum_{v \in V} exp(h_v^T h_{v_i})} \tag{2.9}$$

$h_{v_{i+j}}, h_{v_i}$ are the embeddings of respective nodes. The above equation estimates the probability based on inner-product based similarity of two vectors. Research [56] has shown that Skip-Gram with *negative sampling* and *hierarchical softmax* implicitly factorize matrices $\mathcal{M}$, respectively of the forms,

$$\mathcal{M}_{ij} = log \frac{\mathcal{N}(v_i, v_j).|\mathcal{D}|}{\mathcal{N}(v_i).\mathcal{N}(v_j)} - log\, b \tag{2.10}$$

$$\mathcal{M}_{ij} = log \frac{\mathcal{N}(v_i, v_j)}{\mathcal{N}(v_i)} \tag{2.11}$$

Here a vertex-context pair (v_i, v_j) is associated with b number of negative samples. $\mathcal{N}(v_i, v_j)$ denotes the number of times that vertex-context pair appears in the set of vertex-contexts $\mathcal{D}$ extracted from the random-walk sequences $\mathcal{W}$. Likewise for $\mathcal{N}(v_i) = \sum_{v \in V} \mathcal{N}(v_i, v)$ and $\mathcal{N}(v_j) = \sum_{v \in V} \mathcal{N}(v, v_j)$. $\mathcal{M}_{ij}$ in Eqn 2.10 is identical as the *Pointwise Mutual Information (PMI)* between the vertex-context pair (v_i, v_j) shifted by $log\, b$. Similarly, $\mathcal{M}$ in Eqn 2.11 can be interpreted in terms of *transition probability matrix* $P = D^{-1}A$ of a graph [57] as below,

$$\frac{\mathcal{N}(v_i, v_j)}{\mathcal{N}(v_i)} = \frac{[e_i(P + P^2 + P^3 + ... + P^t)]_j}{t} \tag{2.12}$$

Where e_i is a one-hot vector for node i which denotes the initial state of a random-walker starting walk from node i, thus, $(e_i P)_j$ denotes the probability distribution that the random-walker reaches node j in the next step. $(e_i P^t)_j$ denotes the probability that it reaches node j from node i in the t^{th} step. Therefore, Eqn 2.12 is interpreted as the expectation that v_j appears within the t^{th} neighborhood of v_i. Now, as we can see, Skip-Gram based network embedding methods implicitly factorize matrix of above-shown forms, many matrix factorization works [57, 58, 20] have adopted various matrices of the base-forms shown in Eqns 2.10, 2.11, to reap the benefits of both random-walk and matrix factorization based paradigms.

In this work, we factorize a proximity matrix S up to second-order neighborhood [57, 58] with entries,

$$S_{ij} = \frac{[e_i(P + P^2)]_j}{2} \tag{2.13}$$

We assume that 2− hop neighborhood is sufficient to model the local contexts around a node. We factorize S instead of $log\ S$ to further reduce the complexity of matrix factorization. This is since the complexity of matrix factorization with square loss is proportional to the number of non-zero entries in a matrix [59]. Since the above proximity denotes vertex-similarity based on the average 2−hop transition probabilities of a random-walker, the entries are non-negative. Thus we adopt the NMF paradigm instead of Matrix Factorization (MF), to learn embeddings from S.

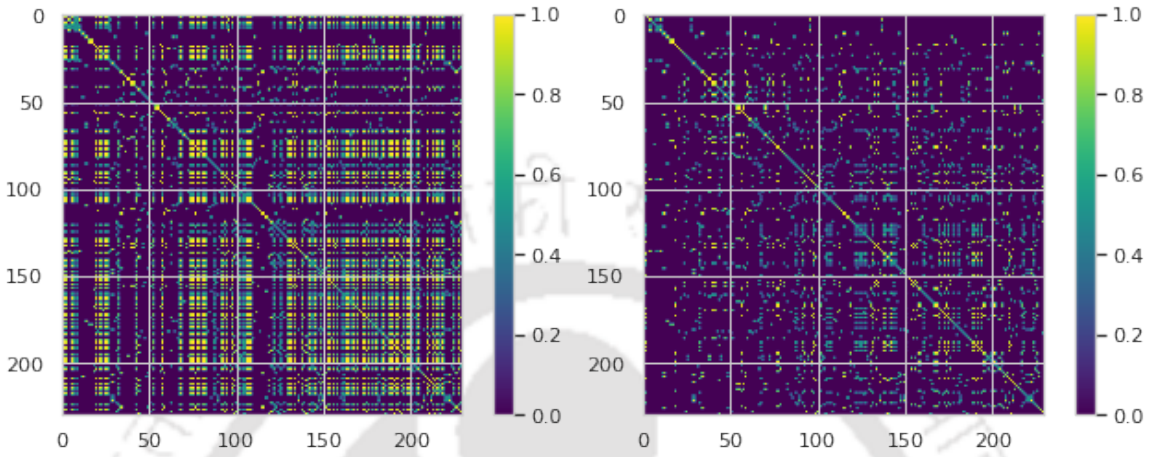
2.1.4 Clusters and Communities

In this thesis, we often refer to incorporating higher-order graph structures to enrich learned node embeddings. And, *Clustering* of nodes in a graph is a key way to define the higher-order neighborhood for a node. Clustering is a generic term that means a logical grouping of nodes in a graph based on certain criteria. This criterion does not make any assumptions on the underlying connectedness of the nodes. For example, groupings of nodes based on — label similarity, centrality based similarity, cosine similarity, etc., are examples of node clustering where network connections may be (second example) or may not be preserved (first and third examples). However, *Communities* in a graph do make assumptions on the underlying connections and link density[1]. Thus, community discovery in a graph is a special case of node clustering. Here, we also distinguish between spontaneous *community discovery* in a graph where the number of communities are unknown at the beginning, and, *partitioning* of a graph into a predefined number of communities. This thesis aims to partition a graph into a predefined number of communities to define a node's high-order neighborhood based on learned node-embedding-based features, trading off on how to preserve various underlying/induced node proximities and network characteristics simultaneously.

A number of node clustering algorithms exist based on — link densities, divergence (modularity-maximization [61], Louvain method [62]); *Expectation-Maximization* algorithm (K-Means [63]); hierarchical grouping (Girvan–Newman algorithm [64]); preserving graph *spectrum*[2] characteristics (Spectral clustering [66]); optimizing entropy (InfoMap [67, 68]). Figure 2.5 plots heatmaps of community membership based node similarities on Washington graph when the original graph adjacency (Figures 2.5a, 2.5b) and label similarity network's adjacency (Figures 2.5c, 2.5d) are given input to Spectral and Modularity-maximization algorithms. The original node connections are shown in Figure 2.4. We observe that — a lot of

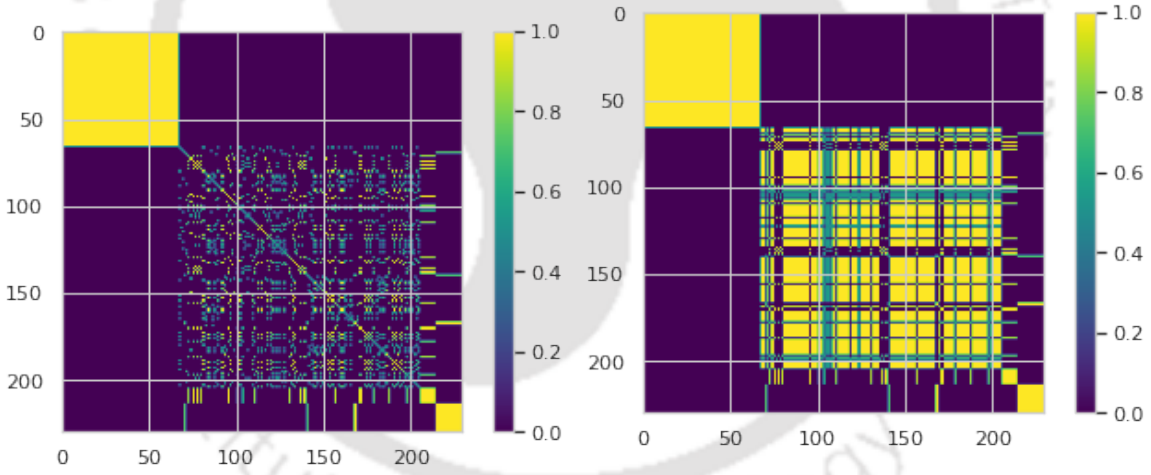
[1] A community is a locally dense connected subgraph in a network. [60]
[2] The set of graph eigenvalues of the adjacency matrix is called the spectrum of the graph. [65]

new connections are introduced based on cluster similarities. Therefore, community detection is an effective strategy to deal with link-sparsity in a graph. We also see that, modularity maximization could completely recover the original connections for label similarity network.



(a) Spectral clustering: adjacency

(b) Modularity maximization: adjacency



(c) Spectral clustering: label-similarity

(d) Modularity maximization: label-similarity

Fig. 2.5 Visualizing community similarity heatmaps on Washington

2.1.5 Semi-Supervised Learning on graphs

The classical graph based semi-supervised learning [69] is a special case of SSL where the underlying data naturally exhibit a graph structure. We consider the problem of classifying

nodes of a partially labeled graph. The loss function can be written as,

$$\sum_{i=1}^{L} \text{Loss}(Y_i, f(X_i)) + \lambda \sum_{i,j \in |V|} A_{i,j} ||f(X_i) - f(X_j)||^2 \tag{2.14}$$

Where X is the node features, L is the set of labeled data points, V is the entire vertex-set related to input graph G, A is the adjacency matrix. $f(.) : X \mapsto Y$ is a classifier function which maps each node to its functional classes, Loss(.) refers to any kind of loss such as log loss, squared loss, hinge loss, cross-entropy loss, etc., to optimize for encoding the supervision knowledge. Note that the first component in Eqn 2.14 denotes the supervised loss with respect to the partial label information available. Whereas the second component refers to the smoothening of label information over the entire graph preserving the underlying connectedness via some form of graph-based regularization. λ weighs the importance of the second component in Eqn 2.14.

Graph-based SSL methods model the underlying data distribution as a graph G, comprising of both labeled and unlabeled data along with any notion of node-to-node associations A. Each pair-wise entry in A corresponds to the thresholded similarity score between a pair of nodes as defined by a similarity kernel on the input space. In the graph-space, we generally consider the adjacency matrix or any network similarity derived from it. The Laplacian, $\Delta(A)$ defined on A, is a key operator for graph-based SSL. It is used to either smooth the target function or to obtain new features. In the first case, the regularizer is added to the classification loss and is jointly minimized as in Eqn 2.14. It solely uses node input features for the class prediction via function $f(.)$. The latter is a two-step process where unsupervised node embeddings are obtained first using an embedding learning function $g(.) : X \mapsto U$ that projects the underlying graph's large feature space X to a latent manifold $U : U \ll X$. And then, obtained features are used to predict labels using $f(.)$ for classification. Both approaches can be seen as enforcing local invariance (A) on a target space $f(X)$. Thus, two design choices are involved, based on — the notion of similarity (A) to use, and, the target space $f(X)$ to be smoothed.

Below, we provide a functional definition of Laplacian regularizer [36, 10, 70, 22] parameterized by the graph of consideration and the space to be smoothed. $LS(A, f(X))$ denotes Laplacian Smoothing (LS) on the target space, $f(X)$ with the Laplacian of A. As mentioned earlier, Laplacian smoothing can be applied on predicted label space $f(X) = \hat{Y}$, on original data X, on data projections $f(X) = U$ or *on any abstract space such as clusters*,

$f(X) = \hat{H}$ as proposed in this work.

$$LS(A, f(X)) = \sum_{i,j \in |V|} A_{i,j} \|f(X_i) - f(X_j)\|^2 = f^T \Delta(A) f \tag{2.15}$$

An end-to-end graph-based SSL is achieved by optimizing a supervised classification loss defined over labeled data coupled with a weighted Laplacian regularizer over all available data (labeled, unlabeled).

2.1.6 Semi-Supervised Node Classification Task

Here we formally define the end task at hand and our learning objective.

Definition 2.1.6 (SEMI-SUPERVISED NODE CLASSIFICATION TASK ON HOMOGENEOUS GRAPHS)**.** Given a homogeneous graph, $G = (V, A)$, a node feature matrix $X \in \mathbb{R}^{|F| \times |V|}$ with a set of features F, a label set, Z and set of labeled nodes, L with ground truth label assignment matrix, $Y \in \{0,1\}^{|Z| \times |V|}$, the task is to predict labels for all unlabeled nodes, $UL = V \setminus L$. For efficient Semi-Supervised Learning (SSL) on homogeneous networks, it is essential to learn a low m-dimensional ($m \ll |F|$) node embedding, $U \in \mathbb{R}^{|V| \times m}$ that encodes relevant structural and label correlation information within the graph, useful for downstream machine learning tasks such as node classification, and clustering. The node embeddings are desired to be *global graph structure-aware*.

Table 2.1 details notations used in this work. Few of the notations are defined in the proposed architecture in Section 3.7 of Chapter 3.

Symbols	Meaning
G	Input graph
V	Nodes
A	Adjacency matrix, $A \in \mathbb{R}^{N \times N}$
Y	Label matrix (binary membership), $Y \in \mathbb{R}^{q \times N}$
	with Z - the set of q possible classes
N	Number of nodes, $\|V\|$
	with L labeled nodes and UL unlabeled nodes
F	A set of features associated with the vertices
$D(A)$	Degree matrix of graph,
	$D_{i,i} = \sum_{j=1}^{N} A_{i,j}$ and $D_{i,j} = 0$ where $i \neq j$
S	Pointwise mutual information matrix, $S \in \mathbb{R}^{N \times N}$
E	Label similarity network over G, $E_{N \times N} = (W \odot Y)^T (W \odot Y)$
	$W \in \mathbb{R}^{q \times N}$ is a weight matrix which zeroes out labels of test data
I	Identity Matrix, $I \in \mathbb{R}^{N \times N}$
m	Size of low-dimensional representation space
k	Number of clusters, is varied for a range of numbers including q
H	Cluster membership matrix, $H \in \mathbb{R}^{k \times N}$
U, M	Node and context embedding matrix of size $\mathbb{R}^{m \times N}$
Q, C	Label and cluster embedding matrix, $Q \in \mathbb{R}^{q \times m}$, $C \in \mathbb{R}^{k \times m}$
P	Transition probability matrix, $P = D^{-1}A$
$\Delta(A)$	Unnormalized graph Laplacian operator, $\Delta(A)_{N \times N} = D(A) - A$
$LS(S, \hat{Y})$	Classical graph based Laplacian regularizer which smoothes
	the predicted label space $\hat{Y}$ over local neighborhood structure S

Table 2.1 Notations used in Chapter 3 and their meanings

2.2 Preliminaries: Multiplex Graphs

Here we explain all the basic notations and definitions related to multiplex graphs.

2.2.1 Multiplex Graphs

A set of entities exhibiting distinct relations among themselves can be modeled as a layered graph, where each layer depicts unique structural dynamics. Figure 2.7 illustrates a few real-world multiplex graphs. Various real-world relational data, such as social networks, collaboration networks, means of transportation, smart city infrastructures, financial networks, brain networks, biological networks, etc., can be modeled in a multiplex way.

Definition 2.2.1 (MULTIPLEX GRAPH)**.** A multiplex graph is defined by the 3−tuple, $\mathcal{G}_M = (\mathcal{V}, \mathcal{R}, A)$. Where, $\mathcal{R}$ is a set of relations that exhibit between vertices, such that $|\mathcal{R}| > 1$. $\mathcal{V}$ is the vertex set that is common to all the $\mathcal{R}$−relational layers. Adjacency A comprises of a set of intra-layer adjacency matrices $\{A_{(r,r)} \in {\mathbb{R}_+}^{|\mathcal{V}|\times|\mathcal{V}|} : r \in \mathcal{R}\}$ – representing within layer edges in r−th layer, and, a set of bipartite inter-layer adjacencies $\{A_{(r,s)} \in {\mathbb{R}_+}^{|\mathcal{V}|\times|\mathcal{V}|} : r,s \in \mathcal{R}, r \neq s\}$ – representing cross layer edges between layer r and s when $r \neq s$.

A multiplex graph is also represented as an $|\mathcal{R}|$-layered graph, $\mathcal{G}_M = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_{|\mathcal{R}|}\}$, where $\mathcal{G}_r = (\mathcal{V}, A_r)$ with adjacency matrix $A_r = \{A_{(r,r)} \cup A_{(r,s)},\ \forall s : r,s \in \mathcal{R}\}$ being the set of all within-layer, cross-layer connections corresponding to the r^{th} relation. Note that $A_{(r,r)}, A_{(r,s)} \in {\mathbb{R}_+}^{|\mathcal{V}|\times|\mathcal{V}|}$ as all the layers share the same set of nodes, $\mathcal{V}$. Often, the nodes are associated with a feature set, $\mathcal{F}$ and the node feature matrix is denoted as $X \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{F}|}$. The node-to-node interactions can be (un)weighted and (un)directed. Further, combining the node interactions from all the relational layers into a single *aggregated adjacency* matrix would result in loss of critical information, such as, relation-wise interaction among the nodes, cross-layer node associations and their interpretations, identity/ role of each node in different layers, etc.

Definition 2.2.2 (SUPRA GRAPH AND SUPRA ADJACENCY MATRIX)**.** A supra-graph $\bar{\mathcal{G}_{\mathcal{M}}}$ is a high-level representation of a multiplex graph $\mathcal{G}_M$. It is defined as $\bar{\mathcal{G}_{\mathcal{M}}} = (\mathcal{V}_{\mathcal{M}}, \mathcal{A}_{\mathcal{M}})$. The vertex set comprises a set of supra-nodes $\mathcal{V}_{\mathcal{M}} = \{\mathcal{G}_l \cup \mathcal{G}_c\}$ denoting various within-layer graphs $\mathcal{G}_l$ and across-layer graphs $\mathcal{G}_c$. Supra-adjacency matrix $\mathcal{A}_{\mathcal{M}}$ comprises component adjacency matrices that define the connections between the supra-nodes in a flattened matrix representation. The intra-layer associations are on the main diagonal, whereas the inter-layer associations are indicated by off-diagonal entries.

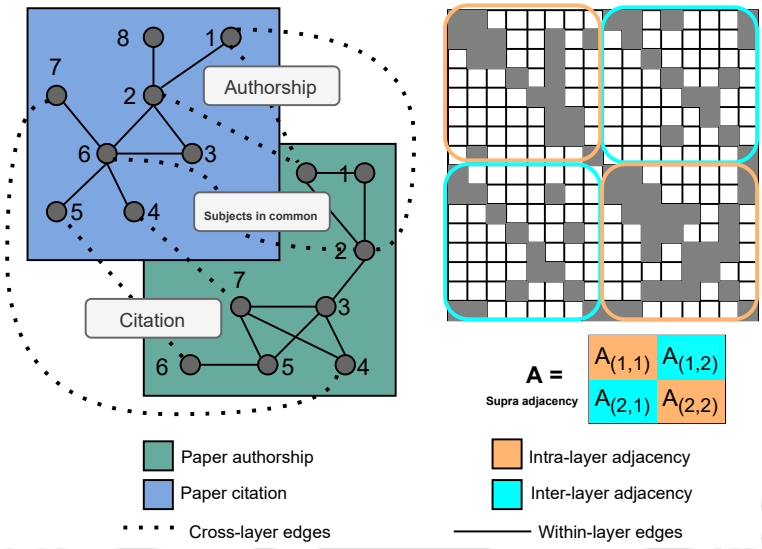


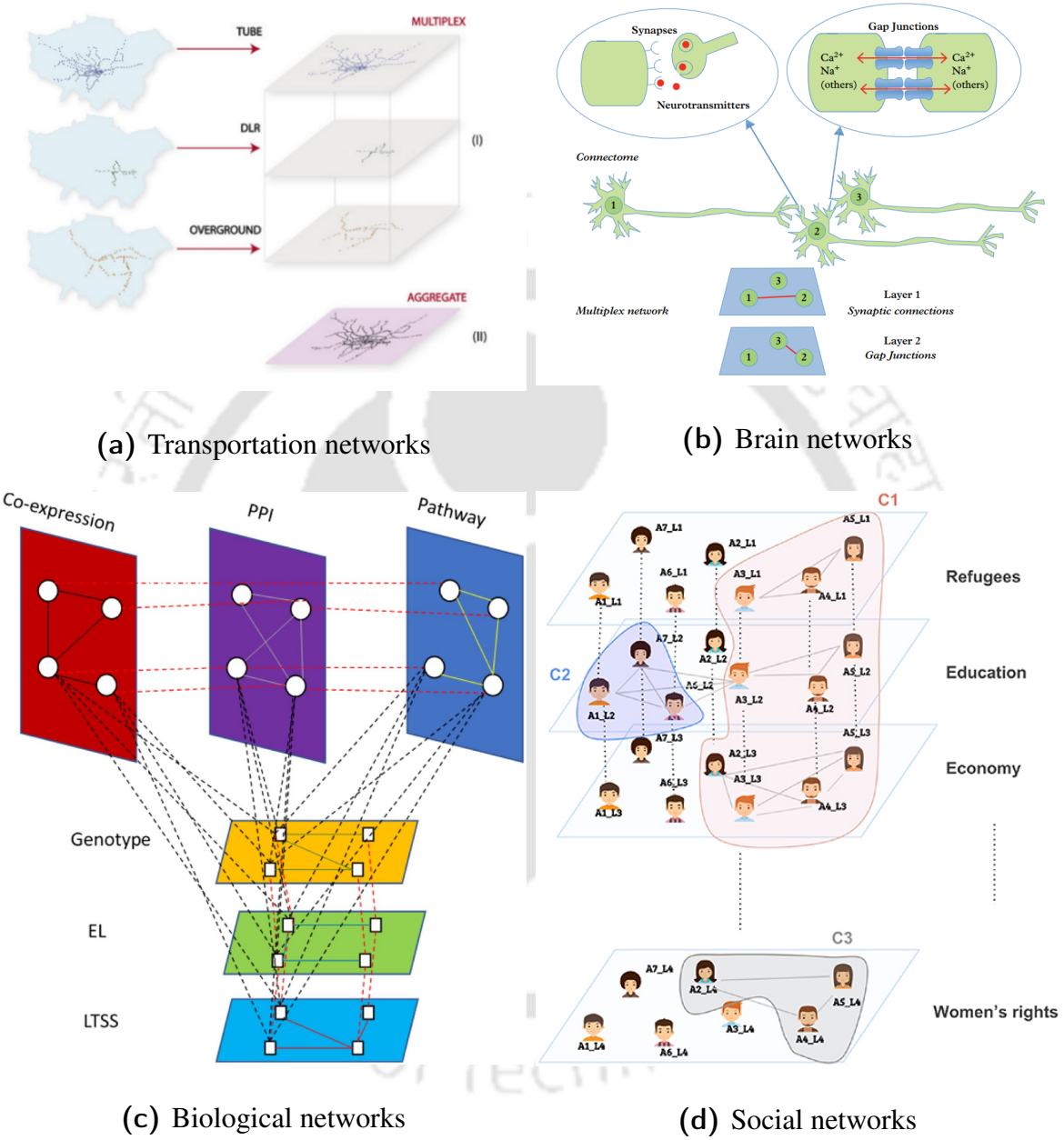
Fig. 2.6 An example bibliographic multiplex graph and the structure of its adjacency matrix

Figure 2.6 shows an example of multiplex graph. It is a two-layer paper multiplex. One layer denotes paper-to-paper associations based on common authorship, and another denotes paper-to-paper associations based on citation relationships. The cross-layer edges connect two papers that have subjects in common. A supra-adjacency representation of this paper multiplex contains two layer-wise graph adjacencies $(A_{(1,1)}, A_{(2,2)})$ along the main diagonal of A, and, the cross-layer adjacencies $(A_{(1,2)}, A_{(2,1)})$ along the off-diagonal of A.

Multilayer vs. Multiplex Graphs. Here, we also distinguish a *Multilayer* graph from a *Multiplex* graph. Unlike multiplex graphs that accommodate the same set of nodes, i.e., replica nodes across the layers, multilayer graphs are more generalized [71, 72]. They do not constrain the graph layers to have the same set of vertices. Thus, multilayer graphs can accommodate different types of nodes in different layers.

Representing relation-wise node interactions within and across layers. The cross-links can be – *trivial* when the edges connect two replica nodes across the layers based on the same identity, or *non-trivial* if the bipartite edges have any other notion of proximity. For example, in a document multiplex where various relations among a set of document nodes are modeled, cross-edges may either represent replica nodes across the layers or represent the cosine similarity between a pair of document nodes based on their component terms. Among the relations, which one to model as relation layers and which one to model as the bipartite connection is a design choice to represent the underlying linked data. In practice [71–73], the sparse interactions are modeled as cross-edges to optimize the cost associated with learning

network embeddings from the multiplex representation. This is since $\binom{\mathcal{R}}{2}$ number of bipartite associations are possible among a set of $\mathcal{R}$ relations.



(a) Transportation networks

(b) Brain networks

(c) Biological networks

(d) Social networks

Fig. 2.7 Multiplex Networks: Examples [Image Source: Internet]

2.2.2 InfoMax Principle for learning network representations

2.2.2.1 Learning network representations on multiplex graphs

Multiplex Network Representation Learning methods encode useful information for all the nodes into a low d-dimensional node embedding, $U_r \in \mathbb{R}^{|\mathcal{V}| \times d}$ for each layer r and then aggregate information across layer by leveraging the cross edges, into a joint embedding, $Z \in \mathbb{R}^{|\mathcal{V}| \times d}$.

Graph Convolutional Networks (GCNs) [18, 19] are widely used node embedding architectures that encode attribute-based local structural information from a node's multi-hop neighborhood. In the context of multiplex networks, GCNs [74, 2, 39] are used layer-wise to obtain node embeddings based on the intra-layer edges. Then, to get a joint node embedding, embeddings from different node counterparts across layers are aggregated via the cross-edges.

2.2.2.2 Encoding global information with InfoMax-based objective

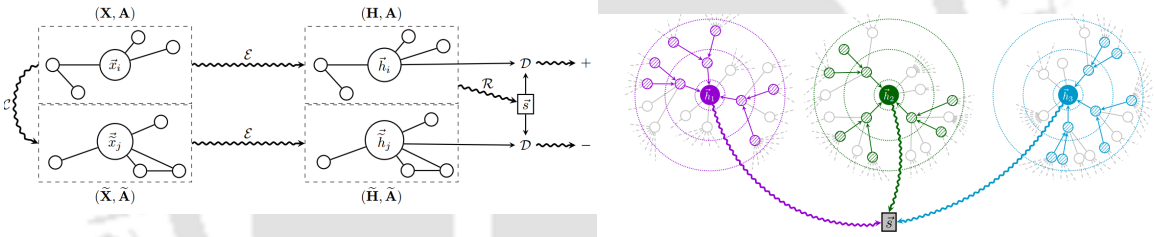
While GCNs are powerful models for encoding local structures, they do not encode global contextual information. On that front, recent efforts in the NRL community have adopted the Mutual Information Maximization (InfoMax) objective, initially proposed in image feature extraction pipelines [75, 76] for learning structurally dependent rich local and global representations of images — to the graph domain to learn rich node [2, 39] and graph-level representations [43]. In problems where the data is sampled from a set of graphs, each data instance is a graph, and the task is to learn a global graph representation wherein the InfoMax-based models learn whole graph representations by maximizing the mutual information (MI) between the global node representations with the local node representations [43]. Nevertheless, in this work, we are interested in the semi-supervised transductive setting for node classification instead of graph classification. Given a partially labeled graph, we look to learn node representations that allow us to predict labels for the rest of the nodes in the same graph. In this setting, existing Infomax models learn the local node representations by maximizing its MI with a single global graph representation [2, 39]. As we argue in Section 4.3 of Chapter 4, where we described this work's intuition, that *this single global representation is often inadequate*.

From the computational aspect, the mutual information between two variables can be maximized by leveraging the KL-divergence between their Joint distribution and the product

of marginals. However, since estimating MI in high dimension and continuous data is complex. In practice, scalable Neural MI estimators [75] that maximize a tractable lower-bound are used. *Noise Contrastive Estimation-based (NCE) loss* [77] that discriminates samples from a true joint distribution against a noisy product of marginals (negative samples) is a simple yet effective way to realize this lower bound. It can be viewed from the predictive coding perspective, where given a global-whole representation, the task is to predict a corresponding local-part representation. This forces the discriminator to provide a high score to a related pair of local-global representations compared to unrelated pairs. Henceforth, unless specified otherwise, we adopt minimizing the NCE loss for this purpose.

Next, we give an overview of InfoMax based models applied to simple homogeneous graphs and how they are trivially extended for learning node embeddings in multiplex graphs for different downstream tasks.

2.2.2.3 InfoMax Principle applied to homogeneous graphs



(a) Summary vectors are obtained by combining several subsampled patch representations

(b) A high-level overview of Deep Graph Infomax

Fig. 2.8 The Deep Graph InfoMax (DGI) Objective [Image Source: DGI [2]]

Deep Graph InfoMax (DGI) [2] is the first-of-its-kind work to propose an InfoMax-based learning objective to learn node embeddings from homogeneous graphs in an entirely unsupervised manner. To do this, it first generates a patch representation $\vec{h}$ of each node capturing its surrounding context using a GCN based encoder $(\mathcal{E})$ as illustrated in Figure 2.8a. The encoder $(\mathcal{E})$ takes graph adjacency matrix A and node feature matrix X as input. It simultaneously employs a *readout* $(\mathcal{R})$ to generate a summarized *global patch representation* $\vec{s}$ from all the generated *local patch representations* $\vec{h}$. Figure 2.8b details the summary vector $\vec{s}$ generation process from the candidate local patch representations surrounding each nodes. Now, to maximize the average MI between the local and global patch representations, a *discrimiantor* $(\mathcal{D})$ is used to discriminate between a pair of true (local, global) patch-pair

representation $(\vec{h},\vec{s})$ from the false ones $(\vec{\tilde{h}},\vec{s})$ as shown in Figure 2.8a. Finally, it employs a noise contrastive estimation loss based on the association probability score generated by $\mathcal{D}$.

$$\mathcal{L} = \frac{1}{M+N}\Big(\sum_{i=1}^{N}\mathbb{E}_{(X,A)}[log\ \mathcal{D}(\vec{h}_i,\vec{s})] + \sum_{j=1}^{M}\mathbb{E}_{(\tilde{X},\tilde{A})}[log\ (1-\mathcal{D}(\vec{\tilde{h}}_i,\vec{s}))]\Big) \tag{2.16}$$

Here, M is the number of negative nodes generated from each node $i \in N$ using a corruption function $\mathbb{E}(.)$. This approach effectively maximizes the MI between $(\vec{h}_i,\vec{s})$ based on the Jensen-Shannon divergence between the joint and the product of marginals [75, 76, 2].

Nevertheless, as we discussed in Section 4.3, the objective of DGI is likely to encode trivial, noisy global information for every node, since the graph-wide global summary is common to all the local patch embeddings even though the local patches are connected in distinct ways with the rest of the network.

2.2.2.4 InfoMax Principle applied to multiplex graphs

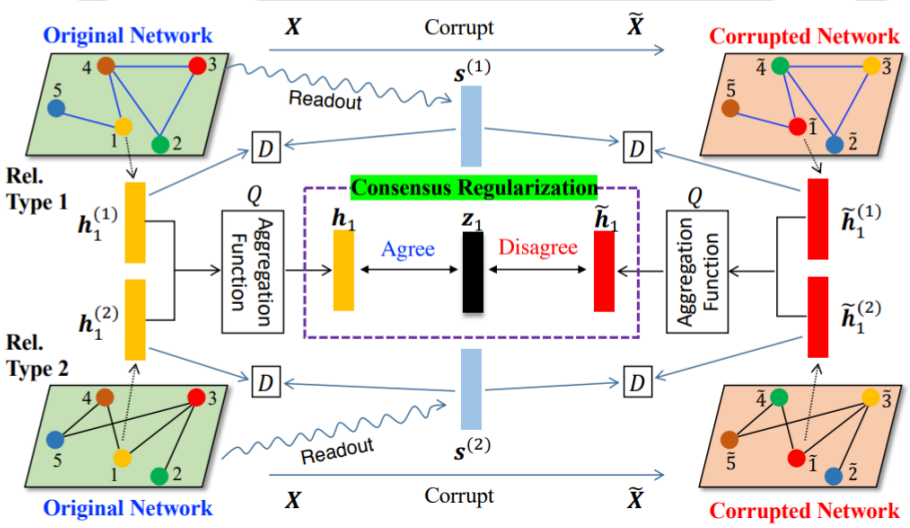


Fig. 2.9 The Deep Multiplex Graph InfoMax (DMGI) Objective [Image Source: DMGI [2]]

Deep Multiplex Graph InfoMax (DMGI) [2] is the first-of-its-kind work to propose an InfoMax based model for the multiplex graphs. Its InfoMax objective is similar to DGI's. It optimizes the average MI between all the (local-global) patch-pair representations in each $\mathcal{R}-$relational layers via a discriminator. This discriminator D as shown in Figure 2.9 is *universal* in essence that it is capable of discriminating between the positive, negative (local-global)

pairs across the layers, thus, more powerful than any set of relation-specific discriminators for this purpose. It then proposes one *consensus regularization* technique, which uses an attention mechanism to meaningfully aggregate relation-specific node embeddings via the *Aggregation Function Q* in Figure 2.9. The consensus regularization strategy minimizes the disagreements among the relation-specific node embeddings to systematically learn a final, unified consensus node representation. DMGI has both unsupervised and semi-supervised attributed variants.

As shown for DGI, DMGI also encodes redundant global graph summaries into unified node representations, clearly lacking in strategy to incorporate meaningful higher-order structures to enrich the node embeddings. Also, DMGI does not have any provision to include cross-layer association and their dependencies, which might benefit any node-wise downstream task.

2.2.3 Semi-Supervised Node Classification Task

Here we formally define the end task at hand and our learning objective.

Definition 2.2.3 (SEMI-SUPERVISED NODE CLASSIFICATION TASK ON MULTIPLEX GRAPHS)**.** Given a multiplex graph, $\mathcal{G}_{\mathcal{M}} = (\mathcal{V}, \mathcal{R}, A)$, a node feature matrix X, a label set, $\mathcal{Q}$ and set of labeled nodes, $\mathcal{L}$ with ground truth label assignment matrix, $Y \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{Q}|}$, the task is to predict labels for all unlabeled nodes, $\mathcal{U} = \mathcal{V} \setminus \mathcal{L}$. For efficient Semi-Supervised Learning (SSL) on multiplex networks, it is essential to learn a low d-dimensional ($d \ll |\mathcal{F}|$) node embedding, $Z \in \mathbb{R}^{|\mathcal{V}|\times d}$ that encodes relevant structural and label correlation information within and across layers, useful for downstream machine learning tasks such as node classification, clustering, and similarity-search.

Table 2.2 details notations used in Chapter 4. Few of the notations are defined in the proposed architecture in Section 4.7 of Chapter 4.

Notation	Meaning
$\mathcal{G}_M$	$(\mathcal{V}, \mathcal{R}, A)$ multiplex graph
$\mathcal{V}$	vertex set, which is common to all the $\mathcal{R}$ layers
$\mathcal{R}$	a set of relations that occur among the vertices
$A_{(r,r)}$	$\in \mathbb{R}_+^{\lvert\mathcal{V}\rvert\times\lvert\mathcal{V}\rvert} : r \in \mathcal{R}\}$ intra-layer adjacency matrix for r layer
$A_{(r,s)}$	$\in \mathbb{R}_+^{\lvert\mathcal{V}\rvert\times\lvert\mathcal{V}\rvert} : r,s \in \mathcal{R}, r \neq s\}$ cross layer adjacency matrix between layer r and s when $r \neq s$
X	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert\times\lvert\mathcal{F}\rvert}$ node feature matrix, associated with a feature set, $\mathcal{F}$
$\bar{\mathcal{G}_M}$	$= (\mathcal{V}_\mathcal{M}, \mathcal{A}_\mathcal{M})$ supra-graph representation of multiplex network
$\mathcal{A}_\mathcal{M}$	supra-adjacency matrix denoting the association among the supra-nodes
U_r, Z	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert\times d}$ layer-wise and unified node embeddings in d latent space, respectively
$\tilde{U}_r$	corrupted local node embeddings
Q	a set of labels
$\mathcal{U}, \mathcal{L}$	a set of unlabeled and labeled nodes, respectively
Y	$\in \{0,1\}^{\lvert\mathcal{V}\rvert\times\lvert\mathcal{Q}\rvert}$ one-hot label matrix for nodes
K	a set of clusters
$\mathfrak{E}_\mathfrak{r}$	relation-wise (r) Graph Neural Encoder to model the local node-contexts upto $M-$hop
S_r	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert\times d}$ relation-wise (r) global node-context
$\mathfrak{D}$	$: \mathbb{R}^{2d} \mapsto \mathbb{R}$ a universal discriminator function to model associations of local and global node-contexts across layers
B	$\in \mathbb{R}^{d\times d}$ bi-linear scoring matrix that capture vector latent interactions
C_r	$\in \mathbb{R}^{K\times d}$ relation (r) specific K cluster embeddings
H_r	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert\times K}$ node-to-cluster probabilistic association matrix
$\mathcal{S}$	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert\times\lvert\mathcal{V}\rvert}$ label-correlated similarity kernel for learning node clusters

Table 2.2 Notations used in Chapter 4 and their meanings

2.3 Preliminaries: Heterogeneous Graphs

This section explains all the basic notations and definitions related to heterogeneous graphs.

2.3.1 Heterogeneous Graphs

Definition 2.3.1. HETEROGENEOUS INFORMATION NETWORK (HIN) is defined as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \psi, \phi)$. $\mathcal{V}$ is the set of vertices. $\mathcal{E}$ is a set of edges. $\psi : \mathcal{V} \mapsto \mathcal{A}$ and $\phi : \mathcal{E} \mapsto \mathcal{R}$ are node and edge type mapping functions, where $\mathcal{A}$ and $\mathcal{R}$ are the set of node types and edge types respectively. In HINs, $|\mathcal{A}| + |\mathcal{R}| > 2$, unlike in homogeneous networks where $|\mathcal{A}| = 1$ and $|\mathcal{R}| = 1$.

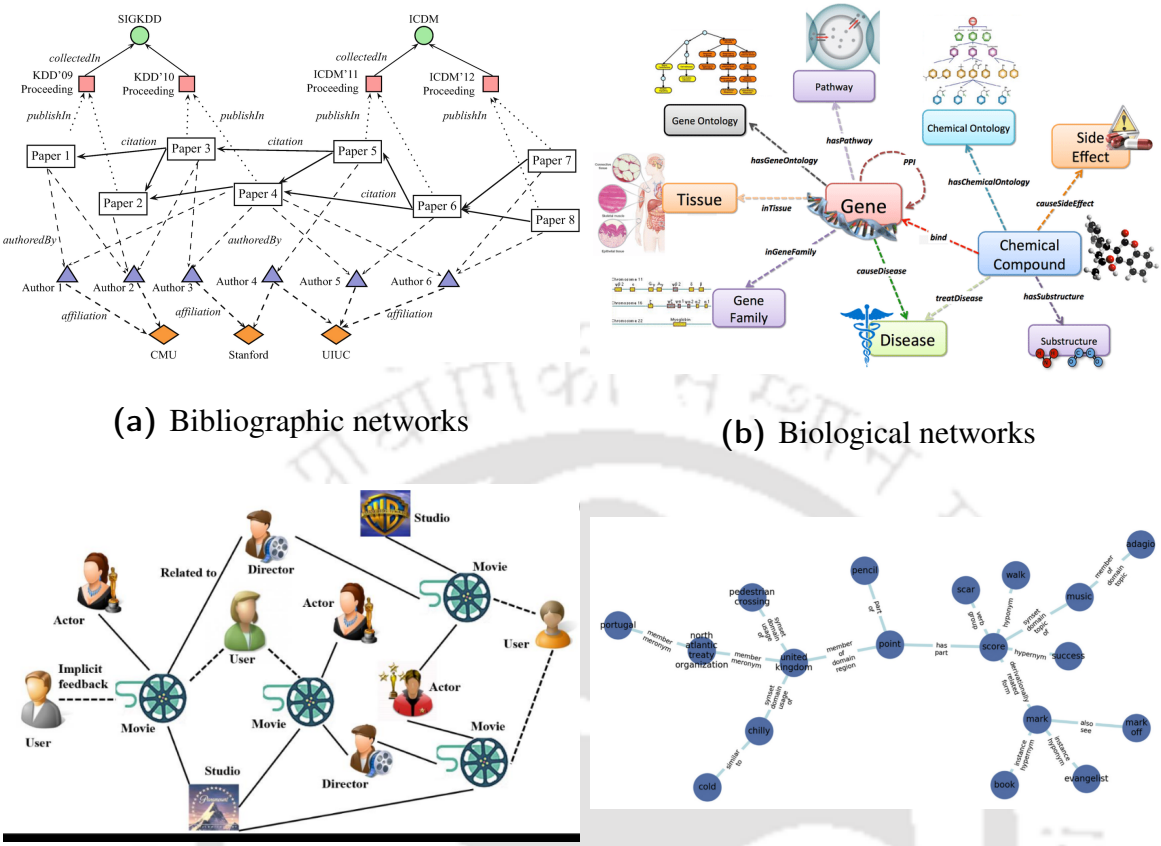
Figure 2.10 depicts a few real-world scenarios that can be modeled as heterogeneous networks. Bibliographic networks, biological networks, movie networks are examples of graphs that comprise different types of vertices and edges of unique semantic meanings. In wordnets, we see nodes are of a single type, but there are several semantic associations among them.

Definition 2.3.2 (HIN SCHEMA)**.** A HIN schema is defined as a meta-graph $\bar{\mathcal{G}} = (\mathcal{A}, \mathcal{R})$. The set of node types $\mathcal{A}$ is the vertex-set, and the set of edge-types $\mathcal{R}$ is the edge-set for $\bar{\mathcal{G}}$. It is an abstract representation of the underlying HIN in typed essence. It provides meta-information regarding the node and edge types. Also, it specifies a set of interaction rules among the nodes based on relation semantics. HIN schema is also known as Metagraph.

Figure 2.11a shows an example HIN schema for bibliographic networks. It is a star schema with the paper node in the center. The paper node is associated with other node types such as authors, venue, terms/ keywords, and subjects/ fields of study. An author is affiliated with a certain Institute. Figure 2.11b details more on how different node types interact with each other.

Definition 2.3.3 (ASPECT)**.** An aspect $\bar{\mathcal{G}}^a = (\mathcal{A}^a, \mathcal{R}^a)$ is defined as a subgraph of the HIN schema $\bar{\mathcal{G}}$. Such that vertex-set $\mathcal{A}^a \subseteq \mathcal{A}$ and edge-set $\mathcal{R}^a \subseteq \mathcal{R}$ denote all the node-types and edge-types involved in the aspect.

An aspect can be thought of as a perspective. Any specific composite semantic meaning is thought to be associated with the participating node types. In Figure 2.11d, we give examples of three such aspects of the example bibliographic HIN schema. Aspect A : models the view



(a) Bibliographic networks

(b) Biological networks

(c) Movie networks

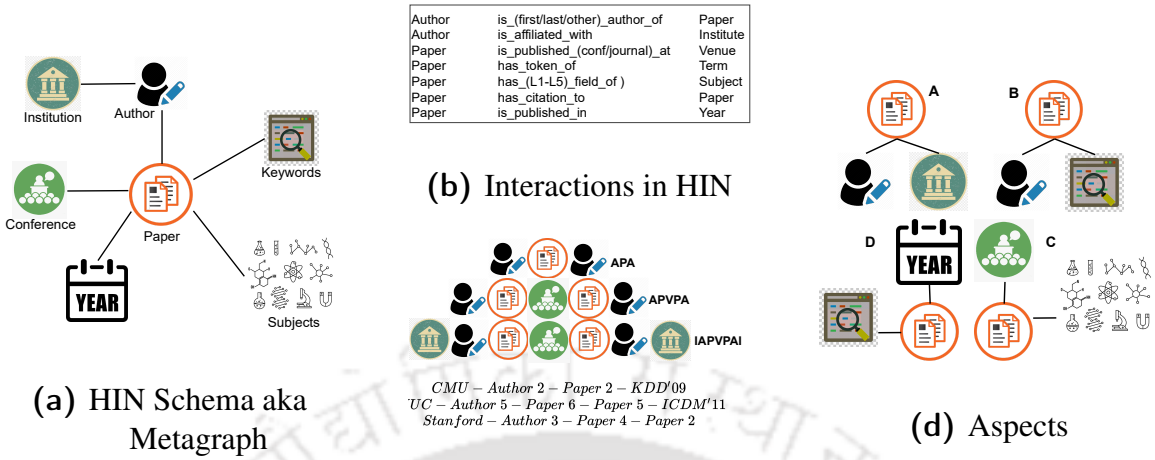
(d) Word networks

Fig. 2.10 Heterogeneous Networks: Examples [Image Source: Internet]

– authors affiliated with certain institutes tend to write research papers. Aspect B : tells us about – authors tend to use certain keywords or terms while writing research papers. Aspect C : models the view – papers belonging to specific fields of study get published in certain venues. Aspect D : tells us about – specific keywords associated with research papers getting trending in a year.

Definition 2.3.4 (PATH)**.** An $M-$length path from a source node s to a target node t in HINs is a sequence of vertices and edges. It is represented as, $s(v_0) \xrightarrow{e_0} v_1 \xrightarrow{e_1} v_2 \ldots v_{M-1} \xrightarrow{e_{M-1}} t(v_M)$, where e_is are edges between $v_i \xrightarrow{e_i} v_{i+1}$.

Definition 2.3.5 (METAPATH)**.** An M-length HIN path between a node-pair (s,t), can be represented in terms of the HIN's meta-level network schema composed of constituent node type (ψ) and edge type (ϕ). This meta-level path representation is called Metapath.



(a) HIN Schema aka Metagraph

(b) Interactions in HIN

(c) Metapaths vs. Paths

(d) Aspects

Fig. 2.11 Heterogeneous Networks: Various components

Let a M-length path in HIN be denoted as, $s(v_0) \xrightarrow{r_0} v_1 \xrightarrow{r_1} v_2 ... v_{M-1} \xrightarrow{r_{M-1}} t(v_M)$, where $r_i = \phi(e(v_{i-1}, v_i))$[3]. The metapath associated with this path is represented as $(r_0, r_1, .. r_{M-1})$.

