COMMUNITY DYNAMICS AND PERSONALIZED RECOMMENDATIONS: ENHANCING LINK PREDICTION IN UNIPARTITE AND BIPARTITE NETWORKS

(TOPLULUK DİNAMİKLERİ VE KİŞİSEL ÖNERİLER: TEK PARÇALI VE ÇİFT PARÇALI AĞLARDA BAĞLANTI TAHMİNİNİN İYİLEŞTİRİLMESİ)

by

ŞÜKRÜ DEMİR İNAN ÖZER, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

in the

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

JULY 2024

Approval of the thesis:

COMMUNITY DYNAMICS AND PERSONALIZED RECOMMENDATIONS: ENHANCING LINK PREDICTION IN UNIPARTITE AND BIPARTITE NETWORKS

submitted by **ŞÜKRÜ DEMİR İNAN ÖZER** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Galatasaray University** by,

Examining Committee Members:

Assoc. Prof. Dr. Günce Keziban ORMAN
Supervisor, **Computer Engineering Department**
Galatasaray University _____

Assist. Prof. Dr. Reis Burak ARSLAN
Computer Engineering Department
Galatasaray University _____

Assist. Prof. Dr. Atay ÖZGÖVDE
Computer Engineering Department
Boğaziçi University _____

Date: _____

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to Dr. Günce Keziban Orman for her invaluable guidance throughout this thesis as my research supervisor. Without her support, this study would not have been possible.

I would also like to thank my friends and family for their continued encouragement during the writing of this thesis.

We thank Setur for providing the hotel sales data that was used in this study.

This thesis is supported by the bilateral project of the Scientific and Technological Research Council of Türkiye (TÜBITAK), under grant number 122N701, with the CampusFrance, within the scope of Hubert Curien Partnerships (PHC) project number 49032VB.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Complex networks serve as suitable models to represent systems involving interactions between entities. The dynamics of these interactions often lead to the formation of communities. Network modeling can be leveraged to tackle a number of tasks, including link prediction, which allows predicting future interactions between entities. Communities can be defined as subsets of the network where the links between nodes are more dense. Therefore, we expect nodes within the same community to have a higher tendency to link with each other. Yet, the dynamics of the relationship between link prediction and community structure remain understudied in the literature. Additionally, there is a lack of comparative studies on link prediction methods for bipartite networks. Our aim is to delve into the link prediction phenomenon, first by revealing its relation to community structure in unipartite networks and, second, by establishing a systematic methodology for improving prediction accuracy in bipartite networks. We present a series of experiments for the former and four link prediction strategies adapted to work in bipartite networks for the latter. Our findings are two-fold. First, the more pronounced community structures in a network, the better the link prediction success, regardless of the identification of the communities. Moreover, performing the link prediction task on a per-community basis improves link prediction success. Second, adaptation of state-of-the-art personalized collaborative filtering methods can perform link prediction successfully in bipartite networks. Lastly, the SPM link prediction method, originally developed for unipartite networks, also demonstrates relatively high accuracy at predicting links in bipartite networks.

Keywords : Complex network, link prediction, community, recommender systems, bipartite networks, node embedding.

RÉSUMÉ

Les réseaux complexes constituent des modèles pour représenter les systèmes impliquant des interactions entre les entités. La nature de ces interactions conduit souvent à la formation de communautés. La modélisation des réseaux peut être utilisée pour accomplir un certain nombre de tâches, comme la prédiction des liens, qui permet de prévoir les interactions. Les liens entre les nœuds sont plus denses dans les communautés. Par conséquent, nous supposons que les nœuds d'une même communauté ont davantage tendance à se lier les uns aux autres. Cependant, la nature de la relation entre la prédiction des liens et la structure des communautés reste peu étudiée dans la littérature. En outre, les comparaisons des méthodes de prédiction des liens pour les réseaux bipartites sont insuffisantes. Notre objectif est d'étudier le phénomène de prédiction des liens, en révélant sa relation avec la structure des communautés et en établissant une méthodologie systématique pour les réseaux bipartites. Nous présentons une série d'expériences dans le premier cas et quatre stratégies de prédiction de liens dans le second. Nos conclusions sont doubles. Premièrement, plus communautés sont prononcées dans un réseau, plus la prédiction des liens est réussie, indépendamment de l'identification des communautés. De plus, la prédiction de liens par communauté plutôt que sur l'ensemble du réseau améliore le succès. Deuxièmement, l'adaptation des méthodes de filtrage collaboratif permet de prédire les liens dans les réseaux bipartites. Enfin, la méthode SPM fait également preuve d'une précision relativement élevée pour prédire les liens dans les réseaux bipartites.

Mots Clés : Réseau complexe, prédiction des liens, commune, systèmes de recommandation, réseaux bipartites, node embedding.

ÖZET

Karmaşık ağlar, varlıklar arasındaki etkileşimleri içeren sistemleri temsil etmek için elverişli modellerdir. Bu etkileşimlerin dinamikleri genellikle ağlar içinde toplulukların oluşmasına yol açar. Ağ modellemesi, varlıklar arasında gelecekteki etkileşimlerin tahmin edilmesine olanak tanıyan bağlantı tahmini de dahil olmak üzere çeşitli problemlerin üstesinden gelmek için kullanılabilir. Bağlantı tahmini ve topluluklar birbiriyle ilişkili iki olgudur. Topluluklar, düğümler arasındaki bağlantıların daha yoğun olduğu ağ alt kümeleri olarak tanımlanabilir. Bu nedenle, aynı topluluk içindeki düğümlerin birbirleriyle bağlantı kurma eğiliminin daha yüksek olmasını bekleriz. Bu eğilimin niteliği topluluk yapısının özelliklerine bağlı olabilir. Buna rağmen, bağlantı tahmini ve topluluk yapısı arasındaki ilişkinin dinamikleri literatürde yeterince incelenmemiştir. Ayrıca, tek parçalı ağlar için bağlantı tahmin yöntemlerinin kapsamlı karşılaştırmaları mevcutken, çift parçalı ağlar için bu tür çalışmalar yetersizdir. Bu tezdeki amacımız, ilk olarak tek parçalı ağlardaki topluluk yapısı ile ilişkisini ortaya koyarak ve ikinci olarak çift parçalı ağlarda tahmin doğruluğunu artırmak için sistematik bir metodoloji oluşturarak bağlantı tahmini olgusunu araştırmaktır. İlki için bir deney tasarımı ve ikincisi için çift parçalı ağlarda çalışmak üzere uyarlanmış dört farklı bağlantı tahmin stratejisi sunuyoruz. Bulgularımız iki yönlüdür. İlk olarak, bir ağdaki topluluk yapıları ne kadar belirginse, toplulukların tanımlanmasına bakılmaksızın bağlantı tahmin başarısı o kadar iyi olur. Dahası, bağlantı tahmin görevini tüm ağ yerine topluluk bazında gerçekleştirmek bağlantı tahmin başarısını artırmaktadır. İkinci olarak, güncel kişiselleştirilmiş kolaboratif filtreleme yöntemleri, çift parçalı ağlara uyarlanarak bu ağlarda bağlantı tahminini başarıyla gerçekleştirebilir. Üstelik, orijinal olarak tek parçalı ağlar için geliştirilen SPM bağlantı tahmin yöntemi, çift parçalı ağlardaki bağlantıları tahmin etmede de nispeten yüksek doğruluk göstermektedir.

Anahtar Kelimeler : Karmaşık ağlar, link tahmini, topluluk, öneri sistemleri, çift parçalı ağlar, düğüm embedding.

1 INTRODUCTION

This thesis is dedicated to deepening the investigation of the well-known link prediction phenomenon in network science. To contextualize this exploration, the following sections provide a preliminary outline of the studies carried out within the scope of this thesis. First, we will go briefly through the overview of the topic, its current relevance, and the motivation behind the studies. Then, under the problem statement section, we will talk about how this thesis will bridge the gap of knowledge on the subject. We will then list the contributions of the study to the literature in general terms. Finally, we will conclude this chapter with a summary of how the rest of the thesis is structured.

1.1 Overview and Motivation

Networks are a very common paradigm to model real-world systems, as they allow explicitly representing the interactions between the objects that constitute these systems (Costa et al., 2011). Representing systems in this manner allows us to analyze the interactions between the objects that constitute those systems. Thorough analysis of networks allows us to make inferences about the dynamics between objects or how systems evolve. Differentiating microbial communities based on topological properties (Barberán et al., 2012), utilizing network centrality for logistic planning (Porta et al., 2006), and botnet detection with clustering (Nagaraja et al., 2010) are examples of cases where network analysis is used to solve real-world problems.

A whole subclass of these systems can be observed to exhibit either a homogenous or a heterogenous structure. In a system with a homogenous structure, all the objects are of the same type. For instance, a social network in which all nodes are individuals and the links indicate friendship can be said to be homogenous. Such systems are represented using networks with one type of node, known as *unipartite networks*. In the case of the systems with heterogeneous structure, there can be several different types of entities. One specific case of these systems involves two types of objects, such as in user-item

purchase interactions or job seeker-job advertisement applications. Their graph-based representation takes the form of *bipartite networks*, which contain two distinct types of nodes, and in which the links always connect nodes of different types (Saoub, 2021). Due to their more constrained structure, bipartite networks require specific processing compared to unipartite ones.

Independently from network type, link prediction is one of the most prominent tasks employed in network analysis. This task consists of determining which links are missing from a network or which links are likely to form in the forthcoming state of an evolving network (Yang et al., 2015). Link prediction is useful in a number of applications, including recommender systems, spam detection, privacy control, network routing, and bibliometrics (Kumar et al., 2020a). Although it has been widely studied on unipartite networks (Lü and Zhou, 2011), there has been a limited number of such works regarding bipartite networks (Kunegis et al., 2010; Benchettara et al., 2010).

Another crucial topic of network science is the identification of community structure. A community can be defined as a subset of nodes where the links between them are more frequent than the links to the rest of the network (Radicchi et al., 2004). Link prediction and community structure characteristics are two topics related by their problem definitions. Despite this, the literature investigating the nature and the dynamics of this relationship remains insufficient (Biswas and Biswas, 2017; Zhang et al., 2022). As will be discussed in Chapter 2, most of the existing work does not directly analyze the relationship between communities in the network and link prediction. Instead, they redefine the link prediction methods by taking the communities to which they belong into account (Iqbal and Latha, 2022; Soundarajan and Hopcroft, 2012). This means that a deeper analysis on this issue is necessary for us to make progress on the link prediction problem.

Similarly, there are few studies that comprehensively address the link prediction problem in bipartite networks (Kunegis et al., 2010). More importantly, standardized, widely accepted methods for bipartite link prediction remain to be established. The lack of a generally accepted approach means the absence of both a method from which future studies can draw inspiration and a baseline against which new methods can be compared. One of the main motivations of this thesis is to bridge this gap by adapting

link prediction methods that work in unipartite networks to bipartite networks.

While there are studies in the literature that are performed with the same motivation with us (Daminelli et al., 2015), they only consider basic link prediction indexes that use only the local neighborhood information. On the other hand, highly sophisticated and successful collaborative filtering techniques for recommendation systems have been published in recent years. Indeed, bipartite link prediction and recommendation are two closely related problems (Wu, Ziteng et al., 2021). That is why we concentrate on recent collaborative filtering methods and adapt them for link prediction in bipartite networks to make more accurate predictions compared to previous works.

The two major motivations of this thesis are to address the gaps in knowledge on *the relation between community structure characteristics and link prediction in unipartite networks* and *link prediction in bipartite networks*.

1.2 Problem statement

The formation of communities in a network may change the expected linking behavior of nodes. For example, in networks without communities, we expect high-degree nodes to link with other high-degree nodes (Barabási et al., 2002). However, in the case of networks with community structure, a high-degree node may instead show a tendency to link with a lower-degree node in its own community rather than with a high-degree node in another community. Not only the formation of communities but also their structural properties affect how the links occur. Link prediction methods rely on an existing network structure to operate on. The more meaningful links this structure contains, the more information we have about the structure, and the more accurate the predictions will be.

Another factor affecting network structure is the object types of the studied system. Indeed, in bipartite networks, where the interacting objects can be divided into two disconnected sets, the link prediction becomes even more complex as the network structure is completely different. That is why we encounter only a limited number of dedicated works (Benchettara et al., 2010; Daminelli et al., 2015). One way to improve this shortcoming is to adapt methods developed for unipartite networks to bipartite networks.

Given that the dynamics of interactions are different in bipartite networks, a straight forward adaptation may not work well.

Collaborative filtering models, which operate entirely over bipartite networks, implement special architectures to capture this unique structure. Their goal is to recommend to a node of one type, e.g., users, in a bipartite network the K most probable nodes of the other type, e.g., items. These recommendations can be considered as links between nodes in the future. Hence, we can liken the recommendation problem to the link prediction problem. Therefore, collaborative filtering models designed to work with bipartite networks can be adapted to solve the link prediction problem.

In this thesis, we interpret the two important problems mentioned above as two specific sub-problems in the context of the effects of the network structure on link prediction. The first problem we tackle is the lack of research on how the link prediction task is affected by the presence of communities in unipartite networks. The second problem is the absence of an established method for bipartite networks, given the gap in the literature on the comparison of link prediction methods for bipartite networks. The development of effective methods for those problems will increase the success of link prediction in the real-life applications in which predicted links can represent an interaction between two people, a job contract between a job seeker and an employer, two proteins interacting with each other, a user watching a TV show, etc.

1.3 Contributions

The main contribution of this study is two-fold, and thus it is more appropriate to analyze them under two different headings.

The contribution of our work on the subject of the relationship between link prediction and communities relies on measuring the link prediction performance of different approaches across varying levels of community cohesiveness and community size. Our work is dedicated to performing a comparative analysis of link prediction methods by overcoming the issues we listed for previous works.

Regarding the work done for bipartite networks, we introduce four main categories

of approaches to study different link prediction methods : (i) traditional link scores, (ii) link prediction via network topological features, (iii) embedding-based link prediction, and (iv) personalized recommendation conversion. During the experiments, these four categories are compared among themselves and with each other over three real-world networks. This comparison was made in terms of both prediction accuracy and prediction speed.

A complete list of all the contributions of this thesis can be given as follows.

- Proposing a taxonomy of link prediction methods in unipartite and bipartite networks.
- Applying link prediction methods on a per-community basis to increase their accuracy of predictions.
- Examine whether various performance metrics used in studies on link prediction are appropriate for evaluating this task based on their sensitivity to varying levels of community cohesiveness.
- Comparing the performance of 29 link prediction methods developed for unipartite networks by executing them on a total of 27 synthetic networks generated with 3 different parameter sets and with different community sizes and cohesiveness levels.
- Assessing the performance of 19 bipartite link prediction methods that we adapted from different techniques on a benchmark of three real-world datasets with varying sizes and densities.
- Repurposing recently developed and well-performing personalized recommendation models to predict links in bipartite networks.
- Evaluating the runtime of different approaches we adapted for link prediction in bipartite networks and examining how they scale in real-world networks.

The rest of the study is organized as follows. In Chapter 2, we review the related work in the literature in two sections. Then, we review the methods used in our experiments

on unipartite networks and the taxonomy proposed for said methods in Chapter 3. Similarly, in Chapter 4, we introduce the methods we apply to bipartite networks, grouped under four different categories. In Chapter 5, we evaluate the different metrics used for performance evaluation of the link prediction task, which is vital for accurate interpretation of the results. The design and results of our experiments are revealed in Chapter 6. We present our findings according to the five research questions we identified to design the experiments. We then provide a detailed interpretation of these findings. Finally, in Chapter 7, we conclude the study by summarizing our work, the findings, the implications of these findings, and future work.

2 LITERATURE REVIEW

In this chapter, we review past work on the two topics that are the focus of this thesis under their respective sections.

2.1 Link Prediction and Communities

Extensive studies comparing various link prediction methods in unipartite networks are available in the literature. In their survey study, Lü and Zhou (2011) presented a taxonomy of link prediction methods and compared the performance of different methods in real-world networks. The methods are analyzed under three categories : Similarity-based algorithms, maximum likelihood methods, and probabilistic models. However, this comparison was limited to similarity-based algorithms. In addition, precision and AUROC metrics, whose capacity to evaluate link prediction success is a matter of debate, were used for the comparison. The work on metrics will be discussed later in this section. After the comparison, the Katz index is found to be successful. Following that, they discuss possible real-life applications of the link prediction task : Network reconstruction, evaluation of evolving models, and labeling nodes.

A survey similar to this study has been conducted more recently by Kumar et al. (2020a). A broader taxonomy was prepared as more method families were included in the study. Differently, embedding methods were also included in the study. The comparison of the success of the methods is also more diverse in terms of the methods used and evaluation metrics. In their experiments, SPM was the most successful of the link prediction indexes. However, when compared to embedding methods, it fell behind node2vec. Finally, this study also talks about possible real-life applications. In addition to the previous work, they address the recommendation and spam detection for possible application areas of the link prediction task.

In a network, community structure is constituted of the node groups with denser inner links (Newman, 2004). Thus, one would expect having a link between two members of the same community to be more likely. Although these two major areas seem to

be related to each other, the works dedicated to revealing the characteristics of such relations are limited. We come across works proposing novel link prediction techniques while considering or leveraging the community structure (Iqbal and Latha, 2022; Soundarajan and Hopcroft, 2012). These works modify some of the existing local similarity-based indexes with the identified community structures. They concluded that community information allows proposed link prediction methods to be more successful. However, they suffer from limited experimental evaluation ; (i) they only consider a small number of link prediction techniques when comparing their proposal, and (ii) the possible relation between communities and link prediction performance is not mentioned or analyzed. Biswas and Biswas (2017) propose three different link prediction methods that consider communities. Unlike the previous works, their experiments are evaluated on both real-world networks and synthetic networks generated by using the Lancichinetti–Fortunato–Radicchi (LFR) algorithm (Lancichinetti et al., 2008) with varying values for the mixing parameter μ and the network size. This enables the exploration of possible performance changes in link predictions according to different types of community structures. However, the experiments suffer from two shortcomings ; (i) being realized on relatively small networks with 500 or 1000 nodes and (ii) not being compared with the existing baselines. Likewise, Zhang et al. (2022) propose a link prediction method that uses community features and compare it to other commonly used link prediction indexes. They conduct their experiments on real-world networks that come from different domains and vary in size. To extract community information from networks, they use Louvain's community detection methods (Blondel et al., 2008). Apart from those works which are mainly focused on proposing novel link prediction techniques, in Xu et al. (2020), the authors analyze the changes in link prediction performance according to different community structure characteristics. However, this work is also limited to two important issues. First, they compare only three link prediction indexes, while there are numerous local and global ones. Second, they use the artificial networks generated by the well-known LFR model with small sizes (around 2000 nodes) and unrealistic benchmark parameters.

Yang et al. (2015) questioned how appropriate the metrics used in scientific studies on link prediction are for measuring the success of link prediction. One of the methods employed in this study is temporal distance. Temporal distance represents how long a link belonging to the test group will exist in the network after the snapshot of

the network in the training phase. Therefore, as the temporal distance increases, we expect the performance to decrease since we are predicting a link that will occur in the distant future. In the experiments, while the AUPR method exhibits this expected decrease, it is observed that increasing the temporal distance sometimes leads to sharp spikes in the AUROC values. They attribute this behavior of AUROC to the class imbalance problem that is commonly encountered in the link prediction task. As a result, precision-recall curve-based evaluation metrics such as AUPR are said to be more suitable for measuring link prediction success.

2.2 Link Prediction and Bipartite Networks

This section of the report provides a detailed analysis of existing studies and methodologies focused on the subject of bipartite node embedding. Examining the advancements and the gaps in the field allows for a better understanding of the subject.

Kunegis et al. (2010) conducted a study comparing link prediction methods used in bipartite networks. This study is important due to the number of datasets used ; 25 real-world bipartite networks of various sizes were used for benchmarking. They compared Odd Polynomials, Non-negative Odd Polynomials, Hyperbolic Sine, Rank Reduction, Odd von Neumann Pseudokernel, and Preferential Attachment methods. Link prediction success was measured using Mean Average Precision. In these measurements, it was found that the Preferential Attachment index yielded the best result. In cases where the best result was achieved by another method, it produced results very close to the best result. Considering the link prediction methods that have been developed since the date of this study, this study is not comprehensive enough in terms of the methods used in today's standards.

Al Hasan et al. (2006) and Benchettara et al. (2010) studied a different approach, namely the use of topological features in combination with supervised machine learning methods for link prediction in bipartite networks. In order to do so, they utilized different topological features. The topological attributes used include the common neighbors index, the Jaccard coefficient, the Adamic Adar index, the Preferential Attachment index, the product of two nodes' PageRank scores, the node's degree, the

shortest distance between two nodes, and the node's clustering index. These attributes are considered as features in the tabular data and a tabular dataset is obtained where each row represents a link. Various supervised machine learning algorithms were then applied to this tabular data, including but not limited to Support Vector Machines, Decision Trees, Multilayer Perceptron, K-Nearest Neighbors with varying K values, and Naive Bayes. Since classification models were used in the studies, metrics such as precision, recall, and F-score were used as measurement metrics.

A closely related topic to the link prediction task is network embedding, an approach we include in our experiments on both subjects. With these methods, nodes on the network can be represented as vectors. In other words, embedding methods can be used to transform the graph data structure into tabular data with numeric variables. This allows for machine learning methods that are normally unable process the graph data to be able to do it. The most recent survey on these methods was conducted by Giamphy et al. (2023). Firstly, the methods proposed so far in the field are identified, grouped, and presented to the readers. The authors mention that the proposed taxonomy is inspired by the work of Yang et al. (2022). This grouping is done by two categories : (1) proximity preserving approaches and (2) message passing approaches. Methods in the former group aim to preserve the characteristics of the graph during embedding by exploiting the local and global properties of graphs. These methods involve the evaluation of first-order and higher-order relations. One of the approaches used by them is the application of the random walk method used in natural language processing to graphs. Researchers mention that this approach has two striking problems. The first is the loss of information due to the indiscriminate evaluation of two node types. The second is the loss of characteristics of graphs such as the node degree distribution obeying the power-law. Message passing approaches are based on the principle that for each node, information about its local neighborhood is iteratively processed and the output is given as input to the next iteration. Figure 2.1 shows how this approach works in bipartite networks. This approach forms the foundation of the Graph Neural Networks (GNNs). The methods mentioned in this survey and others are described in the following paragraphs. After introducing the methods, the survey mentions tasks that can be used during embedding success measurement. The node classification task is similar to a regular supervised classification, but since it is not independently and

identically distributed, the relationships between nodes need to be considered. This task can be used for community detection. Relation prediction, more commonly known as link prediction, is often used in recommendation systems. This task can be described as the prediction of a set of edges given a set of nodes. Graph classification and graph reconstruction are two tasks that are noticeably lacking in the literature. The first one takes many graphs as input and tries to assign labels. This task is used to solve problems such as enzyme identification. The latter is based on predicting the edges of the original graph using vector embeddings. Although the survey is very informative and instructive, no experiments have been carried out and therefore no results can be reported.

