University of Windsor

Scholarship at UWindsor

Electronic Theses and Dissertations Theses, Dissertations, and Major Papers

2023

Learning Networks with Attention Layers for Team Recommendation

Sagar Kaw
University of Windsor

Recommended Citation

Kaw, Sagar, "Learning Networks with Attention Layers for Team Recommendation" (2023). *Electronic Theses and Dissertations*. 9075.
https://scholar.uwindsor.ca/etd/9075

Learning Networks with Attention Layers for Team Recommendation

By

Sagar Kaw

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2023

Learning Networks with Attention Layers for Team Recommendation

by

Sagar Kaw

APPROVED BY:

B. Maheshwari
Odette School of Business

P.M. Zadeh
School of Computer Science

Z. Kobti, Co-Advisor
School of Computer Science

K. Selvarajah, Co-Advisor
School of Computer Science

April 20, 2023

DECLARATION OF CO-AUTHORSHIP/PREVIOUS PUBLICATION

1. Co-Authorship

   I hereby declare that this thesis incorporates material that is the result of research conducted under the supervision of Dr. Ziad Kobti and Dr. Kalyani Selvarajah. In all cases, the key ideas, primary contributions, experimental designs, data analysis, and interpretation were performed by the author, and the contribution of co-authors was primarily through the proofreading of the published manuscripts.

   I am aware of the University of Windsor Senate Policy on Authorship, and I certify that I have properly acknowledged the contribution of other researchers to my thesis and have obtained written permission from each of the co-author(s) to include the above material(s) in my thesis.

   I certify that, with the above qualification, this thesis, and the research to which it refers, is a product of my own work.

2. Previous Publication

   This thesis includes the paper that has been accepted in a peer-reviewed conference, as follows:

Publication Title/Full Citation	Publication Status
S. Kaw, K. Selvarajah and Z. Kobti. Transfer Learning with Graph Attention Networks for Team Recommendation. In 2023 International Joint Conference on Neural Networks, IJCNN 2023. Institute of Electrical and Electronics Engineers (IEEE).	Accepted (To appear in the proceedings of the IJCNN conference, 18-23 June 2023)

   I certify that I have obtained written permission from the copyright owner(s) to include the above-published material(s) in my thesis. I certify that the above material describes work completed during my registration as a graduate student at the University of Windsor.

3. General

   I declare that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained written permission from the copyright owner(s) to include such material(s) in my thesis.

   I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

The Team Formation Problem aims to identify a group of experts who possess the required skills to complete a common goal. Graph-based approaches have been commonly used to solve this problem, but recently, researchers have started exploring this problem from the perspective of social information retrieval and applying neural architectures to recommend teams of experts. However, the learning process of these architectures is faced with several challenges. This includes the inability to handle network modifications after the training process is over as well as the time complexity of the learning process is high, which is proportional to the size of the network.

In this study, we propose a new framework called “*LANT* - *L*everaging Graph *A*ttention *N*etwork for *T*eam formation” which leverages graph neural networks and variational inference to address the challenges faced by existing approaches. The proposed framework utilizes transfer learning and neural team recommendation, with self-supervised learning of node embeddings achieved using Deep Graph Infomax with Graph Attention Networks as an encoder.

We demonstrate empirically how LANT effectively addresses the challenges faced by existing approaches and outperforms state-of-the-art methods on large scale real world datasets. The proposed framework provides an efficient and scalable solution to team formation problems and can be applied in various fields where expert teams are required to achieve a common goal.

DEDICATION

I would like to dedicate this thesis to my family. This achievement would not have been possible without their constant love and support. They have always believed in me and encouraged me to follow my dreams, no matter how impossible they may have seemed. Their unwavering faith and encouragement sustained me through the countless late nights and setbacks, and their sacrifices allowed me to pursue my education. I am forever grateful for all they have done for me and could not have reached this moment without them.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to the following individuals who have provided me with unwavering support throughout my Master's thesis journey.

First and foremost, I would like to thank my supervisor Dr. Ziad Kobti and my co-supervisor Dr. Kalyani Selvarajah, for their guidance and encouragement throughout my research work. They gave me the freedom to choose the subject of my research, and their ongoing support and direction encouraged me to refine my work. They had a deep grasp of research challenges, which taught me a lot, and I appreciate their help and support throughout. I have thoroughly enjoyed discussing and working on my research topic with them. I am grateful for the knowledge they shared with me, the time they invested in me, and the funding they provided.

I would also like to extend my heartfelt thanks to my committee members, Dr. Bharat Maheshwari and Dr. Pooya Moradian Zadeh, for their valuable time and support. Special thanks to Mrs. Christine Weisener and Mrs. Melissa Robinet, who were always there to assist me with various academic issues.

I will always be grateful to my parents for their support, guidance, and understanding as well as for always being there for me. I also like to thank my brother and sister-in-law for encouraging a passion for knowledge and exploration. They constantly motivate me to reach my objectives.