A metapath is a composition of relations that reflects complex semantic meaning associated with its end nodes. It is a simplification of the underlying path to contain only a sequence of edge types information. Figure 2.11c differentiates between a path and metapath. We see examples of a few metapaths with unique interpretations — *APA:* denotes two authors are co-authors of a common paper, *APVPA:* two authors publishing at the same venue, *IAPVPAI:* authors from different institutes are publishing papers at a common venue. A few paths are shown at the bottom of Figure 2.11c. The paths are extracted from the example bibliographic heterogeneous graph as shown in Figure 2.10. Unlike metapaths, paths involve node and edge identities. Therefore, for a fixed length M, the number of unique paths can grow exponentially as compared to the number of candidate metapaths.

Definition 2.3.6 (TRIPLE)**.** An edge, $e(s,t) \in \mathcal{E}$ in a heterogeneous network, $\mathcal{G}$ is represented as a triple, (s,r,t) where $s \in \mathcal{V}$ is the source node, $t \in \mathcal{V}$ is the target node and $\phi(e) = r : r \in \mathcal{R}$ is the type of relation between them.

In HINs, in addition to the existence of an edge (e), the edge type ($r = \phi(e)$) also needs to be identified. We also distinguish here between two kinds of triples — *i)* (s, r_t, t) a relation

[3]Note that, since the relation type in HIN is indicative of the type of nodes it connects, in our work, we simplify the metapath representation by ignoring the node's type information and only considering the relation type between nodes.

triple whose binary existence is to be predicted with r_t being the target relation type; *ii)* (s,r,t) any other relational triple.

2.3.2 Heterogeneous Information for predicting links

Predicting the existence of a link between a pair of nodes requires capturing relevant information surrounding the potential edge (node pairs) to support its possible existence. This involves capturing relevant contextual information for the (potential) edge at multiple scales. Our third contribution focuses on capturing edge information at the (i) finer (meta)-path level, (ii) coarser local-neighborhood level, and (iii) global graph level. Next, we explain the necessary concepts required to capture this different multi-scale information with respect to a heterogeneous network.

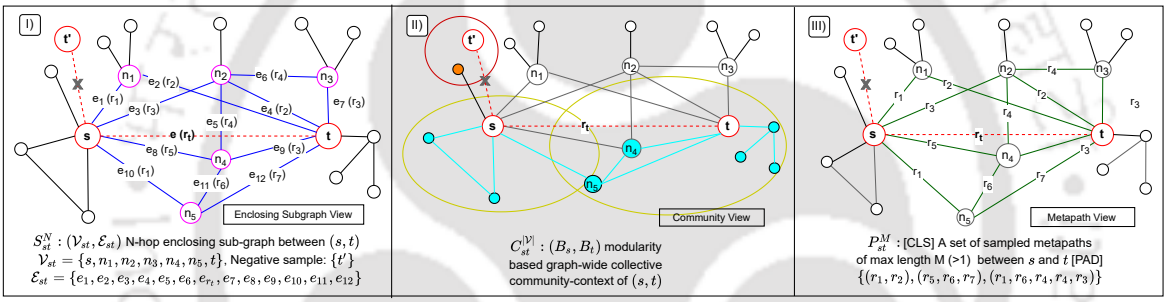


Fig. 2.12 Various views to contextualize a relation triple. [Highlights denote different contexts, I) Blue, Purple: Subgraph, II) Cyan, Yellow: Community, III) Green: Metapaths.]

Definition 2.3.7 (METAPATH CONTEXT OF A TRIPLE)**.** Given a relational triple (s,r_t,t) where source node s is connected to target node t via relation of type r_t, a set of metapaths of length upto M from source s to target t is called as *metapath context of triple*. We denote a metapath context of (s,r_t,t) as, $P^M_{st} = \{ \bigcup_{m=[1,M]} P^m(r_0, r_1, ..., r_{m-1}) : (s \xrightarrow{e_0} v_1) \wedge (v_{m-1} \xrightarrow{e_{m-1}} t)\}$.

Sub-Figure III) of Figure 2.12 shows an example of sampled metapath context for a triple. Out of all available metapaths (highlighted in Green) between (s,t), we randomly sample three metapaths $\{(r_1,r_2),(r_5,r_6,r_7),(r_1,r_6,r_4,r_4,r_3)\}$ to represent the target relation r_t.

Note that we do not use the identities of intermediate nodes while modeling the relational paths. This is to obtain a better inductive bias for inferring unknown links. This also significantly limits the number of unique candidate metapaths to incorporate. Nonetheless, the

number of unique metapaths can still be exponential ($|r|^M$, $M-$length metapaths). However, studies [48] suggest that, in real-world HINs and KGs, most metapaths do not occur; for example, only 3.2% of all possible metapaths of length-2 occur in FB15K dataset (see Table 5.2). Further, the number of relational paths is manageable if small values of path-length (M) and a predefined number of path samples between the node pairs are considered. The number of path samples is a hyper-parameter that can be tuned based on the underlying dataset so that they collectively optimally represent a target relation.

Definition 2.3.8 (ENCLOSING SUBGRAPH CONTEXT OF A TRIPLE)**.** Given a relational triple (s, r_t, t), if $\mathcal{N}^N(s), \mathcal{N}^N(t)$ denote the $N-$hop neighborhood of nodes s and t respectively, an enclosing subgraph is the induced subgraph from the common neighborhood-set $\{\mathcal{N}^N(s) \cap \mathcal{N}^N(t)\}$ and the underlying typed-edges including r_t, of the considered collective heterogeneous subgraph between (s,t). An enclosing subgraph context $S^N_{st} : (\mathcal{V}_{st}, \mathcal{E}_{st})$ denotes the induced common $N-$ hop subgraph comrpising of vertex-set $\mathcal{V}_{st}$ and edge-set $\mathcal{V}_{st}$ between (s,t).

Sub-Figure I) of Figure 2.12 shows an example of enclosing subgraph context for a triple. The participating nodes and edges are highlighted (Purple: intermediate nodes, Blue: edges of certain relations, Red: end nodes and target relations) and are within the $N-$hop neighborhood surrounding (s,t). Clearly, this context takes participating nodes and edges into account, unlike the metapath context, which only considers the sequence of relational semantics between the end nodes.

Definition 2.3.9 (COMMUNITY CONTEXT OF A TRIPLE)**.** A community context $C^{|\mathcal{V}|}_{st}$ of a relational triple (s, r_t, t) represents the triple's global graph-wide connectivity pattern in terms of its participating nodes' (s,t) tendency of forming connections globally with the rest of the nodes $i \in \mathcal{V}$ in the graph with an aim to constitute that triple's higher-order neighborhood.

In this work, we use widely adopted modularity maximization based community learning [29, 34]. If $B \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ denotes the modularity matrix with entries $B_{(i,j)} = A_{(i,j)} - \frac{\text{DEG}(i).\text{DEG}(j)}{2e}$ such that $\text{DEG}(i)$ denotes the degree of node i, $|\mathcal{E}| = e$ is the total number of edges and A is the graph adjacency matrix, we consider $C^{|\mathcal{V}|}_{st} = (B_s, B_t)$ as the B modularity matrix-based community context. Sub-Figure II) of Figure 2.12 shows an example of community context for a triple. The regions highlighted with Yellow boundary depicts the communities surrounding s and t respectively. Sharing of similar characteristics between these two communities is likely to provide us more evidence on the probability of forming an edge between (s,t).

2.3.3 Contextual encoding of triples for Link Prediction

Given an HIN organized as a set of triples $\mathcal{T}$, we generate the contexts of a triple $(s, r_t, t) \in \mathcal{T}$ denoted as $(S^N_{st}, P^M_{st}, C^{|\mathcal{V}|}_{st})$. A context stands for any topological structure extracted from the underlying graph that provide important evidence about the formation of a link between a source s and target t nodes. In this work, we use common subgraph S^N_{st}, relational path P^M_{st} and community $C^{|\mathcal{V}|}_{st}$ based contexts as structural cues for predicting links. This contextualized relational triples of the form $< s, r_t, t, (S^N_{st}, P^M_{st}, C^{|\mathcal{V}|}_{st}) >$ are encoded using any suitable encoding model.

2.3.4 The problem of Structural Link Prediction

Given an observed graph $\mathcal{G}_{\mathcal{T}} = (\mathcal{V}_{\mathcal{T}}, \mathcal{T}, \psi, \phi)$ which is present. And, the future graph for $\mathcal{G}_{\mathcal{T}}$ is $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \psi, \phi)$. $\mathcal{G}_{\mathcal{T}}$ has nodes $\mathcal{V}_{\mathcal{T}} \subseteq \mathcal{V}$ (identical node-set in the transductive, subset of nodes in the inductive setup) and a set of $\mathcal{T} \subset \mathcal{E}$ observed edges. Then the aim of link prediction is to infer the missing future edges in $\mathcal{G}_{\mathcal{T}}$.

This involves discriminating true positive edges $\{\mathcal{E} - \mathcal{T}\}$ from a set of false edges generated using a corruption function $\mathcal{NS}(\mathcal{E} - \mathcal{T}) = \{\mathcal{E} - \mathcal{T}\}' \in \{\mathcal{V} \times \mathcal{R} \times \mathcal{V} - \mathcal{T}\}$. A learning model $\mathcal{L}(\mathcal{G}_{\mathcal{T}}) : \mathcal{V} \times \mathcal{V} \mapsto \mathbb{R}, \mathbb{R} \mapsto \{True, False\}$, which usually comprises an encoder to embed the triples along with their semantic contexts, and, a discriminator to discriminate the true and false edges from the test edge-set, $\mathcal{T}_{test} = \{\mathcal{E} - \mathcal{T}\} \cup \{\mathcal{E} - \mathcal{T}\}'$. The discriminator produces a score to determine the plausibility of a triple. The $\text{SCORE}(s, r_t, t) \in \mathbb{R}$ can be leveraged by any proper objective to train the link prediction model in an end-to-end manner.

We focus on both inductive and transductive link prediction setups. In the inductive setup, $\mathcal{V}_{\mathcal{T}} \subset \mathcal{V}$, whereas, in the transductive setup, $\mathcal{V}_{\mathcal{T}} = \mathcal{V}$. The inductive setup requires predicting links between a pair of newly added nodes or between a new and an existing node. In contrast, with the transductive setup, the vertex set $\mathcal{V}$ is static, and thus the task requires only completing missing links or predicting new links between existing nodes.

Table 2.3 details notations used in Chapter 5. Few of the notations are defined in the proposed architecture in Section 5.7 of Chapter 5.

Notation	Meaning
(s, r_t, t)	(source node, target relation to predict, target node)
Y	$\in \{0,1\}$ edge labels for positive and negative triples
f_r, d	size of relational features and latent representation space
S_{st}^N	enclosing subgraph context of N-hop neighborhood between (s,t)
P_{st}^M	$= \{p_1, p_2, ..., p_{\lvert P_{st}^M \rvert}\}$ set of $M(>1)$ length metapath context between (s,t)
$C_{st}^{\lvert\mathcal{V}\rvert}$	graph-wide community context for edge between (s,t)
$\mathcal{N}_r^N(s)$	N-hop neighborhood of a node s based on r relation type
$\mathcal{NS}(.)$	corruption function to generate negative triples given a positive triple
H, H_r	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert \times d}$, $\in \mathbb{R}^{\lvert\mathcal{R}\rvert \times d}$ node and relation embeddings
C	$\in \mathbb{R}^{K \times d}$ embeddings of K communities
U	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert \times K}$ node-to-community probabilistic association
B	$\in \mathbb{R}^{\lvert\mathcal{V}\rvert \times \lvert\mathcal{V}\rvert}$ Modularity matrix, $B_{(i,j)} = A_{(i,j)} - \frac{\text{DEG}(i).\text{DEG}(j)}{2e}$, A : graph adjacency, $e = \lvert\mathcal{E}\rvert$
CS_i	$\in \mathbb{R}^d$ global community summary of a node $i \in \mathcal{V}$
$\mathcal{S}_{(s,r_t,t)}$	$\in \mathbb{R}^d$ subgraph view of the triple (s, r_t, t)
$\mathcal{P}_{(s,r_t,t)}$	$\in \mathbb{R}^d$ path view of the triple (s, r_t, t)
$\mathcal{C}_{(s,r_t,t)}$	$\in \mathbb{R}^d$ community view of the triple (s, r_t, t)
$\alpha_{\mathcal{SPC}}$	$\in \mathbb{R}^{\lvert\mathcal{T}\rvert \times 3}$ view attention scores for subgraph, path and community views for triples $\in \mathcal{T}$
spc	$\in \mathbb{R}^{\lvert\mathcal{T}\rvert \times d}$ attentively aggregated view representation of train-triples $\in \mathcal{T}$

Table 2.3 Notations used in Chapter 5 and their meanings

Chapter 3

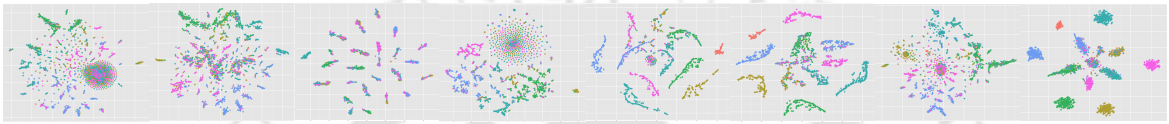
Structure-Aware Network Representation Learning on Homogeneous Graphs

In the first contribution, we propose a Semi-Supervised Learning (SSL) framework, USS-NMF, that allows for explicitly encoding different necessary priors to learn efficient node representations in a graph. USS-NMF specializes in encoding the important yet largely ignored necessary prior for SSL, the cluster assumption. The cluster assumption of SSL requires the existence of well-separated dense regions in a low-dimensional manifold with high label smoothness within each region. USS-NMF encodes this assumption in the form of a proposed semi-supervised cluster invariance constraint, which is a group-level smoothness constraint on nodes. We show that explicitly enforcing this constraint enables learning meaningful node representations from both qualitative (visual) and quantitative standpoints. Specifically, USS-NMF achieves superior performance on semi-supervised node classification and clustering tasks across thirteen datasets from over eight baselines. Also, the learned node embeddings from USS-NMF yield high-quality (well-separated homophilous) clusters in t-SNE visualizations.

3.1 Introduction

Over the last two decades, researchers have increasingly observed the effectiveness of semi-supervised learning (SSL) – the process wherein unlabeled data is judiciously levered with a small amount of labeled data to produce accurate and cost-effective models. An essential class of SSL methods is the graph-based SSL methods that model the underlying

data manifold through a graph and utilize that to enforce smoothness on a target function, typically in the space of labels. The smoothness properties are generally encoded with graph kernels [70]. The graph can either be computed from the data (e.g., nearest-neighbor graphs) or may arise naturally apriori (e.g., social networks). Both labeled and unlabeled data are represented as nodes, and the edges reflect a meaningful notion of similarity. Graph-based SSL methods typically employ a Graph Laplacian-based smoothness regularizer for the classification objective. The regularizer constrains the labels of connected nodes to be similar [78].



(a) MFDW (b) MNMF (c) GEMSEC (d) COM-E (e) MFDW+Y (f) MNMF+Y (g) MMDW (h) USSNMF

Please refer to Section 3.8 for the candidate methods for which the t-SNE visualizations are plotted here.

Fig. 3.1 t-SNE Visualization of Embeddings on Citeseer Dataset for Unsupervised and Semi-Supervised Methods

For graph-based SSL methods, the choice of the underlying data representation (embedding space) is critical. Traditionally, spectral embedding was the de-facto standard for such models. Recently, models that learn distributed node representation based on stochastic flows (random walks) have been popular [16, 17]. Embeddings learned from these models considered a larger multi-hop context and were shown to be more potent than the standard spectral embeddings on tasks like node classification and link prediction. Besides their superior performance over standard spectral embedding, such embedding methods have seen wide adoption for scalability reasons. They leveraged the skip-gram model with negative sampling or noise contrastive estimation to approximate the partition function computation by evaluating only a smaller subset of negative samples [79]. Along these lines, several models have been proposed that replace the random-walk-based objective with the corresponding matrix factorization objective [58, 80, 8]. These models have been shown to work well in practice.

Despite the popularity of node embedding-based methods for the classification task, there exist limited works that learn semi-supervised node embeddings for non-attributed graphs, i.e., graphs with no node or edge features. In this work, we propose a Unified Non-negative Matrix Factorization based framework, USS-NMF, for learning Semi-Supervised node representations. The proposed framework specializes in learning cluster invariant representations of nodes. The learned node embeddings are densely clustered into regions of the same or

similar labels[1]. These node representations provide not only superior visualizations and qualitative clusters, but also improved node classification results.

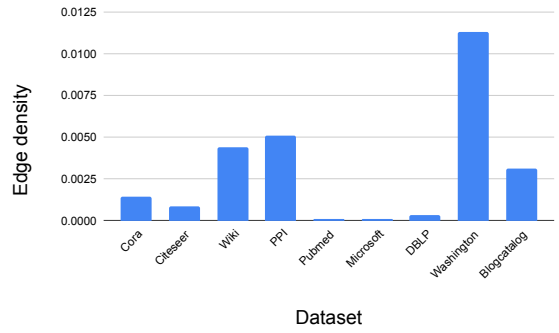
3.2 Challenges

Having motivated the need for incorporating supervision knowledge to enrich network representations, here we discuss *challenges* [13–15, 1, 81] typically exhibited in homogeneous graphs, especially in the context of node-wise downstream tasks.

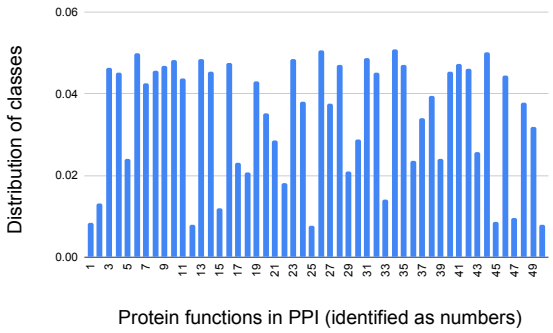
Link sparsity Real-world graphs have a lesser number of observable and/or known edges. Figure 3.2a depicts the real-world scenarios where edge-densities are plotted for several datasets. The metric *edge-density* which is calculated as $\frac{|E|}{\binom{|V|}{2}}$ is an indicator of *link-sparsity* [5]. It measures what fraction is the number of observed edges to the total number of possible edges in a complete graph (clique). Ideally, edge-density is close to 1 for the dense graphs ($= 1$ for the cliques). Nevertheless, we see that a very less number of edges exist for various graphs. The absence of links affects NRL models that heavily rely on the node associations for prediction, special mention of graph convolution neural networks [18, 19] that rely on message-passing mechanisms from the surrounding nodes for learning latent node features of a target node.

Label sparsity Very less number of network components (nodes/edges/graphs) of interest are functionally annotated for a target downstream task (e.g. node/edge/graph classification). This is because the annotation process is costly and resource-consuming. Even if the functional classes exist, they have a skewed distribution [82–84]. This affects the training of decision algorithms due to underfitting on rare classes and overfitting on highly visible classes leading to less generalizability. Figure 3.2b shows us the label-sparsity in protein-protein interaction networks (PPI). It is a multi-label dataset, where significantly fewer nodes are functionally annotated with 50 candidate protein function (candidate classes) — apparent from the Y-axis values. Figure 3.2c shows us the skewed class-distribution of Wiki graph. Wiki is a multi-class graph with 17 functional classes. It has inherent class-imbalance due to the varying number of nodes functionally annotated with class labels leading to a significantly uneven number of representative nodes for each class.

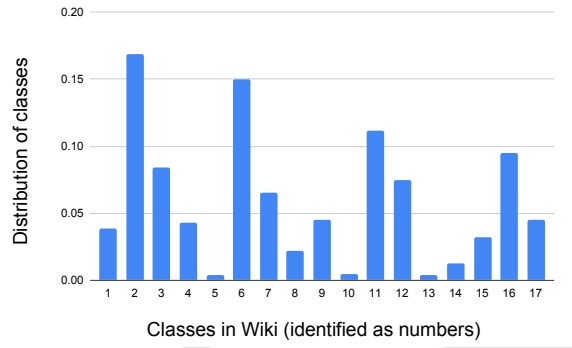
[1]For an initial preview, observe the t-SNE plot presented in Figure 3.1 that visually compares several state-of-the-art methods with the proposed USS-NMF method.



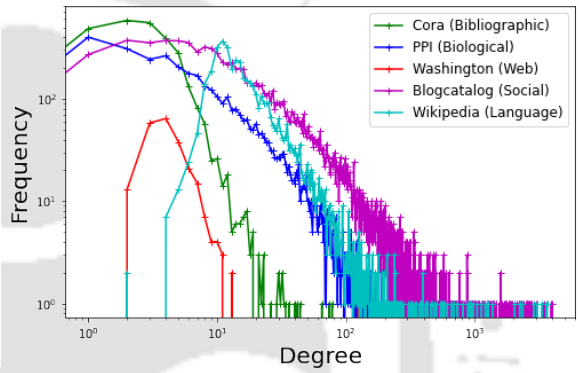
(a) Edge sparsity in datasets



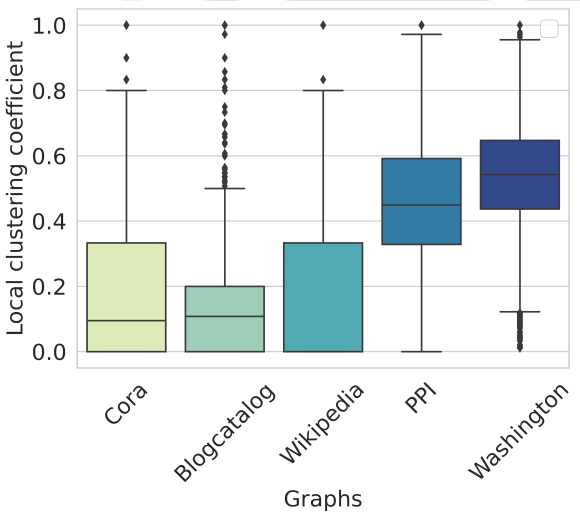
(b) Label sparsity in PPI



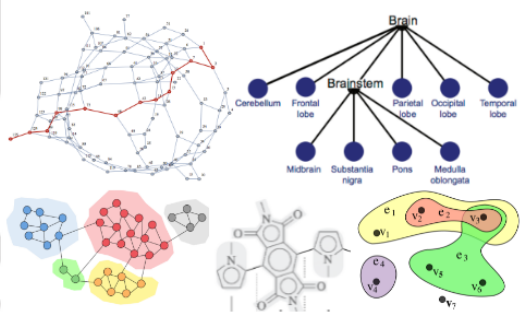
(c) Skewed class distribution in Wiki



(d) Degree distributions in datasets



(e) Distribution of local clustering coefficients



(f) Higher-order structures at multi-scale for homogeneous graphs. Examples include: walks, hierarchies, hyperedges, motifs, communities (in a clock-wise manner). [Image Source: Partially from the Internet]

Fig. 3.2 Challenges in homogeneous graphs [Refer to Table 3.1 for dataset details]

Lack of features The issue of data sparsity exists at various levels — link-level, class-level, and also, at content, aka feature-level. Many network components have missing and/or unknown values for a set of related attributes. These datasets also suffer from fundamental data quality issues [81]. For example, 42% and 65% of nodes have information leakage between node features and class-labels in Cora and Citeseer, respectively. The datasets also suffer from node duplications, redundant and noisy features associated with network components, etc. Due to this, *property prediction at node, edge, and graph levels* [81] are becoming crucial research directions to explore recently.

Preserving network characteristics Graphs in real-world exhibit various *statistical network characteristics* [60, 85, 86] — homophily, heterophily, power-law degree distribution, scale-free property, small-world property, cohesion-coupling, centrality, assortativity-mixing, preferential attachments, information-flow, modularity, etc. are to name a few. Based on the data domains, they exhibit such statistical network properties at varying degrees. For example, technological and biological networks typically show disassortative mixing[2], i.e., the tendency of high degree nodes to connect to low degree nodes. Whereas social networks show highly assortative mixing[3] – where high degree (prestigious nodes) prefer to connect to other high degree nodes. Figure 3.2d shows diverse degree-distribution[4] characteristics of graphs from different domains. We see that Language and Web domains have very different degree-distributions than the rest. Few of the distribution curves are steep (Cora), and few are spread (Blogcatalog). From Figure 3.2e we see varied clustering tendencies[5] in the candidate graphs. Biological network PPI and Web network Washington have more clustering tendencies than Language, Bibliographic domains [29]. For NRL methods, capturing such statistical non-triviality irrespective of the data domains, is crucial to design better decision algorithms for an end task.

Learning from higher-order graph structures Identifying and incorporating various local, topological, and global features into node embeddings are important to aid decision algorithms to predict better [90, 16, 29, 91, 92, 26]. A number of choices exist for selecting high-order graph structures to incorporate — walks, hierarchies of concept in

[2]Tendency of a graph (or network) in which dis-similar nodes are more likely to form connections. [87]

[3]Assortativity, or assortative mixing is a preference for a network's nodes to attach to others that are similar in some way. Though the specific measure of similarity may vary, network theorists often examine assortativity in terms of a node's degree. [87]

[4]Degree distribution is the probability distribution of member nodes' degrees over the whole network. [88]

[5]Clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together. [89]

linked data, communities, subgraphs, motifs, and multi-arity connections (conceptualized as hyperedges in hypergraphs). However, based on the end task, a certain graph structure may prove to be more useful than the rest. Given such a global structure, designing a learning model that efficiently incorporates it into node embeddings remains an open research question.

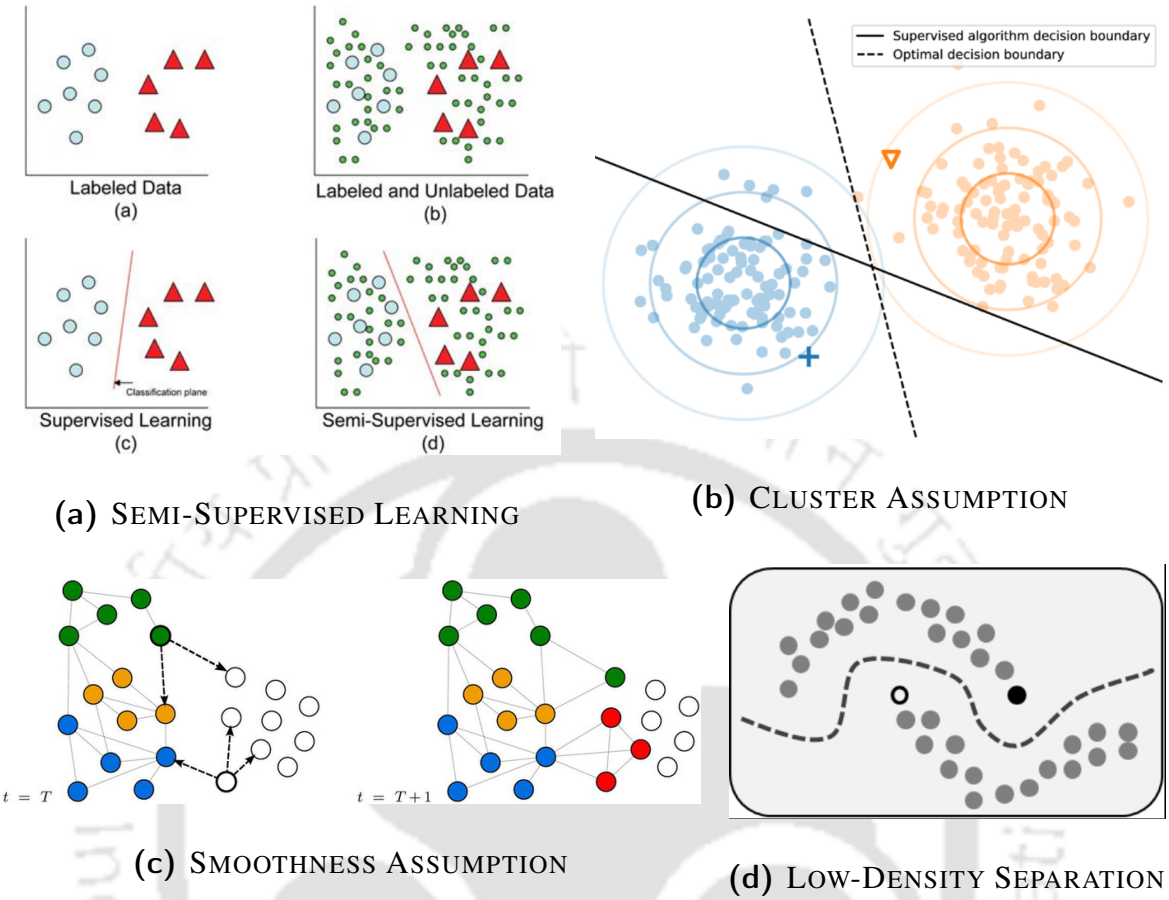
3.3 Motivation

Semi-Supervised Learning (SSL) [36] is a learning paradigm that uses a small amount of labeled data besides the unlabeled for inference. It is halfway between the *supervised* and *unsupervised* learning algorithms. It is a special instance of *weakly-supervised* learning from partially labeled data (often occurs in the real world). Figure 3.3a provides more clarity on the learning principle. In addition to the assumptions of supervised learning, SSL takes the density of underlying data points into account so that the learned decision boundary obtains more insight into the optimal class-boundary separation.

The motivation of our proposed framework USS-NMF stems from the classic assumptions of Semi-Supervised Learning [36], which says, for learning a meaningful inference procedure, a data point x should carry useful information for estimating the target function y, i.e., $\Pr(x)$ should help to infer $\Pr(y|x)$. The principal assumptions that serve as prerequisites for learning useful inference using both labeled and unlabeled include,

SMOOTHNESS ASSUMPTION The target function of two closely connected points in a dense region should also be close. It is often referred to as the *continuity assumption* or as the *label smoothness assumption*.

CLUSTER ASSUMPTION Data points belonging to the same cluster are likely to be of the same class as data from each class tend to form clusters. This, in turn, implies that *the decision boundaries should lie in low-density regions*. It can be seen as a special case of smoothness assumption. Nonetheless, unlike the smoothness assumption, which enforces the smoothness of the target function between any pair of points under consideration, the cluster assumption imposes continuity of the target function on high-density regions or groups. This assumption does not necessarily imply that there should be one compact cluster per class, but instead, there can be multiple small clusters of the same class *with low-density separation among them*



(a) SEMI-SUPERVISED LEARNING

(b) CLUSTER ASSUMPTION

(c) SMOOTHNESS ASSUMPTION

(d) LOW-DENSITY SEPARATION

Fig. 3.3 Semi-Supervised Learning and its principal assumptions [Image Source: Internet]

LOW-DENSITY SEPARATION The decision boundary should lie in a low-density region. The smoothness and cluster assumptions already imply this.

MANIFOLD ASSUMPTION The data lie approximately on a *manifold*[6] of much lower dimension than the input space. This assumption allows learning using distances and densities defined on the manifold, where learning relevant predictive functions without being affected by the *curse of dimensionality* is facilitated.

As shown in Figure 3.3c, the smoothness assumption states that if two nodes are closely connected by any notion of similarity, it is highly likely the connected data points will have

[6]In mathematics, a manifold is a topological space that locally resembles Euclidean space near each point. More precisely, an n-dimensional manifold, or n-manifold for short, is a topological space with the property that each point has a neighborhood that is homeomorphic to an open subset of n-dimensional Euclidean space. [36]

the same labels. It is the basis of *Label Propagation*[7] algorithm. Figure 3.3b illustrates the intuition clearly. The unlabeled data points, connected by paths in dense regions, tend to form groups with coherent label distribution since they follow the continuity assumption. Although, data that share a label may spread across multiple clusters. Thus, an optimal decision boundary should respect the high-density clusters of data points and lie in a low-density region. Figure 3.3d, depicts the preference of decision boundaries for low-density regions.

In this work, we propose a novel semi-supervised cluster invariant constraint that smooths nodes' cluster assignments using a graph Laplacian regularizer to incorporate the cluster assumption of SSL. Though there are previous works, which embed data in high-density clusterable regions, they are primarily unsupervised. In the classical SSL literature, graph-based approaches that preserve the geodesic distance were said to learn high-density clusters. Cluster kernel [10] is one of those famous works that obtains a new representation for (non-networked) data with parameterized spectral embeddings and uses them independently as features for classification. However, they were not used for learning non-attributed network embedding. Recently, there has been an interest in obtaining clusterable embedding by preserving community information in works like MNMF [29], ComE [30], and GEMSEC [35]. They have shown that explicitly adding clustering component to the network learning model help achieve better clusters and better performance on downstream tasks. Nevertheless, these clustering objectives and models are purely unsupervised.

3.4 Research Objective

The objective of this study is to learn higher-order graph structure-aware node embeddings on homogeneous graphs keeping in mind the challenges (refer to Section 3.2) exhibited for NRL tasks. Towards this goal, we find the cluster assumption of SSL as a useful means to learn logical groupings of nodes, especially for the transductive semi-supervised node embedding learning setup, to preserve the common surrounding contexts and the label-correlations between nodes. Learning explicitly from the semi-supervised node clusters enriches the resultant node embeddings. We aim to design an all-encompassing framework that incorporates all the essential learning assumptions from the classic SSL literature.

[7]Label propagation is a semi-supervised machine learning algorithm that assigns labels to previously unlabeled data points. At the start of the algorithm, a (generally small) subset of the data points have labels (or classifications). These labels are propagated to the unlabeled points throughout the course of the algorithm. [36]

3.5 Contributions

Our work unifies a number of existing and novel SSL objectives under this NMF framework, USS-NMF. Our unified approach focuses on learning embeddings that respect class labels as well as cluster structures in the graph. The components of the framework follow - the core concepts of the classic paradigm of Semi-Supervised Learning [36], which states that efficient SSL requires that the data lie in — (1) a low-dimensional manifold, (2) exhibit high label smoothness characterized by homogeneous high-density clusters of the same class and (3) which are well separated from the clusters of different classes. Clusterability, though a well-known prior, was either largely ignored or not explicitly handled in earlier NMF-based SSL methods [93]. To the best of our knowledge, we are the first to explicitly learn semi-supervised cluster invariant node representations. The following are the primary contributions,

Semi-Supervised Cluster Invariance Property for nodes This property enforces a group-level smoothness constraint on nodes based on label-correlations. For this, the framework learns label-informed, semi-supervised clusters to group the nodes in the underlying graph.

Incorporating Cluster Assumption of SSL USS-NMF incorporates the *cluster assumption* from the classic paradigm of Semi-Supervised Learning. Also, it explicitly encodes all necessary priors facilitating SSL.

A novel joint node and cluster representation learning framework Our Unified Semi-Supervised Non-negative Matrix Factorization framework, USS-NMF, specializes in obtaining clusterable node embeddings with high label smoothness within these learned clusters. It is a unified node and cluster representation learning setup that obtains enriched higher-order graph structure-aware node embeddings.

SoTA Performance We demonstrate the effectiveness of USS-NMF for node classification tasks across a range of datasets while comparing it with several State-Of-The-Art (SoTA) baselines. We evaluate the cluster quality of the embeddings with Normalized Mutual Information (NMI) scores. The model obtains high-quality clusters against a range of embedding models. Further, the visualizations obtained with USS-NMF embeddings offer excellent visual separability comparatively (Figure: 3.1).

3.6 Literature Review

In this section, we discuss network representation learning (NRL) literature focused on homogeneous graphs, especially in the context of node-wise downstream tasks.

3.6.1 NRL on homogeneous graphs

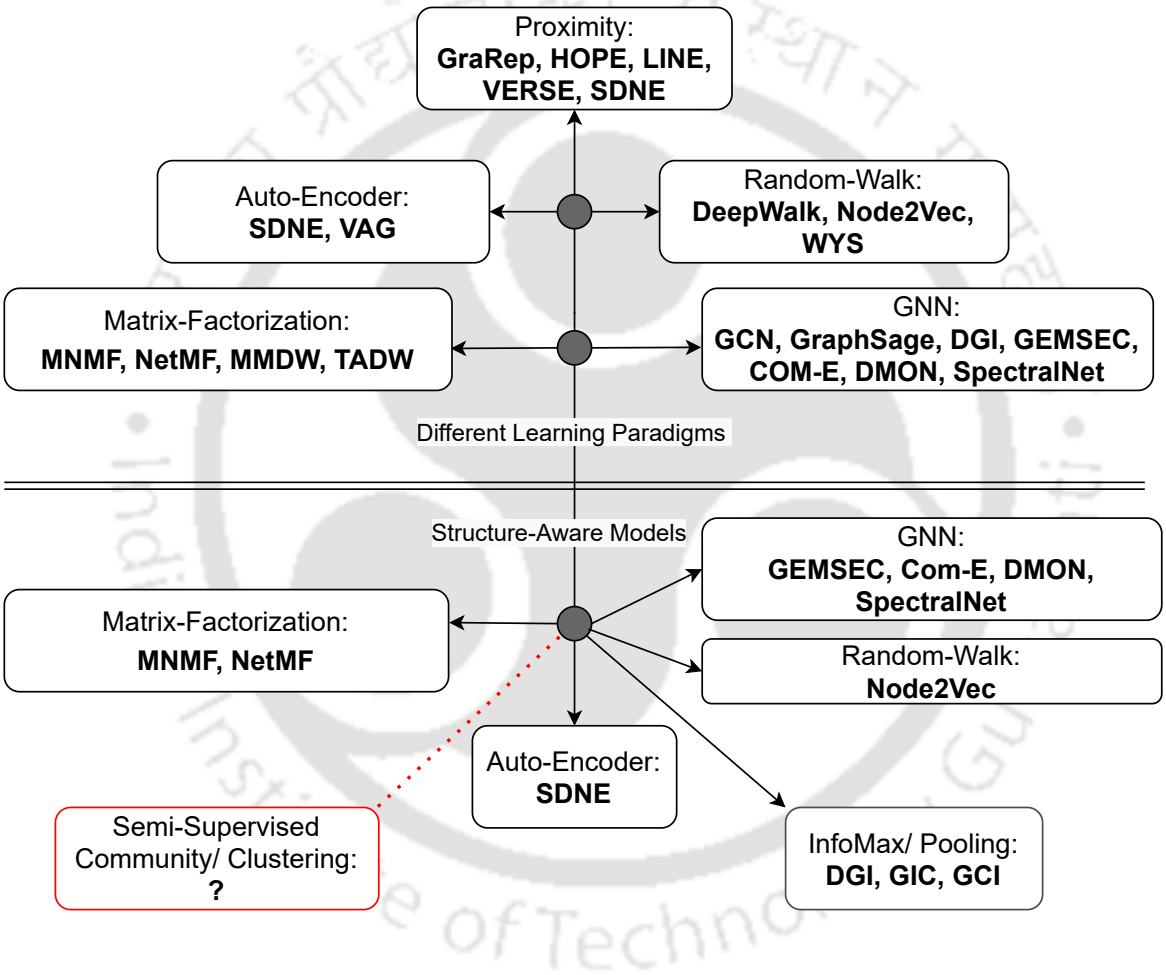


Fig. 3.4 Overview of NRL research on homogeneous graphs

Network Representation Learning (NRL), aka Graph Embedding techniques, has become popular for learning representations for different graph components such as nodes, edges, subgraphs, and the entire graph. Graph embedding models encode different intrinsic and extrinsic properties of the network as continuous low-dimensional vectors. Network representation learning has been realized by a variety of paradigms[14], [94] such as factorization models,

graph kernels, skip-gram-based models, deep learning models, generative models and hybrid paradigms, etc. Figure 3.4 summarizes representative NRL research in various paradigms on homogeneous graphs.

All these paradigms' primary objective is to preserve the neighborhood structure of nodes. These models may differ in the specificity of the neighborhood structure they intend to encode. The widely used skip-gram-based models and the factorization models define them in terms of neighborhood context and proximity matrix. Skip-gram-based models are conventionally scalable as they approximate the partition function by evaluating fewer negative samples. Whereas, Matrix factorization are exact methods that are more interpretable, especially the spectral variants. Standard factorization models and their implementations are not scalable to large graphs. However, they can scale to large graphs by leveraging optimized linear algebra tools such as top-k SVD, sparse SVD, stochastic decomposition [95], etc. Next, we give an overview of NRL research for a few paradigms of interest and discuss the research gaps to address for learning structure-aware network representations.

3.6.1.1 NRL for learning local representations

Here we discuss some prominent works in each of the above-mentioned paradigms.

Factorization based. Factorization models have been widely used to encode different network contexts and couple them with different constraints. Isomap [96], LE [97], LLE [98] are some of the prominent earliest works in this direction. Isomap [96] estimates node embeddings from a $k-$NN graph by first constructing an all-pair shortest-path proximity matrix and decomposing it to principle eigenvectors. Locally Linear Embedding (LE) [97] posits that a node embedding can be intuitively expressed as a linear combination of its neighborhood-based nodes' embeddings. It solves a constrained optimization problem that learns node embeddings in terms of its neighborhood embeddings and constrains the learning objective to have non-trivial, translation invariant solutions. Laplacian Eigenmap (LE) [98] proposes to learn latent node embeddings by *smoothing* them based on the graph Laplacian; that is, embeddings of two nodes are brought closer if their connection weights are high in the underlying similarity network. Graph Factorization (GF) [99] is a simple distributed framework that is the first-of-its-kind to obtain latent node embeddings via factorizing the graph adjacency. GraRep [80] factorizes each of the $k-$step transition matrices into SVD-style decompositions and concatenates the node's latent representation obtained from all the factored matrices. Due to explicitly factoring all the $k-$step transition matrices, this method is not scalable. HOPE [8] also uses multiple higher-order node proximity matrices based on

Katz Index, Rooted Page Rank, Common Neighbors, and Adamic-Adar scores to obtain a common node representation. It uses *generalized SVD* [100] to address the scalability issues related to factoring multiple matrices for learning a common latent embedding. TADW [57] proves the relevance of DeepWalk and network proximity-based matrix factorization methods. It further proposes an NMF framework to jointly model the nodes' textual content alongside the underlying graph's local topologies. DMF [84] improves over the objective of TADW and incorporates the label information. MaxMargin Deep Walk (MMDW) [58], jointly solves the factorization of a Positive Pointwise Mutual Information (PPMI) matrix extracted from k-step random walks and maximizes the margin between support-vectors and classification boundaries by incorporating a hinge loss for label prediction. NetMF [20] is a recent work that unifies the objectives of Deepwalk, Line, PTE, and Node2Vec. Precisely, it analyzes all the network proximity matrices that are implicitly factored in the before-mentioned methods. Further, NetMF comes up with two novel proximity matrix variants that unify the learning objectives of the said methods. NetMF employs a low-rank approximation of the proposed matrices using SVD to obtain node embeddings. Empirically NetMF outperforms the rest of the competing methods by significant margins.

Random walk based. Random walk-based models employ repeated truncated random walks to generate a training set of vertex sequences. It then uses a sliding window-based strategy to obtain context nodes for each node to optimize the skipgram-based learning objective either via negative samples or via hierarchical softmax. The skipgram-based seminal work, Deepwalk [16] and its variants use k-step truncated random walks to define a $k-$th order neighborhood. The probability of (vertex-context) pairs within a context window is then optimized to learn node embeddings in an unsupervised way. Node2Vec [17] is the first to use an informed random walk sampling on nodes based on pre-defined parameters that trade-off between breadth-wise and depth-wise graph explorations. DDRW [101] uses label information of train nodes to minimize the node classification loss based on latent node embeddings learned from optimizing skip-gram objectives on the generated random walks. Tri-DNR [102] incorporates three kinds of information — inter-node relations, node-attribute relations, and label-attribute aka word-sequences correspondence with labels using random walks and explicitly models them. LDE [103] is a negative sampling-based skip-gram optimization framework that jointly models node-node, node-attribute, and node-label correspondence. Both Tri-DNR and LDE are document classification models that consider the underlying network structure of the linked documents. SemiNE [104] proposes an order-sensitive skip-gram architecture to incorporate the ordering information of contextual nodes relative to a target node. PPNE [105] uses joint objectives based on the skip-gram algorithm and pairwise inequality constraints to learn local-context aware and node property-aware

node embeddings. Watch Your Step (WYS) [106] solves the sensitivity issue of random-walk-based models using an attentive mechanism. It proposes learning the hyper-parameters for random walks, such as the number of walks and length of walks, by introducing trainable parameters and back-propagation-based learning. The proposed attention mechanism guides the random walk to focus on short or long-term dependencies pertinent to the input graph.

Deep learning based. A number of Deep Learning (DL) based models have been proposed to learn node embeddings. DNGR [107] uses a random surfing model to capture the contexts of each node in the graph. It further employs a *stacked denoising autoencoder* to effectively decompose, reduce noises and reconstruct the PPMI-based network proximity matrix generated from random surfing of the graph. SDNE [32] preserves the first-order and second-order network proximities in latent dimension via a deep autoencoder that penalizes the reconstruction error of the non-zero elements in the input matrix. Planetoid [22] is a Deep Learning framework that jointly predicts context nodes and labels for a target node. It uses a biased node sampling strategy to generate neighbor-aware and label-aware node contexts, and, predicts the probability of observing them given the target node's latent features and embeddings. It proposes both transductive and inductive formulations for the node classification task. Learning representations from an input matrix of large, sparse graphs can be computationally expensive. Graph Convolutional Neural Networks (GCN) [18] solves this problem by defining spectral convolutions on graphs. Several recent papers [108–110, 23] have proposed methods using convolution on graphs to obtain semi-supervised node embeddings. The approaches either use spectral or spatial filters for convolution operation. Spatial filters operate directly on the original graph and adjacency matrix, whereas, spectral filters operate on the spectrum of graph Laplacian. Towards designing spectral convolution on the graph, Research efforts are also made to investigate finding the importance of the neighboring nodes and the optimal way to aggregate the node embeddings from the neighborhood to learn local context-aware node representations.

Miscelleneous. LINE [111] optimizes an objective function to preserve the first and second-order based network proximity using a KL-divergence-based loss. It proposes to use negative sampling to approximate the optimization for large graphs. VERSE [112] minimizes the KL-divergence-based loss from the similarity distribution in the graph space to the similarity distribution in the latent embedding space. This versatile framework uses various similarity measures such as Personalized PageRank, SimRank, and adjacency matrix-based similarity to represent the similarity distribution in the graph space. Since the objective function is expensive to compute for large graphs, it is approximated by Noise Contrastive Estimation [77]. Besides SDNE, Variational Graph Auto-Encoder (VGAE) [113] is another

autoencoder-based model to preserve the reconstruction loss of the input adjacency matrix. The model uses a GCN encoder to preserve the surrounding neighborhood context for a node and further uses an inner product-based decoder on the obtained node embeddings to reconstruct the adjacency.