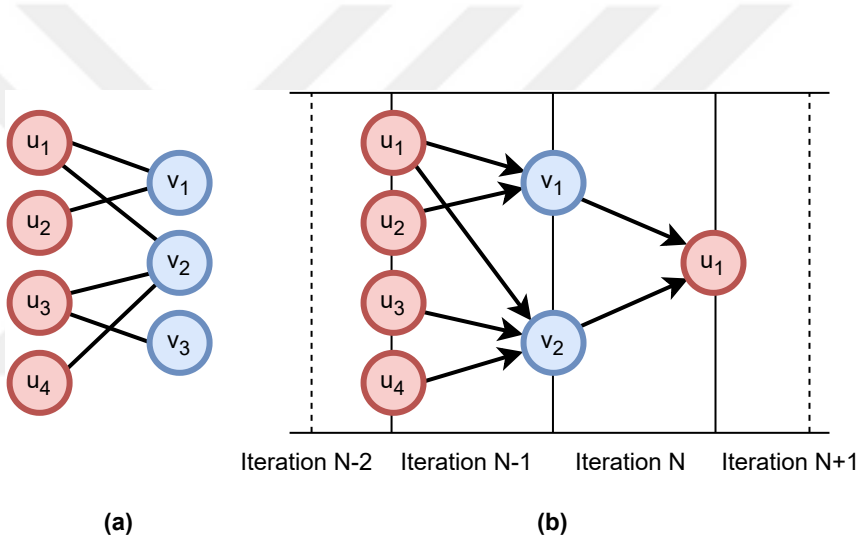


FIGURE 2.1 – The figure illustrating the message passing process as described in Giamphy et al. (2023). Given a bipartite graph shown in (a), for every iteration, the aggregation algorithm takes the information of local neighbors. Then, this information is combined with node attributes in order to generate an output to be used as the input of the following iteration as shown in (b).

Although not directly related to bipartite networks, we will briefly review graph neural network (GNN) based recommendation systems since they are used in this study. Gao et al. (2023) conducted a comprehensive survey on how graph neural networks can be used in recommendation systems, the advantages of using graph neural networks, and the challenges encountered when using them. In the survey, after a brief overview of the concepts studied, they answer the question of why graph neural networks are preferred. They give three main reasons for this. The first one is that GNNs, thanks to their

network-based architecture, can represent all user-item relationships regardless of the domain. The second reason is the ability of GNN-based methods to extract and learn high-order connectivity. Since recommendation accuracy depends on being able to find users with similar tastes and items that face similar demand, high-order connectivity also plays a major role in recommendation accuracy. Finally, it is mentioned that GNNs enrich the data during the learning phase by including secondary behaviors in the data, without depending solely on the target behavior. If we go through the example given by the authors, suppose that the target behavior is the user's purchase of a product. Since this data is sparse, recommender systems that work only on the user's product purchase history will fail. However, GNN-based recommender systems can also use secondary behaviors such as the user's product search history or adding a product to the cart. The authors then present their taxonomy of recommender systems using GNNs and group GNNs from four different perspectives.

3 LINK PREDICTION FOR UNIPARTITE NETWORKS

Link prediction indexes are the metrics that quantify each link $\in L'$ by using the structure of L. These scores are then ranked to choose the most likely links for predictions. To evaluate the performances of different link scores and to compare them with other approaches, we adopt a standard supervised method (Yang et al., 2015). We split L into two sets : L_{train} and L_{test}. We use the graph structure of L_{train} to calculate these scores for all possible links, i.e. the set of $L_{test} \cup L'$. The links in this set are then ranked in decreasing order according to the scores they received. The first $|L_{test}|$ links having the highest scores constitute our prediction set, L_{pred}. The success of predictions is evaluated by comparing the links from L_{pred} with the links from L_{test}. In the following sections, x and y are used to represent any two nodes ; k_x is the degree of node x and $\Gamma(x)$ is the set of the neighbors of node x.

3.1 Neighbor-based Link Prediction

The common property of these methods is that they all use the neighborhood of the nodes when calculating the link prediction score of the node pairs. These link prediction indexes rely on the concept of triadic closure observed in social networks (Opsahl, 2013), which proposes that the more neighbors two nodes share, the more likely they are to be connected.

Common Neighbors Index (CN) is equal to the cardinality of the intersection of the set of the neighbors of x and y. Its formula is given in Eq. 3.1.

$$CN(x, y) = |\Gamma(x) \cap \Gamma(y)| \quad (3.1)$$

Newman proposed this index and demonstrated that collaboration between two nodes is correlated to the number of their common neighbors (Newman, 2001). In the work of Kossinets and Watts (2009), a large-scale social network of student relationships

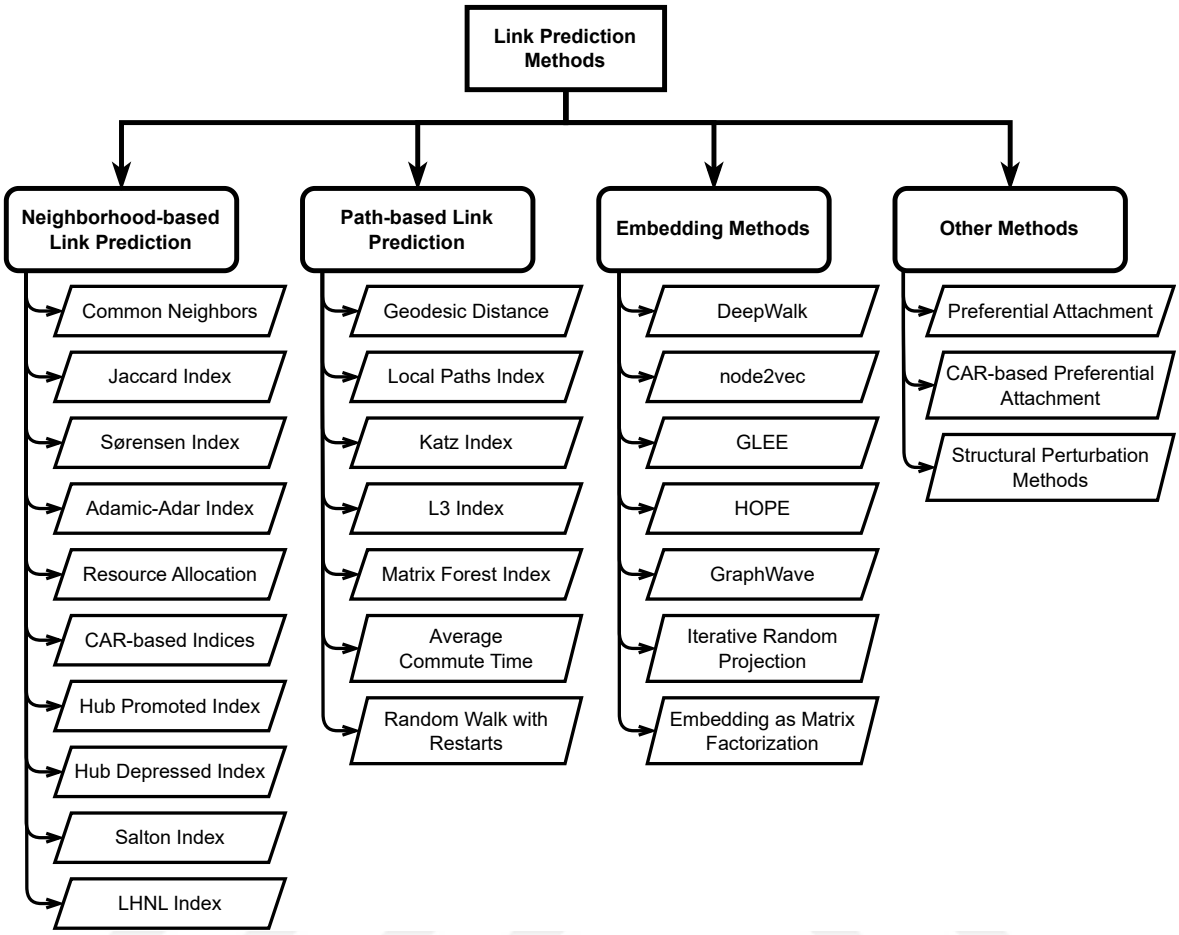


FIGURE 3.1 – Taxonomy link prediction methods applicable for unipartite networks, grouped under four main branches.

was examined. The authors found that a higher number of mutual friends between two students has a positive effect on the likelihood of a friendship between them. This index is preferred in real-life social networks modeling similar situations. The other methods in this section also use this index as the main idea, just with different normalization techniques.

Jaccard Index (JC) is equal to the number of common neighbors of x and y divided by the number of nodes who are neighbors with at least one of them (Jaccard, 1912). Mathematical notation is observed in Eq. 3.2. This can be interpreted as normalization of the CN.

$$JC(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \quad (3.2)$$

With this normalization, JC directly expresses the probability that a random neighbor selected from the set of all neighbors of two nodes is a common neighbor of the two

nodes. The main idea of this index is that if two nodes have a high probability of sharing a common neighbor, the probability of a link between them is also high. However, as shown in the work of Liben-Nowell and Kleinberg (2003), this index lags behind CN in accuracy for some datasets.

Sørensen / Dice Index (DICE) is primarily used for ecological community data (Sørenson, 1948). Shown in Eq. 3.3, Sørensen index is equal to twice the number of common neighbors of x and y divided by the sum of their node degrees.

$$SI(x,y) = \frac{2 \times |\Gamma(x) \cap \Gamma(y)|}{k_x + k_y} \quad (3.3)$$

By doing so, a normalization similar to that of JC is aimed. However, links between nodes with a large number of common neighbors are not penalized as much as in JC.

Adamic-Adar Index (AA) modifies the CN by weighting each common neighbor by the inverse log of its node degree as demonstrated in Eq. 3.4. The lower the degree of a common neighbor, the higher the outcome. This is based on the concept that as the degree of a common neighbor increases, the influence of this common neighbor on the similarity between two nodes decreases (Adamic and Adar, 2003).

$$AA(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log k_z} \quad (3.4)$$

Resource Allocation Index (RA) assumes that a node pair sends resources to each other over their common neighbors, and each common neighbor distributes the resource equally to their own neighbors (Zhou et al., 2009). As with the AA, common neighbors with a large number of links are penalized. The lack of a logarithmic expression in the denominator (Eq. 3.5) penalizes the common neighbors with a larger set of neighbors even more heavily compared to the AA index.

$$RA(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{k_z} \quad (3.5)$$

CAR-based Common Neighbor Index (CAR) based indexes evaluate the similarity of two nodes based on how connected their common neighbors are to the local

community described by the local-community-paradigm (LCP) theory introduced by Cannistraci et al. (2013). When calculating these indexes, the local community links (LCL) value is used. It expresses the connectivity to the local community and is calculated as shown in Eq. 3.6.

$$LCL(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{|\Gamma(z) \cap \Gamma(x) \cap \Gamma(y)|}{2} \quad (3.6)$$

In other words, x and y form a local community with each common neighbor. As the number of neighbors of this common neighbor that are connected to all three nodes increases, the LCL value also increases, and hence the local community is considered more pronounced. This in turn positively affects the similarity score of x and y as seen in Eq. 3.7 below.

$$CAR(x,y) = CN(x,y) \times LCL(x,y) \quad (3.7)$$

CAR-based Adamic-Adar Index (CAA) is calculated using Eq. 3.8. It is an implementation of the AA using the LCP theory (Cannistraci et al., 2013).

$$CAA(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{|\Gamma(z) \cap \Gamma(x) \cap \Gamma(y)|}{log_2 k_z} \quad (3.8)$$

CAR-based Resource Allocation Index (CRA) is calculated by applying the RA to local community links (Cannistraci et al., 2013) as demonstrated in Eq. 3.9.

$$CRA(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{|\Gamma(z) \cap \Gamma(x) \cap \Gamma(y)|}{k_z} \quad (3.9)$$

Hub Promoted Index (HPI) uses the concept of hubs which are the nodes with higher degrees and promotes the linking of hub nodes with lower-degree nodes. Its formula is given in Eq. 3.10. At the same time, it penalizes hubs for establishing links with each other (Ravasz et al., 2002).

$$HPI(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{min(k_x, k_y)} \quad (3.10)$$

This method is based on the hierarchical structure of metabolic networks. In such networks, functional units are formed by the hierarchical combination of modules with high link density.

Hub Depressed Index (HDI) shares the same concept as HPI, but it operates in the opposite way (Ravasz et al., 2002). In other words, the HDI score of a potential link between a node with a low degree and a hub node will be quite low. Its formula is given in Eq. 3.11.

$$HDI(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{max(k_x, k_y)} \quad (3.11)$$

As the degree difference between two nodes decreases, the HDI and HPI scores of the link between them will gradually converge.

Salton Index / Cosine Similarity (COS) is the well-known cosine similarity adaptation to the network node pairs (Salton and McGill, 1983). Cosine similarity finds the similarity of two vectors in vector space by measuring the cosine of the angle between them. That is, this similarity is not related to the magnitude of the vectors, but to their orientation in the vector space. An adaptation of this index can be viewed in Eq. 3.12.

$$COS(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{k_x \times k_y}} \quad (3.12)$$

This equation shares the same formula as the Otsuka-Ochiai (Otsuka, 1936; Ochiai, 1957) coefficient used in biology.

Leicht–Holme–Newman Local Index (LHNL) is similar to COS but this index punishes the highly connected nodes even more severely as there is no square root in the denominator of Eq. 3.13 (Leicht et al., 2006).

$$LHNL(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{k_x \times k_y} \quad (3.13)$$

3.2 Path-based Link Prediction

The indexes that we explain in this section take into account the structural positions of the node pairs in the entire network besides their local neighborhood when finding their similarity. For most of these methods, what matters is the number and length of paths connecting the two nodes, rather than the number of neighbors that the two nodes share.

Geodesic Distance Index (Dist) is the inverse of the geodesic distance, i.e. the length of the shortest path, between two nodes as seen in Eq. 3.14. By taking the reciprocal, we penalize links connecting nodes that have longer paths between them.

$$\text{Dist}(x,y) = \begin{cases} \frac{1}{d(x,y)}, & \text{if } x \text{ and } y \text{ are connected} \\ |E|+1, & \text{otherwise} \end{cases} \quad (3.14)$$

where $d(x,y)$ is the shortest path between x and y.

Local Paths Index (LP) represents the number of two- and three-paths between node pairs, where three-paths are assigned a weight parameter ϵ (see Eq. 3.15) (Lü et al., 2009).

$$LP = A^2 + \epsilon A^3 \quad (3.15)$$

where A is adjacency matrix.

Katz Index (Katz) can be thought of as a generalization of LP. It is the sum of the number of paths of different lengths between x and y, penalized with an attenuation factor α (Katz, 1953). Longer paths are penalized in an exponentially increasing manner. The formula is expressed in Eq. 3.16.

$$Katz(x,y) = \sum_{l=1}^{\infty} \alpha^l (A^l)_{xy}, \quad (3.16)$$

L3 Index (L3) was introduced by Kovács et al. (2019) after they observe that the triadic closure principle was ineffective for certain networks, such as the ones that model protein-protein interactions. L3 is the degree-normalized count of paths of length three connecting two nodes.

$$L3(x,y) = \sum_{u,v} \frac{A_{xu} A_{uv} A_{vy}}{\sqrt{k_u \times k_v}}. \quad (3.17)$$

To compare how the length of the paths affect the link prediction performance, we also propose to use two variations of L3, that we refer to as L5 and L7. Those indexes are computed the same way, except they count the paths of length five and seven, respectively.

Random Walk with Restart Index (RWR) is an adaptation of the PageRank algorithm for link prediction (Tong et al., 2006). Assuming that q_{xy} is the probability

that a random walker who starts in x will end its walk in y, the index between two nodes x and y is expressed as the sum of probabilities q_{xy} and q_{yx}, seen in Eq. 3.18.

$$RWR(x,y) = q_{xy} + q_{yx} \quad (3.18)$$

In this equation q_{xy} is the yth element of $\vec{q_x}$, depicted in Eq. 3.19.

$$\vec{q_x} = pP^T\vec{q_x} + (1-p)\vec{e_x} \quad (3.19)$$

where p is the probability that the random walker will move to a random neighbor and $1-p$ is the probability that it will return to node x. $\vec{e_x}$ is the seed vector that contains zeros except for the x, with a length of the number of vertices in the network. P^T is the transition matrix, which takes the value $P_{xy} = \frac{1}{k_x}$ if x and y are neighbors and 0 if they are not. Further simplifying the expression above reveals Eq. 3.20.

$$\vec{q_x} = (1-p)(I - pP^T)^{-1}\vec{e_x} \quad (3.20)$$

Average Commute Time Index (ACT) is established on the concept of random walkers and signifies the average number of steps it would take a random walker starting from x to reach its target y and then return to x (Liu and Lü, 2010).

Matrix Forest Index (MF) is based on the spanning tree principle. When predicting links between x and y, the matrix forest index uses the ratio of the number of spanning trees with root node x and containing both x and y to the number of all spanning rooted forests in the network. The higher this ratio is, the more likely a link between x and y is considered to exist (Chebotarev and Shamis, 2006).

3.3 Other methods

Preferential Attachment Index (PA) is the multiplication of degrees of two nodes, as shown in Eq. 3.21. The nodes with more neighbors are more likely to be linked to each other (Barabási et al., 2002).

$$PA(x,y) = k_x \times k_y \quad (3.21)$$

Based on the assumption that high ranked nodes will connect with other high ranked nodes, this index was first used for growing scale-free network generation. The higher the score of a link calculated with this index, the higher the probability of adding that link to the network in the next iteration. In this way, a network with the desired number of links is obtained. In our study, the main advantage of this index for link prediction is its low complexity and ease of computation. Naturally, we would expect this index to perform well in networks that are consistent with the assumption behind the index, i.e. networks where higher-order nodes are more densely connected to each other.

CAR-based Preferential Attachment Index (CPA) is a measure obtained by calculating the PA using LCL values (Cannistraci et al., 2013), which results in Eq. 3.22.

$$CPA(x,y) = e_x e_y + e_x CAR(x,y) + e_y CAR(x,y) + CAR(x,y)^2 \quad (3.22)$$

where $e_x = |\{z \in \Gamma(x) : z \notin \Gamma(y)\}|$ and $e_y = |\{z \in \Gamma(y) : z \notin \Gamma(x)\}|$

Structural Perturbation Method (SPM) examines the change in Eigenvalues when the adjacency matrix is perturbed, to measure the structural consistency of the network (Lü et al., 2015). The links to be predicted are ranked according to their scores in the perturbed matrix $\tilde{A}$, which is obtained through first-order perturbation (see Eq. 3.23).

$$\tilde{A} = \sum_{k=1}^{N} (\lambda_k + \Delta\lambda_k) x_k x_k^T, \quad (3.23)$$

where λ_k and x_k are respectively the Eigenvalue and the corresponding Eigenvector of matrix A^R, after the links in the perturbation set are removed from the network. Expressions $\lambda_k + \Delta\lambda_k$ and $x_k + \Delta x_k$ are the corrected Eigenvalue and its corresponding Eigenvector.

To calculate the structural consistency of a network A, a set of links is randomly selected to make up the perturbation set denoted as ΔE while the remaining links make up the E^R. Then, the corresponding adjacency matrix A^R is perturbed with ΔA resulting in the perturbed matrix $\tilde{A}$. U being the set of all links, the links in $U - E^R$ are then ranked by their values in $\tilde{A}$. Of those links, the top-L ranked ones make up

the E^L set, L being equal to $|\Delta E|$. By combining E^R and E^L, the perturbed network is obtained. As the final step of the calculation, structural consistency is defined in Eq.

3.24

$$\sigma_c = \frac{|E^L \cap \Delta E|}{|\Delta E|}. \quad (3.24)$$

The researchers theorize that if a set of links ΔE removed at random did not significantly affect the structure of the network, we can expect the top-ranked links E^L to be approximately the same. Building up on that idea, link prediction using the structural perturbation method treats the training subset of links E^T like the original network A. Instead of calculating σ_c, top-$|E^P|$ links are considered the predicted links, E^P being the probe subset which is $E - E^T$. Essentially, by comparing the properties of the original network with those of the perturbed network, we can predict which nodes are expected to be linked.

3.4 Embedding methods

The indexes listed in this section use embedding vectors, generated by methods of the same name, that represent nodes. Node embedding allows nodes to be represented as n-dimensional vectors, preserving the structural layout of the network and the relationships between nodes. This enables the graph data to be represented in vector space. After the embedding vectors are obtained for each node, cosine similarity between the vector pairs is calculated. For the final step, network homophily is assumed. This is to say that links between similar nodes are given higher prediction scores.

DeepWalk (DW) introduces a node representation learning approach that trains a skip-gram model to extract vector representation of nodes (Perozzi et al., 2014). This approach consists of two primary components : a random walk generator and an update procedure. During random walk generation, a vertex from the graph is selected in a uniformly random manner. This vertex is established as the root of the random walk. Each subsequent step in the walk uniformly samples from the neighbors of the last visited vertex until the walk reaches its maximum length. Using local information from truncated random walks allows for the embedding vectors of nodes that appear together more frequently in random walks to be closer to each other in the vector

space. Hierarchical Softmax (Morin and Bengio, 2005; Mnih and Hinton, 2008) is used for optimization purposes during the training step of the skip-gram model.