Finally, I would like to thank my friends for their continuous support and encouragement as well as for bearing with me during my moments of frustration. It can be extremely challenging to live away from home, but with the support of my friends, it became a lot easier for me. Words cannot express how grateful I am to my parents, siblings, and friends for always supporting me in every situation. This thesis would not have been possible without their support and encouragement.

Sagar Kaw

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

TFP	Team Formation Problem
SNA	Social Network Analysis
GNN	Graph Neural Networks
GCN	Graph Convolutional Networks
GAT	Graph Attention Networks
DGI	Deep Graph Infomax
VBNN	Variational Bayesian Neural Networks
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
MRR	Mean Reciprocal Rank
MAP	Mean Average Precision
NDCG	Normalized Discounted Cumulative Gain
AUC	Area Under the Curve

Chapter 1

Introduction

The Team Formation Problem (TFP) refers to the process of grouping individuals or agents together to form a team for a common goal. This process involves identifying a subset of agents who have the necessary skills, abilities, and preferences to work effectively together.

There are various approaches and algorithms that have been proposed to solve the TFP, such as graph-based methods, mathematical optimization, and machine learning techniques. These approaches aim to balance the needs of the individual agents with the requirements of the team as a whole and to find a solution that is both effective and efficient.

The TFP is relevant to a range of fields and applications, including sports, business, and education. In these fields, the ability to form effective teams can have a significant impact on success, and the team formation process is therefore of great importance.

1.1 Background

Social network refers to the structure of relationships, connections, and interactions between individuals, groups, organizations, and communities. For example, social networking websites such as Facebook, Twitter, LinkedIn etc., have greatly facilitated interactions between web users worldwide. These interactions can be based on a wide range of factors, including shared interests, values, beliefs, and activities, as well as more formal interactions, such as those between co-workers, classmates, or business

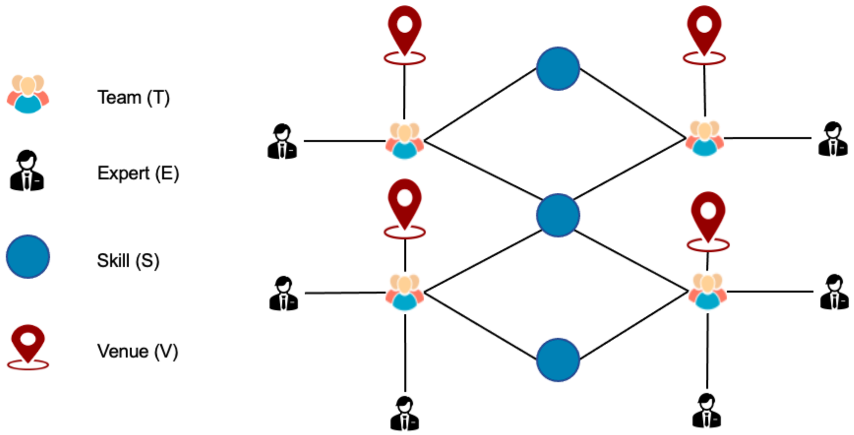


Figure 1.0.1: DBLP Heterogeneous Social Network

partners.

Social networks can be analyzed to better understand the patterns of relationships and interactions within a community and the ways in which these relationships can influence behavior, cognition, and communication. The information gleaned from analyzing social networks is particularly useful in various applications such as team formation, recommendation systems, healthcare, etc.

1.1.1 Social Network Analysis (SNA)

Social network analysis (SNA)[3] is a branch of the social sciences that deals with the study of social relationships and connections between individuals, groups, organizations, and communities. SNA is concerned with the structure and properties of social networks, including their size, shape, and connectivity, as well as the ways in which these networks influence behavior, cognition, and communication.

SNA has roots in sociology and anthropology but has since expanded to encompass a range of disciplines, including computer science, mathematics, and psychology. The field of SNA has grown significantly in recent years, due in part to the increasing availability of large-scale social network data, as well as advances in computational and data analysis methods.

SNA is used to study a variety of social phenomena, including the spread of information, the formation of groups and communities, the influence of social networks on behavior and attitudes, and the formation of opinion and attitudes. SNA is also used to study the structure of complex systems, such as organizations, cities, and economies, and to understand the relationships between individuals and organizations within these systems.

SNA relies on mathematical and computational techniques to analyze and model social networks. This includes the use of graph theory, network algorithms, and machine learning techniques, as well as the use of statistical methods to analyze and interpret the data. SNA also involves the collection and analysis of large-scale social network data, which can be obtained through various sources, including surveys, online platforms, and digital traces.

1.1.2 SNA and Team Formation Problem (TFP)

Social network analysis (SNA) and the Team Formation Problem (TFP) are two related fields that deal with the study of social relationships and connections between individuals, groups, organizations, and communities. SNA focuses on the structure and properties of social networks, while the TFP is concerned with finding the optimal combination of agents to form a team.