NRL with semi-supervised learning objectives. Semi-supervised models can learn discriminative node embeddings that can provide superior node classification results. Discriminative Deep Random Walk (DDRW) [101] jointly learns the topological structures in the graph via random walks and optimizes for the node classification objective to obtain more discriminative node embeddings. Tri-DNR [102] maximizes the probability of observing attributes of a node given the labels of the train nodes. Thus, the node embeddings are influenced by node contexts, attributes as well as supervision knowledge in the graph in a coupled SSL framework. Unlike Tri-DNR, which associates word sequences of a document with label information, DMF [84] reuses TADW's objective to learn text-enriched node embeddings. It then learns a linear classifier to minimize the label prediction loss of the train nodes. LDE [103] conditions on learning document representation itself given the class label of that document. MMDW [58] is a semi-supervised framework that learns a max-margin classifier and employs a biased-gradient-based learning objective to incorporate label information of the train nodes for enriching the node embeddings. SemiNE [104] incorporates node ordering information in the context window to maximize the average probability of observing a target node given the context. It then uses the order-informed contextual node embeddings and projects them using an MLP to optimize node classification objectives for the train nodes. GENE [114] is a semi-supervised method that uses node label information to learn logical groupings of the nodes. It optimizes the probability of observing a vertex given that node's contextual vertex sequence and group labels. However, GENE does not directly optimize for node classification objectives. Planetoid [22] is a recent and notable SSL method that enforces nodes with similar neighbors and labels to be closer in the latent space. It optimizes a cross-entropy loss for label prediction of the train nodes.

NRL on attributed graphs. There are numerous works on learning unsupervised node embeddings for attributed graphs. However, only a few works jointly learn node embedding with side information such as attribute, label, and group information. Planetoid [22] additionally enforces nodes to predict other nodes with similar labels. Planetoid also has a variant for attributed graphs. TADW [57] decomposes the network proximity matrix and learns a low-rank approximation of the input matrix in terms of the latent interaction among two learnable product matrices and the textual attributes of the nodes. DMF [84] incorporates the vertex attributes alongside labels and network proximity in a joint matrix factorization-based

framework. Tri-DNR [102] is a document classification model that uses a random walk exploration strategy to traverse a graph comprised of nodes representing documents and words associated with them. It maximizes the probability of word sequences pertaining to a document node given an observed document class. LDE [103] is also a document classification model that incorporates word attributes for each document via maximizing the probability of observing a word given a context word and the target document. PPNE [105] uses several inequality constraints on the node-to-node attribute correlation matrix based on top$-k$ similar and dissimilar items for each node to incorporate node attributes in the learned embeddings.

3.6.1.2 NRL for learning global representations

Among the skip-gram-based methods, Node2Vec [17] claims to capture higher-order graph structures by balancing the random-walk-based graph exploration in a breath-wise and depth-wise manner. However, learning from the sequences based on the skip-gram objective can not really guarantee capturing a higher-order neighborhood that is non-sequential. NetMF [20] claims the same since it unifies the learning objectives of LINE, PTE, Deepwalk, and Node2Vec. Similarly, VERSE [112] claims to capture communities, roles, and structural equivalence by explicitly learning from various similarity metrics. SDNE [32] claims to capture global graph structures by preserving the second-order neighborhood contexts. However, the discussed methods can not guarantee learning global graph structures since they lack explicit graph-structure modeling objectives.

Recently, a few works have learned enriched node embeddings that explicitly preserve network community information or any higher-order graph structures. MNMF [29] is an NMF-based model that learns node embedding by factorizing the proximity matrix and predicts community assignments for these nodes from the embeddings. The community assignments are learned jointly, maximizing the modularity of the graph. ComE [30], and GEMSEC [35] are two other community-preserving models that learn node embeddings by skip-gram model. GEMSEC is a $K-$means-based adaption for learning node embeddings that jointly learn cluster centers. GEMSEC learns cluster embeddings along with node embeddings and minimizes the distance between the node's embedding and the nearest cluster mean. ComE, along with minimizing the context prediction loss of skip-gram, also maximizes the log-likelihood of generating node embeddings from multiple GMMs. GENE [114] is a random walk-based method that learns the logical grouping of the nodes based on similar neighborhoods and labels. It learns explicit group vectors that are lable-

informed. And, it uses different node sampling strategies to capture within-group and across-group node associations. However, its implementation is not available publicly. HARP [92] learns hierarchical node representations by coarsening the input graph at various levels and recovering the original graph from the coarsened latent representations.

InfoMax-based learning models encode global graph context by maximizing the average Mutual Information (MI) between learned local node representations with a common shared graph summary. DGI [2] is a representative method that computes a global graph summary by naively aggregating all the local node contextual embeddings. Next, the MI is between the (local-global) node representations are maximized. This way, DGI encodes significant noisy global context into the node embeddings. We will discuss more on this in our next Chapter. GraphInfoClust (GIC) [33] and Graph Community InfoMax (GCI) [31] solve this issue of encoding trivial global graph contexts. They employ unsupervised node clustering strategies to generate a personalized global graph context to enrich the local node embeddings via the InfoMax principle.

SAE [109] uses stacked autoencoder-based deep neural networks to learn node embeddings and cluster them using $K-$means algorithm [63] to obtain clusterable representations. The authors demonstrate that SAE-based node embedding learning has theoretical relevance with *spectral clustering* [66]. A few works have explored in the direction of discovering graph communities using Graph Neural Networks DMON [34], SpectralPool [115] are representative works along this direction. DMON proposes *Deep Modularity Networks* – an unsupervised pooling method to cluster the input graph using modularity-maximization [61] based graph partitioning algorithm for learning a pooled graph representation. Whereas, SpectralPool proposes a differentiable *MinCutPool* layer to coarsen the underlying graph using spectral clustering-based objective simultaneously.

Nonetheless, the above-discussed global NRL methods are all unsupervised in nature.

3.6.2 Research Gaps

Since our proposed method is a matrix factorization-based architecture, we only consider representative global NRL methods belonging to the MF paradigm and a few models whose learning objectives are closely similar to ours.

If we are to summarize *the research gaps that exist in learning structure-aware network representations* on homogeneous graphs, the following points give us important glimpses,

- Random-walks, GNNs, auto-encoders and proximity-based methods are limited to only capturing k-hop local contexts for nodes.
- None of the SSL methods incorporate all important priors from the classic SSL Assumptions, especially the cluster assumption.
- Structure-Aware models are all unsupervised in nature – limited by link density and ground-truth network connectivity patterns.

3.7 Proposed Framework: Unified Semi-Supervised Non-Negative Matrix Factorization (USS-NMF)

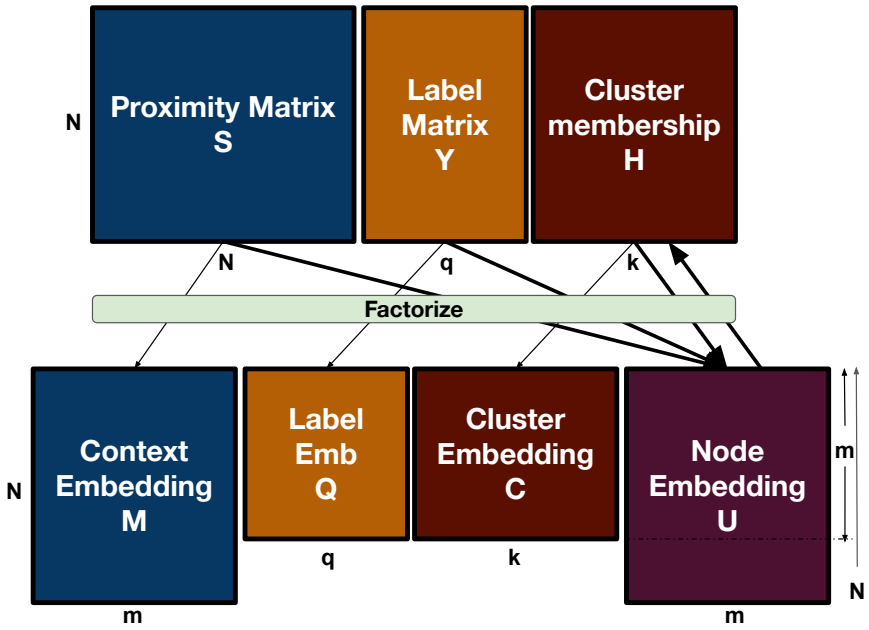


Fig. 3.5 Unified Semi-Supervised Non-Negative Matrix Factorization (USS-NMF) N: number of nodes in graph $|V|$, m: size of low-dimension embedding space, q: number of classes, k:number of clusters. Please refer to Table 2.1 for more details.

In our first contribution, we formulate the clusterability assumption of SSL as a cluster invariance property. We propose a semi-supervised node embedding framework, USS-NMF, that provides a means to enforce this property to learn high-quality clusterable representations of nodes. The proposed framework also incorporates other essential learning principles of SSL. Figure 3.5 illustrates the USS-NMF framework.

3.7.1 Unified Semi-Supervised NMF (USS-NMF)

We incrementally build the proposed framework here.

3.7.1.1 Encoding local invariance aka network structure

This is a fundamental component of any network embedding model. It allows for learning locally invariant node representations, i.e., connected nodes will have similar representations. We obtain locally invariant representations by factorizing a proximity matrix that encodes the similarity between the nodes. Herein, we consider the Pointwise Mutual Information matrix, S, defined as the average transition probability in a window size of (t), in an identical setup [58]. We consider $t = 2$, i.e., taking only the first order and second order average transition probability between a pair of nodes into account [Refer to Section 2.1.3 for details]. Since S is positive semi-definite as the connection weights are non-negative, unlike the formulation in [58, 28], we factorize the proximity matrix S [Refer to Section 2.1.2 for details on NMF Algorithms] into two non-negative basis matrices of dimension m — the node representation matrix $U \in \mathbb{R}^{m \times N}$ and the context representation matrix $M \in \mathbb{R}^{m \times N}$ as with the objective (loss) function below,

$$\mathcal{O}_{network} = \min_{M,U} \|S - U^T M\|^2 : M \geq 0, U \geq 0 \tag{3.1}$$

3.7.1.2 Encoding supervision knowledge

In order to learn semi-supervised representations, we jointly factorize the label matrix, Y, along with S. We define the label matrix factorization term in Eqn 3.2. Where $W \in \mathbb{R}^{q \times N}$ is a weight penalty matrix that zeroes out all the label information of test instances. Specifically, W_i^T is equal to $\vec{0}_q$ vector if the corresponding Y_i^T is unknown and $\vec{1}_q$ vector otherwise. $\odot$ is Hadamard or element-wise multiplication. Thus, element-wise multiplication of the participating matrices in the original label matrix factorization objective $(Y - QU)$ with the penalty-matrix W will result in zero entries in all columns pertaining to the test node instances. This way, we only incorporate the train-label matrix factorization into our objective function. $Q \in \mathbb{R}^{q \times m}$ is the label basis matrix. $U \in \mathbb{R}^{m \times N}$ is the common node embedding matrix. The supervision component is defined as follows,

$$\mathcal{O}_{label} = \min_{Q,U} \|W \odot (Y - QU)\|^2 : Q \geq 0, U \geq 0 \tag{3.2}$$

3.7.1.3 Encoding local neighborhood invariance in label space

This component is the classical Laplacian regularizer for label smoothing, $LS(S,QU)$ [Refer to Section 2.1.5]. Label smoothing regularizer constrains the connected nodes to have similar labels, where QU is the predicted/ reconstructed labels as defined in Eqn 3.3. The learning of reconstructed labels QU is supervised by the network proximity matrix S as below.

$$\mathcal{O}_{LS} = LS(S,\hat{Y}) = LS(S,QU) = Tr\{(QU)\Delta(S)(QU)^T\} \tag{3.3}$$

3.7.1.4 Encoding semi-supervised cluster invariance

Here, we define the key component that allows for learning cluster invariant node representations. Unlike models that enforce Laplacian regularization on the embedding space or label space, we enforce constraints on the abstract space of clusters. Note that the clusters are not provided apriori. The proposed model is made to learn clusters such that nodes with the same/ similar labels are invariant to cluster assignments. This allows the model to learn representations with high label smoothness within the groups. This component primarily comprises two sub-components:

- **Learning cluster assignment via orthogonality constraint:** Let, $H \in \mathbb{R}^{k\times N}$ represents the cluster membership indicator matrix defined for k number of clusters. We obtain H by projecting node embeddings, U on cluster basis C, as $H = CU$. We can restrict the clusters to have different node assignments and have (more or) less overlap with each other via block diagonal constraints. Here, we resort to blocks of size one by enforcing the orthogonal constraints, which encourages every cluster to be different from each other, i.e., we enforce $HH^T = I_k$ similar to [29]. We can obtain cluster assignments as in Eqn 3.4 where β and ζ are Lagrange multipliers.

$$\mathcal{O}_1 = \min_{H,C,U\geq 0} \beta\|H - CU\|^2 + \zeta\|HH^T - I\|^2 \tag{3.4}$$

- **Encoding global context with cluster invariance:** We enforce this constraint by applying a Laplacian regularization on H with label similarity-based proximity matrix E. We define the label similarity network over training data of G as $E = (W \odot Y)^T(W \odot Y) \in \mathbb{R}^{N\times N}$, where $\Delta(E) = D(E) - E$ is the unnormalized Laplacian operator on E. With $\Delta(E)$, we define the cluster smoothing Laplacian regularizer, $LS(E,H)$. The label-based similarity matrix introduces new edges between nodes of similar labels, which may be far away or not even connected in the original network, S. This allows the clusters to enforce a global context.

$$\mathcal{O}_2 = LS(E,H) = Tr((H)\Delta(E)(H)^T) \tag{3.5}$$

Putting it altogether: $\mathcal{O}_{group} = \mathcal{O}_1 + \phi\mathcal{O}_2$, where ϕ is a multiplier. Cluster invariant representations that enforce similar cluster assignments to nodes with the same labels are enabled with this constraint. There is circular enforcement between H and U, i.e., U learns from H, H implicitly learns from U. H explicitly learns from the cluster regularization term and non-overlapping/ orthogonality constraint. Therefore, H pushes two nodes with the same labels and similar neighborhood structures together into the same cluster, thereby leading towards having similar embeddings.

3.7.1.5 USS-NMF Model

In USS-NMF, the node representations, U are learned by jointly factorizing the local neighborhood proximity matrix S, label matrix Y, inferred cluster assignment matrix H and are indirectly influenced by smoothing on the label and cluster space. The joint objective is given below,

$$\mathcal{O} = \alpha\mathcal{O}_{network} + \theta\mathcal{O}_{label} + \delta\mathcal{O}_{LS} + \mathcal{O}_{group} + \lambda L2_{reg} \tag{3.6}$$

where $\alpha, \beta, \theta, \zeta, \phi, \delta, \lambda$ are hyper-parameters controlling the importance of respective terms in the Equation 3.7–3.11. Since the joint non-negative constrained objective is not convex, we can iteratively solve the convex sub-problem for each of the factors $M, U, C, Q,$ & H individually. Next, we derive multiplicative update rules for these factors following the methodologies described in [53].

3.7.1.6 Derivation of multiplicative update rules

Let $\psi_1, \psi_2, \psi_3, \psi_4, \psi_5$ be the Lagrange multipliers for the non-negative constraints on factor matrices M, U, C, Q, H respectively. USS-NMF's loss function in Eqn 3.6 can be expanded and rewritten with Lagrange multipliers as follows:

$$\begin{aligned}
\mathcal{L} = {} & \alpha Tr[SS^T - 2SM^TU + U^TMM^TU] \\
& + \beta Tr[HH^T - 2HU^TC^T + CUU^TC^T] \\
& + \theta Tr[W \odot \{YY^T - 2YU^TQ^T + QUU^TQ^T\}] \\
& + \zeta Tr[HH^THH^T - 2HH^T + I] + \phi Tr\{H\Delta(E)H^T\} \\
& + \delta Tr\{(QU)\Delta(S)(QU)^T\} \\
& + \lambda Tr(MM^T + QQ^T + CC^T + UU^T + HH^T) \\
& + Tr[\psi_1M^T + \psi_2U^T + \psi_3C^T + \psi_4Q^T + \psi_5H^T]
\end{aligned}$$

The KKT conditions for the non-negativity constraints are: $\psi_{1ab}m_{ab} = 0$, $\psi_{2ab}u_{ab} = 0$, $\psi_{3ca}c_{ca} = 0$, $\psi_{4da}q_{da} = 0$ and $\psi_{5cb}h_{cb} = 0$, where $M = [m_{ab}], U = [u_{ab}], C = [c_{ca}], Q = [q_{da}], H = [h_{cb}]$ such that, a, b, c, d are the respective row & column indices. The partial derivatives for each of the factors are:

$$\frac{\partial \mathcal{L}}{\partial M} = -2\alpha US + 2\alpha UU^T M + 2\lambda M + \psi_1$$
$$\frac{\partial \mathcal{L}}{\partial C} = -2\beta HU^T + 2\beta CUU^T + 2\lambda C + \psi_3$$
$$\frac{\partial \mathcal{L}}{\partial Q} = -2\theta(W \odot Y)U^T + 2\theta(W \odot QU)U^T - 2\delta(QUS)U^T + 2\delta(QUD(S))U^T + 2\lambda Q + \psi_4$$
$$\frac{\partial \mathcal{L}}{\partial U} = -2\alpha MS^T - 2\beta C^T H - 2\theta Q^T(W \odot Y) - 2\delta(Q^T QU)S + 2\alpha MM^T U + 2\beta C^T CU + 2\theta Q^T(W \odot QU) + 2\delta(Q^T QU)D(S) + 2\lambda U + \psi_2$$
$$\frac{\partial \mathcal{L}}{\partial H} = 2\beta H - 2\beta CU + 4\zeta HH^T H - 4\zeta H + 2\lambda H + 2\phi HD(E) - 2\phi HE + \psi_5$$

From the partial derivatives, we can derive the following multiplicative update rules for the factor matrices:

$$M = M \odot \left(\frac{\alpha US}{\alpha UU^T M + \lambda M} \right) \tag{3.7}$$
$$C = C \odot \left(\frac{\beta HU^T}{\beta CUU^T + \lambda C} \right) \tag{3.8}$$
$$Q = Q \odot \left(\frac{\theta(W \odot Y)U^T + \delta(QUS)U^T}{\theta(W \odot QU)U^T + \delta(QUD(S))U^T + \lambda Q} \right) \tag{3.9}$$
$$H = H \odot \left(\frac{\beta CU + 2\zeta H + \phi HE}{\beta H + 2\zeta HH^T H + \phi HD(E) + \lambda H} \right)^{1/4} \tag{3.10}$$
$$U = U \odot \left(\frac{\alpha MS^T + \beta C^T H + \theta Q^T(W \odot Y) + \delta(Q^T QU)S}{\alpha MM^T U + \beta C^T CU + \theta Q^T(W \odot QU) + \delta(Q^T QU)D(S) + \lambda U} \right) \tag{3.11}$$

3.8 Evaluation Methodology

This section evaluates and analyzes the proposed model, USS-NMF, against the State-of-The-Art (SoTA) models and numerous proposed adaptions of existing and new semi-supervised models for node classification. Additionally, we also demonstrate the superiority of the discriminative capacity of the learned embeddings by evaluating learned clusters against the ground truth with Normalized Mutual Information (NMI) and t-SNE plots.

In Section 3.9.3, we provide ablation studies to analyze how the inclusion of the cluster information and the supervision information, individually as well as collectively help USS-NMF to achieve better performance. We provide convergence plots (Figure 3.8) as empirical evidence for the correctness of USS-NMF. We also provide an analysis of the sensitivity of different parameters in Section 3.9.7 for our proposed method. In Section 3.9.4, we provide an extensive study of numerous variants of SSL to understand the usefulness of various Laplacian smoothing terms. We also report results for varying label sparsity (Section 3.9.6) and balanced train/test sampling strategy (Section 3.9.5). In all these experiments, USS-NMF comes out as the clear winner achieving SoTA results.

We provide the implementation details in Section 3.8.3 for reproducibility. With our extensive experimental analysis of the sensitivity of different parameters of the models, we found that despite the large number of hyper-parameters, the effective hyper-parameter search space of our model is small (Table 3.3) compared to the full space and is comparable to other SoTA approaches (Table 3.2.)

3.8.1 Datasets

Description of the datasets used is provided below with summary statistics in Table 3.1. **Web**

Dataset	$\|V\|$	$\|E\|$	$\|Y\|$	ML	D | W	Type
Washington	230	596	5	F	F | T	Web
Wisconsin	265	724	5	F	F | T	Web
Texas	186	464	5	F	F | T	Web
Cornell	195	478	5	F	F | T	Web
Wiki	2,405	17,981	19	F	T | F	Bibliography
Cora	2,708	5,278	7	F	T | F	Bibliography
Citeseer	3,312	4,732	6	F	T | F	Bibliography
DBLP	18,721	1,22,245	3	F	F | T	Bibliography
Microsoft	44,034	1,95,361	15	F	F | F	Bibliography
Pubmed	19,717	44,338	3	F	T | F	Bibliography
PPI	3,890	76,584	50	T	F | F	Biological
Blogcatalog	10,312	3,33,983	39	T	F | F	Social
Wikipedia	4,777	92,517	40	T	F | T	Language

Table 3.1 Dataset stats. *ML*: Multi-label, D:Directed, W:Weighted

networks. WebKB [116] consists of four small networks collected from four universities —

Washington, Wisconsin, Texas and Cornell. The networks are a collection of web pages as nodes and hyperlinks as edges. The task here is to predict the type of webpage.
Bibliography networks. Cora [82], Citeseer [117], Wiki [82] and Pubmed [118] are citation networks. Whereas DBLP [119] is a co-author network and Microsoft [112] is a co-citation network. In all these datasets, the task is to predict the research area of the node (paper/author).
Biological network. PPI [120] is a protein-protein interaction dataset where the task is to predict the functional properties of proteins.
Social network. Blogcatalog [121] is a social network dataset with nodes as bloggers, and an edge exists between them if they are friends. The task here is to categorize their blogs.
Language network. Wikipedia [122] is a co-occurrence network of words collected from Wikipedia dump. The task is to predict the POS tags of words in the corpus.

3.8.2 Baselines

Details on Baselines are in Section [3.8.3]. The nomenclature for NMF based models will indicate that they are NMF along with matrices being factored. For example, $NMF : S+Y$ indicates that it factorizes the proximity matrix, S, and label matrix, Y.

The state of the art methods for semi-supervised classification on non-attributed graphs are limited to MaxMargin-DeepWalk (MMDW) [58] and Planetoid [22]. Besides, we also introduce two enhanced SSL alternates of the unsupervised community preserving embedding model (MNMF) [29] and the Non-negative Matrix Factorization version of DeepWalk (NMF:S) [58] viz: (MNMF+Y) and (NMF:S+Y) respectively, where the original objectives are jointly factorized with label matrix (Y) as in Eqn 3.2. For reference sake, we have also included results of the standard random walk sampling-based DeepWalk (DW) [16]. Apart from MNMF [29], we have also included few recent unsupervised community / cluster enhanced node representation learning models, viz, ComE [30] and GEMSEC [35] as baselines.

The Planetoid model was defined for multi-class classification with train-set splits drawn using balanced sampling (which can be unrealistic). We empirically observed that Planetoid-G (Planetoid for non-attributed graphs) performs poorly in comparison to other baselines where the labeled set is randomly drawn (not reported here), and it is not directly extensible to the multi-label dataset (PPI, Blogcatalog, Wikipedia). Hence, we do not report results for Planetoid here. Albeit, to be fair, we provide a comparison against Planetoid-G in their

original balanced train/test split in Section 3.9.5 and also evaluate a proposed NMF variant of Planetoid-G in Section 3.9.4. In both these studies, USS-NMF is the better model.

3.8.3 Experiment Setup

The experiment is set up with 50% labeled data. The results are averaged over 5 different train-test splits. We also extensively searched for optimal hyperparameter values for all the models using 20% of training data as a validation set. More details on the hyper-parameter search space are provided next.

Metrics. We report classification performance with Micro-F1 scores. Additionally, we define two aggregate metrics viz: *Rank* and *Penalty* [123] to measure the overall performance of models across datasets. *Rank* of a model is defined as the average position of the model when their results are ordered in descending order in each dataset. *Penalty* of the model is defined as the average difference from the best performing model in each dataset. The lower the Rank and Penalty, the better is the performance. Let, $\mathcal{E}$ and $\mathcal{D}$ be the set of all embedding methods and datasets respectively with e and d being a member from these sets. $\mathcal{R}_{e,d}$ is the rank of a particular embedding method e on a dataset d when all the competing methods are ranked on each dataset based on their micro-f1(%) scores. Similarly, $\mathcal{S}_{e,d}$ be the score achieved by an embedding method e on a dataset d as how much e diverges from the best performing model e^* on d. The formulae for Mean Rank and Penalty are,

$$\mathcal{MR}_e = \frac{\sum_{d \in \mathcal{D}} \mathcal{R}_{e,d}}{|\mathcal{D}|} \tag{3.12}$$

$$\mathcal{MP}_e = \frac{\sum_{d \in \mathcal{D}} max(\{\mathcal{S}_{e',d}\}) - \mathcal{S}_{e,d}}{|\mathcal{D}|} \quad : e' \in \mathcal{E} \tag{3.13}$$

Our model, USS-NMF, is clearly the winner on the whole across all the tasks [Refer to the Performance Analysis in Section 5.9 in Tables 3.4, 3.5 and other Ablation Studies]. However, we resort to these two aggregate scores to measure the consistency of all the models across datasets. We also provide statistical significance with Wilcoxon signed-rank test [124], the established test for comparing two models on multiple datasets.

Classifier. We learn an external Logistic Regression classifier (LR) to make predictions from model's learned node representations. Though we can obtain label predictions internally for the supervised models by reconstructing the label matrix as in Eqn 3.2 by multiplying label and node embeddings QU, we found that training a classifier based on the node embeddings further improves the performance.

Implementation Details. The details of the hyper-parameters for our model and the baselines are provided below. The hyper-parameter search space for different components in all the models experimented here is tabulated in Table 3.2.

Matrix Factorization based methods								Random Walk/ Other methods			
Co-efficients	NMF:S	NMF:S+Y	MMDW	MNMF	MNMF+Y	MF-Plan	NMF:S+Y +LS(S,Y)	Co-efficients	DeepWalk	ComE	Gemsec
Network	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	p	[1.0]	NA	[0.1, 0.3, 0.5, 0.7, 1.0, 3.0, 5.0, 7.0, 10.0]
Label	NA	0.1-10.0	Max-margin loss based biased gradient: e^[-1, -2, -3, -4, -5]	NA	0.1-10.0	0.1-10.0	0.1-10.0	q	NA	Network: 0.1-10.0	Same as p
Cluster Factorization	NA	NA	NA	0.1-10.0	0.1-10.0	NA	NA	Walk-Length	80	NA	80
Cluster Learning	NA	NA	NA	0.1-10.0	0.1-10.0	NA	NA	No of Walks	40, 80	NA	40, 80
Cluster Orthogonality	NA	NA	NA	$[1e+(0,4,8)]$	$[1e+(0,4,8)]$	NA	NA	Learning Rate	NA	[0.001, 0.025, 0.625, 0.1]	Initial LR: [0.01, 0.1] Minimal LR: [0.0001, 0.001]
Graph Laplacian Reg	NA	NA	NA	NA	NA	0.1-10.0	0.1-10.0	Community Learning	NA	0.1-10.0	[0.01, 0.1, 1.0]
L2 Regularization	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	0.1-10.0	L2 Regularization	NA	NA	0.1-10.0
#Clusters	NA	NA	NA	#Labels(-1, +2)	#Labels(-1, +2)	NA	NA	#Clusters	NA	#Labels(-1, +2)	#Labels(-1, +2)
#Experiments	25	125	125	110	130	130	130	#Experiments	2	125	204

Table 3.2 Hyper-parameter range search for Baselines

The range 0.1-10 refers to the set [0.1, 0.5, 1.0, 5.0, 10.0]. We selected 25 values for k as #Labels(-1, +2), i.e., increasing 2 in the upper range and decreasing 1 in the lower range from a dataset's actual no of labels q (inclusive).

Co-efficients	**USS-NMF** (Effective range)		**USS-NMF** (Entire range)
Dataset (small=<1k, large>1k |V|)	Small	Large	All Datasets
Network	1, 5	1, 5, 10	0.1-10 / [0.1, 0.5, 1.0, 5.0, 10.0]
Label	0.1, 1	0.1, 0.5, 1	0.1-10
Cluster Factorization	0.1, 1	0.1, 0.5, 1	0.1-10
Cluster Learning	10	10	0.1-10
Cluster Orthogonality	1e + (0, 4)	1e + 8	1e + (0, 4, 8)
Graph Laplacian Regularization	0.5, 1	0.5, 1	0.1-10
L2 Regularization	1	1	0.1-10
#Clusters	#Labels	#Labels	#Labels(-1, +2)
#Experiments	32 (Full search)	54 (Full search)	150 (Partial seach)

Table 3.3 Hyper-parameter search space for USS-NMF

We selected 25 values for k as #Labels(-1, +2), i.e., increasing 2 in the upper range and decreasing 1 in the lower range from a dataset's actual no of labels q (inclusive). We also report generic range which effectively works for most of the cases. See results in Table 3.10.

Lagrange Multipliers' Range. We first provide the details of the hyper-parameter search for the Lagrange multipliers followed by model-specific details. For all the matrix factorization baselines, we vary the hyper-parameter values (the respective weightage terms for each component in the objective function) in $[0.1, 0.5, 1.0, 5.0, 10.0]$ except for Wikipedia. In Wikipedia, we found that network information is far more important than other supervision knowledge. So we varied network co-efficient in the range of $[10000, 1000, 100, 10]$ with other weights in $[0.001, 0.01, 0.1, 1.0]$. We fixed the embedding dimension as 128 for all datasets except Blogcatalog, for which the dimension is set as 4096.

DeepWalk and MFDW. For original random-walk based DeepWalk we set the window size to 5. We also have MFDW aka NMF:S in Eqn 3.1 — the objective function for Matrix Factorized DeepWalk, as we build our model incrementally on top of it.

Max-Margin DeepWalk (MMDW). In paper MMDW [58], a max-margin loss is incorporated in the objective function of MFDW to learn discriminative representations of vertices. It has one important hyper-parameter alpha-bias (η) that induces max-margin loss-based bias into the random walk.

NMF:S+Y. We build a variant of MMDW which also incorporates supervised information into node embeddings by jointly optimizing Eqn 3.1 and 3.2. It works competitively as compared to MMDW.

Planetoid and NMF:Planetoid. Planetoid [22] learns an embedding space for nodes by jointly enforcing label and neighborhood similarity. It uses random walks to enforce structural similarity. We derive a matrix-factorized version of Planetoid as an alternative baseline. It enforces matrix E, i.e, train-label similarity on the embedding space U, unlike ours as in Eqn 3.5 which enforces label similarity on the cluster space.

$$\mathcal{O}_{\text{NMF:Planetoid}} = \mathcal{O}(NMF : S+Y) + Tr\{U\Delta(E)U^T\} \tag{3.14}$$

MNMF and MNMF+Y. We build one semi-supervised variant of MNMF, viz. MNMF+Y by jointly optimizing its objective function along with Eqn 3.2. Unlike the original MNMF that factorizes a combination of first-order and cosine similarity based second-order node proximity to learn node representations, here, for the sake of fair comparison, we stick to a combination of first-order and second-order transition probability-based proximity matrix as S, following MMDW [58] as we did for all other comparable methods.

USS-NMF. We used the same range of hyper-parameters as stated in Table 3.3 last column, but instead of searching the optimal combination in the entire range (which is cumbersome), we did a partial range search in steps. Step by step, 1) network + label information weights, 2) cluster matrix factorization + cluster learning weights + orthogonality constraint, 3) label smoothing + L2 regularization weights, and finally, 4) the clusters k for each dataset was varied with an increment of 2 in the upper range and with a decrement of 1 in the lower range from its actual number of labels q (inclusive). In the first step, we fixed all other variable values as 1.0, $k = q$. In later steps, we set already searched parameter values to optimal values found in previous steps to vary the other variables under consideration. In Table 3.3 we have given an effective value range for each of the coefficients, applicable for varying sized datasets. We make the following points based on observation,

- Labels and label-similarity-based clusters gave complementary information to support each other for enhanced prediction (evident from their weight combinations in results).
- For small graphs, network information was more beneficial than other information. Also, small graphs tended to be more sensitive to weight combinations with easily imposed cluster orthogonality constraints (see the generic range). But for large graphs, USS-NMF needed more weights to ensure orthogonal clusters were learned. Higher cluster learning weights (in generic range) indicate that indeed cluster information mattered.
- We found optimal results for clusters same as ground truth labels, very different from labels, multiple optimal clusters — which indicate that a variety of semantically meaningful clusters have been learned. For simplicity, in generic search, we set clusters as the number of labels, which gave decent results. Except for Pubmed and Wikipedia, we do not witness any fluctuation in L2 regularization weights.

With these observations at hand, we narrowed down the effective hyper-parameter search space such that the network has weights (>=1 & <=10), labels & clusters don't overpower network weights, stable values for regularization & orthogonality weights, and the number of ground truth labels as clusters. This ranges' effectiveness can be seen immediately from Table 3.10 where we have used the same effective range to derive results for USS-NMF, which still significantly outperforms other semi-supervised methods (full hyper-parameter search). Thus, we can conclude that the effective hyper-parameter space is not huge for USS-NMF and is comparable to other existing methods (see the number of experiments in the last row of Tables 3.3 and 3.2 for reference).

3.9 Performance Analysis

Here we provide empirical insights into the proposed model's performance and how it compares to the rest of the competing models. We conduct experiments for a number of tasks, including node classification, node clustering, and embedding visualization. A number of conducted case studies bring more clarity on the novelty of USS-NMF. We investigate the following research questions —

RQ1. How well does USS-NMF perform in node classification and node clustering tasks against the SoTA unsupervised and semi-supervised NRL models? [SECTION 3.9.1, SECTION 3.9.2]

RQ2. Are the node embedding visualizations obtained from USS-NMF interpretable and discriminative? [SECTION 3.9.2.2]

RQ3. What are the contributions of the labels and clusters in the final objective function as proposed in Eqn 3.6? [SECTION 3.9.3, SECTION 3.9.3.1, SECTION 3.9.3.2]

RQ4. What are the contributions of the semi-supervised Laplacian smoothing (LS term) in the final objective function as proposed in Eqn 3.6? How do various unsupervised and semi-supervised LS variants perform comparatively? How do various LS variants applied on the latent space of embeddings, labels and clusters perform comparatively? [SECTION 3.9.4]

RQ5. How does USS-NMF perform in label sparsity? [SECTION 3.9.6]

RQ6. How does USS-NMF perform using Planetoid [22]'s publicly available balanced (train/ validation/ test) nodes with equal number node instances per class? [SECTION 3.9.5]

In the following sections we address the discussed research questions elucidating the key takeaways.

3.9.1 Node classification

Node classification results are reported in Table 3.4. The bolded entries in a dataset row denote the best score achieved in that dataset, and the underlined entries denote the second-best scores.

3.9.1.1 Unsupervised Models

NMF based neighborhood embedding model, NMF:S performs similar or better than the sampling-based DW as shown in [16] on all but Pubmed, Co-Author, Blogcatalog, and Microsoft dataset. These models, which do not encode supervision information or any clustering information, are the two least performing models after GEMSEC, as depicted by their Ranks. The superiority of NMF:S over DW is visible from the relatively lower Penalty score. This is consistent with findings in the literature that typically, Matrix factorization models are better. NMF:S is not significantly different from DW for $p < 0.05$ on two-tailed Wilcoxon signed-rank tests. Skip-gram-based models additionally have the power of non-linearity, and they optimize cross-entropy loss when compared to the simple linear squared

	Unsupervised Models					Semi-Supervised Models			
50% splits			Cluster/ Community Models				*Enhanced Alternatives*[2]		Proposed
Datasets	DW	NMF:S	MNMF	GEMSEC	ComE	MMDW	NMF:S+Y	MNMF+Y	USS-NMF
Cora	80.148	80.443	82.066	77.063	82.030	83.838	85.535	85.904	**87.380**
Citeseer	57.272	59.747	61.255	56.403	60.712	66.325	68.618	68.618	**70.187**
Wiki	63.009	64.755	65.254	58.969	63.840	67.997	65.152	66.999	**70.906**
Washington	59.130	62.609	65.217	60.913	62.087	61.739	66.957	65.217	**67.826**
Wisconsin	48.120	50.376	49.624	52.030	51.504	50.376	52.632	54.617	**56.391**
Texas	58.511	59.574	58.511	57.021	59.043	57.447	59.574	58.511	**63.830**
Cornell	38.776	51.020	52.041	51.277	49.490	51.020	52.041	54.082	**56.122**
PPI	21.390	22.467	22.486	21.417	22.373	**23.492**	22.167	22.381	23.433
Wikipedia	49.960	55.637	45.634	50.832	53.112	55.516	56.106	45.870	**57.894**
Pubmed	81.460	78.976	79.523	80.620	81.626	81.430	83.083	83.199	**84.009**
Co-Author	35.860	35.823	36.028	35.512	36.183	36.663	36.04	36.855	**37.858**
Blogcatalog	**41.460**	38.074	38.123	35.935	40.275	36.665	38.974	39.736	**41.254**
Microsoft	46.290	46.083	47.718	46.692	47.001	47.789	47.022	48.184	**49.706**
Rank	7.154	6.077	5.308	7.615	5.385	4.769	3.615	3.231	**1.154**
Penalty	6.590	4.729	4.891	6.339	4.445	3.597	2.551	2.838	**0.020**

Table 3.4 Micro-F1 Scores for Node Classification; *Enhanced Alternatives*[2]: Proposed baseline variants for SSL.

error reconstruction used in our NMF models. We believe factorizing with KL-divergence over squared error will improve the performance of factorization models and match DW in datasets where they currently fail.

Cluster enforcing models are the superior unsupervised models. Among them, MNMF outperforms GEMSEC and ComE on 7/13 datasets except for Wisconsin, Wikipedia, Blogcatalog, and Pubmed, where GEMSEC and ComE outperforms MNMF significantly. This is reflected in the aggregate scores. Despite ComE beating MNMF on a lower number of datasets, its rank is almost similar to MNMF as on these datasets, and it outperforms both MNMF and other models significantly. On the other hand, looking at the Penalty score, MNMF and ComE are similar as both beat each other on a comparable number of datasets by $2-3\%$ points. GEMSEC performs poorly among the community models. MNMF is not significantly different from ComE for $p < 0.05$ under two-tailed Wilcoxon signed-rank tests; this is reflected in their Rank and Penalty scores.

3.9.1.2 Semi-Supervised Models

All semi-supervised models obtain better Rank and lower Penalty over unsupervised models. The Semi-Supervised (SS) variants of the unsupervised models are (statistically) better than their unsupervised counterparts on all datasets, i.e., MNMF+Y > MNMF and NMF:S+Y >

NMF:S. Overall, both enhanced alternatives are better than MMDW. While MNMF+Y is mostly competitive, it falls short of ≈ 10 points of the other two models on the Wikipedia dataset. It is apparent that clustering by maximizing modularity jointly with label matrix factorization may be limiting.

USS-NMF outperforms its base model NMF:S+Y on all datasets, which is a clear indicator that learning cluster and label invariant representations are useful. *USS-NMF is ranked first in* $12/13$ *datasets* while being ranked second on PPI. Thus, obtaining an average rank score of 1.54 and the lowest penalty of 0.020. On the datasets where USS-NMF is ranked first, as per the paired t-test, there exists no case where P-VALUE < 0.05 and T-SCORES are positive (i.e., no competing method significantly beats USS-NMF).

3.9.2 Clusterability of Learned Representations

We validate the superior clusterability of the learned node representations quantitatively in Table 3.5 and qualitatively with t-SNE plots in Figure 3.6.

	Unsupervised Models					Semi-Supervised Models			
			Cluster/ Community Models				*Enhanced Alternatives*[2]		Proposed
	DW	NMF:S	GEMSEC	ComE	MNMF	MMDW	NMF:S + Y	MNMF+ Y	USS-NMF
Cora	34.28	34.40	35.83	41.02	39.29	50.31	51.38	53.21	**57.93**
Citeseer	19.04	17.71	21.42	24.42	29.96	32.70	28.94	41.19	**53.32**
Wiki	32.57	28.31	33.86	32.59	45.62	33.82	47.80	48.38	**61.06**
Washington	2.88	9.93	8.98	5.89	19.90	15.78	18.45	33.52	**40.86**
Wisconsin	5.04	6.09	5.46	5.22	11.20	9.27	6.81	17.89	**33.9**
Texas	2.70	2.85	2.35	3.65	9.00	7.99	10.61	15.14	**35.56**
Cornell	3.53	4.16	3.91	3.35	3.99	**8.76**	4.49	4.14	5.88
PPI	9.44	7.91	9.63	9.07	8.77	8.44	8.26	9.19	**11.48**
Wikipedia	21.57	22.49	31.2	36.94	33.11	25.54	29.94	35.54	**37.98**
Pubmed	20.15	17.28	19.93	29.83	29.77	28.56	29.39	37.32	**38.47**
Co-Author	24.68	24.17	28.02	29.06	31.2	22.14	26.63	30.89	**36.37**
Blogcatalog	3.71	3.06	6.93	5.01	5.94	6.07	6.18	8.61	**14.31**
Microsoft	21.72	19.53	24.01	25.55	23.91	21.42	28.76	29.92	**33.00**
Rank	7.69	7.77	4.46	5.85	5.77	5.38	4.46	2.54	**1.08**
Penalty	20.55	20.82	13.91	17.74	16.87	15.21	13.14	7.97	**0.22**

Table 3.5 (O)NMI Scores for Node Clustering

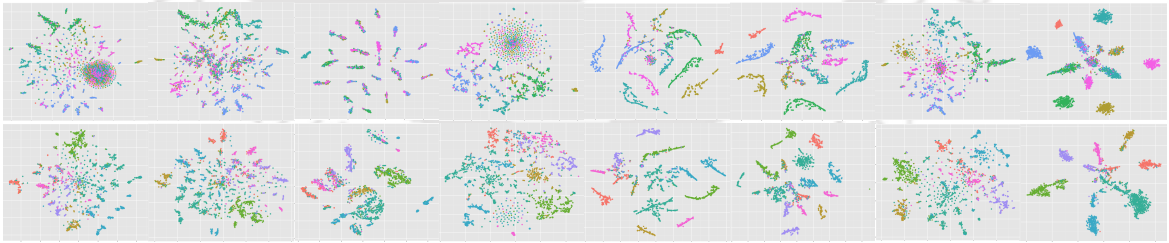
3.9.2.1 Node Clustering

In Table 3.5, we report the average cluster quality of learned embeddings obtained from k-means and Fuzzy c-means algorithms for multi-class and multi-label datasets, respectively.

We take the embeddings learned for node classification and report averaged NMI results over the five different data splits, where the result for each run is an average of 5 other run with different cluster assignment initializations. We used different initialization techniques for initializing the mean of clusters (k-means++, PCA based, random). The optimal number of clusters was obtained using gap statistics [125]. We evaluate the obtained clusters against ground truth classes and report the NMI scores. We used Overlapping NMI (ONMI) [126, 127] for overlapping clusters to evaluate the multi-label datasets.

From Table 3.5, it is evident that USS-NMF performs well in node clustering tasks. It is the best performing model on 12/13 datasets where it beats the second-best model by a large margin of $1\% - 20\%$, and it is the second-best performing model on Cornell dataset where it is falling short of the best by 2.8%. Both NMF:S+Y and MNMF+Y outperform their unsupervised counterparts, NMF:S and MNMF. The supervised MMDW outperforms the simple unsupervised NMF:S in all but Co-author. All the semi-supervised NMF models outperform the unsupervised models except for MMDW, which is outperformed by GEMSEC by the aggregate score metrics. MMDW's max-margin embeddings are seemingly not well clusterable as its counterpart SSL methods. The consistent superior performance of USS-NMF suggests that the label similarity-based clusterability criteria can learn informative node representations beyond the graph structure. This is supported by the t-SNE plots, too, especially that of USS-NMF, which provides superior visualizations of well separable homophilous clusters.

3.9.2.2 t-SNE Visualization



(a) MFDW (b) MNMF (c) GEMSEC (d) COM-E (e) MFDW+Y (f) MNMF+Y (g) MMDW (h) USSNMF

Fig. 3.6 t-SNE Visualization of Embeddings on Citeseer (above) Cora (below) Dataset for Unsupervised and Semi-Supervised Methods. Color-codes denote class-membership of ndoes.

Further, we also present details of t-SNE experiment on the learned node embeddings for Citeseer and Cora dataset in Figures 3.1, 3.6. t-SNE plots are especially well-suited for the visualization of high dimensional data. As the t-SNE algorithm [128] scales quadratically in terms of the number of nodes N, we first reduced the dimension of learned node embeddings to 64, retaining as much information as possible using Principal Component Analysis (PCA). Next, we fed this reduced data to the t-SNE algorithm. In t-SNE, the perplexity term controls the number of neighbors for each sample to take into consideration to preserve the local structure in the reduced dimension space. We experimented with perplexity in the range of $10-100$, increasing by a step size of 10 and found that it did not affect the visualizations much from 30 onwards. It can be seen that our proposed model obtains visually superior clusters compared to all other unsupervised, as well as semi-supervised methods due to the semi-supervised clusterable node representations learned.

3.9.3 Ablation Study

Here, we drill down the components of USS-NMF to analyze the importance of utilizing information about labels and clusters.

3.9.3.1 Importance of Label Information

Since the smoothening on the cluster space is based on the label similarity graph, it is necessary to verify — 1) whether we need to factorize the label matrix and 2) whether we should smooth the predicted label space locally beside this. We report results for this study in Table 3.6, where we subtract the before-mentioned two components from the original model USS-NMF. The model in Column: 3 does not have the label smoothing term, and the model in Column: 4 does not have the label matrix factorization as well as the label-smoothing term. Note that we can not smooth on the label space based on the local neighborhood structure without predicting the labels.

Label smoothing term provides an improvement of up to 3.74 points in Wiki, 2.13 points in Texas, up to 1.5 points for Cora and Wisconsin, and no improvement in Washington and Cornell. On seven datasets, it offers an improvement of $1\%-4\%$, and on three of the other datasets, it provides an increase between $0.3\%-1.0\%$. Thus, the label smoothing term is quite helpful. Moreover, it can be seen from Column: 4, that removing the label matrix factorization, Y has a significant impact on USS-NMF. It results in an average drop in the performance of 2.24% with label matrix factorization solely contributing an average

	USS-NMF	- LS(S, $\hat{Y}$)	- (Y + LS(S, $\hat{Y}$))
Cora	**87.306**	85.883	80.738
Citeseer	**70.187**	69.825	63.109
Wiki	**70.906**	67.168	65.339
Washington	**67.826**	**67.826**	65.217
Wisconsin	**56.391**	54.887	51.281
Texas	**63.830**	61.703	59.574
Cornell	**56.122**	**56.122**	52.020
PPI	**23.433**	23.130	22.346
Wikipedia	**57.894**	56.254	55.578
Pubmed	**84.009**	83.981	78.945
Co-Author	**37.858**	36.972	36.198
Blogcatalog	**41.254**	39.736	38.135
Microsoft	**49.706**	48.669	47.153

Table 3.6 Importance of Label Information.