Node2Vec (N2V) is an improvement made to DeepWalk as it allows for biased random walks (Grover and Leskovec, 2016). The researchers introduce a versatile neighborhood sampling approach achieved through the creation of an adaptable biased random walk procedure, enabling a transition between breadth-first search and depth-first search strategies. The bias is controlled by α as seen in Eq. 3.25. This bias factor is in turn controlled by return parameter p and in-out parameter q. The return parameter controls the probability of returning to the node that was visited in the previous step of the random walk. The in-out parameter allows control over the spread pattern of the random walk. If $q > 1$, the random walk is biased towards visiting the nodes that are closer to the last node visited. If $q < 1$, the nodes that are further away from the last visited node will be more likely to be visited. In the following equation, it is assumed the random walk has just traversed the edge $u - v$ and the current node is v. The random walk will now traverse the edge $v - y$. The bias factor α is then calculated as

$$\alpha(u, y) = \begin{cases} \frac{1}{p}, & \text{if } l_{uy} = 0 \\ 1, & \text{if } l_{uy} = 1 \\ \frac{1}{q}, & \text{if } l_{uy} = 2 \end{cases} \tag{3.25}$$

where l_{xy} is the distance between the nodes x and y. This distance cannot be larger than two, as this would skip a step in the random walk process.

Geometric Laplacian Eigenmap Embedding (GLEE) utilizes the singular value decomposition of the Laplacian matrix of a given graph for node embedding (Torres et al., 2020). The researchers demonstrate that the GLEE embedding of a graph is identical to its simplex projected down to n dimensions. To elaborate, GLEE takes graph G as input. Assuming that the Laplacian matrix of this graph is L, the singular value decomposition can be represented as $L = SS^T$. At this point, the embedding dimension is involved. Suppose that the chosen embedding dimension is $d = 128$, then the first 128 columns of the obtained matrix S, S^d, contain the embedding vectors of the nodes of graph G. For node i, this vector will be the d-dimensional vector s_i^d.

Iterative Random Projection Network Embedding (RN) employs Gaussian random projection on the proximity matrix to map the network into an embedding space (Zhang et al., 2018). In Eq. 3.26, the high-order proximity matrix S is first defined as

$$S = \alpha_0 I + \alpha_1 A + \alpha_2 A^2 + ... + \alpha_q A^q, \quad (3.26)$$

where α represents the predefined weights and q is the order. The researchers cite the Eckart-Young theorem (Eckart and Young, 1936) as proof that the optimal solution of this equation can be reached by singular value decomposition. However, since the computational complexity of this procedure is high, they propose using Gaussian random projection. If $R \in \mathbb{R}$ and all elements of R are assumed to follow a Gaussian distribution, the embeddings can be computed via

$$U = S \cdot R = (\alpha_0 I + \alpha_1 A + \alpha_2 A^2 + ... + \alpha_q A^q)R, \quad (3.27)$$

matrix product operation.

Network Embedding as Matrix Factorization (NM) is a matrix factorization-based network embedding framework (Qiu et al., 2018). It attempts to unify DeepWalk, LINE (Tang, Qu, Wang, Zhang, Yan and Mei, 2015), PTE (Tang, Qu and Mei, 2015), and node2vec by obtaining their closed forms.

High-Order Proximity preserved Embedding (HOPE) addresses the challenge of preserving asymmetric transitivity through approximation of high-order proximities (Ou et al., 2016). They introduce a general formulation of high-order proximities

$$S = M_g^{-1} \cdot M_l, \quad (3.28)$$

where M_g and M_l are polynomial matrices. The researchers apply a generalized singular value decomposition

$$M_g^{-1} \cdot M_l = (V^s \Sigma V^t)^T, \quad (3.29)$$

to the general formulation they described in Eq. 3.28.

GraphWave (GW) has been developed to create structural embeddings of nodes by treating spectral graph wavelets as probability distributions (Donnat et al., 2018). This insight makes it possible to capture similarities between distant nodes with dissimilar

raw spectral graph wavelets. As the nodes with similar neighborhoods have similar spectral wavelet coefficients, the nodes with similar structures have similar embeddings obtained through GraphWave.

3.5 The Relation between Link Prediction and Community Detection

Link prediction and community structure investigation are two of the foremost topics in complex network analysis. Yet, as shown in the previous sections, there are not many studies that attempt to explore the relationship between the two. By definition, we expect communities to influence which two nodes in a network form links over time. Ignoring the community phenomenon observed in real-world networks during link prediction means we are attempting to predict links with incomplete information.

One of the objectives of this study is to reveal possible relations between link prediction and community structure. To achieve this objective, we conduct a wide-ranging comparative analysis between the link prediction techniques introduced previously. We want to explore the relationship through three factors : (i) whether we have access to community information, (ii) how salient the communities in the network are, and (iii) the size of the communities in the network. In order to understand the impact of having access to community information, we run our experiments across the entire network, which we refer to as “network-wide”, and within each separated community of the same network, which we refer to as “community-based”. Thus, we verify whether we can improve the link prediction success by using link prediction methods on a per-community basis or not. By changing the parameters of the network generation algorithm we use to measure the effect of other parameters, we generate networks with different community structures. Then, by performing link prediction on these networks, we observe the performance change of the same link prediction method on networks generated with different parameters.

Synthetic benchmark networks are used in the experimental evaluation to ensure a precise and reliable assessment of community attributes. As in previous works, we employ the LFR model for synthetic network generation, but we generate larger networks with realistic properties in order to mimic the real-world networks more accurately.

3.5.1 Synthetic Network Generation

Since the true community structure of real-world networks is usually unknown, one cannot quantitatively evaluate the effect of community characteristics on methods such as community detection, link prediction, etc. Usually, artificial networks with predefined community structures are used for such evaluations. These networks are generated by mathematical models whose parameters change the final network's structural properties. In this work, we aim to find out the possible effect of community structure characteristics on well-known link prediction methods' performances.

The Lancichinetti-Fortunato-Radicchi (LFR) artificial network generation model (Lancichinetti et al., 2008), extracts networks with a community structure. It is possible to change the characteristics of the community structure with various parameters of the model. For these reasons, we use this model for our experiments. LFR has two steps : first, it generates a scale-free network by using a well-known configuration model, and second, it applies a rewiring process to the links in order to set the communities. It uses several different parameters for controlling the resulting network and community structure. The distribution of community sizes and node degrees adheres to a power law with separate exponents β and γ respectively. The algorithm grants control over community structure with the mixing parameter μ. This parameter represents the average fraction of each node's links with a node that does not belong to its community. A high μ value results in networks where the boundaries of communities become blurred and community structure loses clarity. A low μ value generates networks where community boundaries are clear and each node tends to have neighbors from its community. This effect of different μ values on the network structure is visualized in Fig. 3.2. Apart from the mentioned parameters, LFR allows for the control of the number of nodes, the average and maximum degrees in the network, and the minimum and maximum sizes of the communities.

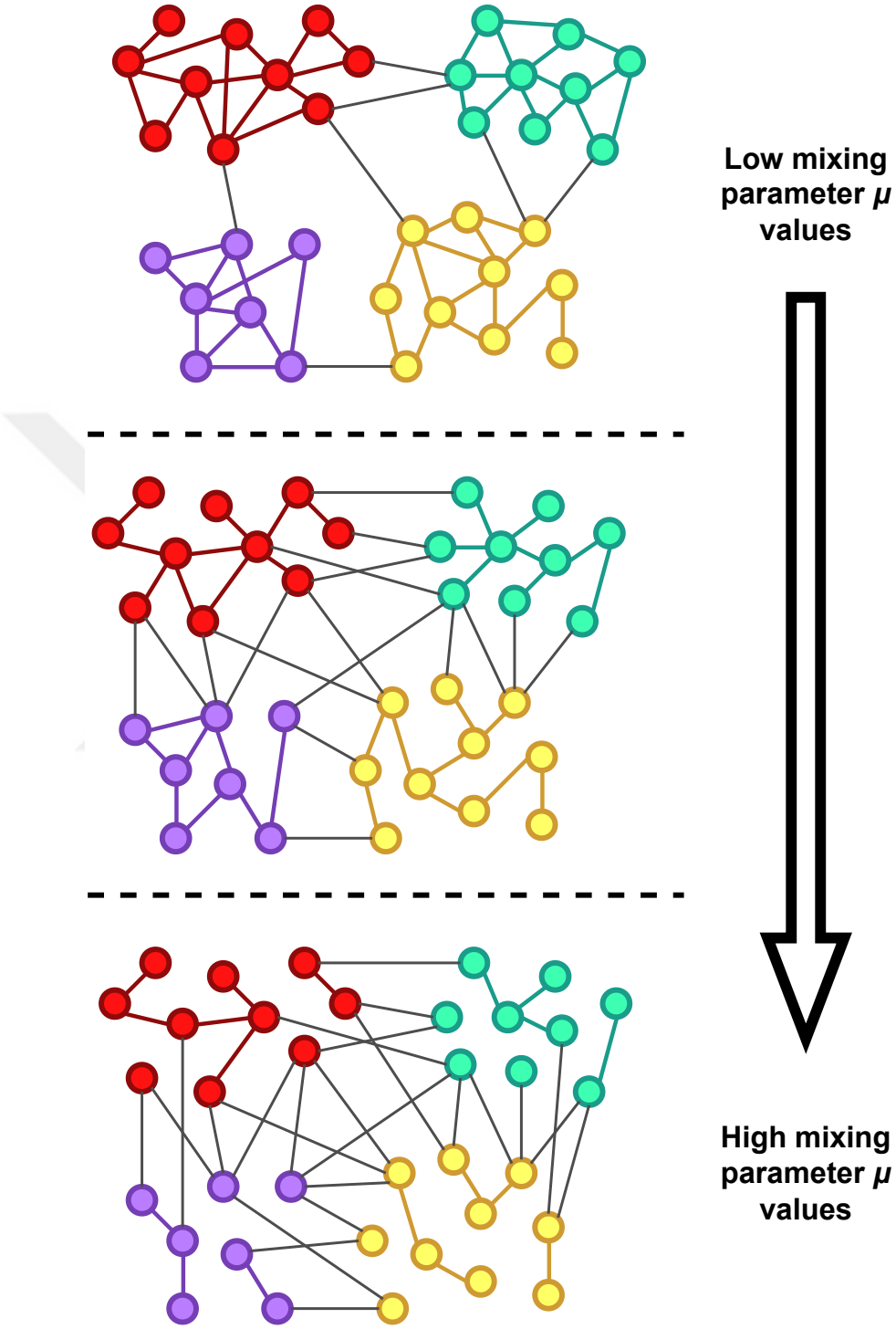


FIGURE 3.2 – An illustration depicting the effect of changing μ value has on the resulting network generated by LFR algortihm.

4 LINK PREDICTION FOR BIPARTITE NETWORKS

As mentioned earlier, bipartite networks require unique approaches. This is notably the case when performing link prediction, a task that consists of determining which links are missing from a network known to be incomplete, or which links are likely to form in the forthcoming state of an evolving network. When it comes to link prediction, one can broadly distinguish three main approaches for bipartite networks : (i) ranking candidate pairs of nodes using a heuristic metric that does not rely on any learning process (Lü and Zhou, 2011), (ii) leveraging the network topological features to train a predictor into predicting such score (Al Hasan et al., 2006), and (iii) perform representation learning to produce graph embeddings and use them during classification instead of features (Grover and Leskovec, 2016). In addition, we propose a fourth approach by leveraging state-of-the-art GCN-based personalized recommendation techniques (Gao et al., 2023) to perform link prediction in bipartite networks.

Bipartite networks are characterized by their nodes being distributed over two mutually exclusive sets U and V, and their edges $L \subseteq U \times V$ connecting only nodes from two different sets. Let us denote $L' = (U \times V) \setminus L$ as the set of links missing from the network. The problem of link prediction can then be defined as finding the proper links in L' that are more likely to occur in the next evolution of the network. This prediction can be performed through many different approaches, many of which rely on the correct scoring of candidate links of L' by learning-based methods. They all use the current network structure, which is encoded in L. In this section, we describe the different approaches that we later assess for the link prediction task in bipartite networks. For the remainder of the thesis, we will refer to the nodes in U as "left nodes" and the others in V as "right nodes".

4.1 Traditional Link Scores

Most of the existing link scores rely on the concept of triadic closure observed in social networks (Opsahl, 2013), which proposes that the more neighbors two nodes share,

the more likely they are to be connected. In bipartite networks, however, two nodes of the same type cannot be directly connected, and there is consequently no triangle. These triangle formations or lack thereof in unipartite and bipartite networks can be seen in Fig. 4.1. In the figure, triangle formations are observed between the nodes

$n_1 - n_2 - n_3$ and $n_7 - n_8 - n_9$ on the unipartite network. In the bipartite network, it can be seen why triangles cannot be formed between nodes $v_1 - u_1 - v_3$ and $u_3 - v_4 - u_5$. This is because $v_1 - v_3$ and $u_3 - u_5$ connections must be realized for the triangle to be completed. However, as mentioned before, in bipartite networks, these triangles cannot be completed because the links between nodes of the same type are prohibited.

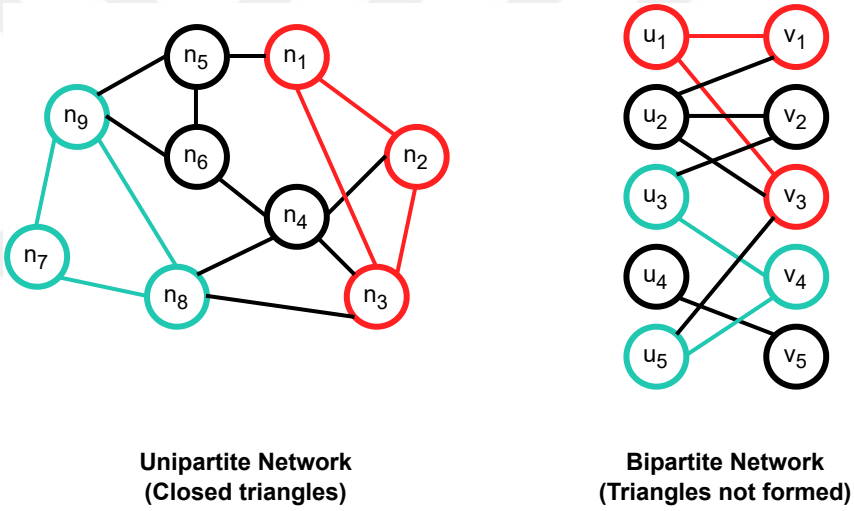


FIGURE 4.1 – Triangle formations shown in a unipartite network and a bipartite network.

Hence, we employ link prediction methods that do not rely on triadic closure : L3, Katz, LP, PA, SPM, and Dist. These link prediction indexes are introduced previously in Chapter 3 and applied as seen in Fig. 4.2. Among these methods, Katz and LP

function slightly differently compared to their unipartite counterparts. In bipartite networks, even powers of A allow counting paths that connect nodes of the same type. In our case, x and y belong to different node sets. As a result, the term $(A^2)_{x,y}$ from

Eq. 3.15 takes 0 for all (x, y). Thus, it is possible to simplify this equation to $LP = A^3$.

It should be noted, however, that this simplified version is not equivalent to L3, as it does not apply degree-normalization to the results. Similarly, Katz will only consider the paths of odd length. For comparing how the length of paths affects link prediction performance, we also propose using two variations of L3, which we refer to as L5 and L7. These indexes are calculated in the same way, except that they count paths of length five and seven respectively.

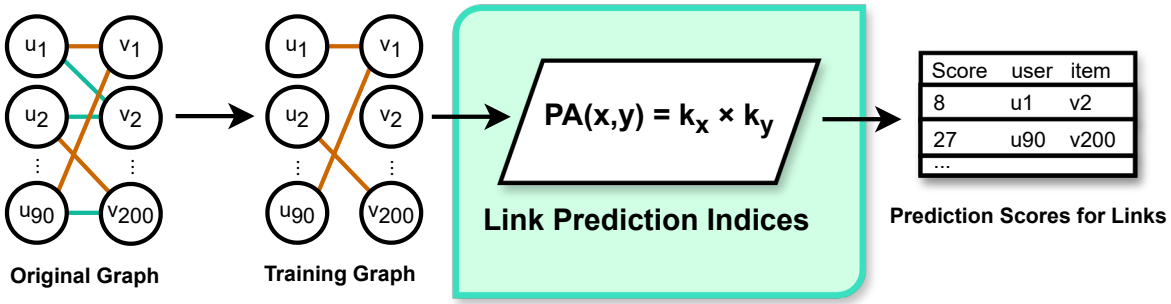


FIGURE 4.2 – Diagram showing how the resulting scores are obtained from link prediction indexes (Section 4.1).

4.2 Link Prediction via Topological Features

One of the common methods of link prediction relies on supervised learning techniques. In this work, we propose to represent each node pair with network-related topological features and label them with the positive or negative classes depending on whether there is a link in L_{train} or not, respectively. Then, we apply classification methods. A detailed schema of the framework we propose can be seen in Fig. 4.3. At first, we split the network set into training and testing subsets. Then, all node pairs from each network are represented by the PageRank (Page et al., 1999) scores of the nodes involved, as well as the PA score of each link.

We also employed the following centrality scores previously reviewed by Landherr et al. (2010) :

— *Degree centrality (DC)* measures the centrality of a node by the number of its neighbors (Nieminen, 1974). In our study, we used a normalized variation of it where it is divided by the number of possible connections the node can make. Assuming a bipartite network with two sets of nodes U and V with $n = |U|$ and $m = |V|$, degree centrality of the node x is calculated as shown in Eq. 4.1 where $k(x)$ is the degree of the node x.

— *Closeness centrality (CC)* assigns higher centrality to nodes with shorter distance to other nodes (Beauchamp, 1965). This is done by taking the reciprocal of the sum of its distance to all other nodes. This value is divided by minimum distance possible as the distance between nodes is affected by the lack of links between same type of nodes in bipartite networks. Continuing the assumption made previously, closeness centrality of the node x is calculated as shown in Eq. 4.2 where d is the sum of the distances from x to all other nodes.

— *Betweenness centrality (BC)* of node x shows how frequently the shortest path between a node pair passes through x. Higher values indicate that x takes the role of a connector between two other nodes more frequently (Freeman, 1978). Assuming the same notation used in previously described centrality measures, the betweenness centrality values of nodes in U are normalized by diving these value by Eq. 4.3 and the values that belong to V are divided by Eq. 4.5.

— *Eigenvector centrality (EC)* assumes connections with high-scoring nodes to be more valuable than those with low-scoring nodes (Bonacich and Lloyd, 2001).

— *Katz centrality (KC)* of a node is the sum of Katz index scores between that node and every other node in the network (Katz, 1953).

$$DC(x) = \frac{k(x)}{m}, \text{ for } x \in U, \text{and} DC(x) = \frac{k(x)}{n}, \text{ for } x \in V \qquad (4.1)$$

$$CC(x) = \frac{m + 2n - 2}{d}, \text{ for } x \in U, \text{and} CC(x) = \frac{n + 2m - 2}{d}, \text{ for } x \in V \qquad (4.2)$$

$$\frac{1}{2}[m^2(s+1)^2 + m(s+1)(2t-s-1) - t(2s-t+3)] \quad (4.3)$$

where,

$$s = (n-1) \div m \text{ and } t = (n-1) \bmod m \quad (4.4)$$

$$\frac{1}{2}[n^2(p+1)^2 + n(p+1)(2r-p-1) - r(2p-r+3)] \quad (4.5)$$

where,

$$p = (m-1) \div n \text{ and } r = (m-1) \bmod n \quad (4.6)$$

For an unbiased training process, we use uniformly random negative sampling to include as many negative instances as positive ones, since the class numbers are highly imbalanced. We employ the XGBoost (Chen and Guestrin, 2016) classifier model for supervised learning. It is an ensemble learning model that trains multiple gradient-boosted decision trees as estimators. We refer to this method as XGB in the rest of the thesis. Another way we represent links with mentioned features is by applying Principal Component Analysis (PCA) to reduce the data to a single dimension. Afterwards, we treat this single dimension as link prediction scores. PCA is a widely used method in data analysis, especially for dimensionality reduction. The goal when applying PCA is to reduce the dimensionality of the data while keeping the variance between instances as high as possible. That being said, PCA does not necessarily reduce the dimensionality of the data. In essence, PCA applies an orthogonal linear transformation in the inner product space (Jolliffe, 2002). This reorients the data to a new coordinate system. In this new coordinate system, the coordinates are called "components". The first component has the highest variance and the variance gradually decreases for each following component. Linear Discriminant Analysis (LDA) is also used in similar manner to PCA for comparison purposes.

Linear Discriminant Analysis (LDA) is a technique used for dimensionality reduction and classification. Unlike PCA, which focuses on maximizing the variance in the data, LDA aims to find the linear combinations of features that best separate the classes. When a dataset with two classes is considered, where each class has its own distribution of instances in the feature space, the goal of LDA is to project these instances onto a

lower-dimensional subspace in such a way that the classes are well-separated. It can be said that while PCA focuses on differences, LDA focuses on similarities. However, since they are both methods that try to find linear combinations of variables, there is overlap in their application areas. It should be noted however, prediction function of LDA is not used to obtain the results in this study. Similar to PCA, fitted LDA model is used for dimension reduction purposes and the resulting single dimensional data is treated as the prediction scores. We evaluate XGB, PCA, and LDA results separately in the experiments.

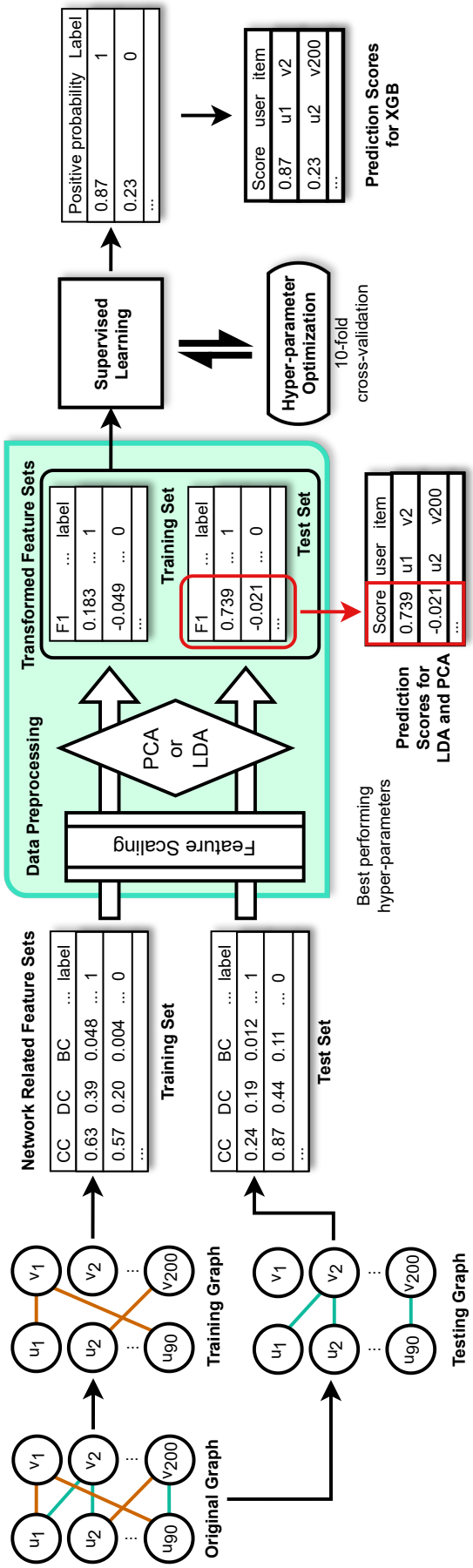


FIGURE 4.3 – Representation of the link prediction method that relies on network topological features (Section 4.2).