Both SNA and the TFP are important in a range of fields, including sports, business, and education, where the ability to form effective teams can have a significant impact on success. In these fields, SNA can provide valuable insights into the social relationships and connections between individuals and can inform the team formation process by identifying potential team members who have strong connections within the network and the necessary skills, abilities, and preferences to work together effectively.

1.1.3 Affinity Seeking and Member Compatibility

Affinity seeking[4] is a concept in team formation that refers to the tendency of individuals to form relationships with others who share similar interests, attitudes, and values. The idea is that individuals with strong connections are more likely to work effectively and efficiently together. This can positively impact team performance and satisfaction.

Affinity seeking can occur naturally through social interactions and relationships that develop outside of work, or it can be facilitated through intentional efforts to match individuals with compatible team members. The concept of affinity seeking is important in team formation because it can help to ensure that individuals are placed in teams where they are likely to work well with others, and this can contribute to the success of the team and the organization as a whole.

Member compatibility refers to the degree to which individuals in a team are similar in terms of their personality traits, skills, and abilities.

1.2 Problem Definition

Neural Team Formation Problem [2]: Given a heterogeneous collaboration graph G, let $\{s_1, s_2, \ldots, s_m\} \subseteq \mathcal{S}$ be a set of m skills and $\{e_1, e_2, \ldots, e_n\} \subseteq \mathcal{E}$ be a set of n experts; $(\langle \mathcal{S}_r, \mathcal{E}_r \rangle)$ is a *Team of Experts* such that $\mathcal{E}_r \subseteq \mathcal{E}$; $\mathcal{E}_r \neq \emptyset$, who collectively hold a subset of skills $\mathcal{S}_r \subseteq \mathcal{S}$; $\mathcal{S}_r \neq \emptyset$. Our aim is to estimate a mapping function f of parameters θ, where f represents a neural network and θ represents the neural network's weights or parameters, from a subset of skills to a set of experts $f : \{\mathcal{S} \rightarrow \mathcal{E}\}$, such that $f(\mathcal{S}_r; \theta) = \mathcal{E}_r$, so that potential individuals can form a team to work on the given set of skills.

Team of Experts: For a given set of n experts $\{e_1, e_2, \ldots, e_n\} \subseteq \mathcal{E}$ and a task that requires a set of m skills $\{s_1, s_2, \ldots, s_m\} \subseteq \mathcal{S}$, a *team of experts* for a task is a set of r skill-expert pairs: $\mathcal{T} = \{\langle \mathcal{S}_1, \mathcal{E}_1 \rangle, \langle \mathcal{S}_2, \mathcal{E}_2 \rangle, \ldots, \langle \mathcal{S}_r, \mathcal{E}_r \rangle\}$, where $\mathcal{E}_r$ represents a set of experts e_i in $\mathcal{E}_r$, $i \in \{1, 2 \ldots, n\}$ that possesses a set of skills s_j in $\mathcal{S}_r$, $j \in \{1, 2 \ldots, m\}$.

1.3 Thesis Motivation

Traditionally, Heuristic [5][6][7], Metaheuristic [8][9][10][11][12] or Rule-based methods have been used to solve TFP. However, these methods are often limited in their ability to handle the complexity and variability of the TFP. For instance, TFP uses large and complex social networks to identify experts for a given task. These networks are often dynamic in nature, meaning that they change over time. Individuals may join or leave a team, the team's objectives and requirements may change, and the skills and preferences of team members may evolve. This can make it challenging for Heuristic and Metaheuristic methods to identify the optimal team composition at a given instance. Some of the known shortcomings of these methods in TFP are:

- Limited information about the network [2], [13]: Heuristic and Metaheuristic methods may not have access to complete or accurate information about the network as it is often constantly evolving in the real world. As these methods are applicable to a snapshot of the network at a given instance of time, therefore they are limited in terms of their ability to accurately identify the optimal team compositions in the real world.

- Computationally expensive [2], [13]: Heuristic and Metaheuristic methods can be computationally expensive, especially when dealing with large and complex networks. This is because they identify subgraphs as teams from the network and subgraph optimization techniques have been shown to be a reduced version of the Steiner Tree problem, which is NP-hard in nature [14]. This can limit the scalability and applicability of these methods in real-world settings.

- Assumptions made may not hold true: Heuristic and Metaheuristic methods often rely on certain assumptions about the relationship between individual characteristics, team dynamics, and team performance. For example, some research works [15], [16], [17] assume that team performance is solely determined by the communication cost between team members, some other research works [18], [19] assume that team members with similar personalities or backgrounds work better together. Although these assumptions are an attempt to simulate favorable scenarios in the real world for TFP they may not always hold true in practice due to the complex and dynamic nature of real-world networks.