USS-NMF: Proposed method, LS(S, $\hat{Y}$): Local smoothing on the predicted label space, Y: Inclusion of label information through factorization. In 3rd and 4th columns the minus sign indicates the respective terms are removed from the objective of USS-NMF.

drop of 1.39%. The performance drop is a maximum of 7.08% on Citeseer, followed by 6.6% in Cora. On $11/13$ datasets, it offers an improvement of $2\% - 9\%$, and on the rest $2/13$ of the datasets, it provides an increment in performance between $1.0\% - 1.5\%$. In comparison, the average reduction in performance when we remove the label smoothing term is approximately 0.85%(ST-DEV:0.8%). This indicates that it is necessary to have both the terms in our objective.

3.9.3.2 Importance of Cluster Information

Here, we try to understand the importance of the proposed cluster smoothing term by removing the cluster smoothing term first and then additionally removing the cluster factorization term too. This corresponds to Column: 3 and Column: 4 in Table 3.7 respectively. Removing the cluster smoothing term results in an average drop of 1.82% across datasets, whereas completely removing all cluster-related components results in an average drop of 2.2% on performance. Cluster smoothing term provides an additional improvement between $2\% - 4\%$ on $5/13$ datasets and $1\% - 2\%$ improvement on other $5/13$ datasets. By removing the cluster assignment/ factorization term, we observe a drop in performance by a maximum of 5.22% on Washington, 4.08% on Cornell and 3.76% on Wisconsin, followed by 3.14% on

	USS-NMF	- LS(E, $\hat{H}$)	- ($\hat{H}$ + LS(E, $\hat{H}$))
Cora	**87.306**	86.159	86.109
Citeseer	**70.187**	69.101	68.697
Wiki	**70.906**	70.407	69.825
Washington	**67.826**	63.478	62.609
Wisconsin	**56.391**	52.632	52.632
Texas	**63.830**	62.766	61.702
Cornell	**56.122**	52.041	52.041
PPI	**23.433**	23.281	23.191
Wikipedia	**57.894**	56.814	56.637
Pubmed	**84.009**	83.540	83.479
Co-Author	**37.858**	36.684	36.200
Blogcatalog	**41.254**	38.505	38.111
Microsoft	**49.706**	47.676	47.008

Table 3.7 Importance of Cluster Information.

LS(E, $\hat{H}$): Label similarity based smoothing on the predicted cluster space, $\hat{H}$: Inclusion of predicted cluster information through factorization. In 3rd and 4th columns the minus sign indicates the respective terms are removed from the objective of USS-NMF.

Blogcatalog, between $1.0\% - 2.5\%$ for $7/13$ other datasets and $< 0.5\%$ for the rest. Note that random cluster assignments with unsupervised clustering objectives (ComE, MNMF, GEMSEC) did reasonably well but only a few times outperformed the second-best model, MNMF+Y in Table 3.4. USS-NMF outperforming all — demonstrates the usefulness of encoding label similarity invariant representations on cluster space. The next Section will further clearly see the benefits of combining label and cluster smoothing terms.

3.9.4 Study on Laplacian smoothing variants

In this Section, we analyze and compare different Laplacian smoothing variants in Table 3.8. The models in the first four columns enforce Laplacian smoothing with the proximity graph, S, and the next four models enforce Laplacian smoothing with the label similarity graph, E. All the models in this Table additionally factorize label matrix, Y. The model in the last column is USS-NMF. *USS-NMF is the winner across the board on all datasets with Rank* 1 *and Penalty* 0.

Model in Column: 3, $NMF : LS(S, \hat{Y}) + Y$, is the standard Label Propagation implemented in NMF style. It does not additionally factorize the proximity matrix, S, and thus does not

Laplacian	Using Neighborhood Similarity Graph (S) : LS(S, *)				Using Label Similarity Graph (E) : LS(E, *)			
Invariant Space	Embedding: U^*	Label: $\hat{Y}^*$	Cluster: $H(U)^*$	Label: $\hat{Y}^*$	Embedding: U^*		Cluster: $H(U)^*$	
	+Y	+Y	+Y	+Y +S	+Y +S	+Y +S + LS(S, $\hat{Y}$)	+Y +S	+Y +S + LS(S, $\hat{Y}$)
Cora	86.568	85.683	85.506	87.109	85.756	86.716	85.883	**87.38**
Citeseer	69.040	68.825	68.678	68.697	68.637	69.039	69.825	**70.187**
Wiki	69.410	69.742	66.885	69.825	67.249	69.493	67.168	**70.906**
Washington	60.000	55.652	66.957	66.957	66.957	65.652	**67.826**	**67.826**
Wisconsin	48.120	49.624	52.880	52.632	52.759	52.880	54.887	**56.391**
Texas	57.447	59.574	60.634	61.702	60.638	61.702	61.703	**63.830**
Cornell	52.041	53.061	52.041	52.041	53.061	54.082	**56.122**	**56.122**
PPI	22.346	22.648	23.087	23.191	22.497	22.886	23.130	**23.433**
Wikipedia	44.277	43.304	56.147	56.637	55.805	56.398	56.254	**57.894**
Pubmed	83.387	83.499	83.151	83.479	82.779	83.801	83.981	**84.009**
Co-Author	36.198	36.127	36.260	36.200	36.684	36.908	36.973	**37.858**
Blogcatalog	36.246	36.103	39.001	39.111	38.983	38.083	39.736	**41.254**
Microsoft	43.220	45.397	48.100	48.108	47.237	48.213	48.669	**49.706**
Rank	6.46	6.31	5.46	3.85	5.62	3.77	2.69	**1**
Penalty	4.5	4.43	2.11	1.62	2.13	1.61	1.13	**0.000**

Table 3.8 Semi-Supervised Learning Analysis | Micro-F1 Scores

have any network embeddings. This is the second-worst performing model with the highest Rank and highest Penalty. From this model, it is evident that we need to learn from the network as well.

Model in Column: 2, $NMF : LS(S,U) + Y$ learns embeddings such that connected nodes have closer embeddings. It is similar in spirit to $NMF : S + Y$, which factorizes the proximity matrix directly. This model has the lowest Penalty.

Model in Column: 6, $NMF : LS(E,U) + S + Y$ (NMF: Planetoid) has the same objective as Planetoid-G but in NMF style. It enforces nodes with similar labels to have similar embeddings. In a head-on comparison with $NMF : LS(S,U) + Y$ (Column: 2), it shows that *enforcing smoothness from a global context has remarkable improvement* in Penalty scores and is statistically better with $p < 0.02$.

Model in Column: 5, $NMF : LS(S,\hat{Y}) + S + Y$, is a powerful model which is second on 5/13 datasets. It can be seen as an improvement of Label Propagation where it additionally factorizes proximity S to learn node embeddings that capture network structure. It clearly beats the base $NMF : LS(S,\hat{Y}) + Y$ in Column: 3.

Model in Column: 7, is an extension of NMF: Planetoid (Column: 6), which includes label smoothing. It can be seen that adding label smoothing provides an improvement on all datasets up to 2.24 points.

Model in Column: 8 is USS-NMF without the label smoothing term. The label smoothing term is crucial as it improves up to 3.74 points in Column: 9 (USS-NMF). *And, similarly, semi-supervised clustering with global context is also important as it provides an improvement*

between $0.5-4.08$ *points over NMF* : $LS(S,\hat{Y})+S+Y$ *(Column: 5) in most of the datasets except Cora, PPI. Adding clustering components to label smoothing term NMF* : $LS(S,\hat{Y})+S+Y$ *(Column: 5) provides an average improvement of* 1.6 *points in Column: 9.*

3.9.5 SSL with balanced dataset

Datasets	Planetoid-G	MMDW	NMF:S+Y	MNMF+Y	NMF:Planetoid	NMF:S+Y+LS(S,$\hat{Y}$)	USS-NMF
Cora	69.1	71.6	74.3	75.9	75.1	75.4	**79.1**
Citeseer	49.3	60.9	62.6	61.6	63.1	63.7	**66.7**
Pubmed	66.4	80.6	80.4	79.6	80.5	80.7	**82.1**

Table 3.9 Node Classification Results | Balanced Sampling | Micro-F1 Scores

Planetoid-G versus all the competing Semi-Supervised models (SoTA Baselines and their variants) for Planetoid paper's train/test splits of graph data

Planetoid [22] was defined for multi-class classification problem on balanced labeled set, i.e., the same number of representative train instances for all labels (one unrealistic setting for real-world data). We empirically observed Planetoid performing poorly in comparison to other baselines when the labeled set is randomly drawn. Also, Planetoid is not directly extensible for multi-label problems. Hence, we define a similar semi-supervised NMF model (NMF:Planetoid) with Planetoid's semi-supervised learning objective, i.e., explicitly enforcing embeddings of nodes of the same label to be similar.

Here, we report results for models on the original balanced train/val/test split given with Planetoid [22]. The non-attributed model of Planetoid, Planetoid-G, performs poorly in comparison to the NMF models even in this setup. Our results for Planetoid on running the author's code are very similar to what is reported in their paper. However, we can see significant improvement in the results for NMF:Planetoid. *The improvement owing to NMF:Planetoid can be attributed to (i) exact computations instead of sampling strategy followed in the original paper and (ii) non-negative embeddings.* From Table 3.9 it is evident that among all the competing methods, USS-NMF does significantly better. We find balanced class distribution to be beneficial for our model as we obtain extraordinary performance improvement on Cora, Citeseer, and Pubmed.

3.9.6 SSL with varying ratio of labeled data

We also report node classification results for varying test-train splits to study and understand how our method does on various label sparsity cases in Table 3.10 against all the semi-

Dataset	Cora						Citeseer						Wiki				
Train (%)	5	10	20	30	40	50	5	10	20	30	40	50	10	20	30	40	50
NMF:S+Y	67.519	76.620	79.012	84.660	84.194	85.535	51.128	54.172	62.068	64.379	68.477	68.618	57.367	60.442	62.945	63.404	65.152
MMDW	67.403	75.101	80.093	82.942	82.889	83.838	47.283	54.509	58.544	63.362	66.265	68.911	56.655	58.120	62.270	63.391	67.007
MNMF+Y	68.947	76.948	81.172	84.396	83.518	85.904	50.906	55.533	63.787	65.689	67.622	69.005	56.536	62.130	64.549	64.751	66.999
USS-NMF	**69.452**	**78.015**	**82.880**	**85.609**	**85.486**	**87.380**	**52.253**	**57.478**	**65.070**	**66.422**	**69.231**	**70.187**	**58.568**	**62.909**	**67.102**	**69.183**	**70.906**

Table 3.10 Node Classification Results | Varying Train-Test Splits | Micro-F1 Scores

supervised State-of-The-Art (SoTA) methods as baselines. The Table is enough to reflect the fact that USS-NMF does well in the case of label sparsity. For varying splits of randomly sampled labeled data, it consistently outperformed the other baselines. *When the labeled data is sparse, the cluster information acts as complementary information to help to predict labels for the unlabeled nodes.*

Impact of the initial training set size: As we varied the size of the train data in the range $5\% - 50\%$ of the entire data-points, we initially observed a more zigzag-like unstable curve when plotted the loss function against iterations for the smaller train data sizes. However, it converged more quickly than the larger train data sizes after the initial iterations.

Application to real-world problems: In many real-world problems, for example, in the Bio-Medical domain, analyzing numerous functions a protein is associated with – poses an important task and has many utilities. However, annotating the protein with a range of biological functions need domain experts. This annotation process is costly and should be executed reliably. Our proposed framework does not merely use the available supervision information along with the unsupervised data. Rather, it captures important diverse signals from the unsupervised data-points shaped up by the labels from the underlying graph. In such a crucial application domain, improving the automatic protein function prediction (node classification) or groupings of proteins for similar functions (node clustering) by a considerable margin will aid the costly annotation process and contribute significantly to various Bio-Medical tasks dependent on the predicted protein functions. In this way, the proposed USS-NMF is capable of contributing to real-world problems.

3.9.7 USS-NMF's sensitivity to number of clusters

We performed an extensive study on how the number of clusters influences node classification performance and whether there is any need for learning clusters at all. For Cora, Citeseer and Wisconsin — three multi-class datasets and Wikipedia — a multi-label dataset, we varied the number of clusters as in Figure 3.7. Blue solid lines indicate the classification performance of our proposed methods USS-NMF. Corresponding Red solid horizontal lines represent

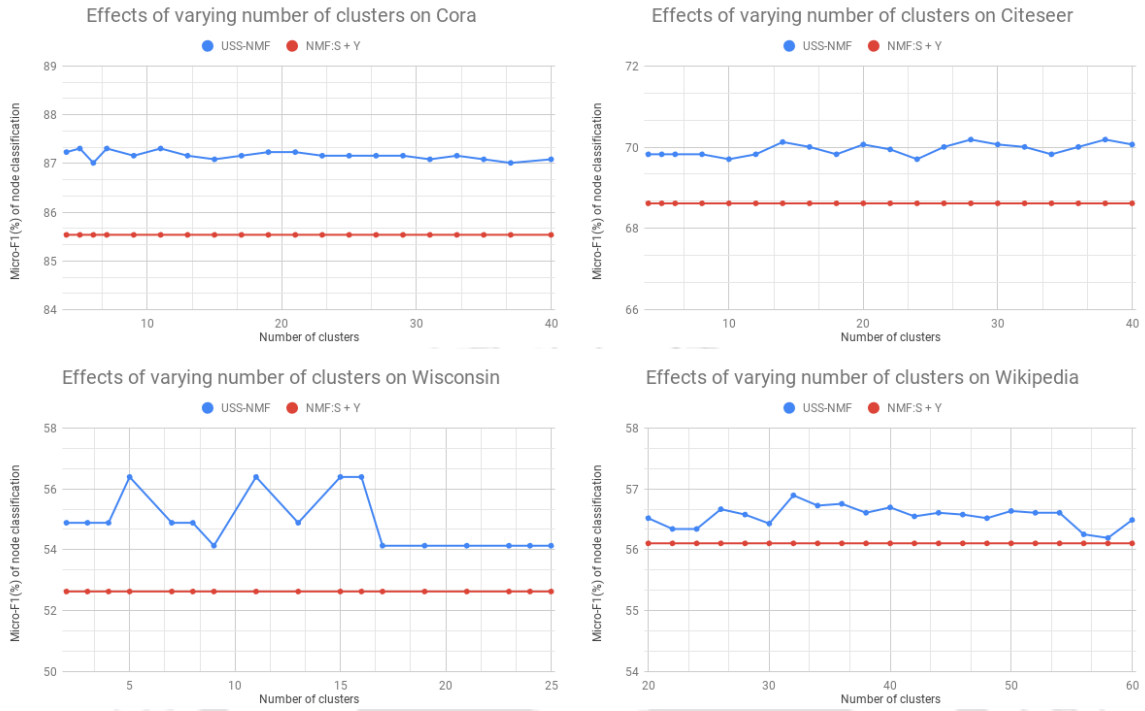


Fig. 3.7 Varying Number of Clusters

respective performances where all the cluster related terms were set to 0 ($\beta = 0, \phi = 0, \zeta = 0$), i.e., learning no clusters (NMF:S+Y best scores).

As we can see, significant portions of the curves are above their respective dotted lines, indicating that learning clusters help in guiding node representations. From the plot, it is interesting that a small dataset like Wisconsin is more sensitive to changing the number of clusters than larger datasets. For Wikipedia, a multi-label dataset, we can clearly see the tendency of learning overlapping clusters as the optimal clusters are lesser than the ground-truth labels. Again, there are optimal clusters very different from the ground truth labels (Citeseer) and multiple numbers of optimal clusters (in Cora, Citeseer, Wisconsin), which indicates that small clusters of same class data having low density separation among them (Refer to t-SNE plots) are being learned.

Impact of cluster-sizes following a truncated power-law: Research in the traditional Network Science domain indicates interesting connections among the group sizes, group dynamics, and the underlying network structures [129–134]. Our proposed framework USS-NMF does not make any assumptions about how the cluster-size distributions are. Given a K (the number of clusters), the cluster sizes will follow along, and it is not guaranteed to exhibit the truncated power-law distribution in the cluster sizes. Also, all our experimental

datasets described in Table 3.1 are real-world that do not guarantee to follow a specific distribution in the ground-truth cluster-sizes. Hence, studying how our model performs when the cluster-sizes follow a truncated power-law is not explored in this thesis. Synthetic graphs may be generated to perform such case-studies toward this objective. It would be an interesting experiment to conduct in the future.

3.9.8 Convergence Analysis for USS-NMF

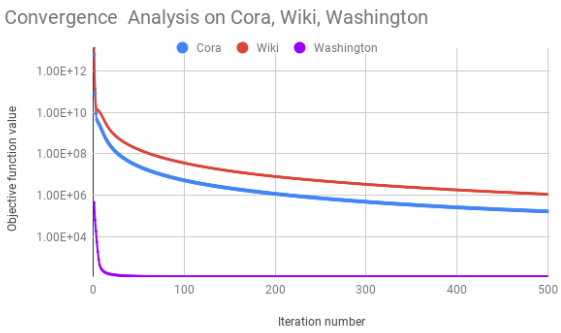


Fig. 3.8 Convergence Analysis of USS-NMF

We study and validate the convergence property of USS-NMF in Figure 3.8, where the objective function values are plotted against iteration numbers. The objective function values are non increasing with iterations and a sharp decrease in the objective function values can be seen within a few iterations between $0-10$, which empirically proves the correctness of our algorithm.

Summary

We proposed a novel cluster learning optimization objective (USS-NMF) based on graph Laplacian regularization, which captures global structure in the graph and ensures label smoothing in a node's local neighborhood and inherent global neighborhood. We developed a semi-supervised transductive framework that incorporates all necessary prior information in non-attributed graph data, for more informative node representation learning. Extensive experimental results on node classification and clustering tasks illustrate the superiority of our method. In this work, our focus has been on developing the unified framework. We can make the framework scale to large networks using ideas from efficient NMF algorithms [135, 136].

Chapter 4

Structure-Aware Network Representation Learning on Multiplex Graphs

Multiplex networks are complex graph structures in which a set of entities are connected to each other via multiple types of relations, each relation representing a distinct layer. Such graphs are used to investigate many complex biological, social, and technological systems. In this work, we present a novel semi-supervised approach for structure-aware representation learning on multiplex networks. Our approach relies on maximizing the mutual information between local node-wise patch representations and label correlated structure-aware global graph representations to model the nodes and cluster structures jointly. Specifically, it leverages a novel cluster-aware, node-contextualized global graph summary generation strategy for effective joint-modeling of node and cluster representations across the layers of a multiplex network. Empirically, we demonstrate that the proposed architecture outperforms state-of-the-art methods in a range of tasks: classification, clustering, visualization, and similarity search on seven real-world multiplex networks for various experiment settings.

4.1 Introduction

Entities in many real-world problems are related to each other in multiple ways. Such relations are often modeled as graph-structured data where the nodes represent entities, and edges between a pair of nodes represent the interactions between the entities. Learning representations for such networked data to mine, analyze and build predictive models has

been gaining a lot of traction recently with the advent of deep learning-based network embedding models [137, 138].

Increasingly such relations are complex, with multiple relationship types linking entities. Such networked data are often naturally represented as multi-layered graphs [71], where each component layer focuses on a specific relation type and can involve different sets of nodes. In this work, we focus on *Multiplex networks*, a special case of multi-layer networks where the graphs in all the layers share the same set of nodes with distinct relations in different layers. Such multiplex network structures are observed in numerous environments, such as bibliographic networks, temporal networks, traffic networks, brain networks, protein-drug-disease interaction, etc. The involvement of the same set of nodes across multiple types of relations, distinctive structures in different layers, and the interplay among various layers of networks — make representation learning of multiplex networks a challenging task.

Existing multiplex Network Representation Learning (NRL) methods learn node embeddings that encode the local relational structure of nodes by using graph convolutions [37–39], or random walks [40, 41] within a subgraph centered at the node of interest. Though there are many powerful models to learn local structures, only a few works encode global structures [38, 39] even in the case of the more widely researched simple homogeneous graphs. Global structural information is encoded in representation learning models through one of three approaches: (i) clustering constraints [29, 109]; (ii) auto-encoding objectives on the adjacency matrix [32, 29, 109], or node embeddings [139]; and (iii) Mutual Information Maximization (InfoMax) [2, 43] objectives that maximize the Mutual Information (MI) between the representations of local nodes and the global summary of the graph derived from the local contexts of all the nodes [2, 43, 39, 44].

Clustering constraints are also often realized with auto-encoding objectives that, in general, are challenging to scale [29, 109]. In contrast to the first two methods, the InfoMax-based approaches use Graph Neural Networks (GNNs) to obtain both local and global context and are potentially more scalable [2, 39, 44]. However, InfoMax objectives that encode global information assume a shared global graph context for all the nodes despite the fact that, in most cases, every node has a different global structure rooted at each node. This calls for a different contextualized global graph representation for each node – which is analogous to the notion of personalization.

4.2 Challenges

Having motivated the need for structure-aware NRL in multiplex graphs and contextualized global graph summaries in InfoMax-based objectives, here, we discuss *challenges* [140, 73, 72] typically exhibited in multiplex networks, especially in the context of learning network representations for node-wise downstream tasks. Apart from the challenges faced in homogeneous graphs (discussed in Section 3.2), multiplex graphs have below unique challenges,

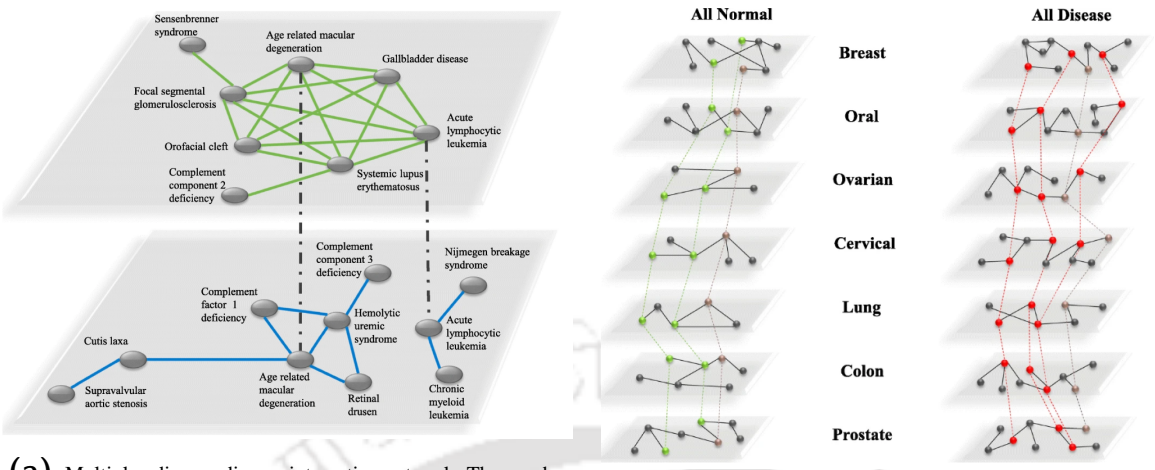
Layers	V	E	D_{Mean}	#CC	LCC	Assortativity	#Claws	#Triangles	Modularity	Avg. clustering coefficient	Power law exponent	Rel. edge distr. entropy
gog	20419	6371558	312.0406494	5959	14454	0.1210141566	817699556367	296136268	0.4467549167	0.001086473384	1.176404984	0.9344384576
gpg	20419	832924	40.79161453	16565	3852	0.1214684564	37926249131	30207063	0.4380525945	0.002389405519	1.205277675	0.7902766049
gdcdg	20419	36190	1.772368908	19825	576	0.2398061504	79681043	398366	0.4985001314	0.01499852355	1.278372358	0.6100376646
gtg	20419	606974	29.7259407	16459	3959	0.02190370239	10421268598	12740991	0.5186753186	0.003667785034	1.215357163	0.8040821361
gdg	20419	14988	0.7340222597	19181	903	0.68626692	1778545	48506	0.8927133879	0.08181856518	1.563538469	0.6775293758
gg	20419	344496	16.87134552	13565	6790	0.06436341977	1695377155	2178390	0.4703936122	0.003854699812	1.320807235	0.8297416882
Aggregated	20419	3948018	401.9359436	4652	15761	0.1185197846	1826224670090	379872417	0.3836856784	0.0006240290528	1.174822063	0.8741877446

Table 4.1 Layer-wise structural diversities in SLAP [Refer to Table 4.4 for details on SLAP]

Metric	Computation	Description
V		The set of vertices in a graph
E		The set of edges in a graph
D_{Mean}	$\text{MEAN}_{v \in V}\, D(v)$	Average degree of all nodes in a graph
#CC		The number of connected components in a graph
LCC	$N_{\text{MAX}} = \text{MAX}_{f \subseteq CC}\ \lvert f \rvert$	Size of largest connected component, where CC are all connected components of the graph
Assortativity	$\rho = \frac{\text{Cov}(X,Y)}{\rho_X \cdot \rho_Y}$	Pearson correlation of degrees of connected nodes, where the (x_i, y_i) pairs are the degrees of connected nodes
#Claws	$\sum_{v \in V} \binom{D(v)}{3}$	Number of claws (3-stars)
#Triangles	$\frac{\lvert\{(u,v,w):(u,v),(v,w),(w,u) \subseteq E\}\rvert}{6}$	Number of triangles in the graph, where nodes u and v are connected via node w
Clustering coefficient	$3 \times \frac{\#\text{TRIANGLES}}{\#\text{CLAWS}}$	The degree to which nodes in a graph tend to cluster together
Modularity	$A_{u,v} - \frac{D(u).D(v)}{2\lvert E \rvert} : u, v \in V$	The modularity of the graph after optimally partitioning it using Louvain algorithm [60]
Power law exponent	$1 + n(\sum_{u \in V} log \frac{D(u)}{D_{\text{MIN}}})^{-1}$	Exponent of the power law distribution, where D_{MIN} denotes the minimum degree in a network
Rel. edge distr. entropy	$\frac{1}{ln\lvert V \rvert} \sum_{v \in V} -\frac{D(v)}{\lvert E \rvert} ln \frac{D(v)}{\lvert E \rvert}$	Entropy of degree distribution, 1 means uniform, 0 means a single node is connected to all others

Table 4.2 Metric definitions and their interpretations [Source: NetGAN [3]]

Preserving layer-wise structural diversities Relational layers of a multiplex graph can be structurally diverse. Table 4.1 provides deeper insights on this. We tabulate values of various topology-based metrics defined in Table 4.2 for the relational layers and the aggregated network of a biological multiplex graph SLAP. We observe that, even though the "gdg" (GENE-DISEASE-GENE) layer has a lesser number of edges, it is more modular, having higher global clustering coefficients than the rest of the layers. Also, it has the highest skewness in the degree distribution apparent from the *Power law exponent* and *relative edge distribution entropy* values. In terms of *assortativity* value "gdg" layer prefers *homophily*, i.e., high-degree nodes tend to attach to high degree nodes. Whereas, the rest of the gene-gene interactions show *disassortativity*



(a) Multiplex disease-disease interaction network. The graph consists of two layers. The nodes are the diseases, connected on the first layer if they share a phenotypic interaction, and on the second if they share a genotypic one.

(b) Multilayer Cancer Cells' Network. The Figure shows two graphs, normal and disease. The nodes in graphs are the proteins, assigned to 7 layers reflecting 7 cancer types.

Fig. 4.1 Inter-dependencies of layers in multiplex graphs [Image Source: Internet]

aka *heterophily* – which is very common in biological networks [60]. We also see that, even though the rest of the layers have a high count of triangles and claws, the nodes in these layers do not tend to form clusters and have fair degree distributions. For an NRL model, capturing this layer-wise diversity is of utmost importance.

Learning from higher-order graph structures Due to the existence of within-layer and across-layer node-to-node associations, defining a higher-order neighborhood for a node can be very challenging [141]. This is since, higher-order graph structures such as walks, relational paths, hyper-edges, communities, and clusters may span across the layers. Even in the case of learning from layer-wise graph structures, combining the global structures across the layers to learn a final consensus pose another challenge.

Unifying node characteristics across layers From Table 4.1, it is evident that the same set of nodes plays unique structural roles in each layer. Non-trivial or trivial cross-layer node-to-node associations do not capture the replica nodes' distinct roles. Thus, designing an effective node aggregation strategy to obtain a unified embedding that not only captures the defined cross-layer interactions, but also incorporates the structural identities of the nodes – is a challenge.

Inter-dependencies of relational layers The presence of layer-correlations and/or layer-dependencies among each other provide important insights for the decision algorithm [142]. Again, given a downstream task, a few layers may prove to be more useful than the rest of the layers. Thus, capturing the relevance of layers for a target task can

aid in the prediction performance even more. Figure 4.1 depicts two such scenarios. In the first example 4.1a, disease-disease node associations are modelled based on genotype and phenotype-based interactions. Now, for the task of predicting disease classes or a drug for a particular disease node – one layer may prove to be more relevant than the other. In the second example 4.1b, the layer-layer dependencies denote various primary and metastatic regions for cancer. The dependencies may influence protein-protein interactions in the participating layers. Capturing such correlations is necessary to learn task-specific enriched network representations.

4.3 Motivation

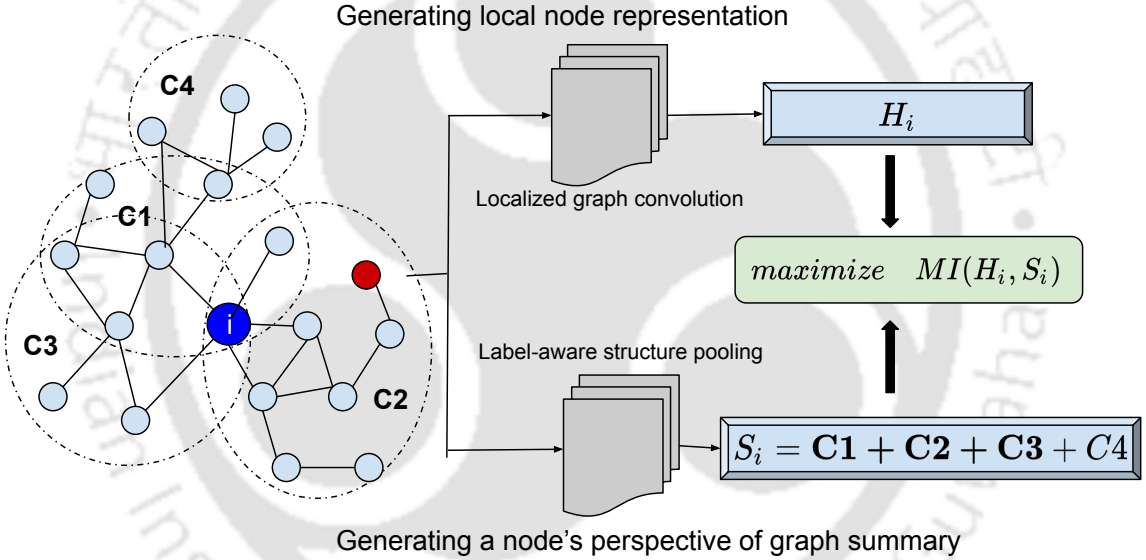


Fig. 4.2 Label correlated structure-aware InfoMax

Here we present some important observations to motivate why it is worthwhile to incorporate personalized global features for node embedding learning in any InfoMax-based methods. In a typical InfoMax-based NRL setup, the global context is defined by all nodes in the graph. Thus, each node in the graph does not have its own contextual view of the graph. Instead has a shared global context that is the same for all the nodes, *even though the nodes may be structurally connected differently within the graph*. For example, from the graph in Figure 4.2, the red node and blue node belonging to the same cluster $C2$ will have different non-local network measures such as betweenness measure, participation coefficient [143], etc., as the structure of the (sub)graph centered around these nodes are differently connected

to the rest of the graph since one is in the center of a cluster, and other is at the end of a whisker.

When a trivial global graph summary function, such as the average of all node embeddings, is used, the global context for all nodes becomes the same — as their global context is isomorphic. Naively maximizing the MI of a node's local representations with a shared global context might bias the model to encode trivial and noisy information found across all the nodes' local information. Albeit, naively defining a different global context for each node, such as a sub-graph-based approach, will shoot down the original objective of learning useful shared information from across the graph.

Thus, this calls for a careful design of a *contextualized representation of shared global information* that facilitates encoding relevant non-trivial shared information present across the graph when maximizing the MI with the local node information. In Figure 4.2, even though the example graph has many clusters, only $C1, C2, C3$ are relevant to the blue node i. Therefore, the global context for node i should be more inclined towards $C1, C2, C3$ instead of a naive summary of all candidate clusters. In light of this simple intuition and motivated by participation scores, we propose a cluster-based InfoMax objective to learn node representations. *The clusters encode shared global graph information, and the node-specific global context is obtained by aggregating information from the clusters with which the node is associated.* In particular, for the semi-supervised classification task, we define label-correlated structure-aware clusters that jointly learn node and cluster representations by optimizing the InfoMax principle.

4.4 Research Objective

In this work, the research objective is to learn higher-order graph structure-aware node embeddings on multiplex graphs keeping in mind the challenges (refer to Section 4.2) exhibited for NRL tasks. We find the InfoMax-based learning strategy for multiplex graphs as a useful means to incorporate global graph structures into the learned node embeddings. However, the InfoMax-based method for graphs, in general, suffers from the problem of trivial, possibly noisy global summaries for the local node embeddings. Here, we aim to bring novelty into existing InfoMax-based methods by proposing a novel non-trivial global graph summary personalized for each node to optimize jointly with the local node embeddings.

4.5 Contributions

The primary contributions of our work are,

Node Contextualized Global Graph Representations Motivated by the need for contextualized global graph representations, we propose a novel *joint* node and cluster representation learning model that defines a structure-aware intra-layer graph context for a node.

Designing Structure-Aware InfoMax In this work, we are the first to propose a novel *Structure-Aware InfoMax* strategy for encoding non-trivial global graph structures into node embeddings. We propose a node-contextualized InfoMax-based semi-supervised learning architecture for multiplex networks.

Learning Label-Informed Clusters In the semi-supervised setting, the cluster constraints are provided by the partial label information. The goal is to learn similar clusters across relational layers in terms of label correlations while preserving the layer-wise structural diversities.

Utilizing Cross-Layer Associations Further, we constrain the nodes connected by cross-edges to have similar embeddings, thereby indirectly influencing the layer-wise InfoMax objective to capture global cluster information across multiple layers.

We evaluate the model on seven multiplex networks for node classification, clustering, and similarity search. Our proposed model achieves the best overall performance outperforming state-of-the-art methods like DMGI [39], MGCN [37], HAN [144]. Also, the learned node embeddings lead to well-separated homogeneous clusters in t-SNE visualizations.

4.6 Literature Review

In this section, we discuss related representation learning literature focused on multiplex networks, especially in the context of node-wise downstream tasks. Figure 4.3 gives us an overview of various paradigms and representative methods for each.

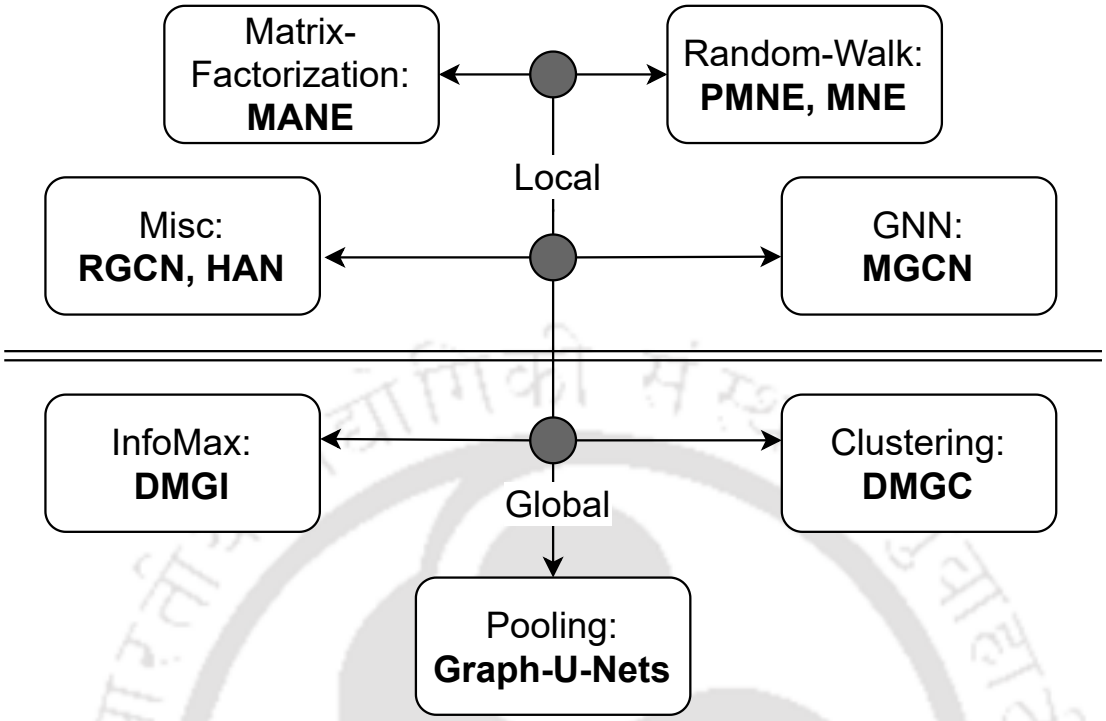


Fig. 4.3 Overview of NRL research on multiplex graphs. Various local and global structure-aware NRL methods are shown.

4.6.1 NRL on multiplex graphs

Network Representation Learning (NRL) methods for multiplex networks use different learning paradigms such as matrix factorization [21]; random-walk based objectives [40, 41] and graph neural network architectures [37–39].

NRL models for multiplex networks have aimed at capturing different aspects of this multi-layered data. Modeling multi-layer data might require one to capture local [21, 145] and global network structures [38, 39] within each layer; leverage cross-layer edges [21, 37, 146, 145] between layers; encode node features [37, 39]; integrate information from multiple layers into a unified feature space [40, 41]; Optimize for single objective [19, 144] or jointly optimize for different objectives at different layers [37, 38].

4.6.1.1 Global context-based NRL

In general, random-walk-based methods and GCNs are limited to capturing the k-hop local contexts of nodes only. While matrix factorization methods embed the entire graph, they are

neither scalable nor more powerful than the other two. Only a few studies capture the global structures into node embedding learning in both homogeneous and multiplex networks.

GUNets [139] proposes a network representation learning framework to facilitate pooling and unpooling operations on multi-graphs through an encoder-decoder architecture using GCNs. Unlike the other graph pooling methods which target graph classification tasks, it can handle both node classification and graph classification tasks. The under-sampling, aka pooling operation, selects a few prominent nodes from all the nodes based on the maximum scalar projection of node embeddings on a learnable projection vector in phases. The unpooling process, aka up-sampling, does the exact opposite of the pooling process, i.e., restores the coarsened graph to its original structure and preserves nodes' local information via a GCN layer. GUNets' prominence-based node pooling is capable of capturing the global graph structures into node embeddings.

Deep Multi-Graph Clustering (DMGC) [38] is the first NRL study to explicitly learn global structures for multilayer networks. It proposes an attentive unsupervised mechanism to encode the cluster structures into multi-graphs based on a similarity-based cluster kernel. It formulates minimum-entropy-based clustering criteria based on the Cauchy kernel to cluster the node embeddings outputted from relation-specific autoencoders. To ensure relation layers' autonomy, it models nodes and clusters in individual feature spaces for each layer and formulates one unsupervised cluster–cluster attention mechanism. The attention weights are used to project cluster centers from unified space to individual graph spaces and to ensure that the cluster assignment of connected cross-layer nodes is brought closer.

Whereas DMGI [39] discussed in Section 2.2.2.3 is also a global context-based NRL method for multiplex graphs that naively captures a global graph representation by applying one mean-pool-based readout function on the learned node representations.

However, the discussed global methods are unsupervised, do not model the cross-layer node-to-node association, and do not offer any useful strategy to combine node embeddings across layers.

4.6.1.2 InfoMax based NRL

Deep Multiplex Graph Infomax (DMGI)[39] employs the InfoMax principle for multiplex networks. It jointly maximizes the MI between the local and global graph patches across the layers of a multiplex graph. It does so by learning a universal discriminator that discriminates positive and negative patch pairs across the relational layers. Simultaneously, it employs

a regularization strategy that attentively aggregates the learned relation-specific node representations by reusing negative node representations used for learning the discriminator weights. Please refer to Section 2.2.2.3 for details. HDGI [44] is a work similar to DMGI, aimed at heterogeneous networks. It adopts a semantic attention mechanism to aggregate metapath-influenced node embeddings and a discriminator-based learning strategy. Unlike DGI [2] which maximizes local and global patch representations in homogeneous graphs for various node-related tasks, InfoGraph [43] adopts a similar objective of simultaneously learning substructure representations at different scales and maximizing the MI between graph-level and substructure-level representations for various graph-related unsupervised, semi-supervised tasks.

4.6.1.3 Semi-Supervised Learning (SSL)

State-of-the-art methods MGCN [37], DMGI [39] are examples of SSL frameworks for multilayer/ multiplex networks. MGCN proposes a layered graph convolution neural (GCN) architecture to preserve the within-layer and cross-layer network structures by leveraging a cross-entropy loss function in each network layer. The framework has one unsupervised part which estimates the structural reconstruction loss, i.e., loss based on within-layer and between-layer network structure reconstruction via an inner-product-based loss function defined on the learned embeddings. It also has one supervised cross-entropy loss from the labels predicted on train instances. DMGI – though originally proposed as an unsupervised method, inculcates a semi-supervised variant that explicitly guides the learning of layer attention weights.

We have HAN [144] and RGCN [19] from the domains of heterogeneous and knowledge graphs, respectively. HAN proposes a GNN architecture based on hierarchical node-level and metapath-level attention mechanisms. First, this model projects typed nodes into a unified feature space with a node-type transformation matrix. Next, node-level asymmetric attention is learned to distinguish a node and the roles played by its heterogeneous metapath-based neighbors for making any inference on that node. Whereas in semantic-level attention learning, rich semantic/metapath-specific node embeddings are fused based on learned metapath importance. The model learns this hierarchical heterogeneous structural attention, guided by a cross-entropy-based semi-supervised prediction loss. In contrast, RGCN proposes a graph convolution-based message passing framework facilitating targetted at multi-relational graph data. To reduce the number of weight parameters owing to the network's highly multi-relational nature and avoid overfitting on rare relations, it proposes two

kinds of weight regularization techniques, namely, basis regularization and block-diagonal decomposition. This weight regularization can be seen as a systematic and effective way of facilitating information flow across the relations. It incorporates supervision information via one cross-entropy-based semi-supervised prediction loss based on train samples.

4.6.2 Research Gaps

If we are to summarize the research gaps that exist in learning structure-aware network representations on multiplex graphs, the following points give us important glimpses,

- Random-walks, GNNs, auto-encoders, and proximity-based methods are limited to only capturing k-hop local contexts for nodes.
- Matrix factorization methods are not scalable to handle node associations from many relational layers.
- Structural pooling-unpooling methods do not capture cross-networks. They also lack an effective node-aggregation strategy across the layers.
- GNN and InfoMax-based methods capture global structures in unsupervised and trivial ways, respectively.

4.7 Proposed Framework: Semi-Supervised Deep Clustered Multiplex (SSDCM)

In this section, we explain step-by-step our proposed approach to learning node representations for multiplex networks. The proposed method *Semi-Supervised Deep Clustered Multiplex (SSDCM)* in Figure 4.4 — (i) learns relation-specific node representation that encodes both local and global information, (ii) enforces cross-edge based regularization to align all nodes connected across layers to lie on the same space, then (iii) learns a joint embedding across layers for all nodes through a consensus regularization and (iv) finally enables label predictions with this joint embedding.

4.7.1 Learning Node Representations

The first component of our model learns relation-specific ($\mathcal{R}$) local node representations, U_r. Then these learned node representations are made aware of their individual global

context, S_r, which summarizes graph-level information. We do this by maximizing the mutual information between them, and this can be realized by minimizing a noise-contrastive estimation such as a binary cross-entropy loss as provided in the equation below,

$$\mathfrak{O}_{MI} = \sum_{r\in\mathcal{R}}\sum_{i\in\mathcal{V}}\Big(\log(\mathfrak{D}(U_r^i, S_r^i)) + \sum_{j=1}^{N}\log(1-\mathfrak{D}(\tilde{U}_r^j, S_r^i))\Big) \tag{4.1}$$

where $\mathfrak{D}: \mathbb{R}^{2d} \mapsto \mathbb{R}$ is a discriminator function that assigns a probability score to a pair of local-global node representations using a bi-linear scoring matrix $B \in \mathbb{R}^{d\times d}$, i.e., $\mathfrak{D}(U_r^i, S_r^i) = \sigma(U_r^{iT} B S_r^i)$, σ being the sigmoid non-linearity. Similar to [39], we learn this discriminator universally, i.e., the weight is shared across all layers with an intention to capture local-global representation correlations across the relations. The discriminator gives a local-global summary a higher value if the local summary is highly relevant to the global summary and, in contrast, assigns a lower score if the local summary is less relevant or irrelevant to the global summary. For every node i in each relation $r \in \mathcal{R}$, N negative local summaries are paired with that node's contextual global summary to train the discriminator $\mathfrak{D}$. Following [2], we create corrupted local patches $\tilde{U}_r^j$ for each relation r by row-shuffling the node features X and passing it through the same local structure encoder.

Having explained the overall structure of our InfoMax objective, we get into the details of how to learn (a) local node representations, (b) global node representations, and (c) the clustering strategy that provides the global context for nodes.