4.3 Embedding-based Link Prediction

In recent years, network representation learning, a.k.a. graph embedding, has gained high attention because it allows for the representation of network objects, i.e., nodes, links, or whole networks, as lower-dimensional real-valued vectors. The aim of these embedding techniques is to reflect the topological differences between the objects in the network into the novel embedded features. Some popular methods developed for this goal include DeepWalk (Perozzi et al., 2014), node2vec (Grover and Leskovec, 2016), GraphSAGE (Hamilton et al., 2017), and LINE (Tang, Qu, Wang, Zhang, Yan and Mei, 2015). These generic embedding methods are all unsupervised. They employ techniques such as random walks, skip-gram models, and neural networks to generate vector representations. However, they are not developed to handle bipartite networks and may overlook the differences in node types. Embedding techniques developed for bipartite networks are rare : in this article, we use metapath2vec, BiNE, and BiGI.

Dong et al. (2017) introduced **metapath2vec (MP2V)** as an unsupervised embedding method for multipartite networks. Taking inspiration from node2vec (Grover and Leskovec, 2016), it applies skip-gram maximization along with a novel approach that employs meta-path guided random walk. These strategies allow capturing both structural and semantic correlations across different node types. Meta-paths have been used in graph algorithms in the past studies (Sun et al., 2012). Using Fig. 4.5 as an example,

in the original graph $u_1 - v_2 - u_2$ is a meta-path that describes the relationship between users via items which is then denoted as $U \to V \to U$. This concept is often used in the study of multipartite networks, as it is used to describe the same type of nodes over different types of nodes. If the original graph included a third node type symbolized by y, we could construct a meta-path $U \to V \to Y \to V \to U$. The meta-path guided random walk technique of metapath2vec works by introducing bias to random walk according to this pattern. When applied to such a network, the random walk will be biased towards selecting a node $y \in Y$ if the current node is $v \in V$ and the last visited node was $u \in U$. Naturally, this process can also be applied to bipartite networks but only with one meta-path pattern. For this reason, we expect the impact of skip-gram maximization to be more pronounced than meta-path guided random walks when it comes to improvement metapath2vec provides over node2vec and DeepWalk

algorithms.

BiNE is bipartite network embedding method that attempts to capture the structural properties of bipartite networks by executing biased random walks (Gao et al., 2018). The bias aspect of the random walks allows BiNE to model the implicit and explicit relationships between the nodes of the network to preserve said structural properties such as the distribution of node degrees. The importance of biased random walks and implicit relations is shown by experiments comparing the results with and without those features. Researchers mention a limitation of BiNE, that it may not be effective for nodes with limited number of or no edges, as it solely relies on the information obtained from observed edges. This limitation arises from the fact that real-world applications often involve missing data, where the observed edges may not provide enough meaningful information about the relationships between vertices.

To overcome the drawback of random walk methods in capturing global structure, Cao et al. (2021) proposed **BiGI** (Bipartite Graph Embedding via Mutual Information Maximization). The graph encoder they implemented with a GNN architecture uses two-hop neighbors for neighborhood aggregation. The authors do not consider using immediate neighbors during aggregation to be the proper approach as they will be of different types than the target node. To illustrate with Fig. 4.4, let's assume that u_3 is our target node. The immediate neighbors of this node are v_1, v_2, v_3 and two-hop neighbors are u_1, u_2, u_4, u_5. For two-hop neighbor aggregation, we first extract the representations of the immediate neighbors of the target node. Contrary to standard aggregation, these nodes' own features are not included. This allows the resulting representation to be obtained without using the features of a node of type v. For example, to obtain the representation of v_1, the features of u_1, u_2, and u_3 are used. Once the temporary representations of immediate neighbors are obtained, the representation for the target node is obtained using their and the target node's features. This is done by performing a non-linear transformation via the LeakyReLU activation function (Maas et al., 2013). After obtaining the local and global representations of the nodes, the global properties of the network are derived using an infomax objective.

As the methods explained above take inspiration from previous node embedding algorithms such as node2vec, they also inherit their generic nature. This certainly allows

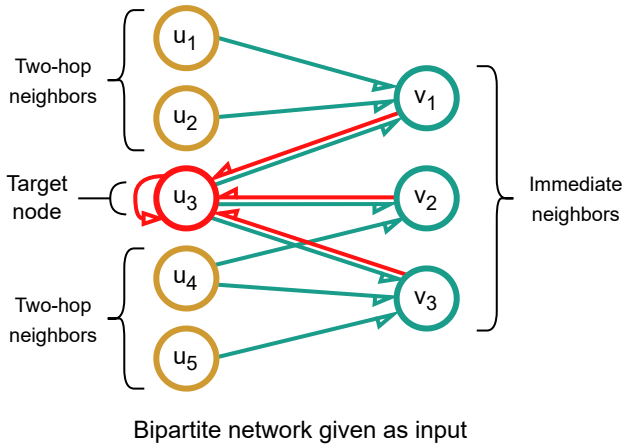


FIGURE 4.4 – The aggregation strategy of BiGI embedding method as described in Cao et al. (2021). The blue arrows pointed towards nodes of type v blue arrows indicate which nodes' features were used to aggregate the representations of those nodes. The red arrows describe the aggregation done to obtain the target node's representation.

for a more versatile algorithm that can cater to a wide array of tasks. At the same time, it hinders the accuracy of the algorithm, as generic approaches are unable to model the fine details that are often crucial.

These embedding methods are not directly developed for link prediction tasks. However, the openly available versions of BiNE and BiGI allow for link prediction by feeding embedding vectors to a logistic regressor while metapath2vec only produces vector embeddings of the nodes. Using the same approach used in BiNE, we utilize embeddings produced by metapath2vec to train a logistic regression model. Each node pair is represented by combining left node and right node embedding vectors. This binary classifier not only labels the node pairs but also generates a prediction score for each of them. This score estimates how likely that node pair is to belong to the positive class, which is to say, how likely it is for a link to exist between them. We treat these scores as link prediction scores and rank them accordingly, as illustrated in Fig. 4.5.

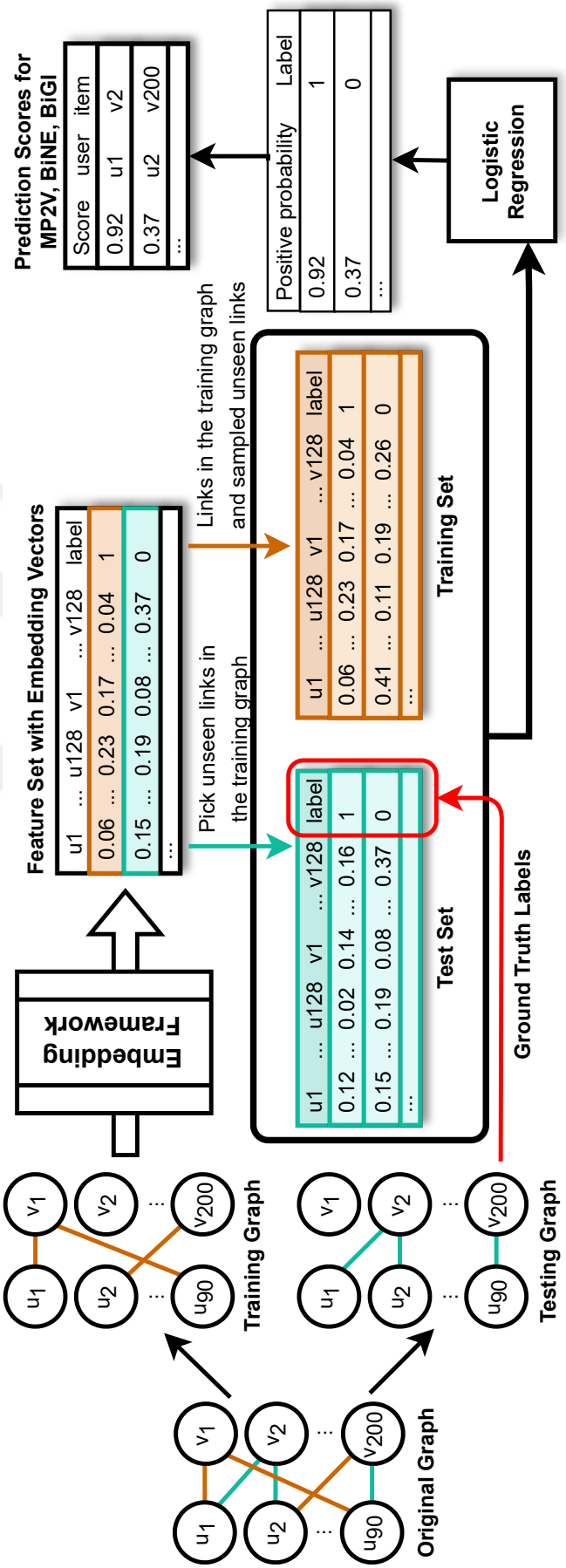


FIGURE 4.5 – Representation of the link prediction method that relies on graph embeddings (Section 4.3).

4.4 Personalized Recommendation Conversion

Graph neural networks (GNN) and graph convolution networks (GCN) have been frequently used to handle bipartite networks in recent years. These techniques are usually employed for recommendation tasks. Basically, the user-item purchase relation is modeled under the bipartite network structure. Then, the main objective of those methods is to rank the k most likely items for each user.

These approaches also generate separate embedding vectors for the nodes in U and V, corresponding to user and item sets, respectively. Unlike the previously mentioned embedding-based methods however, those embedding vectors are generated specifically for the personalized recommendation task and trained with supervised learning. The embedding vectors are extracted through a GCN (or GNN), where each node's embedding vector affects those of its neighbors. On top of that, the training and prediction phases of these models are designed with this task in mind, allowing them to optimize the model for a more accurate ranking of items.

The output of personalized recommendation models is a ranked list of items for each user. These items are sorted based on ranking scores, which are derived from the inner product of user and item embedding vectors. In this work, we reformulate the output sets of these approaches to employ them in link prediction tasks. In order to capitalize on the effective item recommendations made from these methods, we consider the ranking score of an item v for a user u as the predictive score assigned to the connection between nodes u and v, as both user and item representations influence the ranking score of the item. Our proposed approach is shown in Fig. 4.6.

4.4.1 Bayesian Personalized Ranking (BPR)

BPR relies on Bayesian learning (Rendle et al., 2009). In traditional machine-learning-based recommendations, the training set is built up using user-item pairs (Hu et al., 2008). If the user already prefers the item, then the class label is positive, and vice versa. Differently from them, BPR uses the user and two items as training data and optimizes for correctly ranking item pairs for each user instead of scoring a single item.

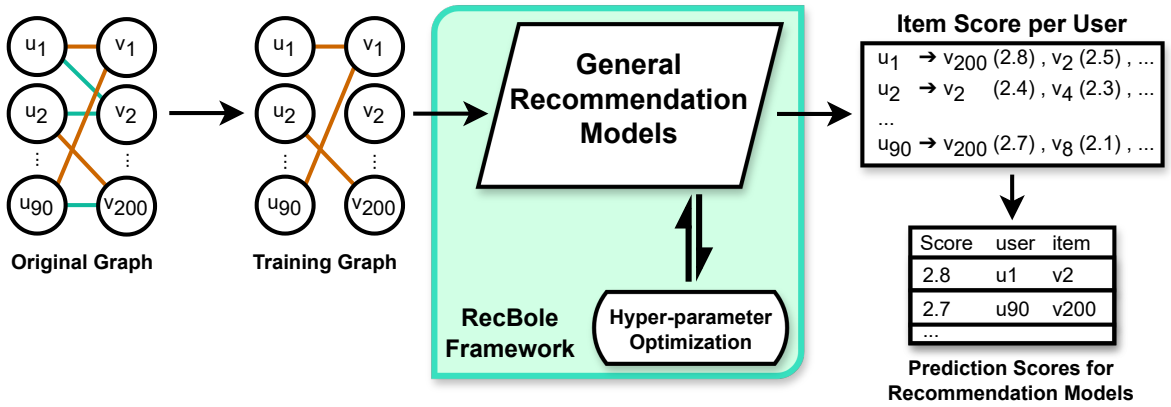


FIGURE 4.6 – Representation of the link prediction method based on the repurposing of item recommendation techniques (Section 4.4).

It optimizes a personalized ranking-focused metric whose equation is given in Eq. 4.7) via stochastic gradient descent. This likelihood function represents a user's preference for an item over another and a prior probability. Here, Θ is the model parameter vector to be inferred, which captures the special relationship $>_u$ of a user u and his item preferences. The reformulation of the training data not only allows for ranking-focused optimization, it also solves the imbalance between positive and negative instances in the training data.

$$\text{BPR}_{\text{Opt}} = \ln p(>_u |\Theta)\ p(\Theta) \qquad (4.7)$$

4.4.2 Neural Graph Collaborative Filtering (NGCF)

NGCF utilizes high-order connectivity in user-item interactions to improve the user and item embeddings (Wang et al., 2019). The researchers identify two key components in collaborative filtering methods : (1) embedding, generating vector representations of users and items ; and (2) interaction modeling, reconstruction of user-item interactions based on the embeddings. Previous methods, such as the MF and NCF models, used the inner product or nonlinear neural networks. The researchers argue that the lack of encoding for collaborative signals during the embedding process hinders the ability of such methods to generate accurate embeddings. In NGCF, the initial embedding is

enhanced through first-order and high-order propagation. This allows for n-hop neighbors to contribute to the embedding of users and items. In other words, the propagation layers can capture the collaborative signal when multiple of them are stacked. Lastly, the prediction layer concatenates the different embedding vectors obtained from different propagation layers for the same user (or item) to produce the user's (or item's) final embedding. The Eq. 4.8 shows the final step of item recommendation, where the inner product is used. $\mathbf{e}_u$ and $\mathbf{e}_u$ are the obtained embedding vectors for user u and item i, respectively.

$$\text{NGCF}(u, i) = \mathbf{e}_u^T \mathbf{e}_i \tag{4.8}$$

4.4.3 Light Graph Convolution Network (LightGCN)

LightGCN attempts to create node embeddings by iteratively performing graph convolution (He et al., 2020). The researchers suggest that feature transformation and nonlinear activation of GCNs (and the NGCF that inherits them) are not necessary for the collaborative filtering task. Due to the lack of user and item attributes in collaborative filtering, performing nonlinear feature transformations slows down the learning process and negatively affects the accuracy of the recommendations. Instead, LightGCN continuously performs weighted sum neighbor aggregation. The convolution operation is described as in Eq. 4.9 by the authors. $\mathbf{e}_u^k$ and $\mathbf{e}_i^k$ represent the embeddings of user u and item i, respectively, after k layers of propagation. N_u and N_i are the sets of items the user u interacted with and the set of users the item i interacted with, respectively.

$$\begin{aligned} \mathbf{e}_u^{k+1} &= \sum_{i \in N_u} \frac{1}{\sqrt{|N_u|}\sqrt{|N_i|}} \mathbf{e}_i^k \\ \mathbf{e}_i^{k+1} &= \sum_{u \in N_i} \frac{1}{\sqrt{|N_u|}\sqrt{|N_i|}} \mathbf{e}_u^k \end{aligned} \tag{4.9}$$

After K layers of propagation, the layers are combined, as seen in Eq. 4.10. The regularization parameter a_k describes the importance of the layer. It can be tuned

automatically or manually, as the authors did by setting it to $1/(K+1)$.

$$\mathbf{e}_u = \sum_{k=0}^{K} a_k \mathbf{e}_u^k$$
$$\mathbf{e}_i = \sum_{k=0}^{K} a_k \mathbf{e}_i^k \tag{4.10}$$

As in NGCF, in LightGCN the inner product is taken for the final embeddings (Eq 4.8). The BPR loss metric is employed during the training process.

4.4.4 Self-supervised Graph Learning (SGL)

SGL aims to improve the recommendation performance of the previous GCN models like LightGCN by using data augmentation techniques tailored for networks (Wu et al., 2021). Frequently used in computer vision tasks, data augmentation allows the model to be trained better by representing the same data in different ways. Researchers use three different approaches to perform data augmentation on the network :

1. Node Dropout. The nodes in the network are randomly removed from the network, with probability ρ, along with the links they have made.
2. Edge Dropout. The edges in the network are dropped from the network with probability ρ.
3. Random Walk. To populate the graph convolutional layers with different subgraphs, random walk is applied for each node in the network to drop edges randomly.

It should be noted that when node dropout and edge dropout are applied, the same edges are present in each layer of the GCN. In other words, these operations create a subgraph by modifying the original graph consisting of the links on the network. Then, the edges in this subgraph remain the same in different layers. In the case of random walk however, the edges may differ from layer to layer. The different techniques applied

can be seen in Fig. 4.7. As shown in the figure, when edge dropout is applied, the link between u_1 and v_3 is dropped in all three layers. During node dropout, u_1 and v_3 are removed from all layers. When random walk is applied, we see that the link between u_3 and v_4 existing in L1 is dropped in L2 and L3.

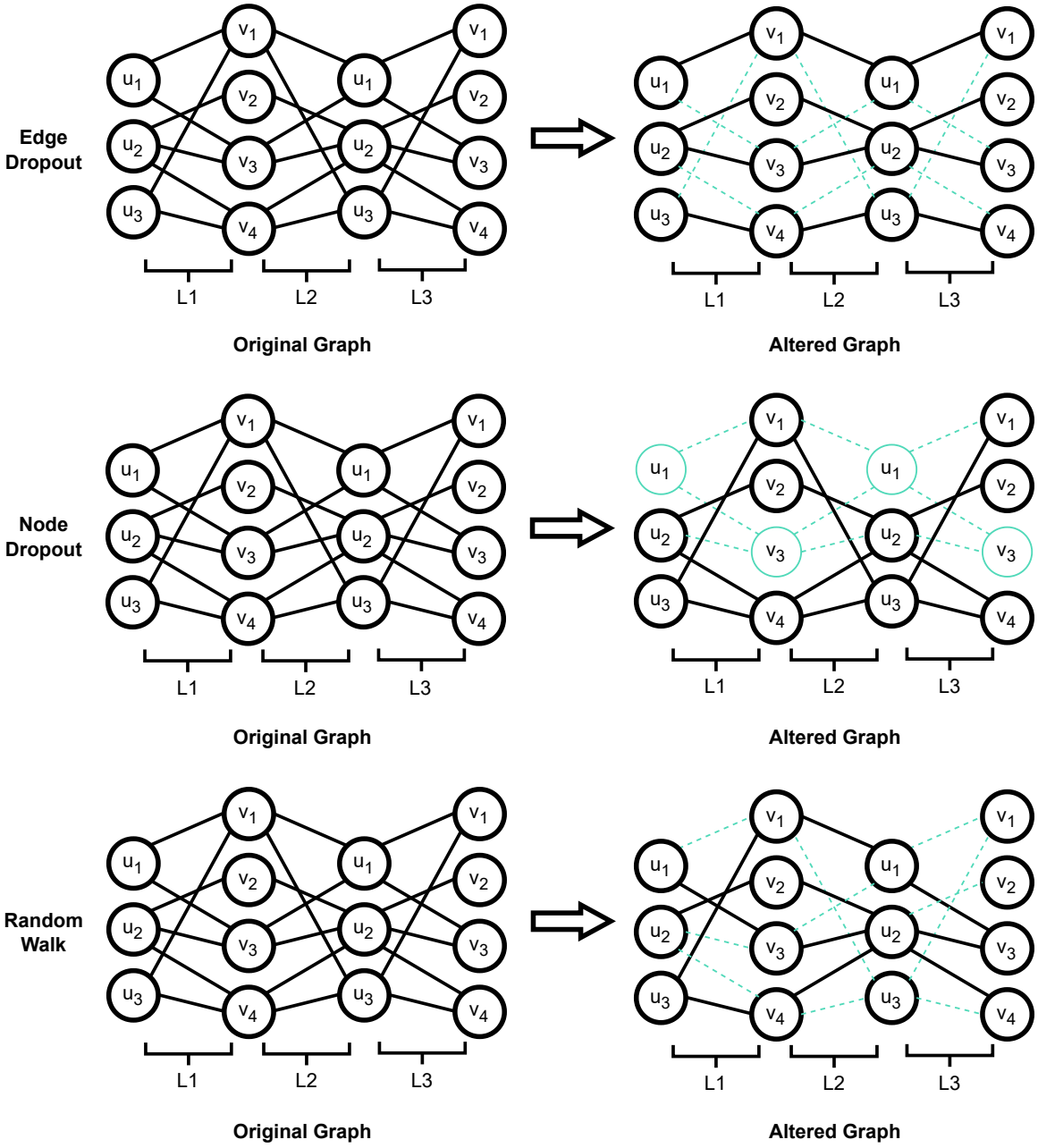


FIGURE 4.7 – Different techniques of data augmentation used in SGL. Light blue color and dashed lines indicate nodes and links that are removed from the graph that describes the network taken as input.

4.4.5 Diffusion Recommender Model (DiffRec)

DiffRec utilizes neural networks for de-noising purposes and obtaining interaction probabilities (Wang et al., 2023). Through the iterative denoising process, the model can learn to generate interactions without being affected by noise in the training data, taking into account complex interaction patterns and dynamics. As the model outputs a probability of occurrence for the interactions it generates, these interactions can be ranked according to their probability and the most probable ones are used in the item recommendation. This is accomplished in two phases. In the first one, called the forward process, the interaction data is corrupted using Gaussian noise. The other phase, called reverse process, aims to learn to reduce this noise in an iterative manner as shown in Eq 4.11. Here, $\mu_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$ are the mean and variance of the Gaussian distribution used to add noise to the data, predicted by the neural network with parameters θ.