- Lack of validation: Heuristic and Metaheuristic methods may not provide a way to validate the quality of the resulting team formation. Here, the quality of the team composition may depend on the optimized value of the objective function (individual factors such as communication cost, geo-proximity, etc., or their combination) as in [15], [16], [17], [18], [19] (and more research works included in table 2.1.1 etc.) involved. But as discussed above, the optimized value might reflect satisfying a favorable scenario but it is not always clear whether the resulting team will be successful or not in the real world.

Recently, researchers have started to see this problem as a social information retrieval and examine it through neural network architectures that recommend the team of experts by learning a relationship between the skills and experts' space [13][2][20][21]. These methods overcome the challenges posed by Heuristics and Metaheuristics such that they are capable of automatically learning and extracting relevant features from large and complex social networks. They also provide a way to validate the resultant teams. However, these methods face the following shortcomings:

- They are unable to handle the modification of a network (i.e., the dynamic nature of a network) once the training process is over.
- The time complexity of the learning process is high and proportional to the size of the network.

Therefore, the motivation of this thesis is to address the challenges faced by current neural network architectures by leveraging state-of-the-art deep learning methods for TFP in any social network.

1.4 Thesis Statement

The objective of this research is to design a novel expert recommendation framework that can efficiently learn the implicit and valuable characteristics underlying the social network. In literature, Graph Neural Networks (GNNs) [22] are designed to perform inference on data described by graphs. This research utilizes Graph Attention Networks (GAT)[23], one of the most popular types of GNN, to recommend suitable experts for TFPs in social networks.

The state-of-the-art in neural TFP [13][2][20] uses random walks based Metapath2Vec [24] for graph representation learning. However, random walk methods suffer from known limitations. To overcome these limitations, our proposed framework, LANT, replaces the random walk-based method Metapath2Vec [24] with Graph Attention Networks(GATs) [23] in the existing state-of-the-art solutions for TFP. The main focus of this study is to demonstrate the effectiveness of GATs in capturing the complex structural information of social networks and to show that they can outperform random walk-based methods in handling dynamic graphs, learning node embeddings, and scalability.

1.5 Thesis Contribution

This thesis addresses the TFP and proposes a novel deep learning framework "LANT - Leveraging Graph Attention Networks for Team Recommendation". The proposed approach optimizes the graph representation learning process in the existing state-of-the-art deep learning frameworks for TFP. GATs take the social network as input and generate low-dimensional vector representations for each node in the latent dimensional space. These vector representations aid in the downstream task of recommending experts in the social network. Our key contributions can be found in the following list:

- We have proposed a new deep-learning framework, LANT, for the team formation problem.
- We have shown that our proposed framework, LANT, outperforms the existing state-of-the-art deep-learning frameworks for TFP in terms of computational efficiency for graph representation learning both in static as well as dynamic social networks.

- We have compared our proposed framework, LANT, against two real-world datasets - DBLP and IMDB.

1.6 Thesis Organization

The rest of the thesis/research work is organized in the following manner.

In chapter 2, We discuss the related works in the field of Team Formation Problem. The literature review comprises classical rule-based approaches including heuristics, meta-heuristics, etc., and the current state-of-the-art deep learning frameworks on TFP.

In chapter 3, We introduce our proposed approach, LANT, to solve the TFP. Basically, LANT leverages Graph Attention Networks (GATs) for the graph representation learning step in the deep learning framework for TFP. This chapter discusses step by step process of our approach and how it optimizes the state-of-art deep learning framework for TFP.

In chapter 4, We provide our experimental setup and environment, which includes the tools and libraries we used to implement our suggested framework (LANT), the system configuration, the dataset information, the hyper-parameters for training, the specifics of the evaluation metrics and the baselines we utilized to assess our model.

In chapter 5, We conducted experiments on two different benchmark datasets DBLP and IMDB. We compared our framework to the existing state-of-the-art methods for graph representation learning in TFP in which our framework outperformed existing architecture in terms of computational efficiency.

In Chapter 6, We conclude our research, explain the insights we gained during our research work, and describe some of the opportunities for future work.

Chapter 2

Related Works

Team recommendation or Team Formation Problem (TFP) is the process of efficiently selecting individuals from a network to form a collaborative group that can achieve shared objectives [15]. Team-based opportunities have become more common in a variety of situations such as manufacturing, games, law, academia, healthcare sectors, in freelancer jobs - Upwork[1] and Guru[2], and other professional organizations [1].