4.7.1.1 Local Node Representations

For each relation $r \in \mathcal{R}$, we obtain an M-hop local node representations $U_r \in \mathbb{R}^{|\mathcal{V}|\times d}$ with a Graph Convolutional Neural Network (GCN) encoder $\mathfrak{E}_r$. GCNs obtain an M-hop local representation by recursively propagating aggregated neighborhood information. Let $\tilde{A}_{(r,r)} = A_{(r,r)} + \varepsilon I_{|\mathcal{V}|}$ be the intra-layer adjacency matrix for relation r with added ε-weighted self-loops (similar to a Random Walk with Restart (RWR) probability kernel). Here, we use the normalized adjacency matrix, $\hat{A}_{(r,r)} = (\tilde{D}_{(r,r)}^{-\frac{1}{2}} \tilde{A}_{(r,r)} \tilde{D}_{(r,r)}^{-\frac{1}{2}})$, as the GCN's neighborhood aggregation kernel, where $\tilde{D}_{(r,r)}$ is the diagonal degree matrix of $\tilde{A}_{(r,r)}$. An M-hop node embedding is obtained by stacking M-layers of GCNs as in Eqn: 4.2. The input to the m^{th} GCN layer is the output of the $(m-1)^{\text{th}}$ GCN layer, X_r^{m-1}, with the original node features X

fed in as input to the first layer.

$$\begin{aligned} X_r^0 &= X \\ X_r^m &= PReLU(\hat{A}_{(r,r)} X_r^{m-1} W_r^m) \\ U_r &= X_r^M \end{aligned} \tag{4.2}$$

where W_r^m is learnable weight matrix for the m^{th} GCN layer corresponding to the r^{th} multiplex layer. Assuming all GCN layers' outputs to be of the same dimension d, we have $X_r^m \in \mathbb{R}^{|\mathcal{V}| \times d}$ and $W_r^m \in \mathbb{R}^{d \times d}$, except for the first GCN layer, whose weights are $W_r^0 \in \mathbb{R}^{|\mathcal{F}| \times d}$. The final M-hop GCN representation for each relation, r is treated as that relation's local node embedding, $U_r = X_r^M$.

4.7.1.2 Contextual Global Node Representations

We learn a contextualized global summary representation, S_r^i for each node i, and for every relation $r \in \mathcal{R}$. In this work, we first capture a global graph level summary by learning K clusters in each relation. Then we leverage these learned clusters to provide a contextual global node summary for all the nodes based on the learned node-cluster associations. We explain the steps in a top-down manner. We first explain how we obtain a contextualized global graph representation given clustering information, and in the following subsection, we explain how to obtain the clusters.

Across all multiplex layers, we learn K clusters in each relation $r \in \mathcal{R}$. We encode the clustering information with relation-specific K cluster embeddings, $C_r = \{C_r^1, C_r^2, ..., C_r^K\}$ with $C_r^k \in \mathbb{R}^{1 \times d}$ and node-cluster assignment matrix, $H_r \in \mathbb{R}^{|\mathcal{V}| \times K}$. Given the learned relation-wise clustering information (C_r, H_r) and local node representations, U_r, we compute the contextual global node representation for a node i as a linear combination of different cluster embeddings, C_r^k weighted by that node's cluster association scores $H_r^i[k], \forall k \in [1, K]$ as mentioned in below.

$$S_r^i = \sum_{k=1}^{K} H_r^i[k] C_r^k \tag{4.3}$$

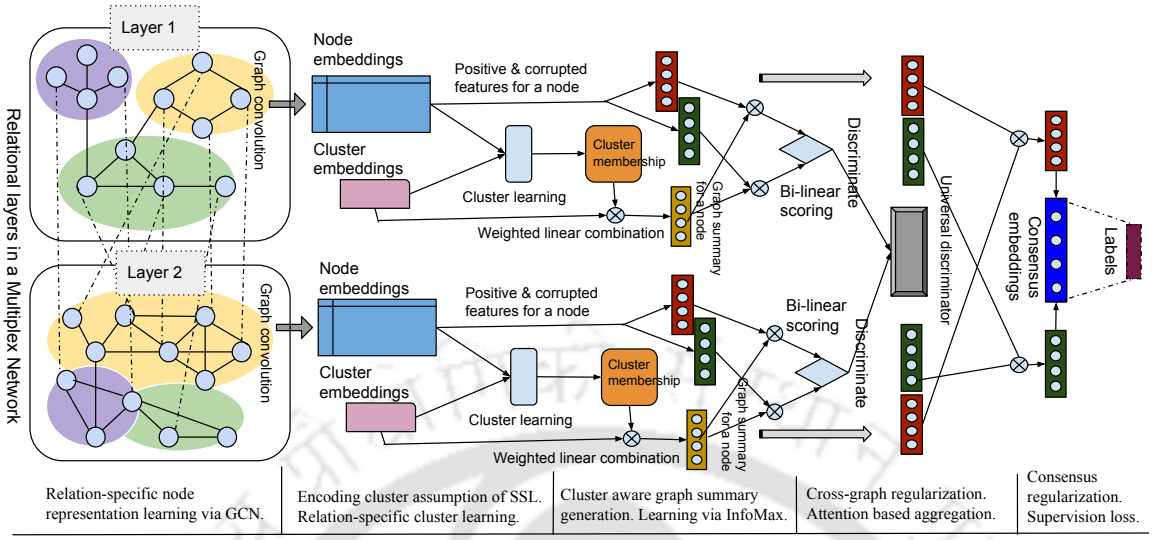


Fig. 4.4 Semi-Supervised Deep Clustered Multiplex (SSDCM) with structure-aware graph summary generation
Explanation of used color-codes. Green: True node embeddings, Red: Corrupted node embeddings, Blue: Final consensus node embeddings. Orange: Learned cluster membership for nodes, Purple: Label information. The color-coded grouping of nodes in relational layers of the example multiplex network denotes different clusters.

4.7.1.3 Clustering

We now describe how to learn clusters that capture useful global information for a node across all relations. Specifically, we aim to capture globally relevant label information that can enrich local node representations for the semi-supervised node classification task when jointly optimized for the MI between them across relations. To achieve this, we adapt [147]'s Non-Negative Matrix Factorization formulation to learn label-correlated clusters to a Neural Network setup as follows.

Cluster Embedding: We randomly initialize the set of cluster embeddings C_r for each relation r and allow them to be updated based on the gradients from the model's loss.

Cluster Assignment: We obtain the node-cluster assignment matrix, H_r, by computing the inner product between node embeddings and cluster embeddings. We then pass it through a softmax layer to obtain normalized probability scores of cluster memberships for each node, see Eqn: 4.4.

$$H_r^i[k] = SoftMax(U_r^i.C_r^{k^T}) \tag{4.4}$$

Non-overlapping Clustering Constraint: To enforce hard cluster membership assignments, we regularize the cluster assignment H_r with block-diagonal constraints. Specifically, we ensure the block size to be one, and the resulting orthogonality constraint enforces less overlap in node assignments between every pair of clusters. This constraint is expressed as a loss function below.

$$\mathfrak{O}_{Orthogonal} = \|H_r^T H_r - I_K\|_F^2 \tag{4.5}$$

Global Label-homogeneous Clustering Constraint: To capture globally relevant information for each relational graph, we group nodes based on an aspect — enforcing homogeneity within clusters. Precisely, we capture global label-correlation information with a similarity kernel, $\mathcal{S} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ and cluster nodes according to it. The label similarity kernel is defined between the labeled nodes $\mathcal{L}$ as $\mathcal{S} = Y_{[\mathcal{L}]}.{Y_{[\mathcal{L}]}}^T$. We use a masking strategy to consider only the label information of training nodes for enforcing this.

We now use a Laplacian regularizer to enforce smoothness on the cluster assignments according to the label-similarity kernel $\mathcal{S}$ as given in the equation below,

$$\mathfrak{O}_{Learn} = Tr(H_r^T \Delta(\mathcal{S}) H_r) \tag{4.6}$$

where $\Delta(\mathcal{S})$ is the un-normalized Laplacian of the similarity kernel. The above Laplacian smoothing constraint enforces nodes with similar labels to lie in the same/similar clusters.

Note that since this is shared across relations, it enforces similar clustering to be learned across relations. The learned clusters can still vary based on the relation-specific node embeddings, thus capturing global shared context across diverse graph structures. More importantly, notice that the label-similarity kernel can connect nodes that may be far away by a distance longer than the local (M) multi-hop context considered and even can connect two nodes that are not reachable from each other.

In the entire pipeline, cluster learning is facilitated by the following loss function: $\mathfrak{O}_{Clus} = (\mathfrak{O}_{Learn} + \mathfrak{O}_{Orthogonal})$

4.7.2 Cross-relation Regularization

Since each multiplex layer encodes a different relational aspect of nodes, it is not straightforward to treat the inter-layer edges the same way as intra-layer edges to aggregate information from cross-linked neighbors. Also, the representation for nodes in different layers lies in different spaces and is not compatible with a naive aggregation of information via cross-edges.

Previous works, incorporated inter-layer (cross-graph) edge information into the learning procedure by adopting either cross-graph node embedding regularization [21, 146] or clustering techniques [38]. Since, in our case, we have the same clustering constraints enforced across layers, we opt to regularize embeddings of nodes connected by cross edges to lie in the same space.

$$\mathfrak{O}_{Cross} = \sum_{r,s \in \mathcal{R}} ||O_{(r,s)}U_r - A_{(r,s)}U_s||_F^2 \tag{4.7}$$

where $O_{(r,s)} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ is a binary diagonal matrix, with $O^{i,i}_{(r,s)} = 0$ if the corresponding $A^{i}_{(r,s)}$ row-wise entries for node i are all-zero and $O^{i,i}_{(r,s)} = 1$ otherwise if the bipartite association exists.This regularization aligns the representations of nodes that are connected by cross-edges to lie in the same space and be closer to each other. Since, in practice, sparse interactions are generally modelled as cross-edges [71–73] [Refer to Section 2.2.1] to optimize the modeling cost – we do not optimize for the particular case where cross-edges are dense. Eqn 4.7 automatically handles the case. In the case of dense cross-layer edges, more entries in O, A matrices make the computation of Eqn 4.7 a costly affair. In our experiments, shown in Table 4.4, we only consider trivial cross-layer edges (node-identity based) in each graph – which accounts for a number of cross-edges proportional to $O(|\mathcal{V}|)$. Therefore, the case of dense cross-layers does not arise here. However, SSDCM is capable of handling both trivial and non-trivial cross-edges irrespective of how dense they are.

4.7.3 Joint embedding with Consensus Regularization

Having obtained rich node representations that incorporated local, global, and cross-layer structural information at every relational layer, we need a mechanism for aggregation of nodes' different relation-specific representations into a joint embedding. Since different layers may contribute differently to the end task, we use an attention mechanism to aggregate information across relations as,

$$\mathcal{J}^i_r = \frac{\exp(L_r \cdot U^i_r)}{\sum_{r' \in \mathcal{R}} \exp(L_{r'} \cdot U^i_r)}, \qquad U^i = \sum_{r \in \mathcal{R}} \mathcal{J}^i_r U^i_r \tag{4.8}$$

where L_r is the layer-specific embedding and $\mathcal{J}^i_r$ is the importance of layer r for node i. The importance score is computed by measuring the dot product similarity between the relational node embedding U^i_r and the learned layer embedding L_r.

Additionally, to obtain a consensus embedding [39], we leverage the corrupted node representations $\tilde{U}_r : r \in \mathcal{R}$ that we computed for InfoMax optimization in Eqn: 4.1. A consensus node embedding $Z \in \mathbb{R}^{|\mathcal{V}| \times d}$ is learned with a regularization strategy that minimizes the dis-agreement between Z and attention-weighted aggregated true node representations U, while maximizing the dis-agreement between combined corrupted node representations $\tilde{U}$ (re-using the same attention weights). The final consensus node embedding Z is generated as,

$$\mathfrak{O}_{Cons} = \|Z - U\|_F^2 - \|Z - \tilde{U}\|_F^2 \tag{4.9}$$

4.7.4 Semi-Supervised Deep Multiplex Clustered InfoMax

We predict labels $\hat{Y}$ for nodes using their consensus embeddings Z. We project Z into the label space using weights $W_Y \in \mathbb{R}^{d \times |\mathcal{Q}|}$ and normalize it with σ, a *softmax* or *sigmoid* activation function for multi-class and multi-label tasks respectively. The prediction function is learned by minimizing the following cross-entropy loss,

$$\mathfrak{O}_{Sup} = -\frac{1}{|\mathcal{L}|} \sum_{i \in \mathcal{L}} \sum_{q \in \mathcal{Q}} Y_{iq} \ln \hat{Y}_{iq} \tag{4.10}$$
$$\hat{Y} = \sigma(ZW_Y)$$

Finally, the overall semi-supervised learning process to obtain rich node representations that capture local-global structures in a multiplex network is obtained by jointly optimizing the equation below that optimizes different necessary components. We leverage hyper-parameters $\alpha, \beta, \gamma, \zeta, \theta$ to fine-tune the contributions of different terms.

$$\mathfrak{O} = \alpha * \mathfrak{O}_{MI} + \beta * \mathfrak{O}_{Cross} + \gamma * \mathfrak{O}_{Cons} + \zeta * \mathfrak{O}_{Clus} + \theta * \mathfrak{O}_{Sup} \tag{4.11}$$

Empirically, we find that our objective function is not very sensitive to variation in α, β values. Therefore, we fix their values as $\alpha = 1.0, \beta = 0.001$. Finally, we only tuned variables γ, ζ, θ in the above objective function to analyze the contributions of network, cluster, and label information. We discuss this further in Table 4.6 of the following section.

4.8 Evaluation Methodology

In this section, we describe the experiment settings for accessing the performance of the candidate methods. We also provide reproducibility information for the shown results and analyses.

Datasets. As mentioned in Table 4.3, we evaluate our proposed algorithm SSDCM on a variety of datasets from diverse domains, containing — both multi-class and multi-label datasets, as well as, attributed and non-attributed datasets. Here in Table 4.4, we provide the

Dataset	Layers	Nodes	Edges(Total)	Features	Labels
ACM [144]	5	7,427	2,45,36,689	767	5
DBLP [144]	4	4,057	1,79,76,710	8,920	4
SLAP [148]	6	20,419	82,07,130	2,695	15
IMDB-MC [39]	2	3,550	80,216	2,000	3
IMDB-ML [149]	3	18,352	25,05,797	1,000	9
FLICKR [38]	2	10,364	5,06,051	–	7
AMAZON [39]	3	17,857	21,94,389	2,395	5

Table 4.3 Summary Statistics of datasets

	Nodes	Layers	Node Types	Intra-Layer Relations	Edges	Features	Weighted	Directed	Multi-Class	Classes
ACM [144]	7427	5	**PAPER (P)** Author (A) Proceeding (V) Institute (I) Subject (S)	PAP PAIAP PSP PVP PP	118453 8353678 14997105 1048129 19324	767 Paper Title & Abstract	True	False	True	5 Conference (C)
DBLP [144]	4057	4	**AUTHOR (A)** Paper (P) Conference (C) Term (T)	APA APAPA APCPA APTPA	11113 40703 5000495 12924399	8920 Paper Title & Abstract	True	False	True	4 Research Field (F)
SLAP [148]	20419	6	**GENE (G)** Gene Ontology (O) Pathway (P) Compound (C) Tissue (T) Disease (D)	GPG GTG GDCDG GOG GDG GG	832924 606974 36190 6371558 14988 344496	2695 Gene Ontology Description	True	False	True	15 Gene Family (F)
IMDB-MC [39]	3550	2	**MOVIE (M)** Actor (A) Director (D)	MAM MDM	66428 13788	2000 Movie Plot & Summary	True	False	True	3 Movie Genre (G)
IMDB-ML [149]	18352	3	**MOVIE (M)** Actor (A) Director (D) Actress (E)	MAM MDM MEM	1455381 923173 127243	1000 Movie Plot & Summary	True	True	False	9 Movie Genre (G)
FLICKR [38]	10364	2	**USER (U)**	Friendship Tag-similarity	390938 115113	NA	True	True	True	7 Social Group (G)
AMAZON [39]	17857	3	**PRODUCT (P)**	Co-purchase Co-view Similar	1501401 590961 102027	2395 Product Description	True	False	True	5 Product Category (C)

Table 4.4 Detailed dataset statistics

detailed statistics of the datasets used for evaluation. We have used two versions of the IMDB

dataset, one multi-class version **IMDB-MC** as used in DMGI, and, another multi-label version **IMDB-ML** from the Column Networks (CLN) [149]. In both versions, movie features are extracted from the movie plot summary with movie genres as functional classes. We used multiplex versions of bibliographic datasets ACM and DBLP. For **ACM** [144], we extracted papers of five conferences[1] and created a multiplex network that includes layers of paper nodes connected by co-authors, similar subjects, similar venues, co-authors belonging from the same institutes, and citation relationships. Here, the task is to classify them according to the conferences as they are published. **DBLP** [144] is a multiplex network of authors. The authors are classified by their field of research-interests[2]. In both the bibliographic datasets, the terms extracted from the paper title and abstract are used as local features for the nodes under consideration. In **SLAP** [90, 148], multiple layers of interactions characterize a gene — including tissue-specific, biological pathways involved, disease associations, phylogenetic profile, gene expression, chemicals involved to treat associated diseases, etc. Each gene has ontology-related terms associated with it as attributes, and it can belong to any of the most frequently occurring fifteen gene Families (F). We have **AMAZON** [150, 39], which is originally multiplex in nature, i.e., the multiplexity is not inferred from composite relations. This network is extracted from the product review metadata of the Amazon website. Target instances, i.e., products exhibit also-bought, also-viewed, and similar-to – three layers of relations among them. Most frequently occurred terms are extracted from product titles as node features. The task is to classify the products into any product categories[3]. **FLICKR** [38] is a non-attributed multiplex social network of users (U) who belong to various communities of interest. It has a friendship layer and a tag similarity-based connection layer among the users. A user is categorized based on their membership in any of the social groups. In all the datasets mentioned in Table 4.4, cross-layer edges link two nodes in different layers if they refer to the same node.

Baselines. We chose State-Of-The-Art (SOTA) competing methods applicable to a diverse range of multi-graph settings. In Table 4.5, we summarize competing methods in terms of important aspects of a multi-graph that they are designed to capture. The compared methods can be roughly categorized into the following classes: multi-layered network-based embedding approaches — DMGC, MGCN; multiplex network embedding — DMGI; heterogeneous network embedding — HAN; multi-relational network embedding — RGCN; pooling method in multi-graph setting — GUNets.

[1] Conferences = ['KDD', 'WWW', 'SIGIR', 'SIGMOD', 'CIKM']

[2] Fields = [Data Mining (DM), Artificial Intelligence (AI), Computer Vision (CV), Natural language Processing (NLP)].

[3] Product Categories in AMAZON Multiplex Network = ['Appliances', 'Automotive', 'Patio Lawn & Garden', 'Pet Supplies', 'Home & Kitchen']

	Methods	Comparison						
		DMGC [38]	DMGI [39]	HAN [144]	MGCN [37]	RGCN [19]	GUNETS [139]	SSDCM
Properties	attributes		✓	✓	✓		✓	✓
	within-network	✓	✓	✓	✓	✓	✓	✓
	cross-network	✓		–	✓	–	–	✓
	labels		✓	✓	✓	✓	✓	✓
	global structure	✓	✓				✓	✓
	aggregation		✓	✓	✓	✓		✓

* Dash marks denote Not Applicable (NA).

Table 4.5 Coverage of multiplex network features.

In Table 4.5, we summarize competing methods in terms of important aspects of a multi-graph that they are designed to capture. These methods either lack strategies for 1) capturing *global structural information*, or 2) *aggregating* node information across different counterparts of the same node from different layers, 3) capturing useful *structures*. Even if there are global NRL methods like DMGI, DMGC, and GUNets — they either use a naive mean-pooling approach for acquiring global graph representations or an unsupervised clustering criteria/ importance pooling strategy to capture global graph structures that might not be useful given the end task is concerned. Our framework, SSDCM, differs in that we build upon a semi-supervised structure-aware version of *InfoMax* — which is first-of-its-kind to the best of our knowledge. Our objective is to learn global-structure enhanced node representations suitable for node-wise tasks capturing all aspects of multiplex graphs.

Experiment Setup. We use a random sampling strategy to split the nodes into train,

Methods	Experiment setup & hyper-parameter range
HAN [144]	l2 coefficient={0.0001, 0.0005, 0.001, 0.005}, learning rate={0.0001, 0.0005, 0.001, 0.005}, attention heads={1,2,4,8}, metapath attention dimension=128
MGCN [37]	network & label coefficient={0.01, 0.1, 1.0, 10.0}, l2 coefficient={0.0005, 0.005}, learning rate={0.0005, 0.001, 0.05, 0.01}, stacked GCNs=2
RGCN [19]	l2 coefficient={0.0005, 0.005}, learning rate={0.0005, 0.001, 0.05, 0.01}, no of bases=no of relations, number of hidden layers=2
GUNETS [139]	l2 coefficient={0.0001, 0.001}, learning rate={0.01, 0.05, 0.001, 0.0005}, depth={3, 4, 5}, pool ratio={0.2, 0.4, 0.6, 0.8}
DMGC [38]	network coefficient={1.0, 0.8, 0.6, 0.4}, cross reg.={0.2, 0.4, 0.6, 0.8}, l2 coef=0.0001, learning rate={0.0005, 0.001, 0.05, 0.01}, stacks in AutoEncoder=2
DMGI [39]	network & label coefficient={0.001, 0.01, 0.1, 1.0}, l2 coefficient={0.0001, 0.001}, learning rate={0.0001, 0.0005, 0.001, 0.005}
SSDCM	network, label & cluster coefficient={0.001, 0.01, 0.1}, l2 coefficient=0.0001, cross regularization=0.001, learning rate={0.0001, 0.0005, 0.001, 0.005}
Default to All	hidden units=64, epochs=10000, patience=20, attention heads=2, non-linearity=prelu, no of clusters=no of classes, $\varepsilon = 3.0$, GCN layers $= 2$, validation set based hyperparameter tuning, features=adjacency for non-attributed graphs, no node aggregation strategy=mean-pooling.

Table 4.6 Experiment setup & hyper-parameter range search for competing methods

validation, and test set. We choose one-third of the labeled examples as train nodes. We

keep the validation set size as half of the train set size. Thus, half of the total nodes are kept for evaluation purposes as test-set. In Table 4.6, we give the details of the hyper-parameter range search for all the competing methods — which is self-explanatory. Also, a generic experimental setup is summarized. For the methods applicable to non-attributed graphs, namely, RGCN and DMGC – we implement attributed versions. For RGCN, we customized the relational GCN to take node features as input. For DMGC that uses relation-specific autoencoders to reconstruct the layer adjacencies, we input another array of feature-specific autoencoders. The feature-based autoencoders jointly learn a common hidden node representation along with the relational autoencoders and reconstruct layers' node features. To set up attributed NRL methods for FLICKR, we leverage the layer adjacencies as node features. We simply obtain the average of layer node embeddings as final representations for the methods with no specific node embedding aggregation strategy.

4.9 Performance Analysis

We demonstrate the effectiveness of the proposed framework on four tasks, namely, node classification, node clustering, visualization, and similarity search. The details of the task-specific experiment setup, along with insights on results, are discussed in the next section. While analyzing performance, we aim to investigate the following research questions —

RQ1. How well does SSDCM perform on a variety of tasks such as node classification, node clustering, visualization, and similarity search? Can SSDCM outperform a diverse range of multi-graph NRL baselines? [SECTION 4.9.1, SECTION 4.9.2, SECTION 4.9.4]

RQ2. What are the contributions of the label-informed clusters in improving the performance of end tasks? [SECTION 4.9.1, SECTION 4.9.2, SECTION 4.9.4, SECTION 4.10.4]

RQ3. What is the novelty of node-contextualized global graph summary over a commonly shared graph summary as proposed in Eqn 4.3? Comparative performance analysis of the mentioned variants. Can visualizing discriminator weights shed more light on the novelty of the proposed graph summary? [SECTION 4.10.1, SECTION 4.10.2]

RQ4. How do various regularizers, such as cross and consensus regularizations, influence SSDCM's performance? [SECTION 4.10.3]

RQ5. What are the important observations on the comparative performance analysis of unsupervised and semi-supervised node clustering strategies? [SECTION 4.10.4.2]

4.9.1 Node Classification

For semi-supervised methods, we use the predicted labels directly to compute the node classification scores based on ground-truth. We train a logistic regression classifier on the learned node embeddings of the training data for unsupervised methods and report the performance of the predictor on the test node embeddings averaged over twenty runs. We report the test-set performance that corresponds to the best validation-set performance for a fair comparison. Micro-F1 and Macro-F1 scores are reported as node classification results in Tables [4.7, 4.8] respectively.

Micro-F1	ACM	DBLP	SLAP	FLICKR	AMAZON	IMDB-MC	IMDB-ML
DMGC	42.822	84.684	29.819	50.308	69.716	56.278	44.765
RGCN	39.118	83.514	26.914	82.69	72.957	62.542	49.802
GUNets	46.428	87.124	32.985	87.607	77.177	52.508	43.988
MGCN	52.458	87.003	29.563	91.307	84.083	63.384	48.059
HAN	77.441	85.989	30.976	89.478	83.77	62.353	47.117
DMGI	81.205	89.43	30.03	91.225	89.422	65.21	53.413
SSDCM	**88.324**	**94.988**	**33.597**	**96.261**	**92.195**	**67.796**	**54.055**

Table 4.7 Node classification results: Micro-F1 scores (%)

Macro-F1	ACM	DBLP	SLAP	FLICKR	AMAZON	IMDB-MC	IMDB-ML
DMGC	39.679	83.279	21.581	46.122	64.013	54.699	29.122
RGCN	38.665	82.86	24.119	81.47	68.323	62.17	45.285
GUNets	41.433	86.426	18.807	85.708	74.332	51.039	27.591
MGCN	46.853	85.462	25.717	91.07	82.349	62.876	38.821
HAN	78.009	85.154	25.413	89.174	82.344	61.891	35.181
DMGI	80.802	88.828	24.854	91.928	88.114	65.066	48.122
SSDCM	**88.571**	**94.681**	**28.072**	**96.147**	**91.973**	**67.803**	**51.756**

Table 4.8 Node classification results: Macro-F1 scores (%)

From Tables [4.7, 4.8], it is clear SSDCM is the best performing model on all the datasets, by a significant margin. In comparison, DMGI gives the second-best performance on most datasets except on SLAP and FLICKR (for Micro-F1 scores). GraphUNets performs competitively in SLAP (in terms of Micro-F1), replacing DMGI as the second-best scoring model. From Tables [4.7, 4.8], it is clear that SSDCM significantly outperforms DMGI in ACM, DBLP and FLICKR. SSDCM beats DMGI on an average of 3.873% points in Micro-F1 scores and by 4.158% points in Macro-F1 scores across all the datasets. To justify

SSDCM's performance improvement over DMGI's, in Sections 4.10.1, 4.10.3, we deconstruct SSDCM's architecture and compare it with DMGI's at various levels showing the usefulness of our learning objective. MGCN and HAN performed competitively when compared with SSDCM, just after DMGI. HAN's semi-supervised hierarchical attention strategy and MGCN's incorporation of layer-wise semi-supervised loss — are arguably the main reasons behind their improved performance. GraphUNets does not perform consistently well in node classification. GraphUNets being node importance-based pooling-unpooling architecture lacks adequate supervision from the structural view of multiplex data – which seems to be the principal reason for its relatively low performance. DMGC being an unsupervised method, is outperformed by the other semi-supervised counterparts.

4.9.2 Node Clustering

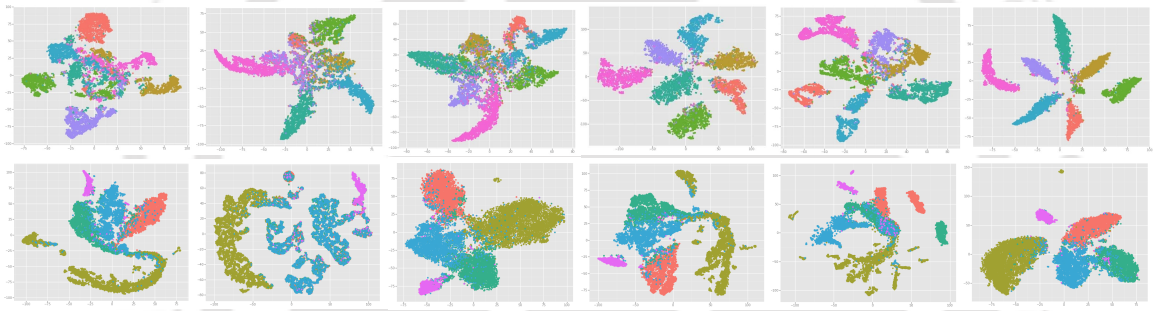
NMI-N	ACM	DBLP	SLAP	FLICKR	AMAZON	IMDB-MC	IMDB-ML
DMGC	0.421	0.532	0.245	0.488	0.468	0.185	0.076
RGCN	0.324	0.559	0.24	0.715	0.405	0.193	**0.102**
GUNets	0.65	0.742	0.251	0.758	0.519	0.108	0.036
MGCN	0.41	0.738	0.278	0.76	0.528	0.195	0.033
HAN	0.939	0.66	0.278	0.639	0.519	0.178	0.055
DMGI	0.837	0.682	0.275	0.644	0.568	0.194	0.056
SSDCM	**0.947**	**0.819**	**0.284**	**0.822**	**0.635**	**0.223**	0.085

Table 4.9 Node clustering results: NMI scores

We only cluster the test nodes to evaluate performance on the node clustering task. We give the test node embeddings to the clustering algorithm as input to predict the clusters. We run each experiment ten times and report the average scores in Table 4.9. K-Means and Fuzzy C-Means algorithms are used to predict clusters in multi-class and multi-label data, respectively. For multi-label data, we take the top q number of predicted clusters, where q is the number of classes that a node is associated with, to compare against the set of ground-truth clusters. We evaluate the obtained clusters against ground truth classes and report the Normalized Mutual Information (NMI) [151] scores. We use Overlapping NMI (ONMI) [126] for overlapping clusters to evaluate the multi-label datasets. Here we consider two kinds of clustering to demonstrate the effectiveness of our method. One is node clustering through clustering algorithms that takes final node embeddings as input. We refer to this clustering score as *NMI-N*. Another is directly predicting clusters from the cluster membership matrices learned during the optimization process and comparing it to the

ground-truth to evaluate the clustering performance. The latter score, referred to as *NMI-C*, is only applicable to SSDCM and DMGC. In Table 4.9, we see that semi-supervised methods, not surprisingly, outperform the unsupervised methods in clustering performance. DMGC performs competitively in movie networks IMDB-ML, IMDB-MC and, in SLAP. MGCN and HAN gave competitive performances among the semi-supervised counterparts, followed by DMGI, GraphUNets, and RGCN. From Table 4.9, we can see that except for IMDB-ML, SSDCM outperforms all the competing methods on the clustering task. It beats the second-best performing model by 0.037 across all datasets on average — a significant improvement. The performance improvement is the highest on the AMAZON dataset, followed by DBLP and FLICKR.

4.9.3 t-SNE Visualizations



(a) RGCN (b) GUNets (c) MGCN (d) HAN (e) DMGI (f) SSDCM

Please refer to Section Baselines for the candidate methods for which the t-SNE visualizations are plotted here. The color codes indicate functional classes (FLICKR: 7, AMAZON: 5).

Fig. 4.5 t-SNE Visualization of node embeddings on FLICKR (top), AMAZON (bottom) for all the SSL methods

We also visualize the superior clusterability of SSDCM's learned node representations for the FLICKR and AMAZON dataset in Figure 4.5 using t-Distributed Stochastic Neighbor Embedding (t-SNE) [128] visualization. The color code indicates functional classes for respective datasets. We chose the node embeddings that gave the best performance in node classification scores for all the competing methods. We can see that all the semi-supervised methods yield interpretable visualizations indicating clear inter-class separation. Among them, SSDCM obtains compact well-separated small clusters of the same class labels, which appear to be visually better separated than the rest of the methods. We see similar trends in visualization for other datasets also (not shown here).

4.9.4 Node Similarity Search

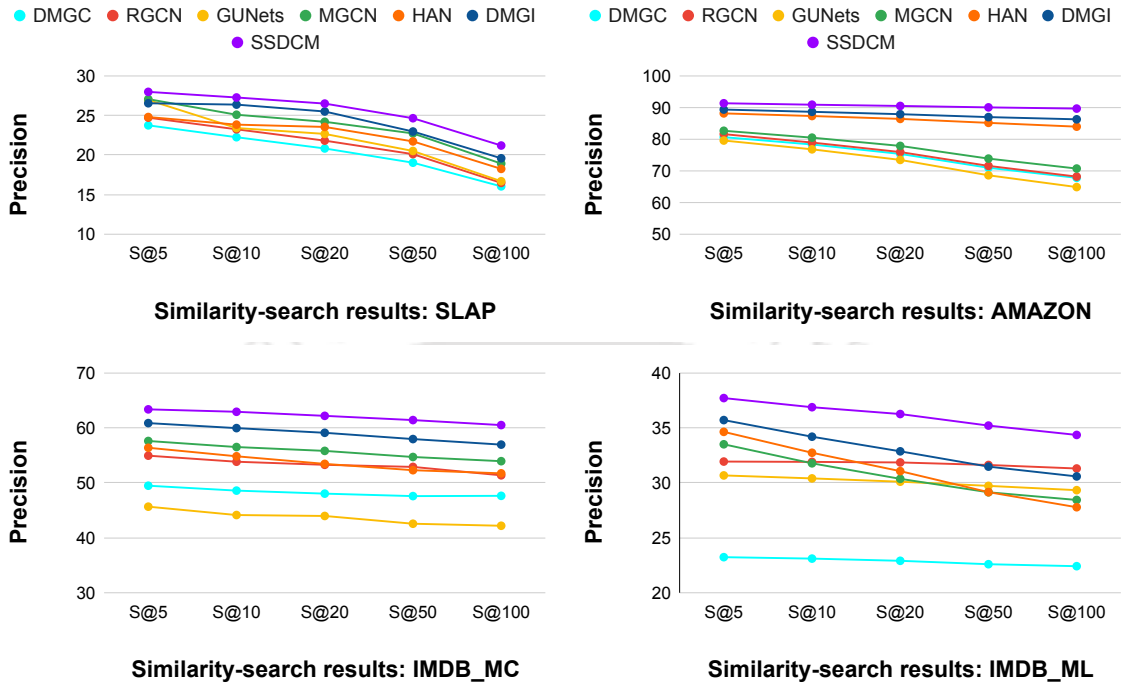


Fig. 4.6 Comparing similarity search results

In a similar setup to [39], we calculate the cosine similarity scores of embeddings among all pairs of nodes. For a query node, the rest of the nodes are ranked based on the similarity scores. We then retrieve top $K = \{5, 10, 20, 50, 100\}$ nodes to determine the fraction of retrieved nodes with the same labels as the query node, averaged by K. For multi-label graphs, instead of exact label matching, we use the Jaccard similarity to determine the relevance of the query and target nodes' label set. We compute this similarity search score for all nodes as a query and report the average. The similarity search results get a significant boost under our framework since our encoding of the SSL clusters puts nodes with similar labels together in the same cluster. Whereas DMGC's clustering criterion, DMGI's global pooling, and GUNet's node importance-based pooling criterion – do not demonstrate a similar benefit. From Tables [4.7, 4.8], we see for SLAP and two versions of IMDB movie networks the classification score of the competing methods are close. But in similarity search, we can differentiate SSDCM as the best performing model among all. DMGI is the second-best performing model in node similarity search, similar to the node classification results. GUNets and DMGC are seen to perform worse than the rest. Among the mentioned methods, DMGC and GUNets don't have any explicit learning objective to keep two nodes' embeddings closer based on label similarity – which explains the reason for their poor performance.

4.10 Ablation Study

Herein we conduct an array of drill-down experiments to shed light on the key components of our proposed SSDCM framework.

4.10.1 Novelty of cluster-based graph summary

Micro-F1 Scores	IMDB_MC	ACM	DBLP	AMAZON	FLICKR
SSDCM [global pool]	65.942	84.176	91.592	90.62	92.698
SSDCM [top-K pool]	63.908	84.218	90.683	90.34	93.714
SSDCM [SAG pool]	66.574	83.176	92.859	90.878	93.015
SSDCM [ASAP pool]	66.365	85.064	91.782	90.844	94.689
SSDCM [cluster aware graph summary]	**67.796**	**88.324**	**94.988**	**92.195**	**96.261**

Table 4.10 Novelty of cluster-based graph summary

In Table 4.10, we delve deeper into how good the cluster-aware graph summary representation (Equation 4.3) is for the universal discriminator (Equation 4.1). We consider alternative SOTA pooling methods — Top-K [139], SAG [152] and ASAP [153] for generating graph summaries in the SSDCM framework. Top-K pool realizes node importance-based pooling strategy via learning a projection vector. In comparison, the SAG pool improves upon the former by encoding structural information from graphs using GNNs. Adaptive Structure Aware Pooling (ASAP) is a new SOTA method that considers the cluster structures from graphs. It proposes a self-attentive GCN architecture *Master2Token* to learn clusters and uses a cluster fitness-based scoring strategy to pool underlying graph structures in phases[4] These pooling strategies generate a common graph summary for the whole graph, which is fed to the discriminator along with the node embeddings. On the contrary, our cluster-aware graph summary has a node's perspective, i.e., the global graph summaries vary from node to node based on their associated cluster structures. For the nodes that share membership under a common set of clusters, the structure-aware graph summaries are similar. That makes the universal discriminator more powerful for discriminating the local and global patch pair representations from the false pairs across the relations.

Here, we keep SSDCM's cluster learning component intact and use various pooling strategies to train the discriminator. The discriminator, thus, does not have any relation to the

[4]We use the Pytorch Geometric [154] library for candidate pooling methods.

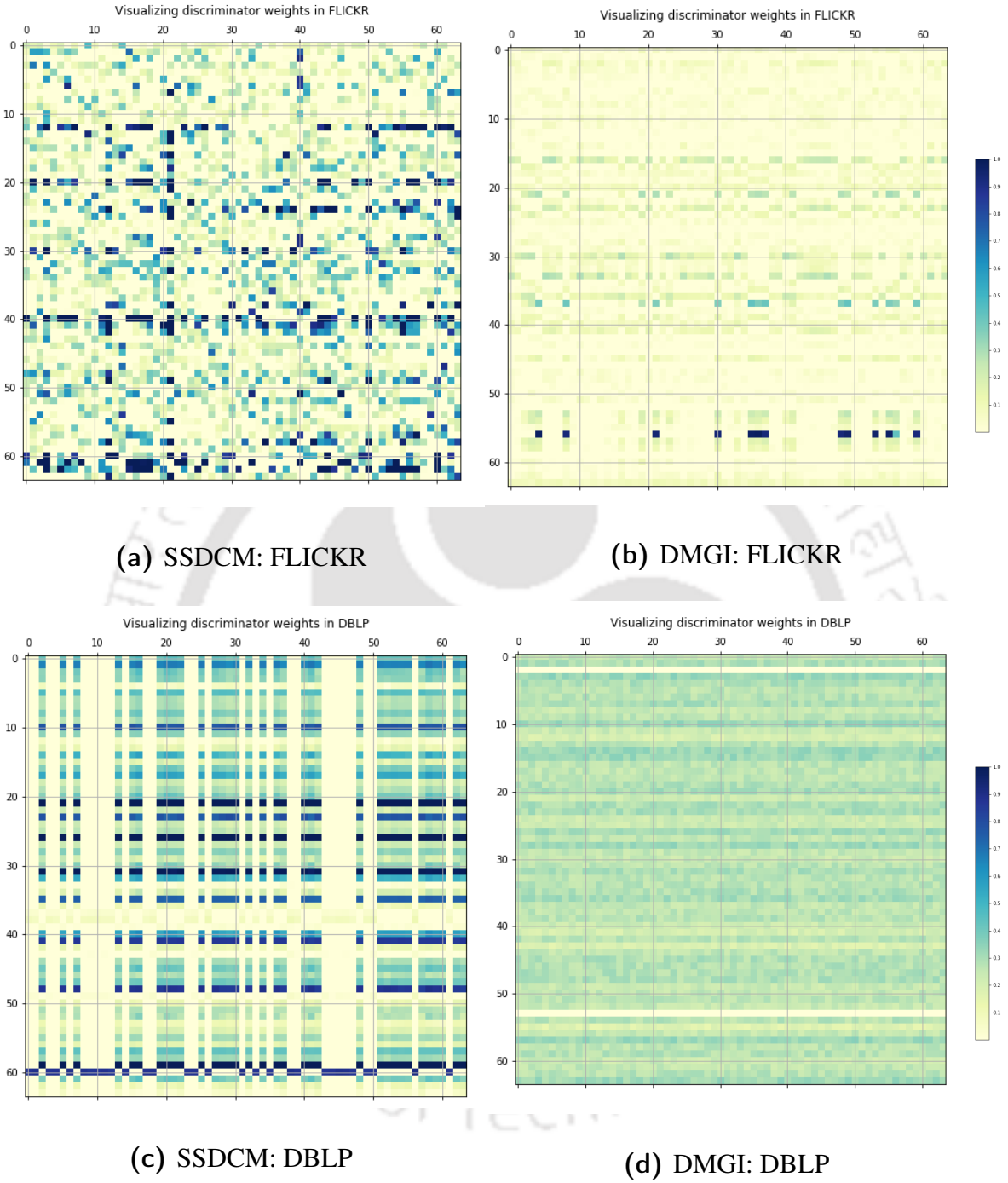
learned clusters and uses a common global summary paired with each node. In Table 4.10, we see, structure-aware pooling methods are beneficial. *We see that the pooling variants with SSDCM have better performance than DMGI's best-reported performance (Refer to Table 4.7), mainly due to learning of the clusters and using advanced pooling strategies* in place of DMGI's mean-pooling. Empirically, we observe that SSDCM with the alternative pooling variants struggle to converge consistently. However, SSDCM with a cluster-based graph summary does not suffer from similar convergence issues. *Our proposed architecture in the last row outperforms all the candidate pooling techniques significantly on every dataset, depicting the effectiveness of our cluster-aware graph summary representations.*

4.10.2 Visualizing discriminator weights

In this interesting setup, we consider the discriminator weights pertaining to the best node classification scores of SSDCM and DMGI for two candidate datasets, FLICKR and DBLP. We plot the discriminator weights in Figure 4.7, which intuitively gives the probabilistic attachment scores between the node and cluster summary representations in the latent space. DMGI uses a single mean-pool based graph summary paired with all the nodes to maximize MI via learning a discriminator. In contrast to this, we use structure-aware global contexts for every node to optimize MI across the relational layers. Thus, *the universal discriminator being learned in our framework is more powerful and discriminative as far as the task of node representation learning is concerned.* Higher similarity scores are depicted using color code varying from low to high as [Yellow–Green–Blue], as shown by the color bar beside plots. We can conclude that *SSDCM's discriminator captures better latent interaction than that of DMGI's*, as shown by DMGI's average local–global embedding attachment scores.

4.10.3 Effect of various regularizations

Here we compare the results of SSDCM without cross regularization with SSDCM to understand the influence of this factor. *We see that removing cross-edge based regularization from the layer-wise node embeddings degrades the performance of the SSDCM considerably,* especially on FLICKR and ACM. Next, we verify the usefulness of learning a final consensus node embedding from the attention-aggregated positive and corrupted node embeddings. Recall that our universal discriminator learns to discriminate between true local–global patches from the corrupted ones with the intuition that the corrupted embeddings seek to improve the discriminative power of the resulting embeddings. *We see that consensus*



(a) SSDCM: FLICKR (b) DMGI: FLICKR

(c) SSDCM: DBLP (d) DMGI: DBLP

Fig. 4.7 Visualization of discriminator weights on $(d \times d)$ x, y axes, where $d = 64$

regularization indeed plays an essential role in enriching the final node embeddings – an observation similar to DMGI's. From Table 4.11, we see that the consensus embeddings improve the performance of Micro-F1 scores by a maximum of 4.425% on FLICKR, followed

Comparison	IMDB-MC		FLICKR		ACM	
	Micro-F1	NMI-N	Micro-F1	NMI-N	Micro-F1	NMI-N
SSDCM	**67.796**	**0.22325**	**96.261**	**0.82171**	**88.324**	**0.94650**
SSDCM−cross	66.613	0.20451	94.182	0.79671	86.371	0.91611
SSDCM−cons	66.069	0.20138	91.836	0.73658	84.889	0.88139
SSDCM−(cons+cross)	64.971	0.18735	88.374	0.71629	83.961	0.85420

Table 4.11 Effect of cross and consensus regularizations. '+' and '-' signs denote augmentation or elimination of the components followed.

by 3.435%, 1.558% improvements on ACM, IMDB-MC respectively. *SSDCM−(cons+cross) gives the worst performance among all the compared variations.* The reasons behind this are self-explanatory – a) no cross-edges to align the relational node representations to each-other, b) it lacks in a discriminative capacity.

4.10.4 Analyzing clusters

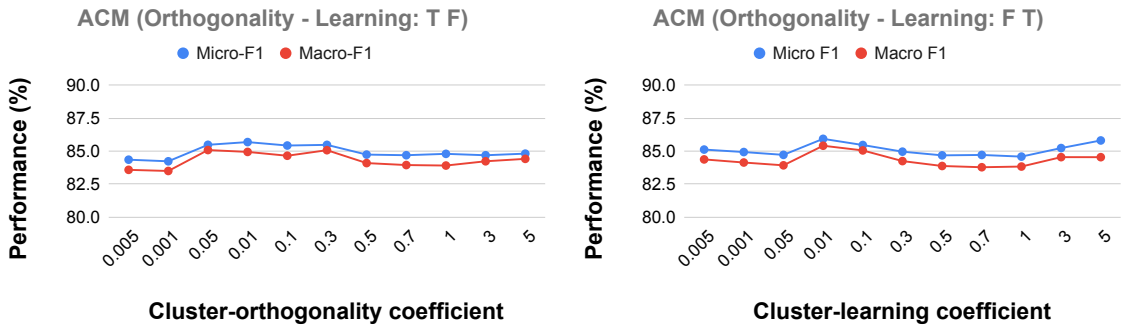


Fig. 4.8 Ablation study of cluster learning components

Variants	Micro F1	Macro F1	NMI-N	NMI-C
SSDCM (OL : FT)	85.584	84.83	0.889	0.518
SSDCM (OL : TF)	84.795	83.907	0.868	0.391
SSDCM (OL : TT)	**88.324**	**88.571**	**0.947**	**0.651**

Table 4.12 Impact of various cluster learning components

Symbol meanings – A: cluster assignment, L: cluster learning, O: cluster orthogonality. T: True, F: False – denotes absence or presence of respective terms.