$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \quad (4.11)$$

The process depicted in the equation recovers the state x_{t-1} without the added noise that is present in state x_t.

5 PERFORMANCE EVALUATION OF LINK PREDICTION

In this thesis, we employ a supervised learning methodology to undertake the link prediction task. links with the highest scores with the links from the test set. First, a fraction of the original network's links are removed to serve as the positive samples of the test set. The absent links, which represent the negative samples, are merged with the removed links to form the test set. Following that, each of the aforementioned indexes is applied to the network formed by the remaining links to produce scores for each link in the test set. Finally, performance evaluation is conducted with the metrics that will be explained below. The evaluation of the link prediction task can be thought of as a binary classification, such that a positive class signifies the existence of a link between two nodes, while a negative class indicates that there is no link.

— True Positive (TP) : Existing link predicted as positive.
— False Positive (FP) : Non-existing link predicted as positive.
— False Negative (FN) : Existing link predicted as negative.
— True Negative (TN) : Non-existing link predicted as negative.

The methods used in the evaluation of the link prediction task can be divided into two categories. First, fixed threshold methods rely on different threshold types such as prediction scores or the number of correctly classified instances. The second category is the threshold curves. Threshold curves do not require a calibration of thresholds and therefore reflect the model's performance more accurately.

5.1 Fixed Threshold Methods

We use **Recall@K** from this category, a metric commonly employed in the information retrieval domain. It is calculated by measuring the well-known recall (Eq. 5.1) but only for the top K predictions rather than the entire prediction set. K, in this context, corresponds to the length of the number of positive samples in the test set (Lü and

Zhou, 2011; Zhou, 2021). Therefore, after a link prediction score is obtained for every node pair, only the top K of these are considered while calculating the Recall@K. Likewise, **Precision@K** is calculated by applying the precision formula shown in Eq. 5.2 to top K predictions.

$$Recall = \frac{TP}{TP + FN} \tag{5.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

$$FPR = \frac{FP}{FP + TN} \tag{5.3}$$

5.2 Threshold Curves

The thresholds used in our work correspond to the different values that the mentioned indexes compute for the possible links. The resulting predictions are then ranked in descending order, based on their scores, for each index. Subsequently, threshold curve methods are applied to the ranked prediction outcomes, wherein predictions with scores lower than the threshold are classified as negative, while those with scores higher than the threshold are classified as positive. In this study, Average Precision, AUROC, and AUPR are used to represent threshold curve methods. We explain them in the following part.

Average Precision is calculated by taking the mean of Precision@K values for each threshold on predicted links which is expressed in Eq. 5.4.

$$Average\ Precision = \frac{1}{|L|} \sum_{K=1}^{|P|} \text{class}_K \times \text{Precision@}K \tag{5.4}$$

where $|L|$ is the number of the positive samples in the test set, $|P|$ is the number of the results which means all possible links, and class_K equals to 1 if K is a positive sample, 0 otherwise.

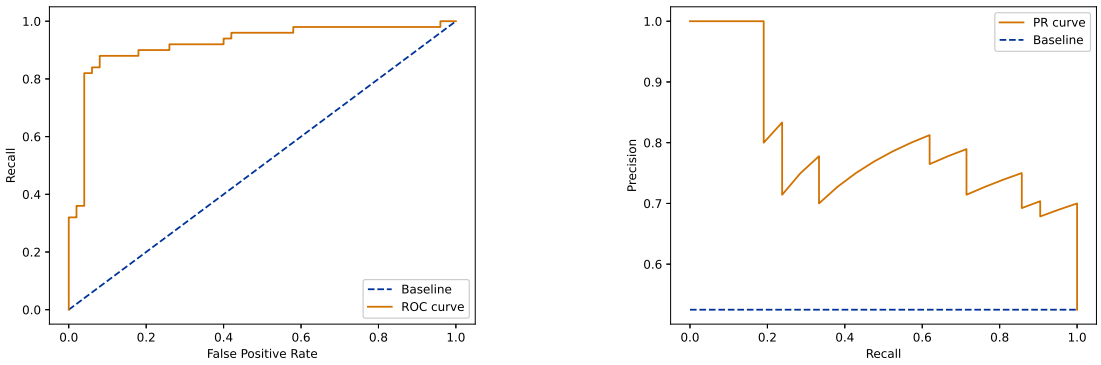
Area under the receiver operating characteristic curve (AUROC) relies on the concept of the receiver operating characteristic curve (ROC). The ROC curve is the plot of the false positive rates (FPR) versus the recall for each threshold. False positive rate is calculated as seen in Eq. 5.3. The AUROC metric refers to the area under the ROC curve. The value of this area is 1 for a perfect classifier and 0.5 for a random classifier. This means that the closer the AUROC value is to 1, the closer the model is to the perfect classifier. Using the AUROC value, we can compare the success of the models without the need to examine the ROC curves individually. Measurement of the area under the ROC curve is done through integral as seen in Eq. 5.5. In Fig. 5.1a, an exemplar of a ROC curve is illustrated.

$$AUROC = \int_0^1 \text{Recall}\, d(\text{FPR}) \tag{5.5}$$

Area under the Precision–Recall curve (AUPR) is obtained the same way as AUROC but on the Precision-Recall curve. In other words, it measures the area under the Preicison-Recall curve, which is the curve drawn with recall on the x-axis and precision on the y-axis. An instance of a Precision-Recall curve is depicted in Fig 5.1b.

AUPR is often used in binary classification tasks where class imbalance is encountered, such as link prediction. Class imbalance occurs when class severely outnumbers another class. In the case of link prediction, the negative class is the total number of every possible link in a network minus the number of existing links. The negative class includes all the links that can exist in the network except the links that actually exist. Since the number of links that can exist in a unipartite network with n nodes is $\frac{n\times(n-1)}{2}$, we see that as the network grows, the negative class can quickly outpace the positive class. AUPR is shown as the most suitable metric for evaluating the success of link prediction methods in Yang et al. (2015).

Both AUPR and AUROC range from 0 to 1. They do not depend on thresholds, offering general assessments that can cover different prediction scenarios.



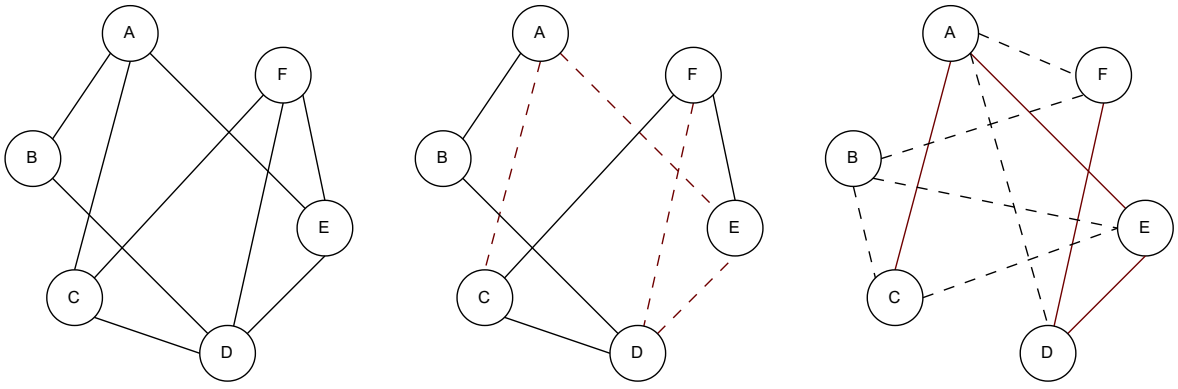
(a) ROC Curve

(b) PR Curve

FIGURE 5.1 – An illustrative example of Receiver Operating Characteristic and Precision-Recall curves. The curves themselves are continuous and colored orange, while the dashed blue line represents the baseline.

5.3 Evaluation example

We demonstrate the link prediction evaluation process using a network with 6 nodes and 9 links. As seen in Fig. 5.2, 4 links are removed from the initial network to be used as positive class instances in the test set. Remaining links make up the training set. Negative class instances in the test set are the non-existent links in the initial network.



(a) Initial network.

(b) Continuous black links represent the network used for training. Dashed red lines are removed to be used in the testing step as positive class instances.

(c) Network used for the testing step. Black dashed links are negative class instances and continuous red lines are positive class instances.

FIGURE 5.2 – Networks used in training and testing phases.

Assume that the method used for link prediction scores the links as shown in Table 5.1. In this experiment, K parameter for Recall@K is equal to 4, which is the number of positive class instances. The recall value obtained from the top 4 highest scoring links is 0.5, as there are 2 true positives (A-E and D-E pairs) and 2 false negatives (A-C and D-F pairs). The result of writing the values for each parameter in the formula for Average Precision is seen in Eq. 5.6. To draw ROC and Precision-Recall curves, we will first display the precision, recall, and false positive rate values for each threshold in Table 5.2. Each column marks the threshold for the lowest score that can be achieved. Links with a score equal to or higher than this value are predicted as positive, while others are predicted as negative. Recall values are used with corresponding precision or FPR values to plot the Precision-Recall curve or ROC curve respectively. The resulting curves are shown in Fig. 5.3 and the area under the curves can be calculated by various methods.

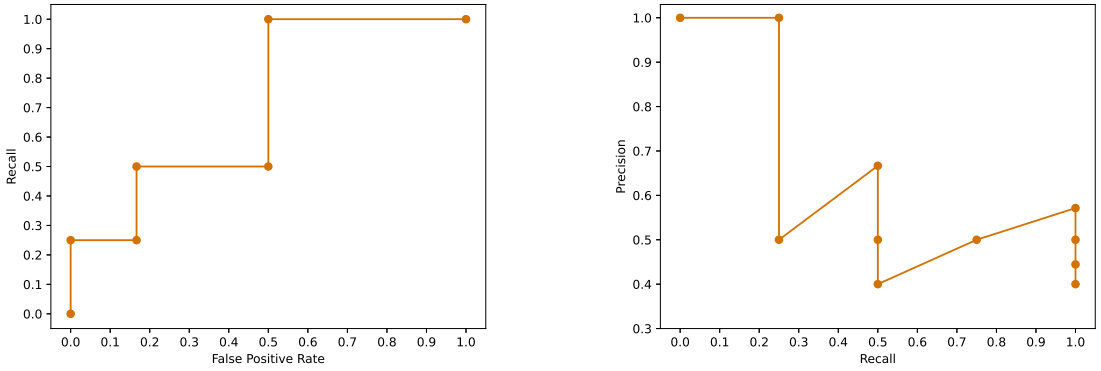
TABLE 5.1 – Prediction scores obtained for each link between given node pairs.

Node pair	A-E	B-F	D-E	B-C	A-D	A-C	D-F	B-E	C-E	A-F
Score	0.96	0.91	0.89	0.78	0.71	0.64	0.57	0.42	0.36	0.13
Real class	1	0	1	0	0	1	1	0	0	0

$$
\begin{aligned}
\text{Average Precision} &= \frac{1}{4}\sum_{K=1}^{10} \text{class}_K \times \text{Precision@}K \\
&= \frac{1}{4} * (1*1 + 0*\frac{1}{2} + 1*\frac{2}{3} + 0*\frac{2}{4} + 0*\frac{2}{5} \\
&\quad + 1*\frac{3}{6} + 1*\frac{4}{7} + 0*\frac{4}{8} + 0*\frac{4}{9} + 0*\frac{4}{10}) \\
&= 0.6845
\end{aligned}
\tag{5.6}
$$

TABLE 5.2 – Recall, precision and FPR values obtained at corresponding threshold.

Threshold	0.96	0.91	0.89	0.78	0.71	0.64	0.57	0.42	0.36	0.13
Recall	0.25	0.25	0.5	0.5	0.5	0.75	1	1	1	1
Precision	1	0.5	0.67	0.5	0.4	0.5	0.57	0.5	0.44	0.4
FPR	0	0.17	0.17	0.33	0.5	0.5	0.5	0.67	0.83	1



(a) ROC Curve

(b) PR Curve

FIGURE 5.3 – The ROC and Precision-Recall curves drawn according to values shown in Table 5.2.

6 RESULTS

Having described the methods and metrics that are relevant to the study, in this section, we will describe how the experiments were designed and the results we obtained. We will discuss the results of the experiments through five research questions : **RQ1** : How does the mixing parameter μ affect the success of link prediction ? **RQ2** : Does community size affect link prediction success ? **RQ3** : Which evaluation metrics allow for a more reliable interpretation of link prediction results ? **RQ4** : Can recommender methods be used for link prediction purposes in bipartite networks ? **RQ5** : How do different link prediction approaches for bipartite networks compare in terms of execution time ? These research questions are related to the two topics we address in this thesis. In our experiments to investigate the relationship between communities and link prediction, we will focus on RQ1, RQ2, and RQ3. Whereas RQ4 and RQ5 are research questions related to our work on link prediction methods that can be used in bipartite networks. Since these two topics require different experimental designs and methods, we will present the experimental setups and results in two sections. Under each heading, we will address the research questions separately. Finally, we will summarize our findings by discussing the results.

6.1 Experimental Setup and Results for RQ1, RQ2, and RQ3

To answer the research questions RQ1, RQ2, and RQ3, we need to perform experiments on datasets where we have access to the complete community information. For this purpose, we use benchmark networks generated with the LFR algorithm, which generates networks with communities. The experiments conducted on this topic of the study can be grouped under two categories. The first case is where it is known which community each node belongs to, which we name community-based experiments. The other is when this information is not accessible, which we name network-wide experiments. The distinction between the two types of experiments can be seen more clearly in Fig. 6.1.

6.1.1 Experimental Setup

The parameter values for the generation of benchmark networks with LFR are shown in Table 6.1. These parameters are based on the values used in previous studies (Kumari et al., 2022). We generate networks with 5 iterations for each set of parameters. Here, we specifically set the μ between 0.1 and 0.9, in steps of 0.1, for a total of 9. We choose network sizes of 5000 and 15000. Since our main goal of these experiments is to investigate the effect communities have on the link prediction task, we generate different networks with small and large maximum community sizes. The set of networks generated at sizes 5000 and 15000 are denoted as Net5 and Net15 for the rest of this work. For the networks in Net15, Net15A refers to the ones with a maximum community size of 200, and set Net15B refers to the ones with a maximum community size of 5000. Among the indexes mentioned in Chapter 3, the damping parameters of LP and Katz are used as $\epsilon = 0.01$ and $\beta = 0.001$, respectively.

Within the scope of these experiments, the previously mentioned link prediction indexes are calculated for each sample network, both network-wide and community-based. Then the success of each index is determined by the mentioned performance evaluation metrics. In the network-wide experiments, we select 0.1 of the links in the entire network

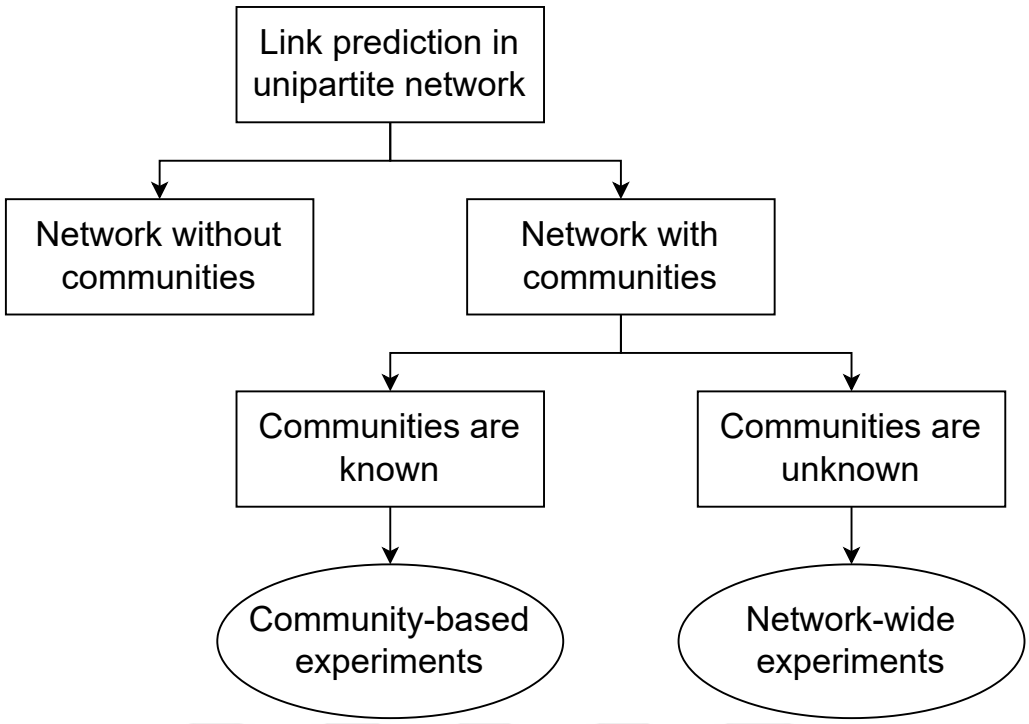


FIGURE 6.1 – Two types of experiments, community-based and network-wide, used in this work.

uniformly randomly and remove them to represent the positive samples of the test set. Removing large numbers of links can disrupt the remaining network structure. In the previous studies, this value was also set to 0.1 (Lü and Zhou, 2011; Zhou, 2021). The remaining pruned network is used as the learning set in which the indexes are calculated for every link in the test set. Lastly, the results are evaluated.

In community-based experiments, each community is treated as a standalone network during the link prediction process. This is to say, for each network, 0.1 of the links between the node pairs where both nodes belong to the same community are used as the test set. After the predictions for each community are evaluated, the results for communities in the same network are averaged as the final results.

6.1.2 Result for RQ1

Due to the large number of indexes used on the data, in order to preserve the readability of the plot, the indexes were divided into six groups according to their AUPR scores in network-wide experiments. Among the metrics, AUPR was chosen for visualization and

TABLE 6.1 – Properties of the datasets used in this study. $\langle k \rangle$ denotes the average degree.

Parameter name	**Net5**	**Net15A**	**Net15B**
Number of nodes (n)	5000	15000	15000
Mixing parameter (μ)	[0.1-0.9]	[0.1-0.9]	[0.1-0.9]
Degree exponent (γ)	3	3	3
Community size exponent (β)	2	2	2
Average degree ($\langle$k$\rangle$)	5.26	5.66	5.66
Maximum degree (k_{max})	35	75	75
Max. size of communities	100	200	5000
Min. size of communities	25	45	45

comparison of the results because it provides a more reliable measurement for the link prediction task (Yang et al., 2015). The representative plots shown in Fig. 6.2 are the outputs of the experiments performed on the networks generated with the parameters in set 2A. Our interpretation is valid not only for these plots but also for all experiments. In each plot, both network-wide (dashed line) and community-based (solid line) results of the same index are shown for comparison. Each data point represents the average score achieved by the index on a network generated using the indicated μ value.

The graphical analysis reveals that SPM and the path-based link prediction indexes Katz and LP are the most successful, while the neighbor-based link prediction indexes DICE, JC, LHNL, HDI along with the embedding methods GW, N2V, DW, GLEE score the lowest. RA, AA, CN, and HPI methods perform relatively better compared to other neighbor-based methods. However, they are unable to achieve the success of SPM and path-based methods, especially in network-wide experiments. This suggests that approaches relying solely on the immediate neighborhood of the nodes overlook important information about the network's structural characteristics, which seems to affect the link prediction task. However, as mentioned in the descriptions of the methods, some embedding methods try to capture high-order node relationships. Nevertheless, these methods also seem to perform poorly. One reason for this may be that cosine similarity is applied between embedding vectors to perform link prediction for these methods. Since cosine similarity is a neighbor-based link prediction index, it may have negatively affected the potential success of these embedding methods.

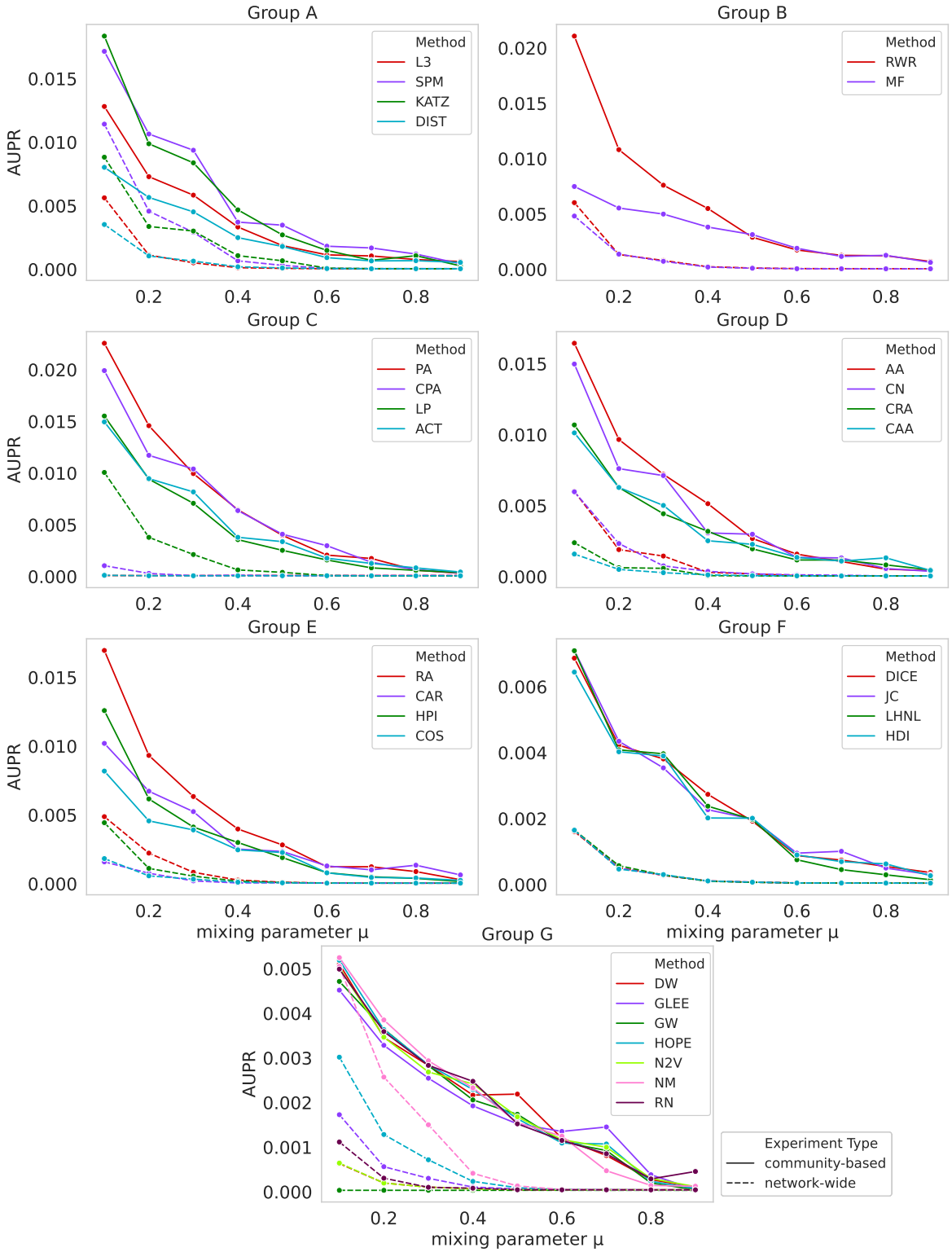


FIGURE 6.2 – AUPR scores for different μ values in set 2A. Solid and dashed lines represent corresponding scores on the community-based and network-wide experiments respectively.