2.1 Classical Approaches

The TFP in social networks has been an area of intense research, with many seminal contributions to the field. Among these, Lappas et al.[15] made the first significant strides in addressing the TFP. In their research, they formulated the TFP using communication costs, which represent the amount of effort required for individuals to interact and collaborate within a team. They identified collaboration and communication between team members as critical factors that impact the success of a project. Communication cost measures the closeness of individuals within a social network, with the aim of identifying the most efficient ways to form a team. By minimizing the communication cost between team members, the authors believed that the team formation algorithms will identify the most efficient team configurations for a given task, which can improve the overall effectiveness and efficiency of a project. Numerous studies in the field of team formation have explored various functions aimed at optimizing communication costs. These functions include but are not limited to the

[1] *https://www.upwork.com/*

[2] *https://www.guru.com/*

minimum spanning tree, the diameter distance, and the sum-of-distance techniques.

- The minimum spanning tree approach involves constructing a shortest-distance path that connects all the members of the team. The idea is to identify the shortest paths between all pairs of team members, thereby minimizing the overall communication cost. This technique is particularly useful when the team members are spread out across the network and need to communicate with each other frequently. Li et al.[25][26], Anagnostopoulos et al.[27], Majumder et al.[28], Basiri et al.[16], Chen et al.[17] are some of the works which used this approach.
- The diameter distance technique involves selecting team members based on their distance to the center of the network, which is the node that has the longest distance from any other node in the network. The aim is to minimize the longest distance between any two team members, which can reduce communication costs. Lappas et al.[15], Selvarajah et al. [29] are some of the works which used this approach.
- The sum-of-distance approach involves selecting team members that minimize the sum of distances between them. The idea is to minimize the total communication cost by selecting team members that are closer to each other, thereby reducing the need for extensive communication. Kargar et al.[30][31][32], Li et al.[33], Selvarajah et al.[34] are some of the works which used this approach.

Furthermore, Lappas et al.[15] demonstrated that the TFP is computationally complex, belonging to the NP-hard class of problems, which are notoriously challenging to solve optimally. Their work laid the foundation for subsequent research in the field and provided a vital framework for further investigation of the TFP using a range of algorithmic approaches, including approximate, heuristic, and metaheuristic techniques.

Authors	Communication Cost	Work Load	Expertise Level	Personnel Cost	Geo Proximity
T. Lappas, K. Liu and E. Terzi [15]	✓				
M. Kargar and A. An [30]	✓				
C.T. Li and M.K. Shan [25][26]	✓				
C.T. Li, M.Y. Huang and R. Yan [33]	✓				
J. Basiri, F. Taghiyareh and A. Ghorbani [16]	✓				
W. Chen, J. Yang and Y. Yu [17]	✓				
K. Selvarajah, A. Bhullar, Z. Kobti and M. Kargar [34][35]	✓				
K. Selvarajah, Z. Kobti and M. Kargar [35]	✓				
A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis and S. Leonardi[27]	✓	✓			
A. Majumder, S. Datta and K. Naidu [28]	✓	✓			
J. H. Gutierrez, C. A. Astudillo, P. Ballesteros-Perez, D. Mora-Melia and A. Candia-Vejar [36]	✓	✓			
M. Kargar, A. An and M. Zihayat [37][32]	✓			✓	
B. Ashenagar, N. F. Eghlidi, A. Afshar and A. Hamzeh [18]	✓			✓	
K. Selvarajah, P. M. Zadeh, Z. Kobti, M. Kargar, M. T. Ishraque and K. Pfaff [38]	✓	✓			✓
Y. Han, Y. Wan, L. Chen, G. Xu and J. Wu [19]	✓				✓
L. Chen, Y. Ye, A. Zheng, F. Xie, Z. Zheng and M. R. Lyu [39]	✓				✓
C. Dorn and S. Dustdar [10]	✓		✓		
H. Zhu, E. Chen, H. Xiong, H. Cao and J. Tian [40]	✓		✓		
M. Neshati, S. H. Hashemi and H. Beigy [41]	✓		✓		

Table 2.1.1: Team Formation Problem based on various parameters [1]

Table 2.1.1 shows some research works which used communication cost in conjuction with the additional parameters such as work load, expertise level etc. for TFP. Anagnostopoulos et al.[27] used heuristic and metaheuristic approaches to solve the multi-task TFP. In particular, they propose a greedy algorithm that employs a number of heuristics to construct initial teams, followed by a local search algorithm that uses a simulated annealing metaheuristic to refine the team composition. Majumder et al.[28] used approximation algorithms to find nearly optimal teams with balanced workloads. Gutiêrrez et al.[36] used a constraint programming approach, a local search heuristic and a variable neighborhood search metaheuristic to solve the multiple tasks TFP. Selvarajah et al.[38] considered geo-proximity in addition to communication cost and workload. The authors used a cultural algorithm as a metaheuristic optimization technique to find the best team composition for palliative care.