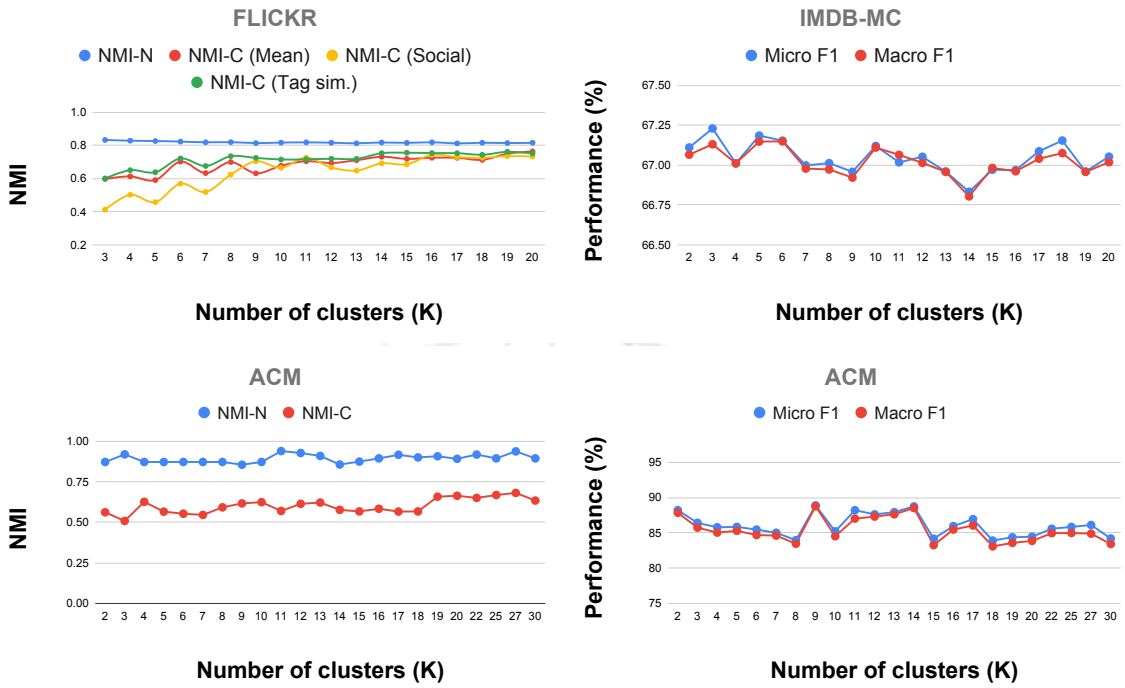
In Table 4.12, we study the impact of cluster-related terms on the end-task performances by removing the relevant terms in two binary combinations. In OL: FT and OL: TF configurations, we remove the cluster orthogonality term and the semi-supervised cluster learning term, respectively. Removing the cluster learning term significantly impacts the NMI N and C scores by reducing the performance by 0.079 and 0.26 points. This configuration moderately affects the F1 scores. Removing the orthogonality term affects the classification performances with $2.74\%, 3.771\%$ reductions in Micro and Macro F1 scores. These reductions are less than the performance drops gotten from removing the cluster learning term in the case of F1 scores but still play a significant role. *The cluster learning term is seen to be more useful than the cluster orthogonality term for learning the cluster membership matrix.*

In Figure 4.8, we consider two possible combinations, namely, node-to-cluster assignment probability score computation in Eqn 4.4 — *i*) with cluster learning (L) as in Eqn 4.6 and *ii*) with cluster orthogonality (O) constraint as in Eqn 4.5 as LO: FT and TF (T: True, F: False), for dissecting the cluster learning objective. We perform a range search to see under which settings the best classification performance is achieved by varying a particular cluster-related term in a range while removing or keeping the rest of the terms intact. The terms are varied in range of $\in \{0.005, 0.001, 0.05, 0.01, 0.1, 0.3, 0.5, 0.7, 1, 3, 5\}$. In FT configuration, cluster orthogonality is varied in the absence of the cluster learning term. It gives best performance for values $\in \{0.05, 0.3\}$. Over a higher range of values, the performances become less fluctuating. In TF configuration, cluster learning is varied in the absence of cluster orthogonality. At 0.01, it gives the best performance in terms of Micro and Macro F1 for ACM. Again, an upward trend in performances can be seen for values $\in [1-5]$.

4.10.4.1 Varying number of clusters

Here we study SSDCM's sensitivity towards varying the number of clusters K. We also verify whether there is a need to learn the cluster structures at all. We take the optimal hyper-parameter combination and vary the number of clusters in the range $[2-20]$ and $[2-30]$ for FLICKR, IMDB-MC, and ACM, respectively. *Compared to DMGI's best performance scores, clear differences can be seen in Figure 4.9 for SSDCM that speaks to the effectiveness of learning clusters to enrich node embeddings.*

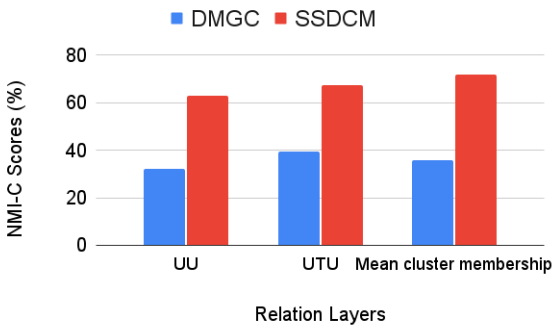
We plot the NMI-N and NMI-C scores (mean and layer-wise cluster memberships) while varying K for FLICKR. We see less perturbation in NMI-N scores than in NMI-C scores here. As K goes higher, the layer-wise and mean cluster membership-based NMI scores increase before flattening at $K = 20$. For IMDB-MC, We can see Micro F1 scores are



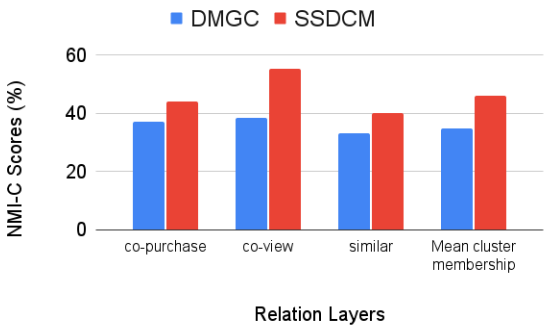
Number of clusters K is varied for FLICKR, IMDB-MC, and ACM. i) Micro, Macro F1 scores (on the right), ii) NMI using node embeddings and cluster memberships (on the left) are plotted.
Best performances of DMGI (no cluster learning) are – a) for FLICKR, NMI-N: 0.644, b) for IMDB-MC, Micro-F1: 65.210, Macro-F1: 65.066, and c) for ACM, Micro-F1: 81.205, Macro-F1: 80.802, NMI-N: 0.837

Fig. 4.9 Varying number of clusters

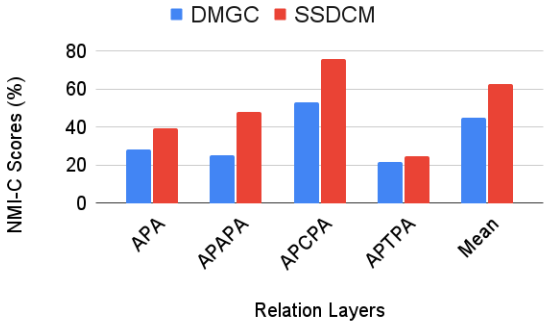
best at $K = Q = 3$, i.e., when the number of classes and clusters are the same. For ACM, varying K improves Micro and Macro F1 scores at $K \in \{2,3\} < (Q = 5)$, i.e., when SSDCM learns high-level clusters. Even when $K \geq Q$, i.e., when SSDCM learns small clusters of the same class data. We see a gradual improvement in both the NMI scores for ACM when $k \in [9-30]$. NMI-N and NMI-C tend to give different NMI scores. The possible interpretation of this performance difference lies in the fact that – in NMI-N, the K-Means algorithm is applied to the node embeddings of considerable hidden dimensions $(d = 64)$, and the NMI scores are calculated for ground truth clusters. Whereas, cluster memberships H are of comparatively low dimensions (K), and in NMI-C, we directly use the learned cluster membership probabilities to derive the NMI scores.



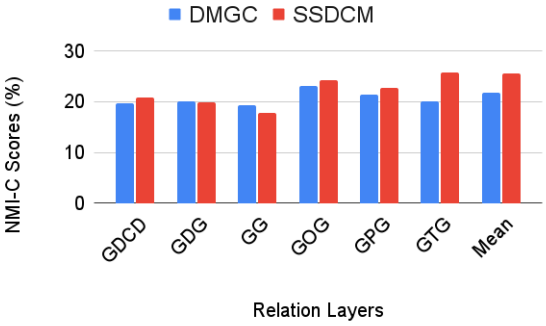
(a) DMGC vs SSDCM: FLICKR



(b) DMGC vs SSDCM: AMAZON



(c) DMGC vs SSDCM: DBLP



(d) DMGC vs SSDCM: SLAP

Fig. 4.11 Relation wise clustering (NMI-C) scores

4.10.4.2 Comparing relation-wise clustering performance

In Figure 4.11, we compare unsupervised and semi-supervised node clustering strategies in a relation-wise manner. Figure 4.11 compares the clustering scores obtained from the layer-wise learned node-community membership matrix $H_r : r \in \mathcal{R}$ for the unsupervised method DMGC and semi-supervised SSDCM. Besides the relational layers, we also compare the clustering performance obtained from the MEAN community matrix $H_{\text{MEAN}} = (\frac{1}{|\mathcal{R}|}\sum_{r\in\mathcal{R}} H_r)$ for the considered methods. This is because we do not employ any effective aggregation strategy for the learned relational clusters. Comparison is made for four candidate datasets – FLICKR, AMAZON, DBLP, and SLAP. We observe that, except for SLAP, SSDCM outperforms DMGC by significant margins in terms of NMI-C scores. However, in SLAP, DMGC is seen to perform competitively in 2 out of 6 relations, but, outperformed by SSDCM in the rest of the relational layers, as well as in H_{MEAN}. *We attribute the performance gain of SSDCM over DMGC to our label-informed cluster learning strategy that preserves label-*

correlations beside the local network contexts while logically grouping the nodes in each layer.

Summary

In this study, we propose a semi-supervised framework for representation learning in multiplex networks. This framework incorporates a unique InfoMax-based learning strategy to maximize the MI between local and contextualized global graph summaries for effective joint modeling of nodes and clusters. Further, we use the cross-layer links to impose further regularization of the embeddings across the various layers of the multiplex graph. Our novel approach, dubbed SSDCM, improves over the state-of-the-art over a wide range of experimental settings and four distinct downstream tasks, namely, classification, clustering, visualization, and similarity search, demonstrating the proposed framework's overall effectiveness. In the future, we hope to extend this work in a couple of ways. First, we hope to improve the scalability of the approach – perhaps by leveraging a graph coarsening and refinement strategy [155] within SSDCM. Second, we propose to see if the ideas we have presented can be generalized for other types of multi-layer graphs (i.e., not just multiplex networks).

Chapter 5

Structure-Aware Network Representation Learning on Heterogeneous Graphs

In this work, we present a novel approach for link prediction on heterogeneous networks – networks that accommodate multiple types of nodes as well as multiple types of relations among the nodes. Specifically, we propose a multi-view network representation learning framework to incorporate structural intuitions from the underlying graph and enrich the relational representations for link prediction. The method relies on the metapath view, the community view, and the subgraph view between a source and target node pair whose linkage is to be predicted. Furthermore, our proposed model leverages a relation-aware attention mechanism to aggregate the candidate contexts in a principled way. Empirically, we demonstrate that the proposed architecture outperforms state-of-the-art transductive and inductive methods in link prediction by a significant margin. A detailed ablation study and attention weight visualizations suggest that the chosen views are complementary and useful to predict links robustly.

5.1 Introduction

Heterogeneous graphs pose unique challenges for the task of Link Prediction (LP), such as — i) node, edge heterogeneity, ii) (under/ over) representations of edges and relational paths due to the type-imbalanced existence of nodes and edges, and, iii) link sparsity. Relying only on the local contexts surrounding a node-pair is often not sufficient to justify the formation of an edge between the end nodes. A rather useful strategy is to analyze various higher-

order associations at multiple scales between the end nodes to evidentially justify the direct interaction between the nodes. Also, distinguishing various graph structure-based cues, understanding their roles, and using them in a principled way can provide more support to the inference algorithm in predicting links, especially in challenging scenarios.

Among the higher-order multi-scale graph structures, metapaths and communities are intuitively useful in characterizing various connectivity patterns between the end nodes. Metapaths are a schematic representation of paths and indicate various composite semantic associations between two nodes. It has been shown that they are helpful for inference [91, 12, 47]. Higher-order associations between the end nodes of a link can also be captured by the community memberships of those nodes. This aids directly in link prediction, especially when the links are sparse.

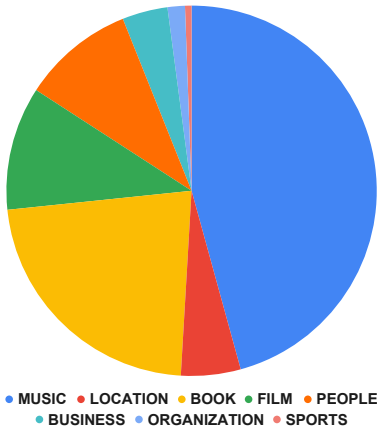
Existing approaches often rely on Network Representation Learning (NRL) methods that learn network embeddings by capturing the local neighborhood contexts [148, 144, 156–158, 19, 45, 47, 159, 26, 48], and subsequently use such representations for link prediction. An alternative class of methods rely on metapath-aware random-walks [12, 91, 46–48], subgraphs [26], and, metagraphs [49, 50] to generate a triplet representation (source, relation, target), subsequently levered for link prediction.

A few works also explore designing expressive Graph Neural Networks (GNNs) [24–26] by learning only from the common subgraph between the end nodes for various link prediction scenarios. They demonstrate that learning from *enclosing-subgraph* can approximately encapsulate all the topological features that support the formation of a link between two nodes. However, these approaches *do not learn explicitly from the network structures or differentiate contributions of various unique structural cues*. The tree-like message passing mechanism is a well-known bottleneck of GNNs and causes the fusing of various structural information from the neighborhoods.

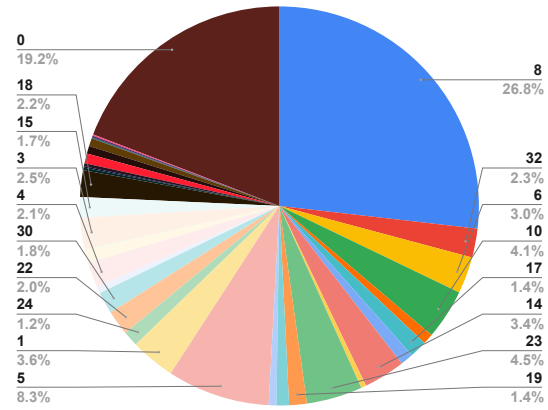
Finally, the role of communities in predicting links has been investigated in traditional network science research works [160–163]. These methods rely on re-designing topological link-prediction metrics to include community information obtained from simultaneous community discovery. However, to the best of our knowledge, no work in recent times has considered communities to obtain a global view of an edge (source, relation, target).

Hence, in this work, we explore and incorporate various graph structure-based cues and analyze their ability to mitigate the challenges (refer to the following section) typically exhibited in *Heterogeneous Information Networks* (HINs) for the task of link prediction.

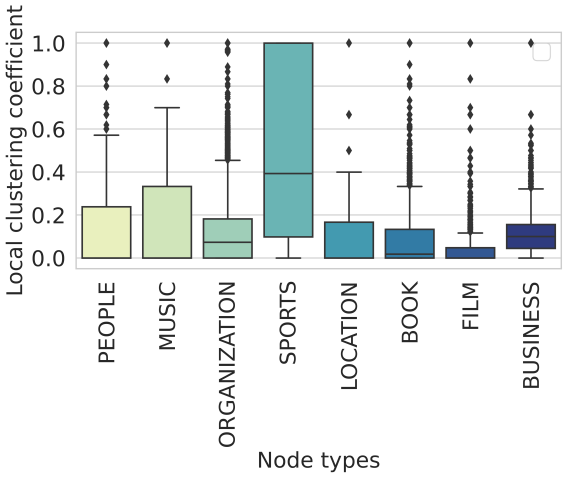
5.2 Challenges



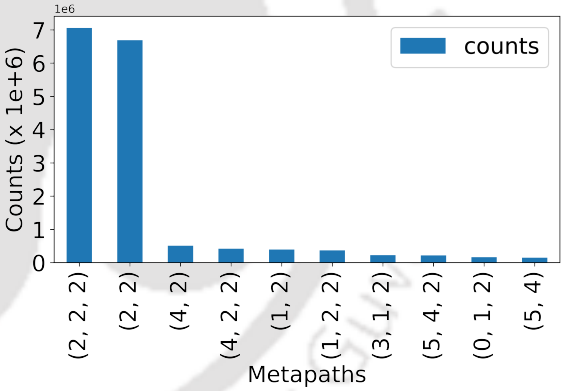
(a) Node heterogeneity in FreeBase.



(b) Edge heterogeneity in FreeBase. Edges of 36 types are identified with numbers.



(c) Node-type specific structural diversity in FreeBase.



(d) (Over/ under) representations of metapaths in PubMed.

Fig. 5.1 Challenges in HINs [Please refer to Table 5.2 for dataset description]

Having motivated the need to use various graph structure-based cues to improve link prediction performance in HINs, here we discuss *challenges* [164, 165, 4] typically exhibited in heterogeneous graphs, especially in the context of the link prediction task.

Underlying heterogeneity Figures 5.1a, 5.1b show the underlying node and edge type heterogeneity in the graph. Also, the nodes and edges have inherent type imbalance, i.e., nodes and edges of some types occur very frequently than the rest. Thus, the learning models, which also encompass the type information, get naturally biased towards the highly visible types of nodes and edges. This results in overfitting on the highly visible node and edge types, and, underfitting on rare types leading to less generalization capability in heterogeneous NRL models.

Type-specific structural diversity In HINs, nodes of various types tend to connect to the rest of the graph very differently. The structural diversities involving various node types can be significant. Figure 5.1c depicts such a scenario. Here we plot the distribution of local clustering coefficients of each node type. The local clustering coefficient [166] of a node quantifies the tendency of nodes to form groups in terms of how close their neighbors are to being a clique (complete graph). We observe that the boxplots are of varying variances with medians located at very different positions – which signifies the diverse tendencies of nodes to connect to the rest of the graph.

(Under/ Over)-representations of relation types and relational paths Due to the type-imbalanced existence of edges, the relational paths extracted from the random exploration of a HIN often see more frequent occurrences of certain relational paths than the rest. The models that operate on learning representations of walks or relational paths often suffer from the issue of (over/ under) representations of the relational paths. Figure 5.1d depicts such a scenario that occurs in PubMed. We plot top-10 frequent metapaths of length ≤ 3. Upon random exploration of PubMed, metapaths involving relation type 2 (DISEASE-CAUSING-DISEASE) occur very frequently, and metapaths $(2,2,2),(2,2)$ are over-represented than the rest of the metapaths involving other node types.

Link sparsity Just like homogeneous and multiplex graphs in our previous chapters, HINs also suffer from link sparsity issues. The link prediction models suffer from less generalizability due to the fewer observed edges of different types.

5.3 Motivation

This chapter proposes a multi-view NRL framework based on subgraph, metapath, and community views for link prediction in heterogeneous graphs. In this section, we explain why it is worthwhile to consider relational paths and communities for predicting links besides the popularly used subgraph view. First, we detail on the views considered. Next, we

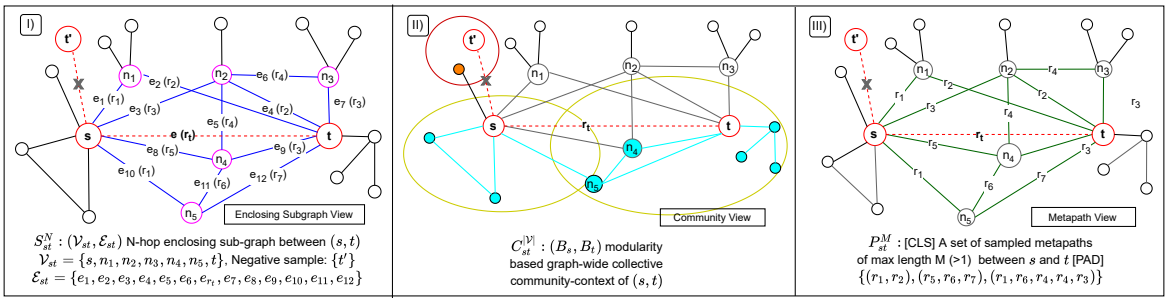


Fig. 5.2 Various views to contextualize a relation triple. [Please refer to Section 2.3.2 for view definitions. Highlights denote different contexts, I) Blue, Purple: Subgraph, II) Cyan, Yellow: Community, III) Green: Metapaths.]

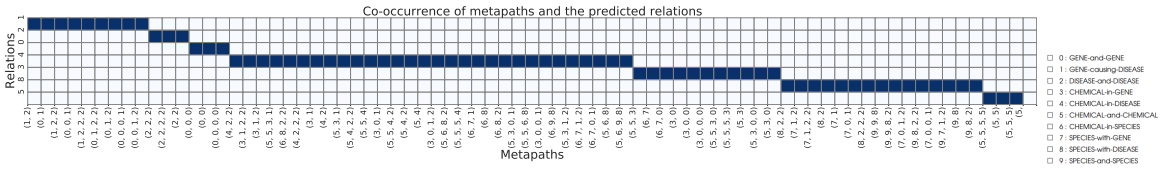
empirically provide some insights on the structural cues that the considered views offer, supporting the formation of a targeted link.

Figure 5.2 illustrates the views for an input relational triple (s, r_t, t). In the *Enclosing Subgraph View*, the common neighborhood of the end nodes is considered, which includes vertex set $\mathcal{V}_{st}$, edge-set $\mathcal{E}_{st}$ and the underlying relation semantics $\mathcal{R}$ associated with the edges. In the *Community View*, the global connectivity patterns of the end nodes are considered based on the ground-truth modularity of the graph. This view imposes higher-order associations of the candidate nodes to the rest of the nodes in the graph based on the edge-formation characteristics of the underlying graph. In the *Metapath View*, a set of sampled relational paths that represent the target relation r_t are considered. The aim is to consider various views of the connectivity pattern exhibited between the end nodes.

As illustrated in Figure 5.2, the proposed candidate views offer very different cues to predict a relation between a (source, target) node-pair. There has been no systematic study to understand the effectiveness and complementarity of each view, and to devise aggregation strategies for combining the views to predict links. Also, no study so far has explored the *community-view* for the task of link prediction in heterogeneous graphs. We elaborate more on this while discussing the usefulness of the views considered and proposing a novel view aggregation strategy to combine them.

5.3.1 Usefulness of metapaths

In a case study shown in Figure 5.3, we plot the co-occurrence characteristics of $1-$hop relation and various metapaths (of length > 1) between the same set of (source, target) node



*Various relations are identified with numbers here. Interpretations of the relations are enlisted on the Right. Interpretations of used color-codes — Dark Blue: co-occurrence, Light Blue: no co-occurrence.

Fig. 5.3 Co-occurrence of predicted relations (y-axis) and sampled metapaths (x-axis) in PubMed. [Please refer to Table 5.2 for dataset description]

pairs for the PubMed dataset. To do this, we explore the HIN using a random walker and slide a dynamic window up to a fixed size over the paths to extract all possible composite relational associations between every node pair. We observe that separate sets of metapaths co-occur with various 1-hop relations, and there is no-overlapping occurrence. For example, complex semantic associations such as — *i) similar species causing similar diseases via common genes* $(9,7,1,2)$ are found to co-occur with the relation *SPECIES-with-DISEASE* $(r=8)$, *ii) chemicals found in certain species are useful in treating certain diseases that affect genes similar to that of the species's* $(6,7,0,1)$ are found to co-occur with the relation *CHEMICAL-with-DISEASE* $(r=4)$. Therefore, without looking at the neighborhood node identities, we can predict links with a high probability between a (source, target) node pair by only looking at the relevant higher-order semantic associations that act as logical evidence to deduce the direct relation.

5.3.2 Usefulness of communities

In an interesting case-study shown in Table 5.1, we consider the train-level heterogeneous graph of ACM and discover communities of nodes via *modularity maximization* algorithm [61]. Now, given a test-triple $\big(s(\text{PAPER}), r_t(\text{HAS-TERM}), t(\text{TERM})\big)$, we enlist all the participating nodes in the $3-$hop surrounding contexts of (s,t) and in the associated metapaths (up to length-5) between (s,t). We remove these nodes from the node-set comprising the members of the target node's t associated community. This gives us all the nodes beyond the subgraph and relational path context of (s,t) from t's community. Now, we check the collective relevance of (s,t) with t's community members and try to understand whether nodes in this community provide any cues to predict the target relation. To do so, we select nodes (in the second row of Table 5.1) from t's community, which are highly similar to both test PAPER and TERM nodes (in the first row of Table 5.1) based on the cosine-similarity of input node features. We observe that, for the candidate test-triple in the first row relevant terms in the second row — *{parameters, probabilistic, robust}*; authors with aligned research interests

Paper: Adaptive event detection with time-varying poisson processes **Term: Learning**
Paper: A bayesian mixture model with linear regression mixing proportions **Paper:** A new efficient probabilistic model for mining labeled ordered trees **Paper:** Community evolution in dynamic multi-mode networks **Paper:** Fast logistic regression for text categorization with variable-length n-grams **Paper:** Reverse testing: an efficient framework to select amongst classifiers under sample selection bias **Paper:** Statistical entity-topic models **Author:** Indrajit Bhattacharya {Machine Learning, Probabilistic Topic Models, Bayesian Nonparametrics, Knowledge Graphs} **Author:** Lise Getoor {Machine Learning, Reasoning Under Uncertainty, Graph & Network Analysis, Information Integration} **Author:** Petros Drineas {Randomized Algorithms, Numerical Linear Algebra} **Term: parameters, probabilistic, unsupervised, series, statistical, significantly, mining, robust, normal, process, partial, estimation, relevance**

Table 5.1 Explainability of formation of a relation (PAPER $\xrightarrow{\text{HAS-TERM}}$ TERM) influenced by community members in ACM [Please refer to Table 5.2 for dataset description]

like *{Bayesian Nonparametrics, Probabilistic Topic Models, Reasoning Under Uncertainty}*; papers on aligned topics *{probabilistic model, statistical model, efficient framework}* are selected. The retrieved nodes' similarity with both the test PAPER and TERM nodes provide strong evidences for the direct relation {HAS-TERM} between them.

5.3.3 Effective aggregation of views to contextualize a triple

We also propose a novel attention mechanism to aggregate the (subgraph, metapath, and community) views based on the target relation. Our modeling intuition is based on the observation that neighboring contexts, metapaths, and community associations are more beneficial for predicting links of a certain kind than others. For example, given the task of predicting the subject of a paper in a bibliographic HIN, the view-contexts based on co-authorship information might be more beneficial than the view-contexts involving citation information; or the community-view of relevant terms for that subject might be more beneficial than complexly correlated metapaths. Again, there are existing challenges [164, 165, 4] in HINs as explained in Section 5.2. Our proposed framework mitigates the mentioned issues by using various complementary views or a combination of them. Section 5.9.4 empirically establishes the robustness of the proposed framework and explains the contributions of chosen views in predicting the plausibility of relations.

5.4 Research Objective

The objective of this study is to understand the roles of various structural cues present in the underlying graph at a multi-scale to support the formation of links in heterogeneous graphs. Towards this goal, we consider semantic contexts for the potential edge at multiple scales, such as — i) finer metapath level, ii) coarser local-neighborhood level, and ii) global graph level. Further, strategically combining the contextual information will give us an enriched view embedding that can optimally support inferring missing and/or future links. The design goals for this multi-view link prediction framework should also consider the challenges present in the graphs as discussed in Section 5.2.

5.5 Contributions

In this work, our contributions are as follows,

Multi-view NRL Framework for LP on Heterogeneous Graphs We propose a multi-view learning framework called *Multi-View Heterogeneous Relation Embedding (MV-HRE)* that advocates for incorporating two additional non-trivial views based on metapath and community contexts between a (source, target) node-pair, alongside the popularly used neighborhood context, for predicting the plausibility of a link between the nodes.

Community-view of an edge for Link Prediction To this end, we propose a novel *community view* for an edge-triple which gives a global context for that triple.

Fine-grained Attention Mechanism for View Aggregation Moreover, we aggregate the candidate views based on our fine-grained relation-aware attention mechanism.

The main advantage of MV-HRE lies in its flexibility to employ any of the candidate contexts or a combination of them, depending on challenges faced in the underlying graph, which allows it to predict links robustly. Empirically, MV-HRE obtains excellent performance for both transductive and inductive link prediction tasks in several challenging experimental setups. Our results and case studies indicate that the chosen views are complementary and useful with regard to the target relation to be predicted, thus boosting the link prediction performance. MV-HRE significantly outperforms 8 competing methods including the most recent *Heterogeneous Graph Benchmark (HGB)* [4] across 11 datasets.

5.6 Literature Review

In this section, we discuss in detail research efforts so far in learning network representations from HINs for the task of link prediction. Figure 5.4 gives us an overview of various NRL models across different paradigms that exist for predicting links on heterogeneous graphs. There exist — proximity-based models [167–170] that preserve chosen network proximities; Random walk based methods [12, 91, 27, 171] which learn network embeddings by optimizing skip-gram-based loss from generated random walks; GNN-based models [19, 144, 46, 45, 159, 47, 4] to learn neighborhood-based representations using (refer to following sub-section for details); miscellaneous methods [49, 50, 172] that learn from pre-defined structures such as aspects, metagraphs, hyperedges (multi-arity relations) exhibited in HINs; optimization-based relation learning approaches [173–178] that explicitly model the relation types of edges via parametric algebraic operators.

5.6.1 Context-based link prediction models

Now, we briefly review the learning (GNN and other paradigms) and non-learning models that consider diverse graph-based contexts for link prediction, especially those related to heterogeneous graphs.

5.6.1.1 Local contexts for link prediction

Existing approaches to link prediction in heterogeneous graphs have explored a variety of local contexts, such as, *i)* surrounding heterogeneous neighborhood (relation-based, metapath-based) [19, 144, 45, 47]; *ii)* enclosing subgraph which is common to both (source, target) node neighborhoods [26]; *iii)* aspect – a subgraph of HIN schema with precise semantic meaning [49], metagraph (synonymous to aspect) [50]; *iv)* random-walks (relation-based, metapath-based) [12, 91, 46–48] etc. Models are either designed in a way to directly learn relational triple embeddings and optimize for the link prediction error in an end-to-end fashion. Alternatively, models learn node embeddings optimized for a particular task (say node classification, clustering, graph-reconstruction) and fuses the end node embeddings (optional relational embeddings, if any) using popular triplet scoring methods — *dot product*, *DisMult*, *Bi-linear*, or any neural (non-linear) projection layer capturing all possible embedding interactions in the latent space. To this end, we also distinguish the local contexts based on scales – finer and coarser. Walk and metapath based NRL models such as Metapath2Vec [12],

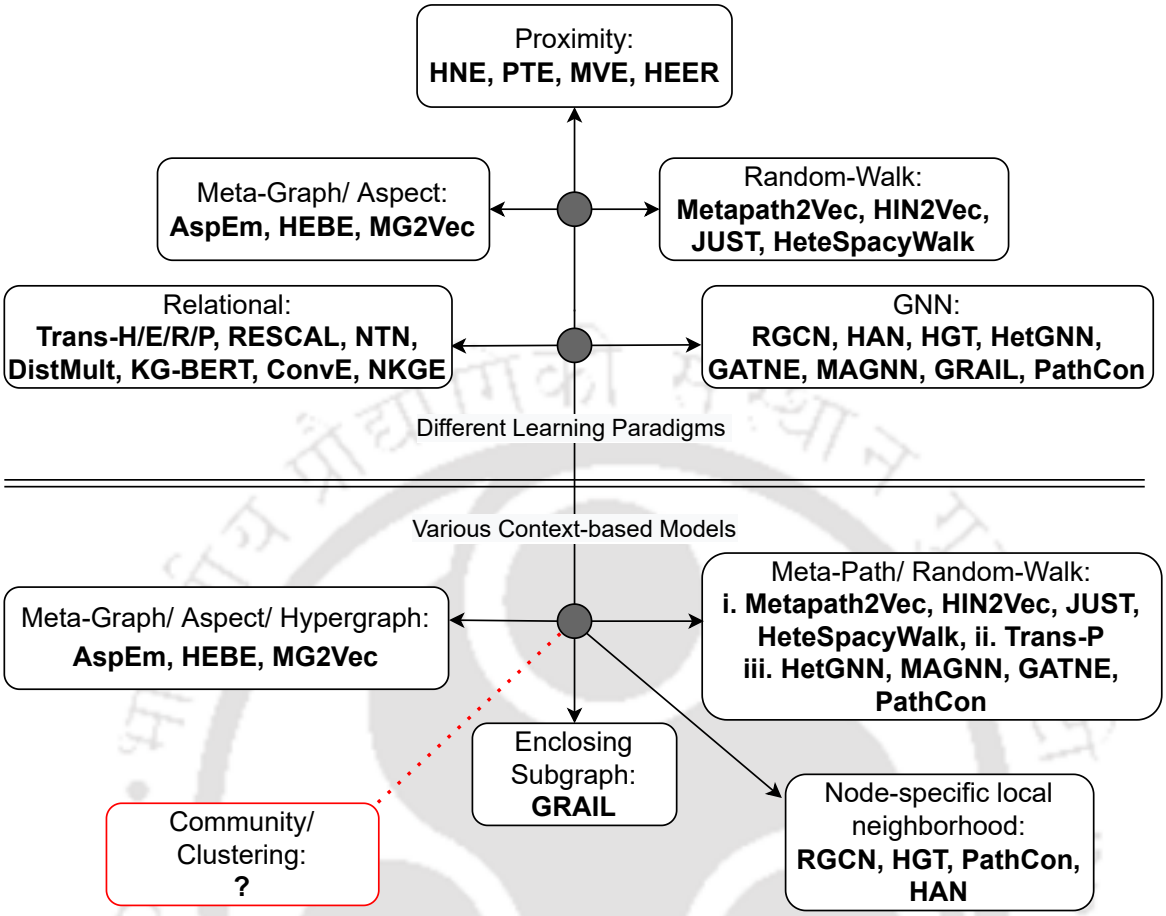


Fig. 5.4 Overview of research for LP on heterogeneous graphs using NRL

Hin2Vec [91], JUST [27], HeteySpacyWalk [171] – learn the node embeddings based on finer scales in HINs. Whereas models like RGCN [19], HAN [144], HGT [45], GRAIL [26], AspEm [49], HEBE [172], Metagraph2Vec [50] learn network representations based on local neighborhood structures, or based on aspects/hyper-edges – on a scale coarser than paths. A few methods such as PathCon [48], MAGNN [47] learn from varying scales, i.e., local neighborhoods and relational paths. However, none of the shown methods in Figure 5.4 learns the likelihood of an edge on a graph-wide global scale.

5.6.1.2 Heterogeneous Link prediction using Graph Neural Networks

Research efforts in designing Heterogeneous GNN have adopted diverse learning strategies such as, multi-relational convolution with regularizations to mitigate over/under-fitting [19],

inter-intra semantic attention based node aggregation [144], attention learning with bi-lstm based contextual encoding of random-walk generated heterogeneous neighbors [46], meta-relation based attentive convolution with subgraph batching and temporal encoding [45], intra-inter metapath aggregated convolution to facilitate both relation and metapath based message passing [47], relation-aware convolution on enclosing subgraphs with double radius node labeling scheme [26], edge type-based convolution with joint learning of path-contexts [48].

On the homogeneous NRL for LP front, a few works [24, 25] have designed expressive Graph Neural Networks that can learn and approximate popular topological heuristics, such as, $1-$hop heuristics common neighbor (CN), preferential attachment (PA); $2-$hop heuristics Adamic-Adar (AA), resource allocation (RA); higher-hop heuristics Katz index, rooted PageRank (PR), SimRank (SR) based on the notion of learning a $\gamma-$decaying heuristic using GNN. Such a class of GNNs operates on enclosed subgraphs extracted around a link of interest. By employing a special node labeling strategy *Double-Radius Node Labelling scheme (DNRL)*, the GNN can distinguish the structural roles of nodes within the subgraph. The learned GNNs are expressive since they learn structural node representations. In the domain of knowledge graphs, GRAIL [26] also uses the enclosing subgraph-based graph convolution with the node-labeling strategy to learn edge embeddings for inductive link prediction. GRAIL also claims to encode and recover relational paths by distinguishing nodes' structural roles in the enclosed subgraphs. However, these approaches do not *learn explicitly from the network structures or differentiate contributions of various unique structural cues*. The tree-like message passing mechanism is a well-known bottleneck of GNNs and causes the fusing of various structural information from the neighborhoods.

5.6.1.3 Global contexts for link prediction

Research efforts have been made to understand the role of communities in aiding link prediction tasks. For homogeneous graphs, a few works [160–163] experiment with various topological metric-based features in a non-representation learning setup to improve the link prediction performance. In these works, community detection methods such as Spectral, Louvain, and Infomap partitioning are applied to the networks, and the node similarity measurements are redefined to account for the detected community structures. A few generative models [179, 180] propose to use Bayesian Modeling to learn a triple scoring function that includes membership distribution of nodes based on network paths and class-labels for supervision to capture protein-protein interactions in PPI networks. However, the neighborhoods, features, incorporating, and capturing correlations among paths are ignored

here. [181] uses dendrogram-based hierarchical clustering to predict missing links. However, to the best of our knowledge, no work in recent times has considered community or clusters as global information to optimize for link prediction objectives in HINs.

5.6.1.4 Research gaps in learning from metapaths

A number of works have incorporated metapath information in network representation learning setup aimed at link prediction. But often, the metapath modeling approaches are implicit [26], are overly simple (and can not capture complex correlations among candidate paths) [48], are not aimed at binary link prediction [48], do not effectively aggregate learned path representations with the neighborhood-context [91], or have limited learning capacity as they depend on domain-knowledge to select metapaths [12, 47, 182]. We implement a *Transformer Architecture* [183–185] for HINs to learn metapath-sequence representations and address the above issues in our metapath learning module elaborated in Sections 5.7.2, 5.7.4.

5.6.1.5 Research gaps in learning from communities

Past research studies the effect of incorporating communities for predicting links in homogeneous graphs, mainly using various topological metrics in a non-representation learning setup [163, 160–162]. However, there is no significant work in the network representation learning space for heterogeneous graphs that considers the community information and its role in predicting a link between the end nodes. To the best of our knowledge, we are the first to propose a novel *community-view* for link prediction in HINs by simultaneously learning communities while optimizing for link prediction loss in a network representation learning setup. Empirical results in Sections 5.9.3, 5.9.4 strongly suggest the usefulness of this *community-view* for inferring links.

5.6.2 Research Gaps: Summary

If we are to summarize *the research gaps that exist in learning structure-aware network representations for link prediction* on heterogeneous graphs, the following points give us important glimpses,

- Most of the studies learn from the local neighborhood contexts for the task of link prediction.

- Expressive GNNs for structural link prediction suffers from the bottleneck of information fusion and morphing from the neighboring nodes since they employ *tree-like message passing mechanism*.
- Very few attempts learn from the relational path or metapath information between the source and target nodes for inferring links.
- There exists no study to investigate the contributions of source and target node's community membership to predict links!
- There exists no study to systematically investigate and incorporate various views exhibited in HINs for predicting links.

5.7 Proposed Framework: Multi-View Heterogeneous Relation Embedding (MV-HRE)

In this section, we explain this framework in a step-by-step manner. First, we obtain the candidate views — subgraph-view ($\mathcal{S}$), metapath-view ($\mathcal{P}$) and community-view ($\mathcal{C}$) for each triplet $(s, r_t, t) \in \mathcal{T}$ in the train edge-set. Next, we attentively aggregate them. Finally, we combine the aggregated view representation with the source, target node embeddings, and target relation embedding to generate a triplet plausibility score for binary link prediction. We simultaneously optimize for link prediction and community learning losses. Figure 5.5 shows the overall architecture of our proposed Multi-View Heterogeneous Relation Embedding (MV-HRE) framework.
The MV-HRE is initially proposed for transductive setup. However, due to its ability to learn node-identity independent network representations by capturing various graph structural cues, this framework is also a suitable candidate for learning inductive network representations. Therefore, it can be applied to both transductive and inductive link prediction setups.

5.7.1 Learning Subgraph View

For learning representations of the subgraph surrounding a triplet (s, r_t, t), we adopt GRAIL [26]'s proposed GNN model, considering the advantage of expressive GNNs and node labeling scheme [24, 25] for predicting links.

Subgraph Extraction. First, $N-$hop neighborhoods $\{\mathcal{N}^N(s), \mathcal{N}^N(t)\}$ for the source s and target t nodes are generated; based on which, an enclosing subgraph context S^N_{st} between

(s,t) is computed from the common neighbor-set $\{\mathcal{N}^N(s) \cap \mathcal{N}^N(t)\}$. Since we are interested in binary link-prediction, the relation r_t from the input triple (s,r_t,t) is also added to the subgraph to facilitate relation-aware message-passing via GNN.

Node Labeling. Following the *double-radius vertex labeling scheme* [24], network-only node features are generated for each participating node $i \in S^N_{st}$ as,

$$x_{ego_i} = \textsc{One-Hot}\big(\textsc{Dis}(i,s)\big) \oplus \textsc{One-Hot}\big(\textsc{Dis}(i,t)\big) \tag{5.1}$$

Here, $\oplus$ denotes the concatenation of one-hot features. $\textsc{Dis}(i,s)$ denotes the shortest path-distance between node-pair (i,s) without considering any path through t (likewise for $\textsc{Dis}(i,t)$). The source s and target t nodes are uniquely identified with labels $(0,1)$ and $(1,0)$. Such a labeling scheme is able to capture the topological position of each intermediate node in the common subgraph relative to the source and target nodes [24]. Thus, this is indirectly capable of capturing the path information embedded in the subgraph context.

Learning via GNN. For learning representation of a node t, first, the messages from t and its neighborhood are attentively aggregated from the previous hidden-layer $(N-1)$ – which captures $(N-1)$ hop surrounding context in HIN. In RGCN [19]-style, messages from heterogeneous neighbors $s \in \mathcal{N}_r(t)$ for each relation are explicitly propagated via relation-specific layer-wise transformation matrix W^N_r. To prevent overfitting on rare relations, basis-sharing [19] and edge-dropout [26] mechanisms are employed. If $h^{N-1}_s \in \mathbb{R}^d$ denote the hidden d-dimension representation of node s in layer $(N-1)$, the messages a^N_t for target node t in next layer are generated as,

$$\begin{aligned} a^N_t &= \textsc{Aggregate}^N\Big(\{h^{N-1}_s : s \in \mathcal{N}(t)\}, h^{N-1}_t\Big) \\ &= \sum_{r=1}^{|\mathcal{R}|} \sum_{s \in \mathcal{N}_r(t)} \alpha^N_{rr_tst} W^N_r h^{N-1}_s \end{aligned} \tag{5.2}$$

The attention score $\alpha^N_{rr_tst}$ is calculated considering the heterogeneous neighborhood surrounding target node t, and, it is edge-type aware. A contextual representation h_{ctx} is generated based on end-node representations (h^{N-1}_s, h^{N-1}_t) from previous layer, attention-based relation embeddings $(e^a_r, e^a_{r_t})$ of neighborhood relation r and target relation r_t. It uses context transformation matrix W^N_{ctx} and $\textsc{Relu}(.)$ activation function. Next, attention score $\alpha^N_{rr_tst}$ is calculated by applying $\textsc{Sigmoid}(.)$ activation function and transformation matrix W^N_{attn} on

the projected context representations. Here, b_1^N, b_2^N are bias vectors in N^{th} layer.

$$\begin{aligned} h_{ctx} &= \text{RELU}\Big(W_{ctx}^N[h_s^{N-1} \oplus h_t^{N-1} \oplus e_r^a \oplus e_{r_t}^a] + b_1^N\Big) \\ \alpha_{rr_tst}^N &= \text{SIGMOID}(W_{attn}^N h_{ctx} + b_2^N) \end{aligned} \tag{5.3}$$

The propagated and aggregated messages a_t^N in Eqn 5.2 are then combined with target node t's previous-layer hidden state h_t^{N-1} via self-transformation matrix W_{self}^N to iteratively update the target node t representation,

$$\begin{aligned} h_t^N &= \text{COMBINE}^N\Big(h_t^{N-1}, a_t^N\Big) \\ &= \text{RELU}\Big(W_{self}^N h_t^{N-1} + a_t^N\Big) \end{aligned} \tag{5.4}$$

Once final $N-$layer representations of all participating nodes in the enclosing subgraph context S_{st}^N are learned, a subgraph representation is obtained by average pooling of all the latent node representations. The subgraph view $\mathcal{S}_{(s,r_t,t)} \in \mathbb{R}^d$ is generated as,

$$\mathcal{S}_{(s,r_t,t)} = \Big(\frac{1}{|S_{st}^N|}\sum_{i \in S_{st}^N} h_i^N\Big) \tag{5.5}$$

5.7.2 Learning Metapath View

Given a training triplet (s,r_t,t), we generate its metapath-context P_{st}^M by selecting a pre-defined number $|P_{st}^M|$ of sampled metapaths of fixed length upto $M(>1)$ between (s,t). We use *Transformer* [183, 184] architecture to encode the sampled M-length metapaths $\{p_1, p_2, ..., p_{|P_{st}^M|}\}$ from the set P_{st}^M. Each path p is represented in terms of its component relation features as $x_p \in \mathbb{R}^{M.f_r}$. Where, f_r is the size of relational feature space. The start of a sequence p is identified with [CLS] and the varied length sequences aka metapaths are padded with [PAD] token identifiers respectively. Thus, the final input relational features $x_p = [\text{CLS}] \oplus x_p \oplus [\text{PAD}] \in \mathbb{R}^{(M+2).f_r}$. The features of entire metapath context P_{st}^M can be expressed in a matrix form as $X_P \in \mathbb{R}^{|P_{st}^M| \times (M+2).f_r}$. The sampled paths $\{p_1, p_2, ..., p_{|P_{st}^M|}\}$ between (s,t) are sequentially passed through the same TRANSFORMER-ENC(.) for encoding

each of the metapath-contexts as below,

$$p_i = \textsc{Pool}\big(\textsc{Transformer-Enc}(x_{p_i})\big),\ 1 \le i \le |P_{st}^M| \tag{5.6}$$

$$\mathcal{P}_i = p_i \oplus h_{p_i}^1 \oplus h_{p_i}^2 \ldots \oplus h_{p_i}^L \tag{5.7}$$

$$\mathcal{P}_{(s,r_t,t)} = \textsc{LeakyReLU}\Big(\textsc{Mlp}\big(\textsc{Dropout}(\bigoplus_{\forall i, 1 \le i \le |P_{st}^M|} \mathcal{P}_i)\big)\Big) \tag{5.8}$$

The *Transformer* architecture is the most recent popular State-of-The-Art (SOTA) sequence modeling approach which uses a multi-head self-attentive encoding mechanism for modeling input sequences. The $\textsc{Pool}(.)$ function provides us a way to obtain the aggregated representation of the entire input sequence via embedding of the token [CLS]. A transformer's self-attention mechanism implicitly models the similarity among the metapath-contexts of (s,t) based on their component relation representations.