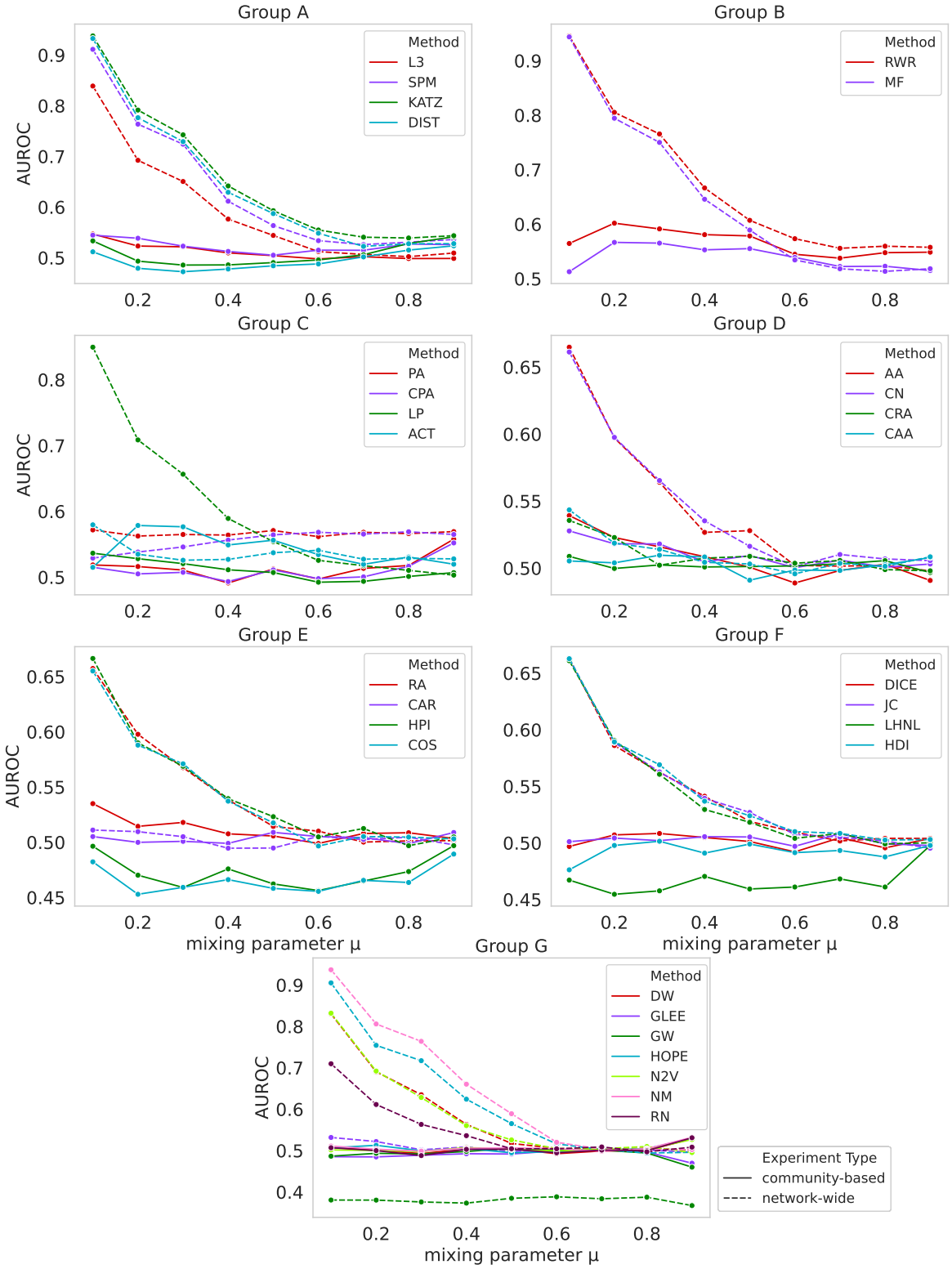


FIGURE 6.3 – AUROC scores for different μ values in set 2A. Solid and dashed lines represent corresponding scores on the community-based and network-wide experiments respectively.

TABLE 6.2 – Values used for experimental settings

Method name	**Parameter**	**Value**
Local Paths Index	damping parameter ϵ	0.01
Katz Index	damping parameter β	0.001
node2vec	dimensions	128
	number of walks	10
	walk length	80
DeepWalk	dimensions	128
	number of walks	10
	walk length	80
GraphWave	polynomial order	100
GLEE	dimensions	128
HOPE	dimensions	128
NetMF	dimensions	128
RandNE	dimensions	128
	smoothing weights	[0.5, 0.5]

Comparing network-wide and community-based results, the link prediction performances of all methods are better in community-based prediction. Since a community is a node subset with denser inner links, the likelihood of having a novel link is higher for the nodes belonging to the same community. Thus, a higher score for all indexes can be expected. Our experiments confirm this fact. Reminding that μ controls the rate of inner and outer links for the community members, an increase in μ leads to the dissolution of communities, disrupting the overall network structure and rendering connections more random. As a result, there are few links left in the community, and after the separation of links to learning and test sets, the number of links on which to perform link prediction is further reduced. This trend becomes particularly prominent when μ reaches 0.9, as the average number of links present in the test sets falls below 2. Consequently, many methods receive their lowest score from the evaluation metrics at this point.

In network-wide experiments, it is noted that an increase in the value of μ yields a decline in the performance of the indexes as seen in Fig. 6.2. For the majority of the indexes, there is a significant difference between the scores obtained from the network-wide experiments with low and high μ values.

TABLE 6.3 – Results of Welch's t-test and Mann-Whitney U test on AUPR values separated by two experiment types, network-wide and community-based, used in the study.

Welch's t-test	
p-value	5.3×10^{-41}
t-statistic value	14.2753
Degrees of freedom	738.39
95% confidence interval	[0.00306, 0.00403]
Estimate for network-wide group	0.00079
Estimate for community-based group	0.00434
Mann-Whitney U test	
p-value	5.7×10^{-109}
W	307544

The resulting AUPR values form a heavily skewed distribution as the AUPR values decrease non-linearly with increasing μ. While the t-test assumes a normal distribution, Welch's t-test has been shown to be valid for large sample sizes even with skewed distributions (Fagerland, 2012). As the sample size of the hypothesis tests is large ($n = 1188$), Welch's t-test is used along with the Mann-Whitney U test, which is a non-parametric alternative to the t-test. The p-value for the Mann-Whitney U test is an approximation and the correction for continuity is not applied. Table 6.4 shows the parameters used to call the R language functions of the tests. The results of both tests can be compared in Table 6.3. According to both tests, there is a significant difference between the AUPR values obtained from network-wide and community-based experiments. The very low p-value obtained from the Mann-Whitney U test may be due to the fact that non-parametric methods, such as the Mann-Whitney U test, produce artificially low p-values at large sample sizes (Fagerland, 2012).

A two-way ANOVA (analysis of variance) test is performed to investigate the connection, or the absence of it, among the variables. The result of the test is compared with the Kruskal-Wallis test, the non-parametric equivalent of a one-way ANOVA test that extends the Mann-Whitney U test. It should be noted that the Kruskal-Wallis test does not explicitly test for interactions between variables. Therefore, only the main

TABLE 6.4 – Parameters used for Welch's t-test and Mann-Whitney U test.

Parameters of Welch's t-test	
x	AUPR values of network-wide experiments.
y	AUPR values of community-based experiments.
alternative	"two.sided"
mu	0
paired	FALSE
var.equal	FALSE
conf.level	0.95
Parameters of Mann-Whitney U test	
x	AUPR values of network-wide experiments.
y	AUPR values of community-based experiments.
alternative	"two.sided"
mu	0
paired	FALSE
correct	FALSE
exact	NULL
conf.level	0.95

effect results are comparable between the two tests. Table 6.5 reveals that both tests confirm that each variable has a significant main effect. The interaction effect results of the two-way ANOVA reveal that there is no indication that the effect of a variable on AUPR is influenced by the other variables. It can be inferred that the μ value has an impact on the link prediction success, regardless of the parameters used when creating the network and whether the experiment is community-based or network-wide.

The results of the statistical tests support the conclusions we made by analyzing the figures. Briefly, we can say that the change in μ has a significant impact on the success of link prediction methods, whether we identify communities or not.

6.1.3 Result for RQ2

In the previous section, we saw that the indexes yield better results in community-wide experiments. In fact, the community-based approach is no different from applying the network-wide approach to each community separately. The mentioned link prediction

TABLE 6.5 – Interaction and main effects of certain variables on the AUPR values, revealed by the two-way ANOVA and Kruskal-Wallis tests. For the interaction effects, the variable names are represented by the symbols and separated by colons. The variable called "experiment type" refers to whether the experiment is done network-wide or community-based. The variable named "network set" is the set of networks used in the experiment, which can take the values Net5, Net15A, and Net15B.

Two way ANOVA		
Variable	p-value	F-value
μ value (M)	$< 2 \times 10^{-16}$	1307.62
Experiment type (E)	$< 2 \times 10^{-16}$	745.39
Network set (P)	$< 2 \times 10^{-16}$	264.18
M : E	$< 2 \times 10^{-16}$	475.95
M : P	$< 2 \times 10^{-16}$	233.00
E : P	$< 2 \times 10^{-16}$	117.53
M : P : E	$< 2 \times 10^{-16}$	74.53
Kruskal-Wallis test		
Variable	p-value	chi-squared
μ value	6.0×10^{-86}	421.00
Experiment type	5.7×10^{-109}	491.82
Network set	9.1×10^{-29}	129.13

indexes find a score for all possible links that are not visible in the network. When the network is taken as a whole, the score is calculated for all node pairs, whereas when the communities are taken separately, the calculation is made only for the pairs of nodes in the same community. The larger the community, the greater the number of possible node pairs. In addition, the smaller the community and the more inner links there are, the more it is possible to find a link between the node pairs in that community. In this case, it can be expected that the community size can also be effective in the link prediction. In order to see if there was a possible effect, we compared the results of the experiments conducted in different community sizes.

Fig. 6.4 shows a comparison of the experiments performed on all three sets of networks.

This includes the small networks (Net5), wide networks with large communities (Net15B) and wide networks with small communities (Net15A). For each evaluation metric, the performances of some path-based indexes and SPM (Group A) are compared across networks generated with different μ values. A similar comparison done using

neighbor-based indexes in Group D is seen in Fig. 6.5. Analyzing the evaluation metric scores, there is no difference between the trends observed in the link prediction results on small and large communities. That said, there is a visible difference between the scores. The results obtained from networks in Net15A are better than those of Net15B. We observe that simply changing the size of the community affects the success dramatically, even though other parameters remain the same. This change is also seen in the results of network-wide experiments. In other words, even when the communities are not identified, the size of the communities in the network can affect the link prediction.

A pairwise comparison between AUPR values achieved from the networks generated with different sets of parameters is seen in Table 6.6 and Table 6.7. The sample size for each group is $n = 396$. The comparison is made using Welch's t-test and Mann-Whitney U test. Two adjusted p-values are obtained for each test. The first one is adjusted with the Holm method to control the family-wise error rate. The second one is adjusted with the Benjamini–Yekutieli (BY) procedure to control the false discovery rate. It is observed that both tests produce low p-values for every pairwise comparison. The statistical tests confirm that there is a significant difference between the success results obtained from all three sets of networks.

TABLE 6.6 – Results of pairwise Welch's t-test on different sets of parameters used to generate the benchmark networks. For each group $n = 396$.

Welch's t-test			
Compared groups	p-value (Holm)	p-value (BY)	t statistic
Net5 - Net15A	9.2×10^{-11}	1.3×10^{-10}	5.4946
Net5 - Net15B	1.8×10^{-29}	3.3×10^{-29}	10.979
Net15A - Net15B	5.3×10^{-7}	9.8×10^{-7}	7.9327

6.1.4 Result for RQ3

Evaluation metrics used to score predictions should also be considered while interpreting our results. The link prediction task primarily focuses on predicting future links,

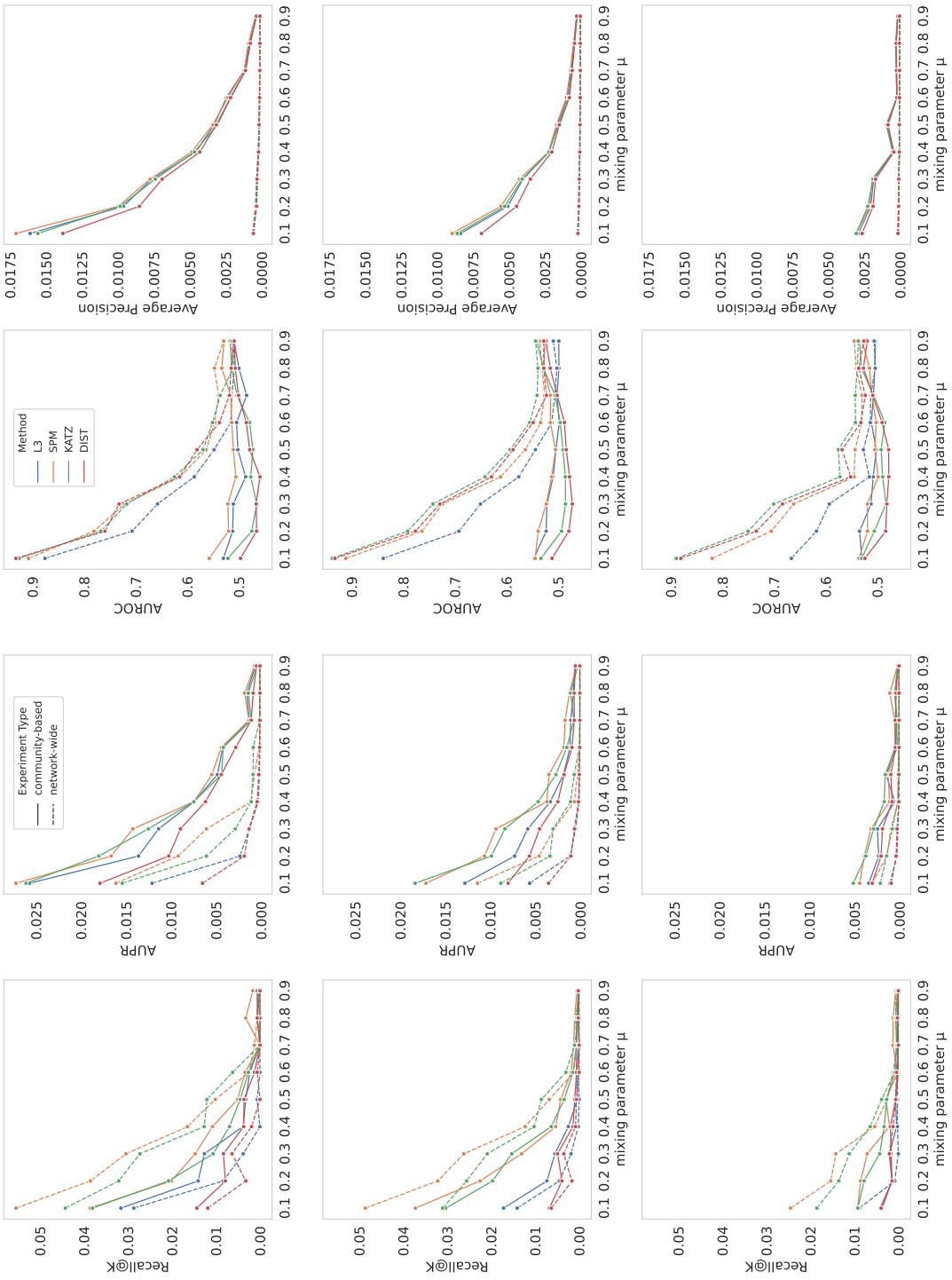


FIGURE 6.4 – The scores obtained by the L3, SPM, Katz, and Dist indexes on different evaluation metrics for link prediction on network sets Net5 (first row), Net15A (second row), and Net15B (third row).

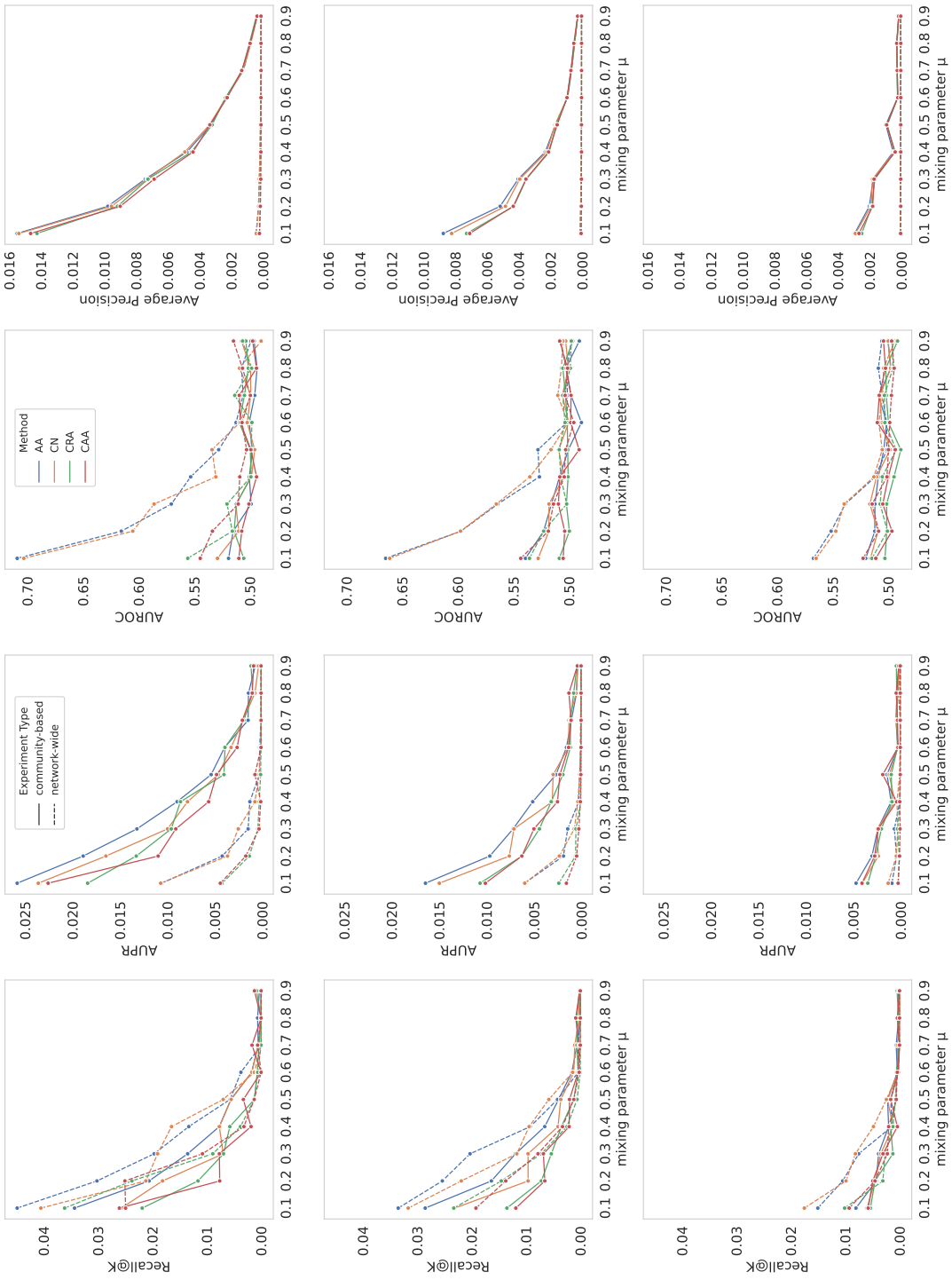


FIGURE 6.5 – The scores obtained by the AA, CN, CRA, and CAA indexes on different evaluation metrics for link prediction on network sets Net5 (first row), Net15A (second row), and Net15B (third row).

TABLE 6.7 – Results of pairwise Mann-Whitney U test on different sets of parameters used to generate the benchmark networks. For each group $n = 396$.

Mann-Whitney U test			
Compared groups	p-value (Holm)	p-value (BY)	W statistic
Net5 - Net15A	1.9×10^{-8}	1.8×10^{-8}	59951
Net5 - Net15B	3.6×10^{-29}	6.7×10^{-29}	114807
Net15A - Net15B	1.9×10^{-8}	1.8×10^{-8}	96873

in other words, positive instances. High AUPR scores indicate that the predictor can maintain a high ratio of true positives to positively predicted items, as the positive class threshold gets lower to include more positives. This is observed as high precision values while recall increases. In our case, the AUPR metric is sensitive to small changes in the number of true positives. In the link prediction task, where there is an abundance of negative instances, AUROC should be handled with care. Due to the high number of negatives, the false positive rate may not vary significantly even after substantial changes in the number of false positives.