Kargar et al.[37][32] combined communication costs with personnel costs to minimize the total cost of forming the team and maximize the quality of the team. To solve the TFP efficiently, the authors propose a two-stage approach. In the first stage, a heuristic algorithm is used to select a set of candidate team members based on their expertise and compatibility. In the second stage, a multi-objective genetic algorithm is used to optimize the team formation problem with respect to both cost and quality objectives. Ashenagar et al.[18] utilized a clustering algorithm and a multi-objective optimization framework that considered both team effectiveness and personnel cost. The clustering algorithm used the local distance metric to group individuals into small subgroups, and the multi-objective optimization framework optimized both team effectiveness and personnel cost to form teams that are both effective and cost-efficient.

Both Han et al.[19] and Chen et al.[39] used genetic algorithm-based optimization functions for forming a team with low communication costs which are geographically closer to each other. Few other research works Dorn et al.[10], Farhadi et al. [42], Zhu et al.[40], Neshati et al.[41], Zhang et al.[12] considered communication cost along with the level of skills using a heuristic based optimization function.

In all these works, researchers used multiple parameters such as communication cost, work load, expertise level, personnel cost, geo-proximity etc. in TFP because they believed these parameters can significantly affect the quality and efficiency of the teams that are formed. For example, Communication Cost can affect the team's ability to collaborate effectively, the Work Load can impact the team's productivity and Availability can influence the team's ability to meet project deadlines. By considering multiple parameters, researchers were able to better model real-world scenarios and create more realistic and effective team formation algorithms. Moreover, different parameters may be more or less important in different contexts, so considering multiple parameters helped researchers design more flexible and adaptable team formation algorithms that can be customized to different situations.

2.2 Deep Learning Approaches

Researchers have recently begun to examine deep learning architectures for team recommendation [43, 13]. Sapienza et al. [43] built a model using a traditional autoencoder to recommend the teams for online multiplayer games. The model was unable to handle the uncertainty in the training data [13]. This problem was then addressed by Rad et al.[13] by employing a variational Bayesian neural architecture in which the network weights are assigned a probability distribution, i.e., uncertainty in weight, which can be used to estimate uncertainty in predictions. However, this work ignored supporting information such as venues for expert recommendation and treated the relationship between the set of skills and the corresponding set of experts as a standalone instance.

The subsequent works by Rad et al. [2] considered a heterogeneous collaboration network to include all the associations between different nodes, i.e., skills, experts, papers, and their corresponding venues in the DBLP dataset. They used a meta-path-based random walks technique *i.e.* Metapath2Vec [24] which generates paths that are able to capture both the semantic and structural relationships between various types of nodes through randomwalk and skipgram. They demonstrated that incorporating contextual data into collaboration network node embedding enhances expert recommendations. In their approach, a meta-path $\mathcal{P}$ is defined as a path with predefined node and edge types pattern based on the DBLP network such that, $\mathcal{P} : V_1 \xrightarrow{r_1} V_2 \xrightarrow{r_2} ... \xrightarrow{r_l-1} V_l$, where $V_i \in${'Expert', 'Skill', 'Paper', 'Venue'} defines the type of node and r_i defines the type of relation [44]. The authors also showed that these composite patterns are useful in capturing the structural and semantic relations in the DBLP collaboration network. Specifically, they defined metapaths "$Expert \leftrightarrow Paper \leftrightarrow Expert$" for capturing the co-author relationship between Experts, "$Expert \leftrightarrow Paper \leftrightarrow Venue \leftrightarrow Paper \leftrightarrow Expert$" for capturing the co-venue relationship between Experts and "$Expert \leftrightarrow Paper \leftrightarrow Skill \leftrightarrow Paper \leftrightarrow Expert$" for capturing the co-skill relationship between Experts. Based on these metapaths, they used random walks [45] for crawling the collaboration network and then trained the node representations using an unsupervised heterogeneous skipgram model. These representations were then used by VBNN in the downstream task for ranking experts. Later in [46], the author incorporated an additional penalty term into the loss function of a VBNN to improve the quality of the recommended expert team by prioritizing candidates with stronger collaboration histories. The modified loss function seeks to minimize the difference between the predicted expert ranking and the actual expert ranking, while also penalizing recommendations of experts with low past collaboration scores in the top positions of the ranking list. More recently, in [20], Rad et al. used subgraph representation learning using the same Metapath2Vec technique, such that the subgraph comprises the set of skills and a corresponding set of experts who have successfully collaborated in the past. More precisely, the author identified the most similar expert subgraph representations to the input skill subgraph representations using the maximum similarity index instead of VBNN.