We obtain a sequence representation for each metapath-context by concatenating $(\oplus)$ the pooled individual path embedding $p_i \in \mathbb{R}^d$ from Eqn 5.6 with all its L hidden-layer states $h_{p_i}^l \in \mathbb{R}^d$ for better discriminative representation as in Eqn 5.7. Finally, we use a $\textsc{Dropout}(.)$ based Linear projection layer $\textsc{Mlp}(.)$ with $\textsc{LeakyReLU}(.)$ activation function to obtain the final aggregated and projected path-view representation $\mathcal{P}_{(s,r_t,t)} \in \mathbb{R}^d$ as in Eqn 5.8.

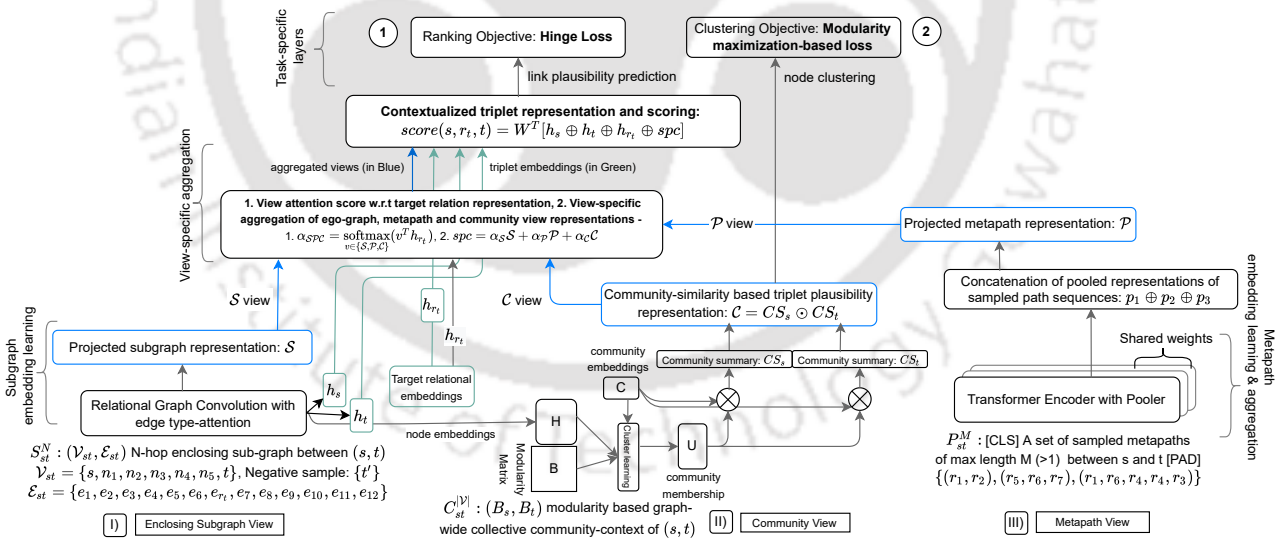


Fig. 5.5 Multi-View Heterogeneous Relation Embedding (MV-HRE)

5.7.3 Learning Community View

To generate a community-view of a triplet (s, r_t, t) — i) we simultaneously learn community structures from the heterogeneous network, ii) generate s and t's graph-wide community summaries, iii) obtain their community-similarity based triplet plausibility representation as the community-view.

Community Representations. We randomly initialize a community embedding matrix $C \in \mathbb{R}^{K \times d}$ for K communities. The row-vectors represent a community in low-dimensional d space. From the latent interaction of node $H \in \mathbb{R}^{|\mathcal{V}| \times d}$ and community embeddings C followed by a SOFTMAX(.) activation function, a node-to-community membership matrix $U \in \mathbb{R}^{|\mathcal{V}| \times K}$ is learned as in Eqn 5.9. The real-valued entries of each row-vector in U denote the probabilities that a node $\in \mathcal{V}$ belongs to any of the K communities. To model the community structures, U obtains further supervision from the modularity matrix B defined for the underlying train heterogeneous graph $\mathcal{G}$ and one orthogonality constraint to minimize the uncertainty associated with the community membership probability distribution. We learn node-community associations U as,

$$U = \text{SOFTMAX}(HC^T) \tag{5.9}$$

Community Membership Learning via Modularity Maximization. We adopt modularity maximization based community learning [29, 34] — as this is one of the most widely used unsupervised algorithm [61] for community detection. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with its adjacency matrix $A \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ and total number of edges $|\mathcal{E}| = e$, the entries of modularity matrix $B \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ are defined as, $B_{(i,j)} = A_{(i,j)} - \frac{\text{DEG}(i).\text{DEG}(j)}{2e}$. Here, (i, j) are two nodes under consideration. $(\text{DEG}(i), \text{DEG}(j))$ are the degrees of (i, j) and the term $\frac{\text{DEG}(i).\text{DEG}(j)}{2e}$ gives us an estimate of the probability of the existence of an edge between (i, j) in a fully random graph. Thus, modularity measures the divergence between the ground-truth intra-cluster edges (edge generation probability denoted by A) from the expected probability distribution of edge generation in a random graph. Since, modularity maximization is an NP-Hard problem, by using *spectral relaxations* [186, 187, 29, 34], the problem of community learning via maximizing modularity can be reformulated as learning the top-K eigenvectors of the modularity matrix B via minimizing the loss below, (Tr is the trace operator on a matrix)

$$\mathcal{O}_{learn} = -\frac{1}{2e} \text{Tr}(U^T B U) \tag{5.10}$$

Orthogonality Constraint. The distribution of node-to-community assignment probability scores $u_i = [u_{(i,1)}, u_{(i,2)}, ..., u_{(i,K)}]$ should be as uneven as possible, so that $u_{(i,k)}$ is clearly distinguishable from $u_{(i,k')}$ for $k \neq k'$, if node i truly belongs to community k and not from community k', i.e., $u_{(i,k)} > u_{(i,k')}$. Following [188, 115] we enforce a soft-orthogonality regularizer that ensures less overlap or less noise in the community assignments,

$$\mathcal{O}_{ortho} = ||U^T U - I_K||_F \tag{5.11}$$

Here, I is an identity matrix of size K and $F(.)$ is *Frobenius* norm of a matrix. By combining all the learning components as in Eqn [5.10, 5.11], the final community learning loss can be expressed as,

$$\mathcal{O}_{community} = \underbrace{-\frac{1}{2e}\text{Tr}(U^T B U)}_{\text{modularity maximization}} + \underbrace{||U^T U - I_K||_F}_{\text{orthogonality regularization}} \tag{5.12}$$

Since Eqn 5.4 already gives us heterogeneous node embeddings that encompass underlying type semantics; we entrust the input node features H in Eqn 5.9 with the task of learning heterogeneous communities in the underlying HIN. We note that several research [189, 190, 187, 191, 192] exist that consider underlying heterogeneity in a graph for learning communities. Nevertheless, we take the most straightforward approach to learning the communities by considering the underlying graph as homogeneous to simplify the designed architecture.

Community Summary Generation for a Node. Next, we obtain the summarized representations of source and target nodes' propensity of belonging to the communities graph-wide. An intuitive way to model this is to get a contextualized global community summary for the said nodes. As in [193], we generate the contextualized global community summary CS_i of a node $i \in \{s,t\}$ as a linear combination of candidate community embeddings C_k weighted by that node's propensity of belonging to any of the communities $U_{(i,k)} \;\; \forall k \in [1,K]$,

$$CS_i = \text{SIGMOID}\Big(\sum_{k=1}^{K} U_{(i,k)} C_k\Big) \tag{5.13}$$

Community-Similarity Based Triplet Plausibility Representation. Given an edge (s, r_t, t), we posit that *the community-view representation of the edge can be modeled as the similarity between the source and target nodes' contextualized community summaries.* It is intuitive because if two nodes do not have any edge between them — they are likely to

be different and may belong to very different community structures. Whereas, two nodes belonging to the same community or similar communities — will have a high probability of forming an edge. Section 5.9.5 elucidates more on this intuition. Therefore, if $\odot$ denotes element-wise multiplication of vectors and (CS_s, CS_t) denote the contextualized community summaries of (s,t) respectively, the community view $\mathcal{C}_{(s,r_t,t)} \in \mathbb{R}^d$ of a triplet is obtained as,

$$\mathcal{C}_{(s,r_t,t)} = CS_s \odot CS_t \tag{5.14}$$

5.7.4 Attentive View Aggregation

We obtain candidate views — subgraph-view ($\mathcal{S}$), metapath-view ($\mathcal{P}$) and community-view ($\mathcal{C}$), for each training triplet $(s,r_t,t) \in \mathcal{T}$. Next, for meaningful view-aggregation, we learn target relation embeddings h_{r_t} using an $\textsc{Mlp}(.) \in \mathbb{R}^{|\mathcal{R}| \times d}$, with respect to which, the attention scores $\alpha_{\mathcal{SPC}} \in \mathbb{R}^{|\mathcal{T}| \times 3}$ for the candidate views are calculated. The ground-truth target relation r_t is treated as observed during the training phase since we are only interested in binary link plausibility prediction and not in relation type prediction of a link. We calculate the attention scores for each view using $\textsc{Softmax}(.)$ and combine the views to obtain attention-weighted view representation $spc \in \mathbb{R}^d$ as,

$$\alpha_{\mathcal{SPC}} = \underset{v \in \{\mathcal{S},\mathcal{P},\mathcal{C}\}}{\textsc{Softmax}}(v^T h_{r_t}), \quad spc = \alpha_{\mathcal{S}}\mathcal{S} + \alpha_{\mathcal{P}}\mathcal{P} + \alpha_{\mathcal{C}}\mathcal{C} \tag{5.15}$$

5.7.5 Triplet Representation and Scoring

For a triplet (s,r_t,t), given the source h_s, target h_t node embeddings, target relation h_{r_t} embedding, and the aggregated view spc embedding, a contextualized triple representation is obtained via concatenation $\oplus$. The link plausibility $\textsc{Score}(s,r_t,t) \in \mathbb{R}$ is calculated based on the contextualized embedding as,

$$\textsc{Score}(s,r_t,t) = W^T[h_s \oplus h_t \oplus h_{r_t} \oplus spc] \tag{5.16}$$

$W \in \mathbb{R}^{4d \times 1}$ are learnable parameters pertaining to the final feed-forward layer that generates real-valued scores for binary link prediction.

5.7.6 Multi-Task Learning Objectives

Link Plausibility Prediction. MV-HRE is trained to discriminate between the existence of a true relational triplet $tp = (s, r_t, t) \in \mathcal{T}$ from the false ones $(tp)'_z = \{(s, r_t, t') \vee (s', r_t, t)\}_z \in \mathcal{T}', z \in \mathcal{NS}(tp)$. For generating invalid triplets, either the target or source node is corrupted using a corruption function $\mathcal{NS}(.)$ with another random node of the same type, which is not connected with the target node under consideration via the relation r_t. Section 5.8 elaborates on this training data generation process. We train our model by scoring the positive triplets higher than negative triplets by a margin of $\gamma \in \mathbb{R}$. We use a *noise contrastive hinge loss* as,

$$\mathcal{O}_{link} = \frac{1}{|\mathcal{T}|} \sum_{tp \in \mathcal{T}} \frac{1}{|\mathcal{NS}(tp)|} \sum_{tp' \in \mathcal{NS}(tp)} \textsc{Max}\Big(0, \textsc{Score}(tp') - \textsc{Score}(tp) + \gamma\Big) \tag{5.17}$$

Community Learning. As we optimize the ranking loss for link prediction, we also simultaneously optimize modularity maximization-based community learning loss as in Eqn 5.12, to generate a community-view for the triplets as in Eqn 5.14.
Therefore, by combining Eqn 5.17 and Eqn 5.12, the final loss of our proposed model is: $\mathcal{O} = \mathcal{O}_{link} + \beta * \mathcal{O}_{community}$. We tune the hyper-parameter β [see Section 5.8] to obtain the optimal contribution of community learning for link prediction.

5.7.7 Inference Complexity

Assume, given a graph $\mathcal{G}$ with vertex-set $\mathcal{V}$ and edge-set $\mathcal{E}$ containing $\mathcal{R}$ distinct relations. The subgraph view-learning model requires i) sampling a subgraph surrounding a triplet (s, r_t, t) and ii) computing shortest paths using the Dijkstra's algorithm in the sampled subgraph — contributing to a time-complexity of $\mathcal{O}(log(|\mathcal{V}|)|\mathcal{E}| + |\mathcal{R}|dL)$ [26] to learn the subgraph $\mathcal{S}$ view embedding. The Transformer architecture contributes to $O(|\mathcal{PS}|.|\mathcal{E}|.\{MLd^2 + M^2d\})$ [183] number of operations for sequence modeling of $|\mathcal{PS}|$ number of sampled metapaths (of length M) per edge $\in \mathcal{E}$ to learn the metapath $\mathcal{P}$ view embedding. Note that we refrain from increasing the complexity of MV-HRE as we only consider metapaths instead of the paths (that include intermediate node identities). This design choice simplifies the model to a large extent. The overall running time of modularity-maximization based community-view $\mathcal{C}$ learning from Eqns 5.9, 5.12 contributes to a factor of $O(|\mathcal{V}|^2 d + |\mathcal{V}|^2 K)$ [29, 34] assuming $d, K \ll |\mathcal{V}|$. We see that community-view learning dominates the overall running time. That is why, we adopt *rank-one degree normalization* trick from DMoN [34] to reduce the computational overhead of matrix-matrix multiplication. From the encouraging results

obtained, we see that introducing metapath and community views do play significant roles in link prediction.

5.8 Evaluation Methodology

In this section, we describe the experiment settings for accessing the performance of the candidate methods. We also provide reproducibility information for the shown results and analyses.

Transductive				
Dataset	Nodes	Node Type	Edges	Relation Type
DDB [48]	9,203	1	44,561	14
ACM [4]	10,942	4	5,47,872	8
IMDB [4]	21,420	1	86,642	6
DBLP [4]	26,128	4	2,39,566	6
PubMed [4]	63,109	4	2,44,986	10
FreeBase [4]	1,80,098	8	10,57,688	36
LastFM [4]	20,612	3	1,41,521	3
Inductive				
Dataset	Nodes	Node Type	Edges	Relation Type
DDB [48]	9,203	1	44,561	14
WN18RR [26]	40,943	1	87,003	11
FB15K-237 [26]	14,541	1	3,10,116	237
NELL995 [48]	63,917	1	1,47,465	198

Table 5.2 Statistics of datasets

Datasets. We evaluate our proposed framework on a variety of publicly available transductive, inductive datasets as mentioned in Table 5.2. For transductive link prediction, we follow the edge-splitting strategy of *Heterogeneous Graph Benchmark (HGB)* [4]. Of total edges, $(81\%, 9\%, 10\%)$ edges are considered as (train/ validation/ test) triplets respectively. For inductive setup, we make use of the publicly available (train/ validation/ test) edge-splits from the papers PathCon [48] and GRAIL [26]. To evaluate both transductive and inductive link prediction tasks, we experiment with two kinds of negative sampling strategies followed in HGB. For each positive test-triplet, we uniformly randomly sample one negative neighbor based on — *i)* $1-$hop, and, *ii)* $2-$hop distances.

Training Data Generation. Since MV-HRE uses relational paths for learning edge embeddings, therefore, to ensure a non-empty relational path-set for each edge, we curate our own (train/ validation/ test) edge-splits using an efficient random-walk-based network exploration proposed in Hin2Vec [91]. Repeated random walks of fixed length l are performed on the ground-truth HIN. For extracting a maximum of M-length relational path between a node-pair (s,t), we slide a window of length up to M on the generated random walks. We then select only a source-target node pair (s,t) that has a direct link of type r_t, as well as higher-order relational paths up to M-length of a pre-defined path sample size – which collectively represent the direct connection between the node-pair. In case of lesser number of path samples, empty sequence marked by [PAD] token identifiers is added to P^M_{st}.

Although Hin2Vec proposes a scheme to eliminate cycles from the generated random walks, it reports no significant difference in link prediction performance for edges with or without cycles in associated path-context $P^M_{s,t}$. Therefore, to ensure a sufficient number of path samples between a node pair, we choose not to remove cycles from the random walks.

Our training data comprises a set $\mathcal{T}$ of contextualized relational triplets of the form $< s, r_t, t, y, (S^N_{st}, P^M_{st}, C^{|\mathcal{V}|}_{st}) >$. In this binary link prediction setup, the edge labels y of positive and negative samples belong to the set of labels $Y \in \{0,1\}$. Here, the link prediction task is formulated as a ranking problem that learns to discriminate between positive and negative triplets. Given a true relational triplet $tp =< s, r_t, t, y, (S^N_{st}, P^M_{st}, C^{|\mathcal{V}|}_{st}) > \in \mathcal{T}$, we use a corruption function $\mathcal{NS}(.)$ to generate false triplets $tp'_z = \{< s, r_t, t', y, (S^N_{st'}, P^M_{st'}, C^{|\mathcal{V}|}_{st'}) >_z \} \in \mathcal{T}', z \in \mathcal{NS}(tp)$ by replacing the destination node t with another random node t' of the same type $\psi(t) = \psi(t')$, which is not connected to the source node s via the relation r_t. Accordingly, negative contexts for node-pair (s,t') are also extracted for the graph.

Experimental Setup. For evaluation, we adopt HGB's [4] experiment setup for both transductive and inductive link prediction. To evaluate the link prediction task as a binary classification problem, we use AUC-ROC and AUC-PR scores. These two metrics give us fair estimates of how good the learned model is — *i)* for classifying both the positive and negative classes, *ii)* in identifying positive class examples, which is also an estimate of the average precision score. Whereas, for obtaining the rank correlation between predictions and targets, we use Mean Rank (MR), Mean Reciprocal Rank (MRR), and Hits@$\{K = 1\}$ scores.

Table 5.3 details the range of hyper-parameters that we search for all the competing methods. For the path-based methods, we adopt their suggested optimal configurations for path-samples, path-length and neighboring hops. We use a patience-based stopping mechanism based on validation loss and obtain the optimal model for evaluation on the

test-set. We set the patience value at 40 and the number of epochs at 300 following HGB. We also use one negative sample per positive test edge for evaluating inference performance. We use only one-hot features for learning the representation of nodes and relations, i.e., feature-less relation-only network representation learning. Unless specified otherwise, a batch size of 128 is used. We vary the community contribution β as explained in Section 5.7.6 in the range of $[0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 10.0]$, out of which, β values within the range $[0.5 - 1.0]$ provide optimal performance for link prediction.

Hyper-parameter	Range
learning rate	[0.0005, 0.0001, 0.005, 0.001, 0.05, 0.01]
weight decay	[0.0001, 0.001, 0.01]
attention heads	[2, 4, 8]
hidden layers	[2, 4, 8]
dropout	[0.1, 0.5]
node hidden dimension	64
edge hidden dimension	64
#clusters (K)	[10 - 100], step-size = 10
batch size	128

Table 5.3 Experiment setup & hyper-parameter range search for competing methods

Baselines. We compare MV-HRE with a number of SoTA frameworks proposed for link prediction on heterogeneous and knowledge graphs. RGCN [19] proposes a graph convolution-based message passing framework with regularization schemes to avoid over-fitting on rare relations. HetGNN [46] uses random-walk with restart strategy to extract type-specific neighborhoods, aggregates them using Bi-LSTMs, and uses attention-based GNN to learn edge embeddings. GATNE [158] uses heterogeneous skip-gram to generate multiple network embeddings from metapath based random-walk explorations on heterogeneous multiplex networks. It then uses graph convolution and attention mechanisms to learn an aggregated edge embedding. Heterogeneous Graph Transformer (HGT) [45] proposes a heterogeneous subgraph sampling strategy, and, a meta-relation aware self-attention mechanism to generate triplet representation. Heterogeneous Graph Neural Network (HGN) [4] is the most recent SoTA NRL model that incorporates edge type-aware multi-head attention with edge residual connections in a GNN architecture to learn edge embeddings. We also consider the GCN and GAT architectures optimized for LP from the HGB benchmark for a fair comparison. HGB concludes that simple GCN and GAT models give fairly competitive performance and remain the second-best competing methods next to HGN (proposed by the HGB benchmark) in their experiment setup [4]. GRAIL [26] is a recent SoTA method

for inductive link prediction in KGs. It proposes to sample enclosing subgraphs between a (source, target) pair and uses edge type-aware attentive GNN with an edge-dropout strategy to learn edge embeddings.

5.9 Results

Here we provide empirical insights into the proposed model's performance and how it compares to the rest of the competing models. We conduct experiments for both the transductive and inductive link prediction setup. A number of conducted case studies bring more clarity on the novelty of MV-HRE. We investigate the following research questions —

RQ1. How well does MV-HRE perform on a variety of classification and ranking metrics in both transductive and inductive setup? [SECTION 5.9.1, SECTION 5.9.2]

RQ2. How useful are the metapath and community views as compared to the subgraph-only view? [SECTION 5.9.3, SECTION 5.9.4]

RQ3. Is the community-view of a triplet proposed in Eqn 5.14 interpretable? [SECTION 5.9.5]

RQ4. How semantically meaningful metapath representations are learned? [SECTION 5.9.6]

RQ5. How semantically meaningful community representations are learned? [SECTION 5.9.7]

In the following sections, we address the discussed research questions elucidating the key takeaways.

5.9.1 Transductive Link Prediction

5.9.1.1 2-Hop random neighborhood sampling

As reported in Tables 5.4, 5.5, we obtain excellent link classification and ranking scores for heterogeneous link prediction using 2-hop negative test-triplets. Our model has more discriminative capacity in identifying 2-hop false neighbors since it explicitly learns from metapaths between end nodes. RGCN, a multi-relational convolution model, performs efficiently and is the second-best method on FreeBase. As observed in the HGB paper,

AUC-ROC	ACM	DBLP	DDB	IMDB	PubMed	FreeBase	AUC-PR	ACM	DBLP	DDB	IMDB	PubMed	FreeBase
GCN	79.701	82.643	80.865	89.183	70.904	76.993	**GCN**	92.435	83.008	74.707	87.101	68.543	74.753
GAT	79.468	81.532	76.564	86.589	71.904	75.131	**GAT**	92.122	74.158	77.389	84.202	71.141	75.976
RGCN	75.707	82.908	82.338	73.429	76.335	77.133	**RGCN**	78.751	84.648	79.033	78.269	69.445	74.692
GATNE	76.285	83.348	77.984	88.23	75.06	71.979	**GATNE**	77.438	83.415	78.849	86.562	74.397	74.935
HetGNN	88.784	83.915	78.349	86.951	71.632	70.203	**HetGNN**	90.057	85.696	79.483	86.15	74.271	74.898
HGT	75.691	85.136	87.393	75.024	79.007	73.244	**HGT**	78.466	80.883	87.168	77.508	71.252	72.252
HGN	75.205	86.11	83.879	89.04	78.911	72.033	**HGN**	76.553	81.798	84.464	88.578	75.165	76.648
GRAIL	88.753	88.205	90.804	90.565	95.059	71.676	**GRAIL**	89.075	86.013	91.131	89.649	95.309	76.435
MV-HRE	**97.15**	**92.064**	**97.45**	**92.323**	**96.778**	**84.453**	**MV-HRE**	**97.368**	**92.069**	**97.224**	**92.702**	**96.781**	**83.774**

Table 5.4 Transductive link prediction classification scores (%): 2−Hop negative triples

MRR	ACM	DBLP	DDB	IMDB	PubMed	FreeBase	HITS@1	ACM	DBLP	DDB	IMDB	PubMed	FreeBase
GCN	96.702	92.548	92.789	93.048	83.2	87.198	**GCN**	94.209	91.159	86.73	92.177	68.359	74.879
GAT	97.227	94.322	87.058	92.548	89.359	86.715	**GAT**	95.38	94.741	75.394	93.267	79.13	73.913
RGCN	97.693	95.344	94.268	86.5	82.059	92.11	**RGCN**	95.584	92.802	89.159	73.665	69.706	84.541
GATNE	97.972	91.605	90.729	90.797	86.047	85.588	**GATNE**	96.44	93.255	88.493	91.995	74.121	71.981
HetGNN	97.756	93.24	89.478	93.32	87.5	86.393	**HetGNN**	95.881	86.585	83.9	86.682	75.52	73.43
HGT	94.916	94.283	94.764	85.709	88.328	87.739	**HGT**	91.088	88.772	90.238	72.214	77.366	76.738
HGN	94.051	93.667	93.323	95.68	96.745	87.923	**HGN**	89.511	92.37	87.045	91.446	93.647	76.329
GRAIL	99.821	94.997	98.883	96.212	97.794	90.419	**GRAIL**	99.842	94.942	97.796	92.432	95.588	81.159
MV-HRE	**99.961**	**95.069**	**99.82**	**97.012**	**98.529**	**94.203**	**MV-HRE**	**99.921**	**95.451**	**99.64**	**93.662**	**97.059**	**88.406**

Table 5.5 Transductive link prediction ranking scores (%): 2−Hop negative triples

GCN and GAT consistently perform well and outperform HGN on the ACM dataset for all compared metrics. As GRAIL uses structurally expressive GNN [24, 25] learned using enclosing subgraphs, it performs effectively - the second-best model in our evaluation. Since GRAIL and MV-HRE are optimized based on ranking objectives, they also consistently perform well on ranking metrics. MV-HRE outperforms the second-best methods across datasets by an average of 5.66% and 0.796% in AUR-ROC and MRR scores, respectively. At the same time, MV-HRE beats GRAIL by an average of 5.859% and 1.078% in AUR-ROC and MRR scores, respectively. We attribute this performance improvement to enriched attentive contextual triplet representation learning.

5.9.1.2 1-Hop random neighborhood sampling

AUC-ROC	ACM	DBLP	DDB	IMDB	PubMed	FreeBase	AUC-PR	ACM	DBLP	DDB	IMDB	PubMed	FreeBase
GCN	91.745	96.396	95.246	95.175	94.436	91.854	**GCN**	93.36	96.043	94.165	93.83	92.929	89.751
GAT	88.288	96.189	92.19	93.991	91.641	92.405	**GAT**	89.663	95.023	90.429	93.049	90.97	90.768
RGCN	97.925	93.583	87.129	83.483	82.978	91.379	**RGCN**	97.887	93.442	84.46	83.183	82.231	91.2
GATNE	97.14	92.572	90.865	95.728	85.014	81.409	**GATNE**	97.426	92.665	89.46	96.278	83.487	80.866
HetGNN	96.299	94.855	89.32	96.79	81.655	86.794	**HetGNN**	96.494	94.27	87.422	97.455	83.162	85.79
HGT	98.009	93.62	91.438	86.128	84.312	86.131	**HGT**	**97.983**	92.726	90.88	87.939	86.541	87.482
HGN	**98.064**	97.098	94.089	98.102	93.52	91.794	**HGN**	97.971	97.034	94.659	98.332	90.36	92.562
GRAIL	88.68	96.393	90.036	97.657	97.593	86.692	**GRAIL**	88.781	95.85	90.101	98.996	97.908	86.926
MV-HRE	97.612	**98.903**	**97.979**	**99.063**	**98.626**	**94.711**	**MV-HRE**	97.548	**98.445**	**97.798**	**99.745**	**98.93**	**93.884**

Table 5.6 Transductive link prediction classification scores (%): 1−Hop negative triples

MRR	ACM	DBLP	DDB	IMDB	PubMed	FreeBase	HITS@1	ACM	DBLP	DDB	IMDB	PubMed	FreeBase
GCN	98.083	99.45	99.198	99.383	98.912	96.338	**GCN**	98.671	98.906	98.471	98.786	97.824	96.675
GAT	98.071	99.104	98.793	99.252	99.044	95.358	**GAT**	97.715	98.215	97.616	98.544	98.121	95.272
RGCN	99.894	97.278	95.732	91.675	91.042	98.51	**RGCN**	99.824	94.626	91.903	83.722	82.474	97.02
GATNE	98.062	96.428	93.816	96.371	92.937	88.928	**GATNE**	98.553	94.22	89.055	97.193	86.348	85.507
HetGNN	98.97	97.552	95.09	98.551	94.051	90.095	**HetGNN**	98.454	95.121	90.281	97.115	88.293	89.901
HGT	99.959	96.959	96.639	94.812	94.777	89.544	**HGT**	99.785	93.975	93.612	89.943	89.752	87.567
HGN	99.859	98.826	97.331	99.196	98.236	95.97	**HGN**	99.753	97.678	94.737	98.393	96.495	92.239
GRAIL	99.965	99.962	98.685	99.46	99.786	97.848	**GRAIL**	99.93	99.923	98.37	98.92	99.572	95.695
MV-HRE	**99.982**	**99.981**	**99.843**	**99.947**	**99.92**	**98.834**	**MV-HRE**	**99.965**	**99.962**	**99.685**	**99.893**	**99.84**	**98.669**

Table 5.7 Transductive link prediction ranking scores (%): 1−Hop negative triples

Tables 5.6, 5.7, show the transductive link prediction performance of the candidate methods for 1−hop negative neighbors. We observe that, though the margin of improvement is lower than that of the performance on 2−hop negative triples, MV-HRE still outperforms the rest of the competing methods. In Table 5.6, HGN is seen to give a competitive performance for the classification metrics, outperforming MV-HRE in ACM. However, MV-HRE remains the best and second-best performer across most of the datasets for ranking metrics. RGCN is consistently seen to perform better in Freebase. In this case, the GNN baselines generally give competitive performance compared to their performances in the 2−hop case. This is since the methods are originally trained based on positive, negative edges and not based on other higher-order graph topologies.

5.9.1.3 Performance on Benchmark splits

Transductive	2-Hop				1-Hop			
	LastFM		**PubMed**		**LastFM**		**PubMed**	
	AUC	**MRR**	**AUC**	**MRR**	**AUC**	**MRR**	**AUC**	**MRR**
RGCN	57.21	77.68	78.29	90.26	81.9	96.68	88.32	96.89
GATNE	66.87	85.93	63.39	80.05	87.42	96.35	78.36	90.64
HetGNN	62.09	83.56	73.63	84	87.35	96.15	84.14	91
MAGNN	56.81	72.93	-	-	76.5	85.68	-	-
HGT	54.99	74.96	80.12	90.85	80.49	95.48	90.29	97.31
GCN	59.17	79.38	80.48	90.99	84.71	96.6	86.06	**98.8**
GAT	58.56	77.04	78.05	90.02	83.55	91.45	87.57	98.38
HGN	67.59	90.81	83.39	92.07	91.04	99.21	91.4	96.04
MV-HRE	**90.954**	**98.243**	**88.147**	**94.41**	**96.667**	**100**	**93.596**	98.796

Table 5.8 Performance on the Heterogenous Graph Benchmark's Transductive Link Prediction Task [4]. Results of competing methods are taken from the Benchmark.

In Table 5.8, we evaluate MV-HRE's performance using the (train/ validation/ test) edge-splits from HGB [4] Benchmark. Given the input triplets, we perform repeated random walks to generate representative path contexts. Note that this does not guarantee to cover all the input triplets with a path context. We report all the competing methods' performance from the original paper [4]. On two datasets, LastFM and PubMed, using HGB's splits, we show that MV-HRE offers significant margins of improvement in both classification and ranking scores for the 2-hop followed by $1-$hop negative sampling cases. Specifically, MV-HRE beats HGN (the best performer in LP on LastFM and PubMed) by 23.364% and 4.757% in AUC-ROC scores on LastFM and PubMed. We see $2.5\% - 7.5\%$ performance improvement in MRR scores by introducing MV-HRE. MV-HRE also outperforms HGN using 1-hop negative sampling test instances consistently. $2\% - 5.5\%$ improvements can be seen in the AUC-ROC scores for LastFM and PubMed. This study strongly speaks for the usefulness of various contexts in link prediction, as well as, the robustness of MV-HRE in a diverse range of experimental settings.

Clarification on high-performance: Since MV-HRE is optimized for the ranking loss, its ranking performances are always on the higher side. MV-HRE's link prediction performance on PubMed is comparable with the rest of the methods that gave competitive performances. However, on LastFM, MV-HRE gives excellent link prediction results, far better than the other competing methods. We found a number of issues in the HGB-provided splits of the LastFM dataset — the existence of a large number of singleton nodes, skewness in the node degree-distribution, less modular structures in the graph, and leakage of train edges. We assume the overlapping train and test edges are the probable cause of such high scores for the methods HGN and MV-HRE in classification and ranking performance metrics on the LastFM dataset. We tried resolving the data leakage issue in LastFM. But it resulted in very small sample sizes for the train/ validation/ test edges. Therefore, we did not proceed further and reported here the results obtained on the original edge splits provided by HGB.

5.9.2 Inductive Link Prediction

Since MV-HRE has the ability to learn useful connectivity patterns for edge representation that are node-independent, we also evaluate its performance for inductive LP. We consider the SoTA inductive method GRAIL and benchmark method HGN for this small-scale study. Another related inductive method Pathcon [48] could not be compared with, since it is a multi-class relation prediction model. Tables 5.9, 5.10 detail the inductive LP performance. MV-HRE outperforms HGN by a significant margin of $9.496\%, 10.663\%$ on average AUC-

	AUC-ROC				AUC-PR			
	DDB	**WIN**	**FB**	**NELL**	**DDB**	**WIN**	**FB**	**NELL**
HGN	77.106	74.378	85.821	68.015	75.015	72.344	85.437	66.673
GRAIL	83.097	76.093	77.033	74.379	79.355	75.898	77.396	76.258
MV-HRE	**94.151**	**81.197**	**87.805**	**80.151**	**91.28**	**80.124**	**92.91**	**77.808**
	MRR				**HITS@1**			
	DDB	**WIN**	**FB**	**NELL**	**DDB**	**WIN**	**FB**	**NELL**
HGN	89.353	89.025	92.91	85.609	88.929	90.447	85.821	71.251
GRAIL	97.694	92.584	91.519	82.665	95.423	91.759	83.177	72.821
MV-HRE	**98.459**	**93.393**	**95.838**	**88.054**	**97.942**	**94.166**	**91.85**	**76.305**

Table 5.9 Inductive link prediction classification and ranking scores on Test Set: 2-Hop

	AUC-ROC				AUC-PR			
	DDB	**WIN**	**FB**	**NELL**	**DDB**	**WIN**	**FB**	**NELL**
HGN	78.268	**97.222**	97.015	88.752	77.873	**98.148**	**98.507**	91.887
GRAIL	95.628	82.437	93.279	85.647	95.674	83.517	91.948	89.701
MV-HRE	**96.844**	86.126	**97.727**	**91.937**	**96.546**	85.54	94.547	**92.488**
	MRR				**HITS@1**			
	DDB	**WIN**	**FB**	**NELL**	**DDB**	**WIN**	**FB**	**NELL**
HGN	91.126	95.37	98.507	94.919	90.225	98.148	97.015	89.843
GRAIL	99.163	99.671	98.959	95.103	98.348	99.605	**99.168**	91.477
MV-HRE	**99.587**	**99.704**	**99.584**	**97.897**	**99.174**	**99.941**	98.919	**95.82**

Table 5.10 Inductive link prediction classification and ranking scores on Test Set: 1-Hop

ROC and AUC-PR scores, respectively, across the datasets for the 2−hop case. However, this margin of improvement is low in FB, where HGN competitively performs. In Table 5.10 for FB dataset, HGN is seen to outperform the rest in AUC-PR score. Similar trends followed in the classification metrics on the WIN dataset for HGN's performance. We attribute HGN's competitive performance on WIN and FB datasets to relatively small network sizes and high edge densities [refer to Table 5.2 for dataset details]. In DDB and FB, we see significant performance improvements (11.054%, 10.772% in AUC-ROC scores) of MV-HRE over GRAIL in the 2−hop case.

5.9.3 Ablation Study

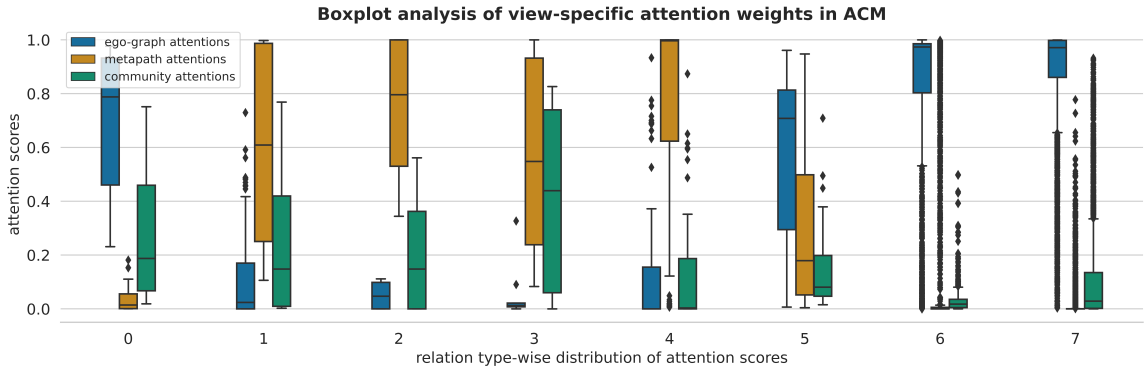
	ROC-AUC (%)	ACM	DDB	PubMed	FreeBase
[1]	GRAIL ($\mathcal{S}$)	88.753	90.804	95.059	71.676
	MV-HRE ($\mathcal{P}$)	92.489	91.807	95.631	74.213
	MV-HRE ($\mathcal{C}$)	92.994	95.189	95.864	77.429
[2]	MV-HRE ($\mathcal{SP}$)	95.494	93.143	95.943	82.079
	MV-HRE ($\mathcal{SC}$)	96.526	96.249	95.988	82.878
	MV-HRE ($\mathcal{PC}$)	96.087	95.822	96.048	74.712
[3]	MV-HRE ($\mathcal{SPC}$)	**97.15**	**97.45**	**96.778**	**84.453**

Table 5.11 Ablation Study: Variants of MV-HRE. ($\mathcal{S}$: Subgraph View, $\mathcal{P}$: Path View, $\mathcal{C}$: Community View)

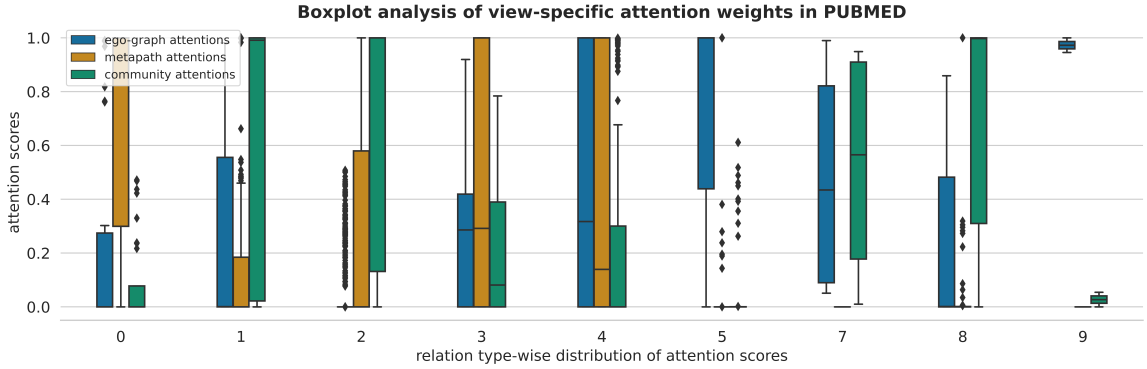
We now drill down to demonstrate the value of multiple views in link prediction on heterogeneous networks. Results are summarized in Table 5.11. We see that all three proposed views of a relational triplet, namely, *i)* the subgraph-view $\mathcal{S}$ – that includes the nodes' relative topological positions and enclosing subgraph, *ii)* the path-view $\mathcal{P}$ – that includes the composite higher-order semantic relations, and, *iii)* the community-view $\mathcal{C}$ – that considers higher-order neighborhood based on link-density surrounding end nodes — are complementary and aid in link prediction. In Table 5.11, we compare GRAIL and view-specific variants of MV-HRE. The respective views are specified in the round brackets in each row. Among the single-view variants in the [1] first grouped rows, we see that the community view of a triplet (s, r_t, t) is the most helpful. It beats the subgraph and path views by an average of 3.796% and 1.834%, respectively, in terms of AUC-ROC scores. In group [1], we see introducing path and community views improve over the subgraph-only view. In terms of performances, $\mathcal{C} > \mathcal{P} \geq \mathcal{S}$. In the [2] second group of rows, as any two views are combined, we see an increasing trend in the classification and ranking performances. Augmenting any view with the $\mathcal{C}$ view improves the performance of the former view. Specifically, the subgraph-community $\mathcal{SC}$ view becomes the second-best performing variant only except in PubMed, where $\mathcal{PC}$ outperforms $\mathcal{SC}$. Each higher-level group outperforms the lower-level group, i.e., $[3] \geq [2] > [1]$, MV-HRE ($\mathcal{SPC}$) being the best performer.

5.9.4 Visualizing view-wise attention weights

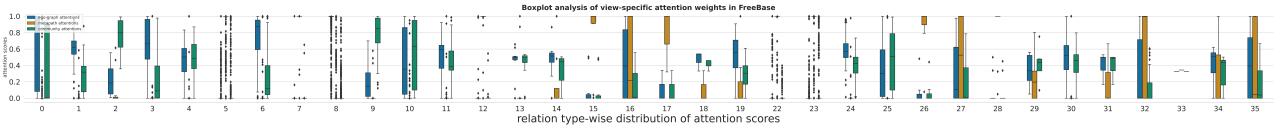
In Figure 5.6, we analyze the distribution of view-specific attention weights in a relation-wise manner for datasets ACM, PubMed, and FreeBase. In the subplots 5.6a, 5.6b, 5.6c, it is



(a) Boxplot analysis of view-specific attention weights in ACM



(b) Boxplot analysis of view-specific attention weights in PubMed



(c) Boxplot analysis of view-specific attention weights in FreeBase

Fig. 5.6 Attention-weight visualization of view embeddings in relation-wise manner Specification of color-codes used. Blue: Subgraph view, Orange: Metapath view, Green: Community view

very interesting to observe how different views contribute for different types of relationships whose binary existence is to be predicted. This is indeed our fine-grained relation-specific view aggregation objective as described in Equation 5.15. The novelty here lies in our model's ability to mitigate most of the challenges that exist for HINs elaborated in Section 5.2. MV-HRE is flexible and robust enough to use various complementary views depending on the situation. In the ACM dataset metapaths play a significant role since they have more attention

weights than subgraph-view and community-view in predicting links. In the PubMed dataset, we see that the community-view plays an essential role for four such relations and has more weight than the subgraph-view. For FreeBase in Figure 5.6c, we observe lesser variance in the boxplots with distinct ranges of attention-weights in 26 out of total 36 relations.

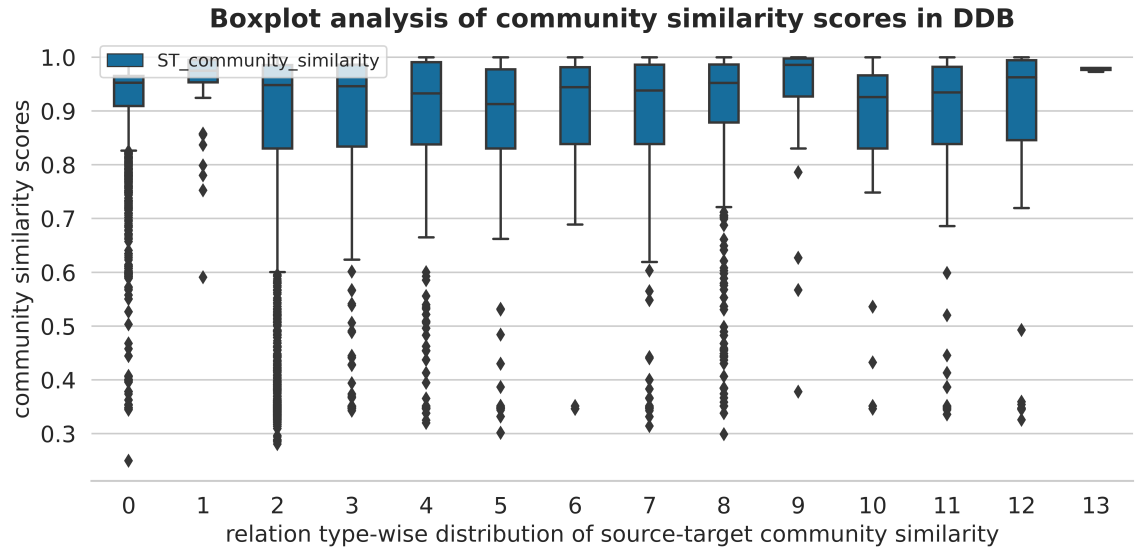
5.9.5 Similarity analysis of node communities

Here, we empirically verify the intuition behind proposing the community-view of an edge in terms of the similarities of the end node-pairs' community-summaries as explained in Equation 5.14. We argue that two nodes will have a high propensity of forming an edge between them if they are part of the same or similar communities. We consider the learned node-to-community association matrix $U \in \mathbb{R}^{|\mathcal{V}| \times K}$ and compute all-pair cosine similarity of the cluster memberships to obtain a node-to-node similarity matrix of cluster associations. For DDB in Figure 5.7a, the similarity scores are distributed within the range $[0.8 - 1.0]$. DDB has very compact boxplots indicating less variance in similarity scores with median values on or above 0.9. For PubMed in Figure 5.7b, this distribution is varied in the range $[0.5 - 1.0]$. One striking observation is — the similarity score distributions vary based on the relation types; that is, for some relations, the similarity scores are significantly higher than the other relations. Figure 5.7b depicts such similarity-score distribution diversity on ACM. This calls for modeling fine-grained community summary association, instead of similarity, between the source and target nodes conditioned on each relation-types — which we keep as an important future direction to explore.