To see how the different metrics react to a decrease in μ, we compare them in two tables. Table 6.8 shows, for each μ value, the evaluation metric results from the network-wide experiments. By comparing these values, we try to see which evaluation metrics react more strongly to changes in μ. If a metric does not react to large changes in μ, it means that the link prediction success is independent of μ or that the evaluation metric is not sensitive enough to changes in μ and hence, to changes in link prediction success.

On the table, we see that the difference in success between the cases where μ is 0.8 and 0.9 is quite small. When the results obtained from other μ values are analyzed, we see that the most sensitive evaluation metric to the change in μ is Recall@K with a 49.4% drop in success between levels. It is followed by AUPR with a decrease of 45.7%. The least sensitive metric is AUROC, with an average drop between levels of only 4.7%. To illustrate this in another way, the largest change from one μ level to another in the AUROC metric is the 10.5% drop from 0.1 to 0.2 μ. The smallest changes in the Recall@K and AUPR metrics are the decreases of 28.1% and 22.7% respectively.

TABLE 6.8 – The average results of the evaluation metrics used in network-wide experiments at different μ values.

μ value	Recall@K	Average Precision	AUROC	AUPR
0.1	166.79×10^{-4}	21.94×10^{-5}	0.6916	407.45×10^{-5}
0.2	95.85×10^{-4}	15.43×10^{-5}	0.6192	143.48×10^{-5}
0.3	68.94×10^{-4}	13.38×10^{-5}	0.5944	80.32×10^{-5}
0.4	33.28×10^{-4}	10.44×10^{-5}	0.5495	29.26×10^{-5}
0.5	19.43×10^{-4}	9.23×10^{-5}	0.5355	17.79×10^{-5}
0.6	6.76×10^{-4}	8.34×10^{-5}	0.5221	10.64×10^{-5}
0.7	2.23×10^{-4}	7.78×10^{-5}	0.5167	8.23×10^{-5}
0.8	1.71×10^{-4}	7.83×10^{-5}	0.5160	8.55×10^{-5}
0.9	1.39×10^{-4}	7.65×10^{-5}	0.5149	8.07×10^{-5}

To confirm these findings, we compare again using the same values from the community-based experiments in Table 6.9. The first striking fact is that the AUROC values are almost insensitive to the μ level. In the AUROC metric, a value of 0.5 is used as a baseline and represents a random classifier. Not only are these values very close to 0.5, but there is an increase in AUROC values after 0.6 μ. Without further analysis, we can assert that the AUROC metric was not affected at all by the changing μ values in these experiments. As for the most sensitive metrics, the other three metrics exhibited similar behavior. The Recall metric is again the most sensitive : The average drop between different μ values is 37.6%. The AUPR and Average Precision metrics show an average decrease of 35.1% and 34.6% respectively for the same value.

Considering the results produced by all metrics in response to varying μ values, we can interpret the reason for the insensitivity of the AUROC metric in community-based experiments as the inability of the AUROC metric to measure link prediction success. The most sensitive metric was Recall@K. However, when using Recall@K, it is important to remember that this metric is a threshold-based metric and the accuracy of the metric in measuring success is based on this threshold. The AUPR metric, on the other hand, was the most sensitive of the metrics that could work without depending on the threshold.

TABLE 6.9 – The average results of the evaluation metrics used in community-based experiments at different μ values.

μ value	Recall@K	Average Precision	AUROC	AUPR
0.1	177.05×10^{-4}	85.72×10^{-4}	0.5121	132.02×10^{-4}
0.2	100.01×10^{-4}	54.61×10^{-4}	0.5100	84.34×10^{-4}
0.3	70.85×10^{-4}	43.17×10^{-4}	0.5074	64.50×10^{-4}
0.4	39.34×10^{-4}	24.97×10^{-4}	0.5019	40.69×10^{-4}
0.5	27.74×10^{-4}	19.97×10^{-4}	0.5006	29.67×10^{-4}
0.6	10.54×10^{-4}	11.60×10^{-4}	0.4982	18.13×10^{-4}
0.7	5.76×10^{-4}	6.97×10^{-4}	0.4994	9.85×10^{-4}
0.8	4.78×10^{-4}	5.02×10^{-4}	0.5034	7.54×10^{-4}
0.9	3.37×10^{-4}	2.62×10^{-4}	0.5111	3.82×10^{-4}

6.2 Experimental Setup and Results for RQ4 and RQ5

In our experiments, we use three bipartite networks coming from different domains. *MovieLens* contains users' ratings of movies on a scale of 1 to 5. We do not consider the ratings and only work with the interaction network. *Setur*, supplied by the Turkish travel agency Setur, contains user-hotel interactions. *LastFM* dataset contains the interactions of users with songs. As weighted links are not supported by some of the methods, the repeating interactions between the same user and song are ignored. We ignore all node and link attributes, besides node types. We apply a minimum degree filter to *Setur* and *LastFM* that removes left nodes (user nodes) with fewer than 8 interactions, for the sake of supervised learning experiments and prediction tasks. This filter is not applied to *MovieLens*, as the lowest degree observed for the left nodes is already 20. The topological properties of the networks used in the study are given in Table 6.10.

6.2.1 Experimental Setup

We split the network into training and test networks with 90% and 10% of the existing links, respectively. Instead of uniformly randomly selecting 10% of all the links, we draw 10% of the links connected to each left node. Thus, in both the training set and

the test set, none of the left nodes are isolated. As a result, during prediction, we are able to rule out the well-defined cold-start problem in recommender systems, which could create additional difficulties for any of our methods.

We assess prediction quality using two key metrics : *Area Under the Precision-Recall Curve* (AUPR) and *Area Under the Receiver Operating Characteristic Curve* (AUROC). These metrics are previously explained in Chapter 5. It should be noted that, in practice, AUPR takes much lower values than AUROC when assessing link prediction (Yang et al., 2015; Kumar et al., 2020b).

TABLE 6.10 – Properties of the datasets used in this study. $\langle k \rangle$ denotes the average degree.

Dataset	#Left nodes	#Right nodes	#Links	$\langle k \rangle$	Density
Setur	324	1,305	3,798	4.18	0.00898
MovieLens	943	1,682	100,000	68.88	0.06305
LastFM	512	4,014	18,492	5.91	0.00900

For the implementation of personalized recommendation models, we use the RecBolelibrary, with the suggested configuration settings. They are tuned via random search with early stopping, using NDCG@10 as the validation metric and a training-to-validation data ratio of 8 :1. We have obtained the best performances of the learning with these parameters. The NetworkX Python package is used to calculate the network topological features described in Section 4.2. Link prediction indexes do not require hyper-parameter tuning. The same goes for PCA and LDA methods. However, a random search with 10-fold cross-validation is applied to the XGB method to obtain the best performance. During the validation step, the F1-score of the positive class is used to evaluate the success of the parameters, as the prediction of negative links did not matter. Embedding-based methods are executed with the hyper-parameters suggested in their respective papers to generate vector embeddings with a length of 128.

6.2.2 Result for RQ4

The AUPR and AUROC results of our experiments on three different datasets are presented in Table 6.11. It is split into four parts, according to the different categories of methods introduced in Chapter 4. In addition to predictive performance, the table shows the execution time of every method on all three datasets. These do not include the time spent tuning hyper-parameters.

For *MovieLens*, SPM (category *Traditional Link Scores*) gives the best results, while for *Setur* and *LastFM*, NGCF and DiffRec (*Personalized Recommendation Conversion*) are the best, respectively. We cannot generalize this success to their whole respective categories, though, because methods from the same category exhibit a high variability in terms of performance. Still, categories *Link Prediction via Topological Features* and *Embedding-based Link Prediction* seem to be one step behind the others, overall. Independently of the method category, SPM and DiffRec seem to be the best-performing methods. Especially, DiffRec obtains significantly higher AUPR scores for *LastFM*. It achieves an AUPR of 0.4061, while the highest score achieved by methods from other categories is 0.1315, representing a 208.8% increase. Dist, BiGI, and MP2V seem to get the lowest scores in general.

Evaluating each method individually, in the *Personalized Recommendation Conversion* category, LightGCN performs best for *Setur*, but receives relatively low AUPR scores for both other datasets. On top of obtaining the best AUROC scores for *LastFM* and *Setur*, NGCF also manages to consistently be in the top 3 when the methods are ranked according to AUPR scores. BPR, the only recommendation model that does not employ GCNs, is ranked the lowest for *MovieLens* and *Setur*, as expected. However, it manages to obtain the third-highest AUPR and second-highest AUROC scores for *LastFM*. In category *Traditional Link Scores*, SPM outperforms the others, most notably for *LastFM*, where SPM receives a 32.43% higher AUPR score compared to the second best-performing method. Following SPM, LP is consistently one of the best-performing methods. In all three datasets, there is a small difference in the AUPR scores of Katz and LP indexes (ΔAUPR= 0.2%). L5 and L7 score worse than L3 for every dataset on every metric except once, where L5 achieves a negligibly higher

TABLE 6.11 – Performances of the methods used on different datasets evaluated by AUPR and AUROC metrics.

	MovieLens		*Setur*		*LastFM*	
Method	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC
Katz	0.0560	0.8648	0.1394	0.6286	0.0992	0.7847
SPM	**0.0778**	0.8879	**0.1412**	0.6472	0.1315	0.8153
L3	0.0621	**0.8962**	0.0096	0.6950	0.0444	0.8154
L5	0.0532	0.8759	0.0077	0.6515	0.0447	0.7921
L7	0.0490	0.8648	0.0074	0.6379	0.0421	0.7921
PA	0.0464	0.8552	0.0867	0.6348	0.0480	0.7803
Dist	0.0130	0.7181	0.0026	0.6168	0.0073	0.7641
LP	0.0561	0.8652	0.1398	0.6976	0.0993	0.8132
XGB	0.0179	0.7788	0.0039	0.7207	0.0337	0.7983
PCA	0.0119	0.5560	0.0553	0.6191	0.0376	0.7576
LDA	0.0151	0.7417	0.0061	0.6453	0.0139	0.7416
DiffRec	0.0613	0.8462	0.0541	0.6368	**0.4061**	0.8309
LightGCN	0.0368	0.8266	0.1316	0.7126	0.0730	0.8342
BPR	0.0270	0.7944	0.0473	0.7009	0.2152	0.8678
SGL	0.0443	0.8143	0.0508	0.7066	0.0652	0.8674
NGCF	0.0324	0.8048	0.0724	**0.7250**	0.2174	**0.8732**
BiNE	0.0275	0.8091	0.0080	0.5334	0.0250	0.7632
BiGI	0.0368	0.3423	0.0065	0.4434	0.0082	0.5379
MP2V	0.0078	0.5469	0.0012	0.5692	0.0220	0.6298

AUPR score than L3 for *LastFM*. This shows us the ineffectiveness of using paths longer than three since we proposed paths longer than five and seven specifically for this work. Moreover, L3 performed better than LP for *MovieLens*, but worse for both other datasets. We attribute this to degree normalization in L3, as both methods are identical otherwise. Finally, Dist performed the worst.

In category *Link Prediction via Topological Features*, the highest AUPR scores are achieved by PCA. Only for *MovieLens* does XGB get better results. XGB also consistently ranks highest in terms of AUROC. Finally, in category *Embedding-based Link Prediction*, for *MovieLens*, BiGI has a better AUPR score than others, although it has the lowest AUROC score in the entire table. For the other datasets, it cannot replicate this success and is surpassed by BiNE.

Examining the experimental results, it is evident that there is a considerable variation in the scores obtained from the performance metrics depending on the dataset. To make an interpretation of the results, we take into account the properties of the datasets seen in Table 6.10. Indeed, as the density of the data declines, so does the accuracy of the predictions made by the L3 and Dist link prediction indexes. This decrease can be explained by the difference in L3 index scores between low-probability and high-probability links becoming less pronounced. Such a decrease, however, is not observed for the SPM, Katz, LP, and PA indexes. Nevertheless, they seem to be affected by the size of the network. *Setur* and *LastFM* have similar densities, but the former is smaller. This could be the reason for their success with *Setur*. Another group of methods that does not see a decline in evaluation metrics in sparse networks is *Personalized Recommendation Conversion*. On the contrary, their performance seems to increase as the data becomes sparser. This may be attributed to their unique architecture. The *Personalized Recommendation* models used in our study, except for BPR, employ GCNs to generate representations of the nodes. While metapath2vec and BiNE methods also generate node embeddings, they rely on random walks to do so. BiGI utilizes GNNs, whose effectiveness is apparent in the AUPR score achieved on the *MovieLens* dataset, matching LightGCN's AUPR score.

Moreover, here, *Personalized Recommendation Conversion* methods are optimized for scoring items to make accurate recommendations to users. In contrast, *Embedding-based Methods* treat the link prediction task as a binary classification problem by fitting a logistic regression model to make predictions. They are similar in this way to methods that use topological network features. It is also possible that the success of these methods is limited by the simple classifier used during the prediction step.

The methods that rely on topological features do not apply graph representational learning at all. It could be argued that this puts them at a disadvantage compared to methods that can exploit the network structure through path-based algorithms or graph representational learning. While topological features may accurately describe a node's role in the network, they do not offer any insight into how two specific nodes interact with each other. The same line of thought could be applied to *Embedding-based* methods. They indeed generate embedding vectors, but no matter how successful this process is, those embeddings are generalized representations of nodes. By contrast,

Personalized Recommendation Conversion models generate embeddings for the explicit purpose of assigning a score to a pair of nodes. As a result, although we do not use exact-ranked items but convert them to link prediction, it seems to be effective in terms of AUPR and AUROC accuracy.

6.2.3 Result for RQ5

The employed methods' execution times in seconds for the three datasets are shown in Table 6.12. Regarding these results, generally, it can be seen that the more successful methods, such as SPM and GCN-based, take longer to make predictions. Most methods with low execution times perform poorly on the link prediction task.

The methods in the *Traditional Link Scores* category operate on the adjacency matrix. As we recall, this matrix contains all nodes in the network in its rows and columns. The code used to execute these methods is not specific to bipartite graphs. In other words, these scores are calculated without taking into account that the graph is bipartite. We simply filter the scores to exclude links that cannot exist in a bipartite network during evaluation. Therefore, the structure of the adjacency matrix does not change. Hence, we can assert that the execution time of most of these methods depends on the number of nodes in the network, independent of the number of links present. As can be seen in the measurement results, for all methods in this group, the dataset with the longest running time is *LastFM* with the highest number of nodes, not *MovieLens* which has the highest number of links. The time complexity of the algorithms is also in line with the findings of the experiment. As can be seen in Table 6.13, which shows the time-complexity of the methods used, all of the methods except Dist scale independently of the number of links based on the number of nodes.

The quickest category is *Link Prediction via Topological Features* with PCA and LDA having much lower execution times compared to other methods. Since PCA is unsupervised, it does not go through any training phase, whereas for LDA, having a closed-form

TABLE 6.12 – The execution time of considered methods on three selected datasets in seconds.

Method	MovieLens	Setur	LastFM
Katz	16.51	4.00	25.27
SPM	1189.00	200.00	4024.00
L3	4.92	1.35	10.60
L5	6.10	1.40	13.92
L7	7.48	1.61	12.84
PA	2.20	1.24	6.24
Dist	5.00	2.09	15.55
LP	5.95	3.60	14.50
XGB	183.00	40.41	133.00
PCA	1.92	0.32	1.47
LDA	**1.43**	**0.25**	**1.35**
DiffRec	534.00	239.00	791.00
LightGCN	184.00	13.52	9.64
BPR	36.17	15.47	79.00
SGL	763.00	92.00	443.00
NGCF	477.00	36.31	413.00
BiNE	3966.00	650.00	2002.00
BiGI	4834.00	204.00	1066.00
MP2V	4011.00	643.00	1912.00

solution makes it easy to compute. For these reasons, LDA and PCA allow us to obtain results in a very short time. Compared to these two methods, XGB takes longer to run as it goes through a real training process and already includes PCA in this training process. When these three methods are considered as a whole, we see that the number of links in the network has a significant impact on the running speed of the methods. All three of these methods spent the most time on *MovieLens* data. This is because each link in the network corresponds to an instance in the tabular data. In other words, the length of the training data in these methods is twice the length of 90 percent of the links in the network, since negative sampling is done in a way that provides one negative instance for every positive one. This suggests that the training time is scaled by the number of links in the network. However, it should be noted that the number of nodes in the network also plays an important role. Namely, in the input

TABLE 6.13 – The time complexity of traditional link scores used in the study except SPM shown in big O notation, courtesy of Martínez et al. (2016). The letters n, e, and k represent the number of nodes, number of edges, and maximum degree of a node respectively. The table is simplified to exclude the methods that were not used in this study.

Method	Time complexity
Katz	$O(nk^3)$
L3	$O(n^3)$
L5	$O(n^3)$
L7	$O(n^3)$
PA	$O(nk^2)$
Dist	$O(ne \times log(n))$
LP	$O(nk^3)$

that these methods receive in the prediction phase, all possible links in the network are considered as instances to be predicted. Since all possible links in a bipartite network are calculated as $U \times V$, the prediction time is scaled by the number of user and item nodes. It is true that the actual number of links to be predicted is all potential links minus the links used as positive training instances. However, considering the density of the networks, the impact of this difference on the runtime is quite small. To sum up, for these algorithms, the runtime of the model fitting phase depends on the number of present links. The prediction time, which measures how long the transformation phase takes for LDA and PCA, depends on the number of all possible links which is dictated by the number of nodes.

Embedding-based Link Prediction methods take longer to work than other methods, despite their low prediction accuracy. This can be attributed to the fact that they use logistic regression to obtain prediction scores, which may not be able to handle the embedding dimensions. In the prediction phase of these methods, it is more appropriate to use models such as XGBoost or artificial neural networks to classify the embedding data. Especially neural network models, because they are more successful when trained with high-volume data.

To briefly summarize our findings on this research question, the *Link Prediction via Topological Properties* group seems to contain the fastest methods. Following this, the *Traditional Link Scores* group is the second fastest while providing accurate predictions.

Personalized Recommendation Conversion methods are in third place. *Embedding-based Link Prediction* methods can be considered the slowest. It should be noted that these results may change when the experiment is repeated on different datasets. *Traditional Link Scores* are not affected by the number of links already in the network and depend on the number of nodes. Other approaches are affected by both the number of links and the number of nodes.

6.3 Discussion

In this section, we discuss the outcomes of our experiments on a topic-by-topic basis.

We will discuss the results of the study to explore the relationship between link prediction and communities based on the research questions we have defined. First of all, the variation of the mixing parameter μ has a significant impact on the success of community-based link prediction. This was an expected result because increasing the mixing parameter decreases the link density in the communities. Since links are used to calculate indexes, this means that the amount of prior information available during link prediction is reduced. Methods that have to rely on less information naturally make less accurate link predictions. However, interestingly, the success of network-wide link prediction also decreases. This decline occurs even though there is no significant change in the number of links across the network. For two networks with the same number of nodes and links, the fact that link prediction success is higher in the network with more precisely defined communities shows that communities can affect link prediction success regardless of whether we have access to community information or not. The first conclusion we can draw from this is that, during the evaluation of link prediction success, how clearly the communities of the network are separated should be taken into account. A second conclusion we can draw from these experiments is that when predicting links in a network, treating the community in which a link is located as if it were a standalone network increases link prediction success. This conclusion also holds for experiments with low mixing parameter μ values. That is, even when communities are not very precisely defined, this approach can improve link prediction success. Experiments where the network characteristics are kept the same and the community size is changed show that network size does indeed affect success.

In the metrics used, unlike the other three metrics, we see that AUROC is not much affected by the μ change. Especially in community-based experiments, the deteriorating community structure is expected to decrease link prediction success. Since we do not observe this decrease, we conclude that it is not appropriate to use AUROC for link prediction success prediction. According to our experiments, Recall@K is the most sensitive metric to changing μ levels. However, studies in the literature do not recommend using fixed-threshold methods unless the threshold is set by including domain-specific information. Therefore, we propose the AUPR metric, which is both sufficiently sensitive and based on a threshold curve, as the most appropriate metric.

The findings of the experiments conducted on bipartite networks suggest that the most successful method categories are *Personalized Recommendation Conversion* and *Traditional Link Scores*. It can be seen that the SPM method achieves the highest link prediction accuracy compared to the other methods in its group. In contrast, for methods that rely on personalized recommendations, no single method that achieves the best results on every dataset can be named. This finding suggests that the success of *Personalized Recommendation Conversion* methods can be increased by combining them under an ensemble method.

Methods that rely on bipartite network embedding and network topological features for link prediction task produce the worst results. This may be attributed to the fact that they perform link prediction using more general approaches that are not specific enough to the problem at hand. Instead of representing how two different nodes might interact, these methods focus on the position and role of nodes on the network in a general way. This study could be further expanded by including more datasets with varying network sizes, link counts, and densities. An ensemble model could be used for personalized recommendation models, since our results imply that different recommendation models were able to better capture the structural properties of different networks.

Furthermore, the outcomes of this study indicate that personalized recommendation models may also be successfully applied as link predictors. This insight could be used to employ these models in reciprocal recommendations, where the goal is to make two-way recommendations that satisfy both parties. The task of reciprocal recommendation resembles both personalized recommendation, as each user is given a list of

recommendations, and link prediction because the problem definition implies a directionless relationship between two nodes.

Concerning the execution times of the methods, we can observe the following. The methods in the *Embedding-based Link Prediction* category run much slower than all other methods due to the logistic regression they use during prediction. Considering their low performance, we conclude that they are not suitable for link prediction in real-world networks. Following them, the methods in *Personalized Recommendation Conversion* category are the second slowest. Since these methods are not affected much by the change in the number of nodes, they are suitable for networks with a large number of users and items. Their prediction accuracy also makes them a particularly suitable choice. *Traditional Link Scores* can be calculated quickly since they do not undergo any training. They are convenient for providing instant results on small and medium-sized networks. However, it is evident that they are affected significantly by the increase in the number of nodes. In much larger networks that have not been used in the study, the calculation of these scores may become difficult. It is not recommended to use these methods in networks with a large number of nodes, for example, where user-item matching is performed. *Link Prediction via Topological Features* methods can be considered the quickest. However, when compared to the methods from *Traditional Link Scores*, the reduction they offer in execution time is not enough to compensate for the lost accuracy. Therefore, the use of these methods is also not recommended.