Although the models in [2][46][20] have modest variations, the evaluation metrics for these models do not seem to benefit from their procedures. Moreover, these models use contextual information in the DBLP heterogeneous network using Metapath2Vec [24] to learn the representations of nodes in the collaboration system. But since Metapath2Vec uses random walks [45] to generate node embedding, it comes under the category of shallow graph embedding approaches [47] and is a self-supervised learner. The shallow graph embedding approaches have been shown to face the following challenges, which we address in this paper:

- (C1) The number of parameters used to train the neural network in shallow embedding methods is directly proportional to the size of the network, so, the complexity is $\mathcal{O}(|V|)$, where $|V|$ represents the number of nodes in the collaboration network. It is a computationally inefficient method [47].
- (C2) The shallow embedding methods are inherently transductive, *i.e.*, they cannot generate embeddings for previously unseen nodes unless additional rounds of optimization are performed to optimize embeddings for these nodes. This is problematic for evolving collaboration networks because they are huge in size, therefore, cannot be entirely stored in the memory [47].
- (C3) These approaches fail to leverage node attributes during encoding, and in many collaboration networks, the attribute information is often highly informative with respect to the node's position and role in the graph [47].

Lv et al. [48] assert that the premise of Graph Neural Network (GNN) is to avoid the feature engineering process by automatically extracting the implicit and valuable characteristics underlying the network. In contrast, if the model uses Meta-path2Vec approach, it then involves human engineering to define the Meta-paths and the number of Meta-paths. They further prove that graph representation learning in heterogeneous collaboration networks using GNNs such as Graph Convolutional Networks [49] and Graph Attention Networks [23] outperforms meta-path-based random walks.

Therefore, in order to circumvent the shortcomings of the previous methodologies, this research work proposes a novel framework, LANT, to learn graph structure features from the heterogeneous network using Graph Attention Networks (GAT) and recommend experts for teams. GAT basically aggregates the node's neighborhood information and uses both the attention-based aggregation function and shared parameters to generate embedding for every node in the collaboration network [50].

Because GAT incorporates parameter sharing during the information aggregation with other nodes, it increases the computational efficiency, and hence it alleviates the challenge C1 posed by current solutions. As a result, LANT can generate embedding for nodes not observed during training [50] and hence solves the challenge C2 posed by the current state-of-the-art. Further, LANT incorporates the collaboration network graph structure by leveraging its node's attributes [50], and hence it can overcome challenge C3. We demonstrate empirically that our suggested technique outperforms the state-of-the-art methods for ranking experts in collaboration networks in terms of computational efficiency, functionality in inductive contexts, and improved quantitative performance.

Chapter 3

Proposed Approach

3.1 Introduction

Graph Neural Network (GNN) is a type of neural network designed to operate on graph-structured data, where each node in the graph is associated with a feature vector. The goal of a GNN is to learn a low-dimensional vector representation for each node, and/or the entire graph, that captures both the local and global structural information of the graph[51]. GNNs use node features to learn a vector representation of node, h_v, or the entire graph, h_G. A GNN usually consists of *graph aggregation layer* to aggregate node-level features into a graph-level feature vector, i.e., it employs a neighborhood aggregation technique, which involves iteratively updating a node's representation by aggregating representations of its neighbors. During each message passing iteration in GNN, the updated representation of the target node $\{h_v \in \mathbb{R}^F | \ v \in V\}$ associated with each node $v \in V$ updates after each iteration based on the information aggregated from v's neighborhood $\mathcal{N}(v)$, where F is the number of features in each node. The embedding or updated representation of target node v can be expressed as follows:

$$h_v = \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} h_u \tag{1}$$

where h_u is the feature vector of node u.

Graph Attention Networks [23] (GATs) are one of the most popular types of Graph

Neural Networks. Unlike other GNNs, which give equal importance to all neighbors and assigns static weights based on node degrees, GAT assigns dynamic weights to node features because some neighbors are more important than others. This process in GAT is called self-attention, which allows each node to weigh its own features, as well as the features of its neighbors in the graph, in order to compute an updated representation of itself. Specifically, the self-attention mechanism computes a set of attention coefficients, which represent the importance of each neighboring node to the current node. In team recommendation, the heterogeneous networks comprise several information in the neighbors such as publications (i.e., existing successful teams), skills, experts, and venue. LANT leverages the potential benefits of GAT to learn the graph features efficiently so that this information can be used in the team recommendation.