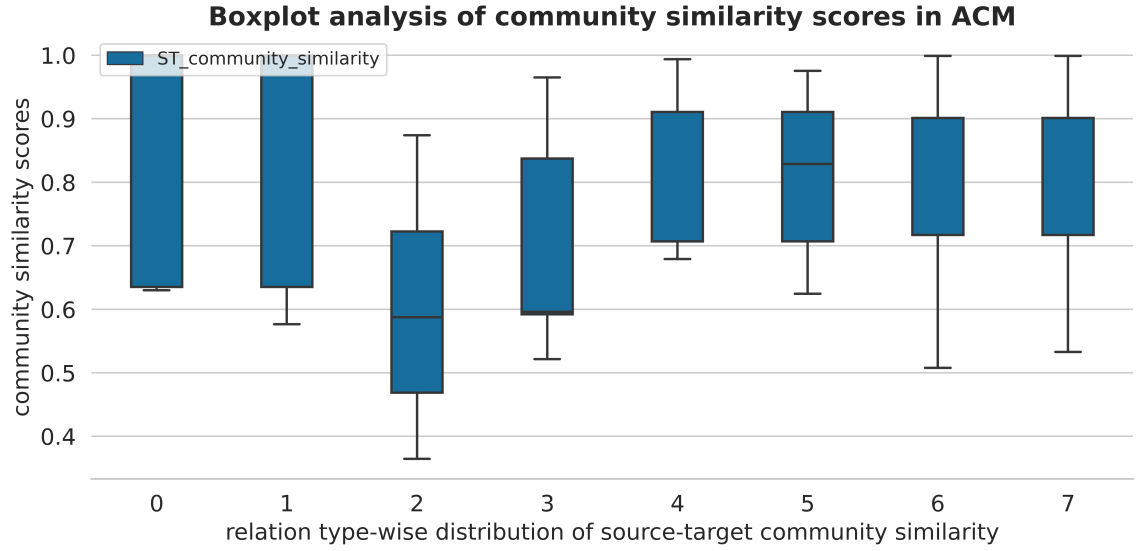
5.9.6 Metapath clustering

We conduct a thorough analysis of how semantically meaningful metapath representations are being learned as a by-product of our proposed architecture MV-HRE, as described in Equations 5.6, 5.7. Not only does MV-HRE distinguish various complex semantic associations in terms of relational paths between two nodes, it also conditions learning of one-hop relation based on the higher-hop associations by a collective representation of relational paths as in Equations 5.8, 5.15.

In an intrinsic evaluation setup, we consider all the $2-$hop metapaths of ACM. Next, we cluster the metapath embeddings using the K-Means algorithm [63] into five clusters. Clusters of metapaths having distinct semantic associations are obtained. Table 5.12 enlists in the left column – various relation types exhibited on ACM dataset, and, enlists in the right



(a) Boxplot analysis of community similarity scores in DDB



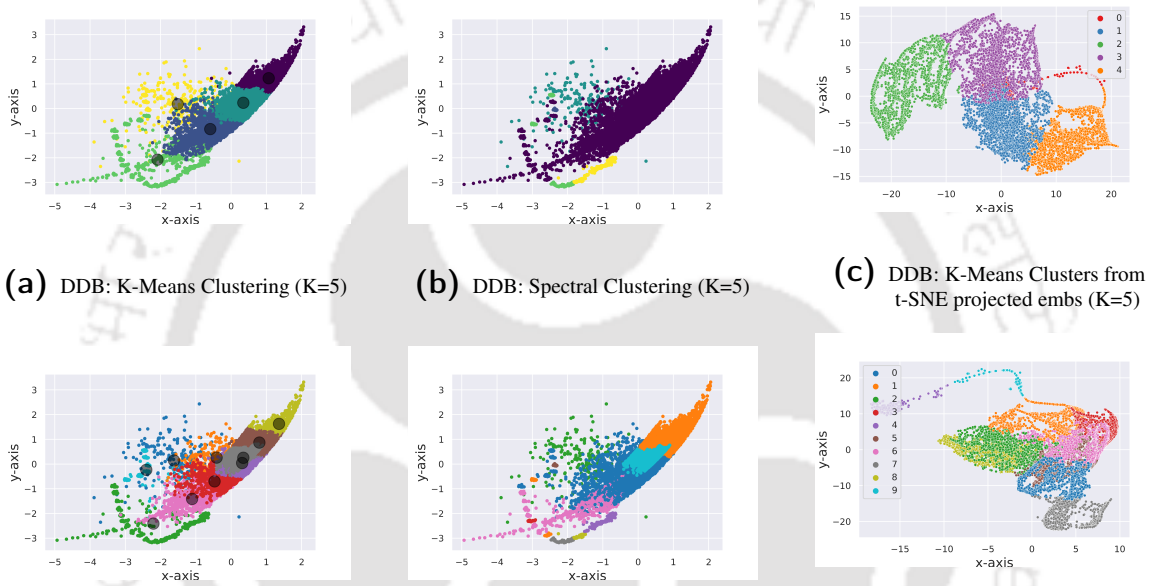
(b) Boxplot analysis of community similarity scores in ACM

Fig. 5.7 Distribution of source and target nodes' community similarity in relation-wise manner

column – the obtained clusters of metapaths. We are able to associate composite high-level semantic meanings with each cluster.

Relation	Meaning	Clusters of metapaths	Semantic meaning
0	paper-cite-paper	0-6, 7-0, 1-6, 7-1, 6-7, 5-1, 1-4, 0-4	various relationship among terms, subjects and papers
1	paper-ref-paper	5-0	subjects related to cited papers
2	paper-author	3-0	authors related to cited papers
3	author-paper	4-5, 1-2, 2-3	papers related by common subjects and authors
4	paper-subject	1-0, 0-1, 0-0, 1-1	papers related by common references and citations
5	subject-paper		
6	paper-term		
7	term-paper		

Table 5.12 K-Means clustering of 2-hop metapaths on ACM and their interpretation



(a) DDB: K-Means Clustering (K=5)

(b) DDB: Spectral Clustering (K=5)

(c) DDB: K-Means Clusters from t-SNE projected embs (K=5)

(d) DDB: K-Means Clustering (K=10)

(e) DDB: Spectral Clustering (K=10)

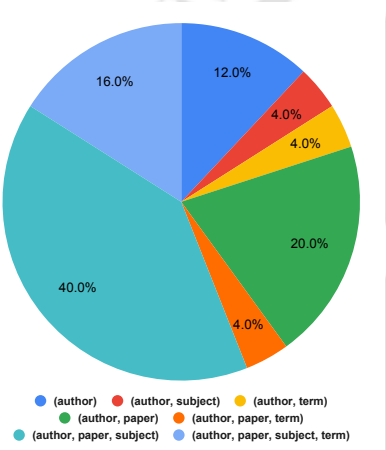
(f) DDB: K-Means Clusters from t-SNE projected embs (K=10)

Fig. 5.8 Clustering of metapath embeddings on DDB

In an extrinsic evaluation setup, plotted in Figure 5.8, we consider the best performing MV-HRE model on DDB dataset and cluster the resultant metapath embeddings using K-Means [63] for $K = \{5, 10\}$ shown in Figures 5.8a, 5.8d. The output cluster-centroids are highlighted as black dots in the figures. Metapath instances are colored based on their associated clusters. Since for metapath clustering evaluation, we do not have any ground-truth labels available, we consider the predicted cluster-memberships from the K-Means algorithm as the ground-truth. Next, we evaluate whether — *i)* other clustering algorithms, such as Spectral [66] clustering, and *ii)* embedding visualization techniques such as t-SNE [128], can conform to the obtained cluster labels from K-Means. Thus, in Figures 5.8b, 5.8e, we cluster the same metapath embeddings using the Spectral method (nearest-neighbor based affinity)

and plot the metapath instances colored by their K-Means cluster labels. In Figure 5.8d, 5.8e, we observe similar clustering tendencies (metapaths belonging to the same K-Means clusters are plotted closer) by two different algorithms on the same set of relational path embeddings — which empirically verifies the semantic relatedness of the learned embeddings. In a more difficult setup, we project the metapath embeddings using the t-SNE algorithm. We cluster the projected embeddings using K-Means and color-code the instances based on cluster labels. In Figures 5.8c, 5.8f, we see metapaths belonging to the same clusters are projected closer. Hence, we cross-validate the semantic relatedness of the metapath embeddings via K-Means clustering, Spectral clustering, and t-SNE visualizations.

5.9.7 Intrinsic evaluation of communities



(a) Distribution of heterogeneous communities discovered in ACM (K=25)

Label-Homogeneous paper community-classes	
database	{1, 10}
wireless communication	{3, 8}
data mining	{23}
Label-Heterogeneous paper community-classes	
{database, wireless communication}	{2, 4, 15}
{database, data mining}	{7, 11, 18, 19}
{wireless communication, data mining}	{11, 13, 18, 19}
{database, wireless communication, data mining}	{11, 18, 19}

(b) Relevance study: ground-truth labels (Left) vs. learned communities (Right) of papers in ACM (K=25). Discovered communities (on the Right) are identified by numbers in the range $\in [1:25]$.

Fig. 5.9 Intrinsic evaluation of learned communities

We evaluate the characteristics and quality of the learned communities in a few interesting setups. We analyze what kinds of heterogeneous communities are being learned on ACM. As we already discussed in Section 5.7.3, we learn the modularity maximization-based communities that do not consider underlying heterogeneity. And, we rely on the generated heterogeneous node embeddings for learning heterogeneous communities. Here we consider the learned node-community membership matrix U of ACM and extract top-most communities based on node-wise community probabilities. We empirically verify in Figure 5.9a the kinds of communities being learned. We see that except for the homogeneous cluster comprising author nodes, the rest of the groupings of nodes contain varied types of vertices –

when 25 communities are discovered using MV-HRE. Top-3 frequently occurring communities involve nodes from — (author, paper, subject)[40%], (author, paper)[20%] and (author, paper, subject, term)[16%] respectively in descending order of frequency.

In this interesting pilot-study, we obtain ground-truth class-labels *{database, wireless communication, data mining}* of paper nodes from the HGB Benchmark node-classification task data-splits. Now, we consider all the clusters involving paper nodes from the previous experiment – discovering 25 communities on the ACM dataset using MV-HRE. We now analyze the distribution of class-labels of paper nodes in ACM from the obtained clusters identified by numbers $\in [1:25]$ in Figure 5.9b. From the upper part of the Figure, label-homogeneous groupings of paper nodes are obtained from communities $\{1,3,8,10\}$. Whereas, we observe groupings of paper nodes belonging to a combination class-labels forming heterogeneous communities related to some latent composite semantic meanings. *This case-study is insightful since it is able to interpret the meaningful communities learned by MV-HRE. Even though MV-HRE has not used any class-label information of the paper nodes in HIN, it can still recover the label-homogeneous communities for the paper nodes.*

Summary

This study considers the task of link prediction in heterogeneous graphs. The study proposes two complementary perspectives– metapath view and community view, besides the standard enclosing subgraph view of a node-pair to learn meaningful edge embeddings for link prediction. From the encouraging results obtained, this study concludes that effective aggregation of multiple views aid in link prediction task by a huge margin. Even as a by-product of this simple, intuitive framework, meaningful relational path representations and node communities are learned. Analysis of view importance suggests that all the chosen candidate views are indeed important and complementary in achieving the best performance on heterogeneous link prediction.

Chapter 6

Conclusions

This dissertation aims to incorporate non-trivial higher-order structures of graphs into NRL frameworks to foster the learning of useful network representations. Graphs of varying and diverse structural complexities, such as — homogeneous graphs, heterogeneous graphs, and multiplex graphs, are chosen as the subjects to incorporate such global structures. Figure 6.1

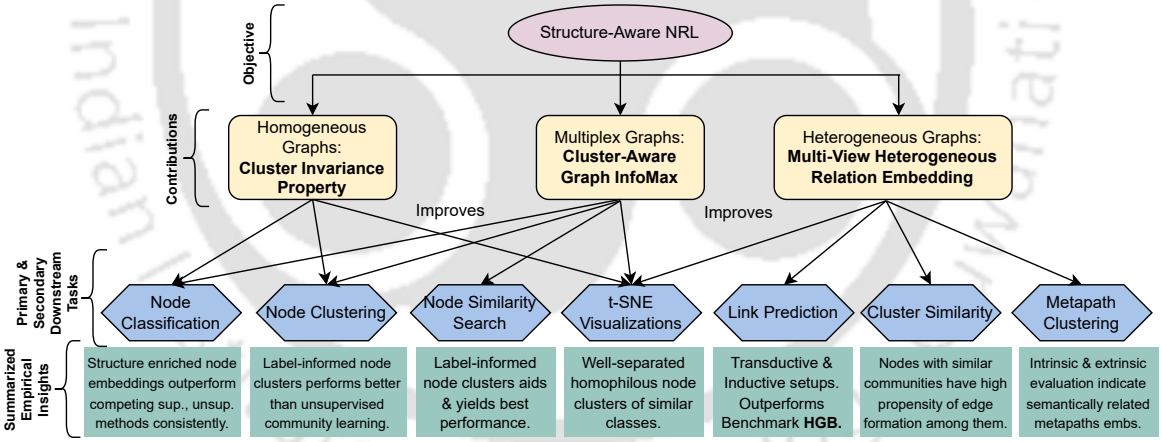


Fig. 6.1 Summary of contributions in the dissertation

recaps the contributions made towards this dissertation. Next, we enlist a summary of contributions, limitations of the proposed frameworks and conducted studies (if any), and prominent future directions to explore.

6.1 Structure-Aware NRL on Homogeneous Graphs

This thesis first investigates the incorporation of non-network node proximity measures such as partially available supervision information to group nodes in a homogeneous graph. The proposed method simultaneously learns node embeddings from the cluster structures as well as from the local neighborhood contexts. In this work, the focus has been entirely on proposing a unified framework that encompasses all the essential learning principles of semi-supervised learning. Empirical results show that embedding nodes with non-local neighborhoods improve the learned node representations' discriminative capacity for node classification and clustering tasks. The usefulness of such embeddings is verified in several challenging scenarios, such as various test-train node sampling strategies and label sparsity.

Matrix factorization frameworks are not scalable for large graphs. A novel framework may be designed to address the scalability issues using ideas from efficient NMF algorithms. It is fascinating to see how the proposed framework performs for link prediction, given the fact that past research efforts have positively acknowledged the role of communities in aiding link prediction performance.

6.2 Structure-Aware NRL on Multiplex Graphs

Further, this thesis considers multiplex graphs with a number of relational layers among the nodes and investigates the applicability of InfoMax principle-based learning to include global structures. To this end, it proposes a structure-aware InfoMax based framework to effectively incorporate global graph structures into local-neighborhood-based node representation learning. It is evident from various experimental analyses that the framework improves the performance of a number of downstream tasks, such as node classification, clustering, and similarity-search. Also, interpretable visualizations from the node embeddings are obtained, which speaks for the informativeness of the learned embeddings.

Encouraging future directions exist for this contribution. It is useful to apply the idea of proposed *Cluster-Aware InfoMax* to other types of graphs such as homogeneous, heterogeneous, and multilayer graphs. The challenge lies in coming up with an apt learning objective capable of capturing the uniqueness of such graph representations. Also, for similar reasons as stated in the previous section, gaining insights on how the proposed InfoMax learning performs for link prediction will be of interest.

6.3 Structure-Aware NRL on Heterogeneous Graphs

Finally, this thesis considers the task of link prediction in heterogeneous graphs, which comprise node and edge types. The study incorporates multiple views, including the metapath view and rarely used community view, for the task of link prediction on HINs. To this end, it proposes a unique community view for LP. From the encouraging results obtained, this study concludes that effective aggregation of multiple views aid in link prediction tasks by a huge margin. Even as a by-product of this simple, intuitive framework, meaningful relational path representations, and node communities are learned. Analysis of view importance suggests that all the chosen candidate views are indeed important and complementary in achieving the best performance on heterogeneous link prediction.

Our study proposes a trivial community view based on simplistic embedding interactions of two end nodes graph-wide community summaries. By parameterizing the embedding interactions – more complex interactions can be modeled, which might provide a finer community view representing an edge. Target relation embedding should also be a part of the proposed community view. Also, it will be interesting to see if incorporating supervision knowledge into the clusters can have more benefits for link prediction. This way, supervision knowledge can directly justify the link formation in graphs. Generalizing the proposed framework for multi-view link prediction on other types of graphs is also of interest.

6.4 Publications

Asterisk (*) denotes equal contributions.

6.4.1 From Thesis

Workshop: **Anasua Mitra**, Priyesh Vijayan, Srinivasan Parthasarathy, and Balaraman Ravindran. "Semi-supervised learning for clusterable graph embeddings with NMF." In NeurIPS Relational Learning Workshop, 2018.

Conference: **Anasua Mitra**, Priyesh Vijayan, Srinivasan Parthasarathy, and Balaraman Ravindran. "A Unified Non-Negative Matrix Factorization Framework for Semi Supervised Learning on Graphs." In Proceedings of the 2020 SIAM International Conference on Data Mining, pp. 487-495. Society for Industrial and Applied Mathematics, 2020.

Conference: **Anasua Mitra**, Priyesh Vijayan, Ranbir Sanasam, Diganta Goswami, Srinivasan Parthasarathy, and Balaraman Ravindran. "Semi-Supervised Deep Learning for Multi-

plex Networks." In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 1234-1244. 2021.

Conference: **Anasua Mitra**, Priyesh Vijayan, Ranbir Sanasam, Diganta Goswami, Srinivasan Parthasarathy, and Balaraman Ravindran. "Revisiting Link Prediction on Heterogeneous Graphs with A Multi-view Perspective." to appear in the Proceedings of 22nd IEEE International Conference on Data Mining (ICDM 2022).

6.4.2 Outside Thesis

Short: **Anasua Mitra**, and Amit Awekar. "On low overlap among search results of academic search engines." In Proceedings of the 26th international conference on world wide web companion, pp. 823-824. 2017.

Journal: Gurukar, S., Vijayan, P., Parthasarathy, S., Ravindran, B., Srinivasan, A., Bajaj, G., Cai, C., Keymanesh, M., Kumar, S., Maneriker, P., **Mitra, A.** and Patel, V., Benchmarking and Analyzing Unsupervised Network Representation Learning and the Illusion of Progress.

Conference: Loitongbam Gyanendro Singh*, **Anasua Mitra***, and Ranbir Sanasam Singh. "Sentiment Analysis of Tweets using Heterogeneous Multi-layer Network Representation and Embedding." In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8932-8946. 2020.

Conference: Bornali Phukon*, **Anasua Mitra***, Ranbir Sanasam, and Priyankoo Sarmah. "TEAM: A multitask learning based Taxonomy Expansion approach for Attach and Merge." In Findings of the Association for Computational Linguistics: NAACL 2022, pp. 366-378. 2022.

6.5 Github Repositories

6.5.1 From Thesis

Python implementation of Max-Margin DeepWalk (MMDW)

Python implementation of Modularity Maximization based NMF (MNMF)

Python implementation of Matrix Factorized Planetoid (MF-Planetoid)

Unified Semi-Supervised Non-Negative Matrix Factorization (USS-NMF)

Semi-Supervised Deep Clustered Multiplex (SSDCM)

Multi-View Heterogeneous Relation Embedding (MV-HRE)

6.5.2 Outside Thesis

Academic Search Engine Disagreement

Python implementation of MultiRank Centrality Algorithm

MultiRank Centrality Algorithm for Heterogeneous Multi-layer Networks

Sentiment Analysis of Tweets using Heterogeneous Multi-layer Networks

TEAM: A multitask learning based Taxonomy Expansion approach for Attach and Merge

6.6 Miscellaneous Research Activities

6.6.1 Research internships

Dec'21–May'22 **Advanced Analytics and Data Science group at Eli Lilly, Bangalore, India**
• Exploration in both NLP and NRL domains for predictive analytics in Biomedical Knowledge-Base. `Heterogeneous semantic search in Knowledge Graphs:` Designed graph learning models to improve the semantic-search capability of in-house Biomedical Search Engine via learning entity representations from Clinical Knowledge Graph (CKG).

May–Aug '18 **I.B.M Research Lab (I.R.L), Bangalore, India**, under Dr. Sreyash Kenkre
• `Question Answering with Explainable Inference:` Built a Question-Answering solver that can reason and come up with rational explanations behind its predicted answers. We formulated QA with Explanation task as a variant of Multiple-Premise Textual Entailment task – which is the first of its kind. We used conic optimization to extract a subset of premises as justification for the correct hypothesis aka answer. • Presented **[Poster]** at I.R.L's Research Poster Presentation session.

Jan'16–Aug'17 **R.I.S.E Lab, Indian Institute of Technology Madras, India** under Prof. Balaraman Ravindran • Developed a framework for multi-label learning on multi-view, multi-relational data via coupled tensor and matrix factorization. • Wrote python version of two recent State-of-the-Art Network Representation Learning algorithms as my contribution to the *NRL Research Communtity* (as the papers' implementations were not available online) — Max-Margin DeepWalk **[MMDW Code]**, Modularity Maximized NMF **[MNMF Code]**.

6.6.2 Invited Talks

- Presented a talk and a poster on **Semi-Supervised Deep Learning for Multiplex Networks [Slide] [Poster]** at the Eleventh and Tenth RBCDSAI Workshop on Recent Progress in Data Science and AI, Annual Research Showcase at **[RBC-DSAI]**, IIT Madras on 17th November, and 29th May, 2021, respectively.
- Presented a talk on **Learning Semi-Supervised Cluster Invariant Node Representations with NMF [Poster]** at the third Indian Workshop on Machine Learning **[iWML]**, IIT BHU on 3rd July, 2018.
- Delivered a talk on **Real-time Crime Mapping for Delhi** an award winning mobile app idea at **Make Delhi Smarter** workshop, IIIT Delhi on 13th Feb, 2016.

6.6.3 Service

- Co-organizer for **3rd Graphs and more Complex structures for Learning and Reasoning (GCLR) Workshop** at **AAAI 2023**.
- Awarded the prestigious **Google Travel Grant** to attend **ICDM 2022** in Florida, USA.
- Awarded travel grants to attend: **NAACL 2022**, **KDD 2021**, **EMNLP 2020**, **CODS-COMAD** (2022, 2020, 2019).
- Reviewer: **ICLR 2022**.
- Volunteer: **ICDM 2022**, **NAACL 2022**, **KDD 2021**.
- National Award in 2016 | **Real-time Crime Mapping for Delhi** – A mobile app idea, awarded one of the best eight ideas in **Make Delhi Smarter** – a National Workshop by IIIT Delhi & Govt. of Delhi.

References

[1] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[2] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *International Conference on Learning Representations*, 2019.

[3] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann, "Netgan: Generating graphs via random walks," in *International Conference on Machine Learning*. PMLR, 2018, pp. 610–619.

[4] Q. Lv, M. Ding, Q. Liu, Y. Chen, W. Feng, S. He, C. Zhou, J. Jiang, Y. Dong, and J. Tang, "Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks," in *ACM SIGKDD*, 2021, pp. 1150–1160.

[5] J. A. Bondy, U. S. R. Murty *et al.*, *Graph theory with applications*. Macmillan London, 1976, vol. 290.

[6] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to wordnet: An on-line lexical database," *International journal of lexicography*, vol. 3, no. 4, pp. 235–244, 1990.

[7] C. Berge, *Hypergraphs: combinatorics of finite sets*. Elsevier, 1984, vol. 45.

[8] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 1105–1114.

[9] S. Bhagat, G. Cormode, and S. Muthukrishnan, "Node classification in social networks," in *Social network data analytics*. Springer, 2011, pp. 115–148.

[10] O. Chapelle, J. Weston, and B. Schölkopf, "Cluster kernels for semi-supervised learning," in *Advances in neural information processing systems*, 2003, pp. 601–608.

[11] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American society for information science and technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[12] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *ACM SIGKDD*, 2017, pp. 135–144.

[13] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE transactions on Big Data*, vol. 6, no. 1, pp. 3–28, 2018.

[14] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[15] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[16] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *SIGKDD*, 2014.

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

[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[19] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *ESWC*, 2018, pp. 593–607.

[20] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec," in *Proceedings of the eleventh ACM international conference on web search and data mining*, 2018, pp. 459–467.

[21] J. Li, C. Chen, H. Tong, and H. Liu, "Multi-layered network embedding," in *SDM*. SIAM, 2018, pp. 684–692.

[22] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," *ICML*, 2016.

[23] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[24] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," *NeurIPS*, vol. 31, 2018.

[25] M. Zhang, P. Li, Y. Xia, K. Wang, and L. Jin, "Labeling trick: A theory of using graph neural networks for multi-node representation learning," *NeurIPS*, vol. 34, 2021.

[26] K. Teru, E. Denis, and W. Hamilton, "Inductive relation prediction by subgraph reasoning," in *ICML*. PMLR, 2020, pp. 9448–9457.

[27] R. Hussein, D. Yang, and P. Cudré-Mauroux, "Are meta-paths necessary? revisiting heterogeneous graph embeddings," in *Proceedings of the 27th ACM international conference on information and knowledge management*, 2018, pp. 437–446.

[28] C. Tu, H. Wang, X. Zeng, Z. Liu, and M. Sun, "Community-enhanced network representation learning for network analysis," *arXiv preprint arXiv:1611.06645*, 2016.

[29] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[30] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, "Learning community embedding with community detection and node embedding on graphs," in *CIKM*, 2017.

[31] H. Sun, Y. Li, B. Lv, W. Yan, L. He, S. Qiao, and J. Huang, "Graph community infomax," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 16, no. 3, pp. 1–21, 2021.

[32] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *ACM SIGKDD*, 2016, pp. 1225–1234.

[33] C. Mavromatis and G. Karypis, "Graph infoclust: Leveraging cluster-level node information for unsupervised graph representation learning," *arXiv preprint arXiv:2009.06946*, 2020.

[34] A. Tsitsulin, J. Palowitch, B. Perozzi, and E. Müller, "Graph clustering with graph neural networks," *arXiv preprint arXiv:2006.16904*, 2020.

[35] B. Rozemberczki, R. Davies, R. Sarkar, and C. Sutton, "Gemsec: Graph embedding with self clustering," in *Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining*, 2019, pp. 65–72.

[36] O. Chapelle, B. Scholkopf, and A. Zien, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 542–542, 2009.

[37] M. Ghorbani, M. S. Baghshah, and H. R. Rabiee, "Mgcn: semi-supervised classification in multi-layer graphs with graph convolutional networks," in *Proceedings of the 2019 ASONAM*, 2019, pp. 208–211.

[38] D. Luo, J. Ni, S. Wang, Y. Bian, X. Yu, and X. Zhang, "Deep multi-graph clustering via attentive cross-graph association," in *WSDM*, 2020, pp. 393–401.

[39] C. Park, D. Kim, J. Han, and H. Yu, "Unsupervised attributed multiplex network embedding." in *AAAI*, 2020, pp. 5371–5378.

[40] W. Liu, P.-Y. Chen, S. Yeung, T. Suzumura, and L. Chen, "Principled multilayer network embedding," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2017, pp. 134–141.

[41] H. Zhang, L. Qiu, L. Yi, and Y. Song, "Scalable multiplex network embedding." in *IJCAI*, vol. 18, 2018, pp. 3082–3088.

[42] R. Linsker, "Self-organization in a perceptual network," *Computer*, vol. 21, no. 3, pp. 105–117, 1988.

[43] F.-Y. Sun, J. Hoffman, V. Verma, and J. Tang, "Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," in *International Conference on Learning Representations*, 2019.

[44] Y. Ren, B. Liu, C. Huang, P. Dai, L. Bo, and J. Zhang, "Hdgi: An unsupervised graph neural network for representation learning in heterogeneous graph," 2020.

[45] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *The Web Conference*, 2020, pp. 2704–2710.

[46] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *ACM SIGKDD*, 2019, pp. 793–803.

[47] X. Fu, J. Zhang, Z. Meng, and I. King, "Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding," in *The Web Conference*, 2020, pp. 2331–2341.

[48] H. Wang, H. Ren, and J. Leskovec, "Relational message passing for knowledge graph completion," in *ACM SIGKDD*, 2021, pp. 1697–1707.

[49] Y. Shi, H. Gui, Q. Zhu, L. Kaplan, and J. Han, "Aspem: Embedding learning by aspects in heterogeneous information networks," in *SIAM SDM*. SIAM, 2018, pp. 144–152.

[50] Y. Fang, W. Lin, V. W. Zheng, M. Wu, J. Shi, K. C.-C. Chang, and X.-L. Li, "Metagraph-based learning on heterogeneous graphs," *IEEE TKDE*, vol. 33, no. 1, pp. 154–168, 2019.

[51] F. R. Chung and F. C. Graham, *Spectral graph theory*. American Mathematical Soc., 1997, no. 92.

[52] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[53] ——, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[54] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Computational statistics & data analysis*, vol. 52, no. 1, pp. 155–173, 2007.

[55] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[56] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," *Advances in neural information processing systems*, vol. 27, 2014.

[57] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[58] C. Tu, W. Zhang, Z. Liu, M. Sun *et al.*, "Max-margin deepwalk: Discriminative learning of network representation." in *IJCAI*, vol. 2016, 2016, pp. 3889–3895.

[59] H.-F. Yu, P. Jain, P. Kar, and I. Dhillon, "Large-scale multi-label learning with missing labels," in *International conference on machine learning*. PMLR, 2014, pp. 593–601.

[60] A.-L. Barabási, "Network science," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 371, no. 1987, p. 20120375, 2013.

[61] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the national academy of sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[62] H. Lu, M. Halappanavar, and A. Kalyanaraman, "Parallel heuristics for scalable community detection," *Parallel Computing*, vol. 47, pp. 19–37, 2015.

[63] K. Krishna and M. N. Murty, "Genetic k-means algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 3, pp. 433–439, 1999.

[64] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[65] N. Biggs, N. L. Biggs, and B. Norman, *Algebraic graph theory*. Cambridge university press, 1993, no. 67.

[66] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[67] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the national academy of sciences*, vol. 105, no. 4, pp. 1118–1123, 2008.

[68] M. Rosvall, D. Axelsson, and C. T. Bergstrom, "The map equation," *The European Physical Journal Special Topics*, vol. 178, no. 1, pp. 13–23, 2009.

[69] A. Subramanya and P. P. Talukdar, "Graph-based semi-supervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 8, no. 4, pp. 1–125, 2014.

[70] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning theory and kernel machines*. Springer, 2003, pp. 144–158.

[71] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks," *Journal of complex networks*, vol. 2, no. 3, pp. 203–271, 2014.

[72] G. Bianconi, *Multilayer networks: structure and function*. Oxford university press, 2018.

[73] E. Cozzo, G. F. De Arruda, F. A. Rodrigues, and Y. Moreno, *Multiplex networks: basic formalism and structural properties*. Springer, 2018.

[74] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *Bioinformatics*, vol. 34, no. 13, pp. i457–i466, 2018.

[75] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, and R. D. Hjelm, "Mine: mutual information neural estimation," *arXiv preprint arXiv:1801.04062*, 2018.

[76] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," *arXiv preprint arXiv:1808.06670*, 2018.

[77] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 297–304.

[78] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, "Deep learning via semi-supervised embedding," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 639–655.

[79] C. Dyer, "Notes on noise contrastive estimation and negative sampling," *arXiv preprint arXiv:1410.8251*, 2014.

[80] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *Proceedings of the 24th ACM international on conference on information and knowledge management*, 2015, pp. 891–900.

[81] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," *Advances in neural information processing systems*, vol. 33, pp. 22 118–22 133, 2020.

[82] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, 2008.

[83] Q. Wu, Z. Wang, C. Li, Y. Ye, Y. Li, and N. Sun, "Protein functional properties prediction in sparsely-label ppi networks through regularized non-negative matrix factorization," in *BMC systems biology*, vol. 9, no. 1. BioMed Central, 2015, pp. 1–14.

[84] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Collective classification via discriminative matrix factorization on sparsely labeled networks," in *Proceedings of the 25th ACM international on conference on information and knowledge management*, 2016, pp. 1563–1572.

[85] E. D. Kolaczyk and G. Csárdi, *Statistical analysis of network data with R*. Springer, 2014, vol. 65.

[86] R. Zafarani, M. A. Abbasi, and H. Liu, *Social media mining: an introduction*. Cambridge University Press, 2014.

[87] M. E. Newman, "Mixing patterns in networks," *Physical review E*, vol. 67, no. 2, p. 026126, 2003.

[88] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *science*, vol. 286, no. 5439, pp. 509–512, 1999.

[89] P. W. Holland and S. Leinhardt, "Transitivity in structural models of small groups," *Comparative group studies*, vol. 2, no. 2, pp. 107–124, 1971.

[90] X. Kong, P. S. Yu, Y. Ding, and D. J. Wild, "Meta path-based collective classification in heterogeneous information networks," in *CIKM*, 2012, pp. 1567–1571.

[91] T.-y. Fu, W.-C. Lee, and Z. Lei, "Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning," in *CIKM*, 2017, pp. 1797–1806.

[92] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, "Harp: Hierarchical representation learning for networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[93] H. Lee, J. Yoo, and S. Choi, "Semi-supervised nonnegative matrix factorization," *IEEE Signal Processing Letters*, vol. 17, no. 1, pp. 4–7, 2009.

[94] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques and applications," *arXiv preprint arXiv:1709.07604*, 2017.

[95] T. A. Snijders and K. Nowicki, "Estimation and prediction for stochastic blockmodels for graphs with latent block structure," *Journal of classification*, vol. 14, no. 1, pp. 75–100, 1997.

[96] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[97] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[98] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[99] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 37–48.

[100] C. F. Van Loan, "Generalizing the singular value decomposition," *SIAM Journal on numerical Analysis*, vol. 13, no. 1, pp. 76–83, 1976.

[101] J. Li, J. Zhu, and B. Zhang, "Discriminative deep random walk for network classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 1004–1013.

[102] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-party deep network representation," *Network*, vol. 11, no. 9, p. 12, 2016.

[103] S. Wang, J. Tang, C. Aggarwal, and H. Liu, "Linked document embedding for classification," in *Proceedings of the 25th ACM international on conference on information and knowledge management*, 2016, pp. 115–124.

[104] C. Li, Z. Li, S. Wang, Y. Yang, X. Zhang, and J. Zhou, "Semi-supervised network embedding," in *International Conference on Database Systems for Advanced Applications*. Springer, 2017, pp. 131–147.

[105] C. Li, S. Wang, D. Yang, Z. Li, Y. Yang, X. Zhang, and J. Zhou, "Ppne: property preserving network embedding," in *International Conference on Database Systems for Advanced Applications*. Springer, 2017, pp. 163–179.

[106] S. Abu-El-Haija, B. Perozzi, R. Al-Rfou, and A. A. Alemi, "Watch your step: Learning node embeddings via graph attention," *Advances in neural information processing systems*, vol. 31, 2018.

[107] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[108] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[109] F. Tian, B. Gao, Q. Cui, E. Chen, and T.-Y. Liu, "Learning deep representations for graph clustering," in *AAAI*, vol. 28, no. 1, 2014.

[110] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in neural information processing systems*, vol. 29, 2016.

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

[112] A. Tsitsulin, D. Mottin, P. Karras, and E. Müller, "Verse: Versatile graph embeddings from similarity measures," in *Proceedings of the 2018 world wide web conference*, 2018, pp. 539–548.

[113] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[114] J. Chen, Q. Zhang, and X. Huang, "Incorporate group information to enhance network embedding," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 1901–1904.

[115] F. M. Bianchi, D. Grattarola, and C. Alippi, "Spectral clustering with graph neural networks for graph pooling," in *ICML*, 2020, pp. 874–883.

[116] S. Chakrabarti, B. Dom, and P. Indyk, "Enhanced hypertext categorization using hyperlinks," in *ACM SIGMOD Record*, 1998.

[117] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of internet portals with machine learning," *Information Retrieval*, 2000.

[118] G. Namata, B. London, L. Getoor, B. Huang, and U. EDU, "Query-driven active surveying for collective classification," in *10th International Workshop on Mining and Learning with Graphs*, 2012.

[119] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015. [Online]. Available: http://networkrepository.com

[120] C. Stark, B.-J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, "Biogrid: a general repository for interaction datasets," *Nucleic acids research*, vol. 34, 2006.

[121] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *ACM SIGKDD'09*, 2009.

[122] M. Mahoney, "Large text compression benchmark," *URL: http://www. mattmahoney. net/text/text. html*, 2011.

[123] P. Vijayan, Y. Chandak, M. M. Khapra, S. Parthasarathy, and B. Ravindran, "Fusion graph convolutional networks," *arXiv preprint arXiv:1805.12528*, 2018.

[124] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *JMLR*, 2006.

[125] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2001.

[126] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics*, 2009.

[127] A. F. McDaid, D. Greene, and N. Hurley, "Normalized mutual information to evaluate overlapping community finding algorithms," *arXiv preprint arXiv:1110.2515*, 2011.

[128] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[129] A. Mashanova, T. H. Oliver, and V. A. Jansen, "Evidence for intermittency and a truncated power law from highly resolved aphid movement data," *Journal of the Royal Society Interface*, vol. 7, no. 42, pp. 199–208, 2010.

[130] R. Jovani, D. Serrano, E. Ursúa, and J. L. Tella, "Truncated power laws reveal a link between low-level behavioral processes and grouping patterns in a colonial bird," *PloS one*, vol. 3, no. 4, p. e1992, 2008.

[131] P. Minasandra and K. Isvaran, "Truncated power-law distribution of group sizes in antelope," *Behaviour*, vol. 157, no. 6, pp. 541–558, 2020.

[132] T. Maschberger and P. Kroupa, "Estimators for the exponent and upper limit, and goodness-of-fit tests for (truncated) power-law distributions," *Monthly Notices of the Royal Astronomical Society*, vol. 395, no. 2, pp. 931–942, 2009.

[133] Á. Corral and Á. González, "Power law size distributions in geoscience revisited," *Earth and Space Science*, vol. 6, no. 5, pp. 673–697, 2019.

[134] M. Bellingeri and F. Scotognella, "The influence of a power law distribution of cluster size on the light transmission of disordered 1-d photonic structures," *Journal of Lightwave Technology*, vol. 33, no. 19, pp. 3980–3985, 2015.

[135] C. Wang, S. Yan, L. Zhang, and H. Zhang, "Non-negative semi-supervised learning," in *Artificial Intelligence and Statistics*, 2009, pp. 575–582.

[136] H. Lee, J. Yoo, and S. Choi, "Semi-supervised nonnegative matrix factorization," *IEEE Signal Processing Letters*, vol. 17, no. 1, pp. 4–7, 2010.

[137] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[138] W. L. Hamilton, *Graph Representation Learning*. Morgan and Claypool, 2020.

[139] H. Gao and S. Ji, "Graph u-nets," in *international conference on machine learning*. PMLR, 2019, pp. 2083–2092.

[140] S. Boccaletti, G. Bianconi, R. Criado, C. I. Del Genio, J. Gómez-Gardenes, M. Romance, I. Sendina-Nadal, Z. Wang, and M. Zanin, "The structure and dynamics of multilayer networks," *Physics reports*, vol. 544, no. 1, pp. 1–122, 2014.

[141] M. Magnani, O. Hanteer, R. Interdonato, L. Rossi, and A. Tagarelli, "Community detection in multiplex networks," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–35, 2021.

[142] M. De Domenico, V. Nicosia, A. Arenas, and V. Latora, "Structural reducibility of multilayer networks," *Nature communications*, vol. 6, no. 1, pp. 1–9, 2015.

[143] R. Guimera and L. A. N. Amaral, "Functional cartography of complex metabolic networks," *nature*, vol. 433, no. 7028, pp. 895–900, 2005.

[144] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *WWW*, 2019, pp. 2022–2032.

[145] R. Matsuno and T. Murata, "Mell: effective embedding method for multiplex networks," in *Companion Proceedings of the The Web Conference 2018*, 2018, pp. 1261–1268.

[146] J. Ni, S. Chang, X. Liu, W. Cheng, H. Chen, D. Xu, and X. Zhang, "Co-regularized deep multi-network embedding," in *WWW*, 2018, pp. 469–478.

[147] A. Mitra, P. Vijayan, S. Parthasarathy, and B. Ravindran, "A unified non-negative matrix factorization framework for semi supervised learning on graphs," in *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 2020, pp. 487–495.

[148] Y. Zhang, Y. Xiong, X. Kong, S. Li, J. Mi, and Y. Zhu, "Deep collective classification in heterogeneous information networks," in *WWW*, 2018, pp. 399–408.

[149] T. Pham, T. Tran, D. Phung, and S. Venkatesh, "Column networks for collective classification," in *AAAI*, 2017.

[150] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *WWW*, 2016, pp. 507–517.

[151] C. D. Manning, H. Schütze, and P. Raghavan, *Introduction to information retrieval*. Cambridge university press, 2008.

[152] L. Zhang, X. Wang, H. Li, G. Zhu, P. Shen, P. Li, X. Lu, S. A. A. Shah, and M. Bennamoun, "Structure-feature based graph self-adaptive pooling," in *WWW*, 2020, pp. 3098–3104.

[153] E. Ranjan, S. Sanyal, and P. P. Talukdar, "Asap: Adaptive structure aware pooling for learning hierarchical graph representations." in *AAAI*, 2020, pp. 5470–5477.

[154] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs*, 2019.

[155] J. Liang, S. Gurukar, and S. Parthasarathy, "MILE: A multi-level framework for scalable graph embedding," *in ICWSM*, 2021.

[156] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," *NeurIPS*, vol. 32, pp. 11 983–11 993, 2019.

[157] S. Zhu, C. Zhou, S. Pan, X. Zhu, and B. Wang, "Relation structure-aware heterogeneous graph neural network," in *IEEE ICDM*. IEEE, 2019, pp. 1534–1539.

[158] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, "Representation learning for attributed multiplex heterogeneous network," in *ACM SIGKDD*, 2019, pp. 1358–1368.

[159] H. Hong, H. Guo, Y. Lin, X. Yang, Z. Li, and J. Ye, "An attention-based graph neural network for heterogeneous structural learning," in *AAAI*, vol. 34, no. 04, 2020, pp. 4132–4139.

[160] S. Soundarajan and J. Hopcroft, "Using community information to improve the precision of link prediction methods," in *The Web Conference*, 2012, pp. 607–608.

[161] S. Mallek, I. Boukhris, Z. Elouedi, and E. Lefevre, "Evidential link prediction in social networks based on structural and social information," *Journal of computational science*, vol. 30, pp. 98–107, 2019.

[162] M. Koptelov, A. Zimmermann, B. Crémilleux, and L. Soualmia, "Link prediction via community detection in bipartite multi-layer graphs," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020, pp. 430–439.

[163] X. Feng, J. Zhao, and K. Xu, "Link prediction in complex networks: a clustering perspective," *The European Physical Journal B*, vol. 85, no. 1, pp. 1–9, 2012.

[164] Y. Dong, Z. Hu, K. Wang, Y. Sun, and J. Tang, "Heterogeneous network representation learning." in *IJCAI*, vol. 20, 2020, pp. 4861–4867.

[165] C. Yang, Y. Xiao, Y. Zhang, Y. Sun, and J. Han, "Heterogeneous network representation learning: A unified framework with survey and benchmark," *IEEE TKDE*, 2020.

[166] M. E. Newman, A.-L. E. Barabási, and D. J. Watts, *The structure and dynamics of networks*. Princeton university press, 2006.

[167] J. Tang, M. Qu, and Q. 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*, 2015, pp. 1165–1174.

[168] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 119–128.

[169] M. Qu, J. Tang, J. Shang, X. Ren, M. Zhang, and J. Han, "An attention-based collaboration framework for multi-view network representation learning," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 1767–1776.

[170] Y. Shi, Q. Zhu, F. Guo, C. Zhang, and J. Han, "Easing embedding learning by comprehensive transcription of heterogeneous information networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2190–2199.

[171] Y. He, Y. Song, J. Li, C. Ji, J. Peng, and H. Peng, "Hetespaceywalk: A heterogeneous spacey random walk for heterogeneous information network embedding," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 639–648.

[172] H. Gui, J. Liu, F. Tao, M. Jiang, B. Norick, and J. Han, "Large-scale embedding learning in heterogeneous event data," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 907–912.

[173] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," *Advances in neural information processing systems*, vol. 26, 2013.

[174] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," *Advances in neural information processing systems*, vol. 26, 2013.

[175] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Icml*, 2011.

[176] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," *arXiv preprint arXiv:1412.6575*, 2014.

[177] K. Wang, Y. Liu, X. Xu, and D. Lin, "Knowledge graph embedding with entity neighbors and deep memory network," *arXiv preprint arXiv:1808.03752*, 2018.

[178] L. Yao, C. Mao, and Y. Luo, "Kg-bert: Bert for knowledge graph completion," *arXiv preprint arXiv:1909.03193*, 2019.

[179] X. Wang, P. Hu, and L. Hu, "A novel stochastic block model for network-based prediction of protein-protein interactions," in *International Conference on Intelligent Computing*. Springer, 2020, pp. 621–632.

[180] L. Hu, X. Wang, Y. Huang, P. Hu, and Z.-H. You, "A novel network-based algorithm for predicting protein-protein interactions using gene ontology," *Frontiers in Microbiology*, p. 2441, 2021.

[181] N. Clauset, Moore, "Hierarchical structure and the prediction of missing links in networks," in *Nature 453*. Nature, 2008, p. 98–101.

[182] C. Mavromatis and G. Karypis, "Hemi: Multi-view embedding in heterogeneous graphs," *arXiv preprint arXiv:2109.07008*, 2021.

[183] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008.

[184] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2018, pp. 4171–4186.

[185] J. Zhang, H. Zhang, C. Xia, and L. Sun, "Graph-bert: Only attention is needed for learning graph representations," *arXiv preprint arXiv:2001.05140*, 2020.

[186] M. E. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical review E*, vol. 74, no. 3, p. 036104, 2006.

[187] X. Li, B. Kao, Z. Ren, and D. Yin, "Spectral clustering in heterogeneous information networks," in *AAAI*, vol. 33, no. 01, 2019, pp. 4221–4228.

[188] N. Bansal, X. Chen, and Z. Wang, "Can we gain more from orthogonality regularizations in training deep networks?" *NeurIPS*, vol. 31, 2018.

[189] R. Ghosh and K. Lerman, "Structure of heterogeneous networks," in *IEEE ICCSE*, vol. 4, 2009, pp. 98–105.

[190] Y. Huang and X. Gao, "Clustering on heterogeneous networks," *Data Mining and Knowledge Discovery*, vol. 4, no. 3, pp. 213–233, 2014.

[191] J. Zhang and Y. Chen, "Modularity based community detection in heterogeneous networks," *Statistica Sinica*, vol. 30, no. 2, pp. 601–629, 2020.

[192] V. Satuluri, Y. Wu, X. Zheng, Y. Qian, B. Wichers, Q. Dai, G. M. Tang, J. Jiang, and J. Lin, "Simclusters: Community-based representations for heterogeneous recommendations at twitter," in *ACM SIGKDD*, 2020, pp. 3183–3193.

[193] A. Mitra, P. Vijayan, R. Sanasam, D. Goswami, S. Parthasarathy, and B. Ravindran, "Semi-supervised deep learning for multiplex networks," in *ACM SIGKDD*, 2021, pp. 1234–1244.