7 CONCLUSION

In this thesis, we present how the link prediction problem is affected by the network structure. We analyze two main factors that affect network structure : first, the characteristics of community structure in unipartite networks, and second, the unique connectivity patterns of bipartite networks. In accordance with our findings, we propose strategies to improve the success of link prediction in both unipartite and bipartite networks.

To explore how communities affect network structure and thus link prediction success, we conduct community-based and network-wide experiments on synthetic networks generated by the LFR model with different μ values. This parameter allows us to control the level of community cohesiveness. To conduct the experiments, we use various link prediction indexes that we categorize into four groups : (i) neighbor-based, (ii) path-based, (iii) embedding, and (iv) other methods. We measure their success with Recall@K, Average Precision, AUROC, and AUPR performance evaluation metrics. These experiments show that link prediction is more successful when communities in the network are distinctly separated from each other. This is also true when the communities are not identified and the experiment is conducted network-wide. If communities are known, however, prediction success can be improved by performing community-based link prediction. By comparing the results obtained from the networks we generated with different parameters with each other, we observe that the success of link prediction is affected by both the size and the cohesiveness level of the communities.

Comparing different categories, path-based and some of the other methods exhibit higher prediction accuracy compared to neighbor-based and embedding methods. We get the best results from the SPM, Katz, and LP indexes. Embedding methods resulted in extremely lower prediction accuracy in all experimental cases compared to others. We predicted links based on the cosine similarity of the resulting embedding vectors. However, using these embedding vectors, link prediction could be achieved through a sophisticated supervised learning process as well. Indeed, the purpose of the embedding process itself is to convert graph data into lower-dimension numerical data and to better

utilize supervised learning methods. Nevertheless, from the perspective of our study, we present a general overview of the relationship between link prediction and community structure by using dozens of synthetic networks. We did not focus on tuning each link prediction method and did not provide a contribution to the improvement of link prediction via embedding, which can be the subject of another study. At this point, we just want to emphasize that embedding methods have the potential for improvement.

Another important result of our experiments on the relationship between community characteristics and link prediction concerns the reliability of performance evaluation metrics. According to our experimental results, even when community cohesiveness changes, AUROC does not react to this change. In contrast, Recall@K, Average Precision, and AUPR reacted similarly to each other to the change in cohesiveness, even when their resulting scores were different. AUPR is the performance metric that gives us the most reliable results. It allows us to see the effects of community characteristics most clearly.

In addressing the second problem discussed in this thesis, we have adapted 19 different link prediction methods, some originally designed for unipartite networks and others being collaborative filtering recommendation systems, to bipartite networks. We divide them into four categories : (i) traditional link scores, (ii) link prediction via topological features, (iii) embedding-based, and (iv) personalized recommendation conversion. Most of the methods that we used were not previously applied to link prediction in bipartite networks. Hence, our work carries especially experimental novelties for link prediction methodologies in bipartite networks. As a major contribution, we suggest repurposing recommendation models that generate personalized ranked lists of items for link prediction tasks in bipartite networks. We present their comparative evaluation over the experiments on three different real-world datasets of varying sizes and densities. Our experiments show that the proposed conversion of personalized recommendations outperforms almost all the other methods. Specifically, the DiffRec model achieved an AUPR score of 0.4061 on the *LastFM* network, which has a low density (0.009) and a high number of nodes ($4,526$). Nevertheless, the SPM, which is a traditional link score, outperforms most other methods. Although this method was not developed for bipartite networks, our experiments show that it can be a very good link predictor for bipartite networks as well.

We have addressed two specific problems and evaluated the results obtained for them separately. These problems intersect on the subject of the network structure's effects on link prediction. From this perspective, the results surprisingly reveal that SPM is the least affected by structural changes in the network compared to all the other methods we have used. In fact, most of the 19 methods we used for bipartite networks and the 29 methods we used for unipartite networks have very different strategies. Yet, we were able to use SPM in experiments designed for both problems, which are the paradigms of the relation with community structure characteristics and accurate link prediction in bipartite networks, and it proved to be one of the most successful methods in both groups. Most link prediction methods make assumptions about the network structure. For example, neighbor-based indexes that we apply in unipartite networks assume the formation of triangles in the network. They cannot work in bipartite networks where these triangles do not exist. Since SPM does not make such assumptions, it can perform link prediction without being significantly affected by the conditions that cause changes in the network structure, such as the existence of communities or the network being bipartite. It should be noted, however, that in some cases, it is more appropriate to use specialized methods for the network structure : In our experiment on LastFM data, SPM ranks fourth in terms of success, behind recommendation models designed to work on bipartite networks.

Some future perspectives of this study can be listed as follows : First, the comparative work conducted on unipartite networks would benefit from experiments being repeated by using recent link prediction techniques such as graph neural networks. Second, the comparative performance evaluation that we realized here can be repeated on real-world networks, but community structure in such networks is usually unknown, and this will pose a challenge. To address this issue, different community detection algorithms can be used, such as the Louvain method. Third, here, the learning set includes 90% of the existing links. Changing this amount can be effective on the link prediction task because the network structure is directly related to the link amount. Thus, a complementary experiment on this topic can reveal interesting results. Especially on the subject of the evaluation metrics' reliability since it is safe to assume that removing a higher percentage of links from the network would result in a decrease in link prediction accuracy.

The experiments that involve bipartite networks could be further expanded by including more datasets with varying network sizes, link counts, and densities. The question of whether these methods are scalable on big data can be better answered by using much larger real-world networks, especially for the execution time measurement of the models. Diversification of the bipartite networks studied could be another improvement. In all the bipartite networks used, one node type is a person and the other is a product. Since there may be differences in the structure of networks sourced from different domains, increasing the variety of networks will make the results of the study more generalizable. An ensemble model could be used for personalized recommendation models since our results imply that different recommendation models were able to better capture the structural properties of different networks.

REFERENCES

Adamic, L. A. and Adar, E. (2003). Friends and neighbors on the web, *Social Networks* **25**(3) : 211–230.

Al Hasan, M., Chaoji, V., Salem, S. and Zaki, M. (2006). Link prediction using supervised learning, *SDM06 : workshop on link analysis, counter-terrorism and security*, Vol. 30, pp. 798–805.

Barabási, A., Jeong, H., Néda, Z., Ravasz, E., Schubert, A. and Vicsek, T. (2002). Evolution of the social network of scientific collaborations, *Physica A : Statistical Mechanics and its Applications* **311**(3) : 590–614.

Barberán, A., Bates, S. T., Casamayor, E. O. and Fierer, N. (2012). Using network analysis to explore co-occurrence patterns in soil microbial communities, *The ISME Journal* **6**(2) : 343–351.

Beauchamp, M. A. (1965). An improved index of centrality, *Behavioral Science* **10**(2) : 161–163.

Benchettara, N., Kanawati, R. and Rouveirol, C. (2010). Supervised machine learning applied to link prediction in bipartite social networks, *2010 International Conference on Advances in Social Networks Analysis and Mining*, pp. 326–330.

Biswas, A. and Biswas, B. (2017). Community-based link prediction, *Multimedia Tools and Applications* **76**(18) : 18619–18639.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R. and Lefebvre, E. (2008). Fast unfolding of communities in large networks, *Journal of Statistical Mechanics : Theory and Experiment* **2008**(10) : P10008.

Bonacich, P. and Lloyd, P. (2001). Eigenvector-like measures of centrality for asymmetric relations, *Social Networks* **23**(3) : 191–201.

Cannistraci, C. V., Alanis-Lobato, G. and Ravasi, T. (2013). From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks, *Scientific Reports* **3**(1) : 1613.

Cao, J., Lin, X., Guo, S., Liu, L., Liu, T. and Wang, B. (2021). Bipartite graph embedding via mutual information maximization, *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, New York, NY, USA, p. 635–643.

Chebotarev, P. and Shamis, E. (2006). The matrix-forest theorem and measuring relations in small social groups, *Automation and Remote Control* **58**.

Chen, T. and Guestrin, C. (2016). Xgboost : A scalable tree boosting system, *Proceedings of the 22nd ACM SIGKDD*, p. 785–794.

Costa, L. d. F., Oliveira Jr, O. N., Travieso, G., Rodrigues, F. A., Villas Boas, P. R., Antiqueira, L., Viana, M. P. and Correa Rocha, L. E. (2011). Analyzing and modeling real-world phenomena with complex networks : a survey of applications, *Advances in Physics* **60**(3) : 329–412.

Daminelli, S., Thomas, J. M., Durán, C. and Cannistraci, C. V. (2015). Common neighbours and the local-community-paradigm for topological link prediction in bipartite networks, *New Journal of Physics* **17**(11) : 113037.

Dong, Y., Chawla, N. V. and Swami, A. (2017). metapath2vec : Scalable representation learning for heterogeneous networks, *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, New York, NY, USA, p. 135–144.

Donnat, C., Zitnik, M., Hallac, D. and Leskovec, J. (2018). Learning structural node embeddings via diffusion wavelets, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, Association for Computing Machinery, New York, NY, USA, p. 1320–1329.

Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank, *Psychometrika* **1**(3) : 211–218.

Fagerland, M. W. (2012). t-tests, non-parametric tests, and large studies—a paradox of statistical practice ?, *BMC Medical Research Methodology* **12**(1) : 78.

Freeman, L. C. (1978). Centrality in social networks conceptual clarification, *Social Networks* **1**(3) : 215–239.

Gao, C., Zheng, Y., Li, N., Li, Y., Qin, Y., Piao, J., Quan, Y., Chang, J., Jin, D., He, X. and Li, Y. (2023). A survey of graph neural networks for recommender systems : Challenges, methods, and directions, *ACM Trans. Recomm. Syst.* **1**(1).

Gao, M., Chen, L., He, X. and Zhou, A. (2018). Bine : Bipartite network embedding, *The 41st International ACM SIGIR*, SIGIR '18, Association for Computing Machinery, New York, NY, USA, p. 715–724.

Giamphy, E., Guillaume, J.-L., Doucet, A. and Sanchis, K. (2023). A survey on bipartite graphs embedding, *Social Network Analysis and Mining* **13**(1) : 54.

Grover, A. and Leskovec, J. (2016). Node2vec : Scalable feature learning for networks, *Proceedings of the 22nd ACM SIGKDD*, p. 855–864.

Hamilton, W. L., Ying, R. and Leskovec, J. (2017). Inductive representation learning on large graphs, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, p. 1025–1035.

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y. and Wang, M. (2020). Lightgcn : Simplifying and powering graph convolution network for recommendation, *Proceedings of the 43rd International ACM SIGIR*, p. 639–648.

Hu, Y., Koren, Y. and Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets, *2008 Eighth IEEE ICDM*, pp. 263–272.

Iqbal, M. M. and Latha, K. (2022). An effective community-based link prediction model for improving accuracy in social networks, *Journal of Intelligent & Fuzzy Systems* **42** : 2695–2711. 3.

Jaccard, P. (1912). The distribution of the flora in the alpine zone, *New Phytologist* **11**(2) : 37–50.

Jolliffe, I. T. (2002). *Principal component analysis for special types of data*, Springer.

Katz, L. (1953). A new status index derived from sociometric analysis, *Psychometrika* **18**(1) : 39–43.

Kossinets, G. and Watts, D. J. (2009). Origins of homophily in an evolving social network, *American Journal of Sociology* **115**(2) : 405–450.

Kovács, I. A., Luck, K., Spirohn, K., Wang, Y., Pollis, C., Schlabach, S., Bian, W., Kim, D.-K., Kishore, N., Hao, T., Calderwood, M. A., Vidal, M. and Barabási, A.-L. (2019). Network-based prediction of protein interactions, *Nature Communications* **10**(1) : 1240.

Kumar, A., Singh, S. S., Singh, K. and Biswas, B. (2020a). Link prediction techniques, applications, and performance : A survey, *Physica A : Statistical Mechanics and its Applications* **553** : 124289.
URL: *https ://www.sciencedirect.com/science/article/pii/S0378437120300856*

Kumar, A., Singh, S. S., Singh, K. and Biswas, B. (2020b). Link prediction techniques, applications, and performance : A survey, *Physica A : Statistical Mechanics and its Applications* **553** : 124289.

Kumari, A., Behera, R., Sahoo, B. and Sahoo, S. (2022). Prediction of link evolution using community detection in social network, *Computing* **104**.

Kunegis, J., De Luca, E. W. and Albayrak, S. (2010). The link prediction problem in bipartite networks, *in* E. Hüllermeier, R. Kruse and F. Hoffmann (eds), *Computational Intelligence for Knowledge-Based Systems Design*, pp. 380–389.

Lancichinetti, A., Fortunato, S. and Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms, *Phys. Rev. E* **78** : 046110.

Landherr, A., Friedl, B. and Heidemann, J. (2010). A critical review of centrality measures in social networks, *Business & Information Systems Engineering* **2**(6) : 371–385.

Leicht, E. A., Holme, P. and Newman, M. E. J. (2006). Vertex similarity in networks, *Phys. Rev. E* **73** : 026120.

Liben-Nowell, D. and Kleinberg, J. (2003). The link prediction problem for social networks, *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, Association for Computing Machinery, New York, NY, USA, p. 556–559.

Liu, W. and Lü, L. (2010). Link prediction based on local random walk, *Europhysics Letters* **89**(5) : 58007.

Lü, L., Jin, C.-H. and Zhou, T. (2009). Similarity index based on local paths for link prediction of complex networks, *Phys. Rev. E* **80** : 046122.
URL: *https ://link.aps.org/doi/10.1103/PhysRevE.80.046122*

Lü, L., Pan, L., Zhou, T., Zhang, Y.-C. and Stanley, H. E. (2015). Toward link predictability of complex networks, *Proceedings of the National Academy of Sciences* **112**(8) : 2325–2330.

Lü, L. and Zhou, T. (2011). Link prediction in complex networks : A survey, *Physica A : Statistical Mechanics and its Applications* **390**(6) : 1150–1170.

Maas, A. L., Hannun, A. Y. and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models, *ICML*, Vol. 30, Atlanta, GA, p. 3.

Martínez, V., Berzal, F. and Cubero, J.-C. (2016). A survey of link prediction in complex networks, *ACM Comput. Surv.* **49**(4).

Mnih, A. and Hinton, G. E. (2008). A scalable hierarchical distributed language model, *in* D. Koller, D. Schuurmans, Y. Bengio and L. Bottou (eds), *Advances in Neural Information Processing Systems*, Vol. 21, Curran Associates, Inc.
URL: *https ://proceedings.neurips.cc/paper_files/paper/2008/file/1e056d2b0ebd5c878c550da6ac*
Paper.pdf

Morin, F. and Bengio, Y. (2005). Hierarchical probabilistic neural network language model, *in* R. G. Cowell and Z. Ghahramani (eds), *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, Vol. R5 of *Proceedings of Machine Learning Research*, PMLR, pp. 246–252. Reissued by PMLR on 30 March 2021.
URL: *https ://proceedings.mlr.press/r5/morin05a.html*

Nagaraja, S., Mittal, P., Hong, C., Caesar, M. and Borisov, N. (2010). Botgrep : Finding P2P bots with structured graph analysis, *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, USENIX Association, pp. 95–110.
URL: *http ://www.usenix.org/events/sec10/tech/full_papers/Nagaraja.pdf*

Newman, M. E. J. (2001). Clustering and preferential attachment in growing networks, *Phys. Rev. E* **64** : 025102.

Newman, M. E. J. (2004). Detecting community structure in networks, *The European Physical Journal B* **38**(2) : 321–330.

Nieminen, J. (1974). On the centrality in a graph, *Scandinavian Journal of Psychology* **15**(1) : 332–336.

Ochiai, A. (1957). Zoogeographical studies on the soleoid fishes found in japan and its neighhouring regions-ii, *NIPPON SUISAN GAKKAISHI* **22**(9) : 526–530.

Opsahl, T. (2013). Triadic closure in two-mode networks : Redefining the global and local clustering coefficients, *Social Networks* **35**(2) : 159–167. Special Issue on Advances in Two-mode Social Networks.

Otsuka, Y. (1936). The faunal character of the japanese pleistocene marine mollusca, as evidence of climate having become colder during the pleistocene in japan, *Biogeograph Soc Japan* **6**(16) : 165–170.

Ou, M., Cui, P., Pei, J., Zhang, Z. and Zhu, W. (2016). Asymmetric transitivity preserving graph embedding, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, Association for Computing Machinery, New York, NY, USA, p. 1105–1114.

Page, L., Brin, S., Motwani, R., Winograd, T. et al. (1999). The pagerank citation ranking : Bringing order to the web.

Perozzi, B., Al-Rfou, R. and Skiena, S. (2014). Deepwalk : Online learning of social representations, *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, Association for Computing Machinery, New York, NY, USA, p. 701–710.

Porta, S., Crucitti, P. and Latora, V. (2006). The network analysis of urban streets : A primal approach, *Environment and Planning B : Planning and Design* **33**(5) : 705–725.

Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K. and Tang, J. (2018). Network embedding as matrix factorization : Unifying deepwalk, line, pte, and node2vec, *Proceedings of the*

Eleventh ACM International Conference on Web Search and Data Mining, WSDM '18, Association for Computing Machinery, New York, NY, USA, p. 459–467.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. and Parisi, D. (2004). Defining and identifying communities in networks, *Proceedings of the National Academy of Sciences* **101**(9) : 2658–2663.
URL: *https ://www.pnas.org/doi/abs/10.1073/pnas.0400054101*

Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N. and Barabási, A.-L. (2002). Hierarchical organization of modularity in metabolic networks, *Science* **297**(5586) : 1551–1555.

Rendle, S., Freudenthaler, C., Gantner, Z. and Schmidt-Thieme, L. (2009). Bpr : Bayesian personalized ranking from implicit feedback, *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, AUAI Press, Arlington, Virginia, USA, p. 452–461.

Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval*, International student edition, McGraw-Hill.

Saoub, K. R. (2021). *Graph Theory : an introduction to proofs, algorithms, and applications*, Chapman and Hall/CRC.

Sørenson, T. (1948). *A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and Its Application to Analyses of the Vegetation on Danish Commons*, Biologiske skrifter, I kommission hos E. Munksgaard.
URL: *https ://books.google.com.tr/books ?id=rpS8GAAACAAJ*

Soundarajan, S. and Hopcroft, J. (2012). Using community information to improve the precision of link prediction methods, *Proceedings of the 21st International Conference on World Wide Web*, WWW '12 Companion, Association for Computing Machinery, New York, NY, USA, p. 607–608.

Sun, Y., Norick, B., Han, J., Yan, X., Yu, P. S. and Yu, X. (2012). Integrating meta-path selection with user-guided object clustering in heterogeneous information networks, *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Dis-*

covery and Data Mining, KDD '12, Association for Computing Machinery, New York, NY, USA, p. 1348–1356.

Tang, J., Qu, M. and Mei, Q. (2015). Pte : Predictive text embedding through large-scale heterogeneous text networks, *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, Association for Computing Machinery, New York, NY, USA, p. 1165–1174.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J. and Mei, Q. (2015). Line : Large-scale information network embedding, *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, p. 1067–1077.

Tong, H., Faloutsos, C. and Pan, J.-y. (2006). Fast random walk with restart and its applications, *Sixth International Conference on Data Mining (ICDM'06)*, pp. 613–622.

Torres, L., Chan, K. S. and Eliassi-Rad, T. (2020). GLEE : Geometric Laplacian Eigenmap Embedding, *Journal of Complex Networks* **8**(2) : cnaa007.

Wang, W., Xu, Y., Feng, F., Lin, X., He, X. and Chua, T.-S. (2023). Diffusion recommender model, *Proceedings of the 46th International ACM SIGIR*, SIGIR '23, Association for Computing Machinery, New York, NY, USA, p. 832–841.

Wang, X., He, X., Wang, M., Feng, F. and Chua, T.-S. (2019). Neural graph collaborative filtering, *Proceedings of the 42nd International ACM SIGIR*, p. 165–174.

Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J. and Xie, X. (2021). Self-supervised graph learning for recommendation, *Proceedings of the 44th International ACM SIGIR*, SIGIR '21, Association for Computing Machinery, New York, NY, USA, p. 726–735.

Wu, Ziteng, Song, Chengyun, Chen, Yunqing and Li, Lingxuan (2021). A review of recommendation system research based on bipartite graph, *MATEC Web Conf.* **336** : 05010.

Xu, X.-K., Shang, K.-K. and Xiao, J. (2020). Quantifying the effect of community structures for link prediction by constructing null models, *IEEE Access* **8** : 89269–89280.

Yang, C., Xiao, Y., Zhang, Y., Sun, Y. and Han, J. (2022). Heterogeneous network representation learning : A unified framework with survey and benchmark, *IEEE Transactions on Knowledge and Data Engineering* **34**(10) : 4854–4873.

Yang, Y., Lichtenwalter, R. and Chawla, N. (2015). Evaluating link prediction methods, *Knowledge and Information Systems* **45**.

Zhang, W., Li, B., Zhang, H. and Zhang, L. (2022). Community and local information preserved link prediction in complex networks, *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7.

Zhang, Z., Cui, P., Li, H., Wang, X. and Zhu, W. (2018). Billion-scale network embedding with iterative random projection, *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 787–796.

Zhou, T. (2021). Progresses and challenges in link prediction, *iScience* **24**(11) : 103217.

Zhou, T., Lü, L. and Zhang, Y.-C. (2009). Predicting missing links via local information, *The European Physical Journal B* **71**(4) : 623–630.

BIOGRAPHICAL SKETCH

Ş. D. İnan Özer received the Bachelor's degree in Computer Engineering from the Faculty of Engineering and Technology, Galatasaray University, İstanbul, Turkey in 2022. He is a Master's student in Computer Engineering at Graduate School of Science and Engineering of Galatasaray University. He has been working as a research assistant since 2022 in Galatasaray University. His areas of research include network science, artificial intelligence, and deep learning.

PUBLICATIONS

— Ş. Demir İnan Özer and G. Keziban Orman, "Enhanced Item Recommendation via Graph Properties in Sparse Data," *IFIP advances in information and communication technology*, vol 714 pp. 111–124, June 2024. Springer, Cham. doi : 10.1007/978-3-031-63223-5_9

— (Accepted) Ş. Demir İnan Özer, G. Keziban Orman, and Vincent Labatut, "Link Prediction in Bipartite Networks" 2024 International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES), Seville, Spain, 2024

— (Submitted) Ş. Demir İnan Özer and G. Keziban Orman, "Experimental Evaluation of the Effect of Community Structures on Link Prediction", Information Sciences, 29 May 2024