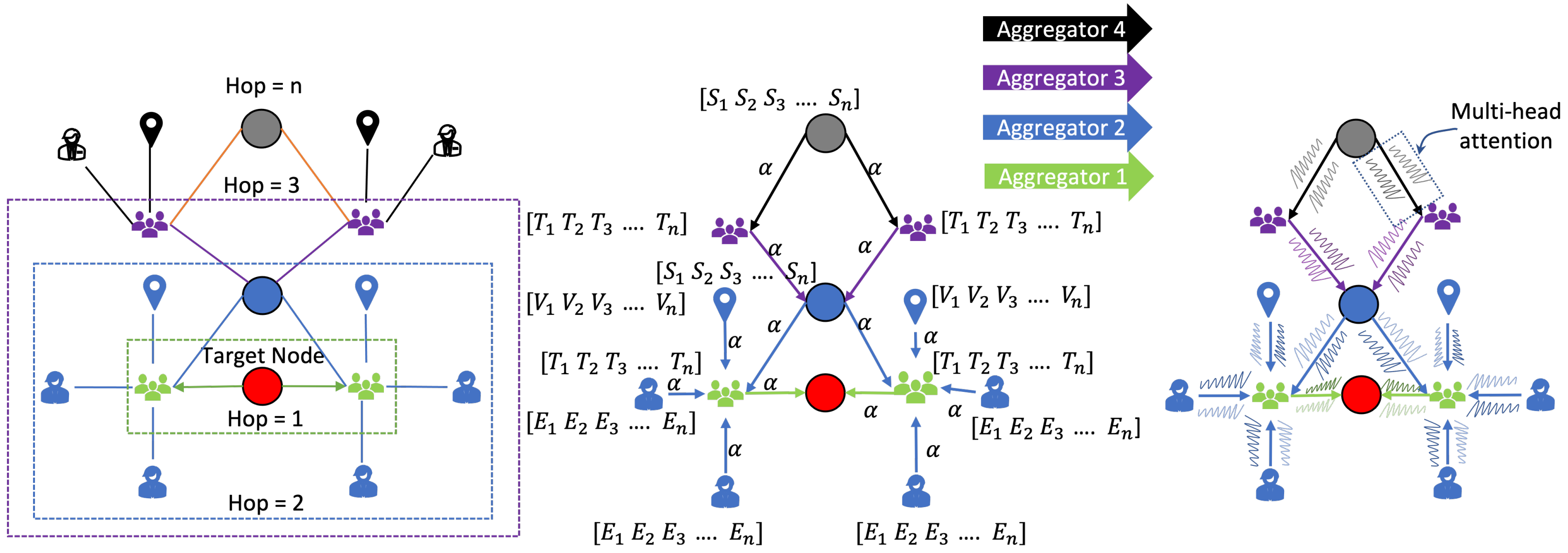
GNNs are supervised or semi-supervised algorithms for learning node representations that require data with class labels. Since LANT addresses an unsupervised task, where no label information can be used to train a graph neural network, we need to learn node embeddings in an unsupervised manner. Recently, Deep graph Infomax(DGI)[52] has achieved fairly high performance for unsupervised graph representation learning.

3.2 Team Recommendation Model

Our aim is to build a neural model for the team recommendation problem. We leverage the concept of Rad et al.[2] to define our architecture, consisting of two primary phases: 1) Transfer learning and 2) Neural Team Recommender.

3.2.1 Transfer learning in LANT

Transfer learning defines the learning process of the local substructure of nodes. For the team recommendation problem, we aim to learn the representations of experts and associated skills in a heterogeneous collaboration network. Because of the challenges with Metapath2Vec in capturing the node features and generating meta-paths,



(a) Sample Neighborhood in DBLP

(b) Aggregating information from other nodes

(c) Multi-head attention mechanism

Figure 3.1.1: GAN Architecture for Transfer Learning: (a) Sample Neighborhood in DBLP for a target node type → let's assume, Skill (Red) where green represents nodes at Hop 1, blue represents nodes at Hop 2, and purple represents nodes at Hop 3, respectively, (b) Aggregate information from the sample neighborhood using an attentive mechanism where α is an attentive co-efficient, n is the number of experts, and (c) Multi-head attention mechanism for independent attention (heads = 2 in the given figure) computations.

we decide to use GNN, especially Deep Graph Infomax [52] combined with Graph Attention Network (GAT) [23]

As shown in Figure 3.2.1, LANT uses GAT as an encoder in DGI to learn node embeddings in a self-supervised manner. G is a true graph with true information of nodes, edges, and features associated with each node, while H is a corrupted graph where the nodes and edges have been changed using a *corrupted function*, $\mathcal{C}$. Then, an encoder $\mathbb{E}$, GAT in our case, generates the node embeddings of G (i.e., h_i), and H (i.e., h_j). The node embedding of each node in G is summarized into a single embedding vector T_G, a global graph summary, by using the read-out function R. D is a discriminator, a logistic non-linear sigmoid function, that compares h_i and h_j against T_G using the following loss function [53].

$$L = \frac{1}{N+M}(\sum_{i=1}^{N} \mathbb{E}_{(h,A)}[logD(h_i, T_G)] + \sum_{j=1}^{M} \mathbb{E}_{(h',A')}[log(1 - D(h_j, T_G))]) \tag{2}$$

The encoder GAT in DGI updates the representation of a target node using a function called *attention function* $e(h_u, h_v)$ as shown in equation 3.

$$h_v = \sum_{u \in \mathcal{N}(v)} e(h_u, h_v) h_u \tag{3}$$

The attention function considers the embedding of both source and target nodes. It allows the weight to depend not only on the number of neighbors but also on capturing the features and local structures. So, the attention function allows the nodes to attend to some important nodes more than others. The normalized updated representation of a target node v can be written as:

$$h_v = \sum_{u \in \mathcal{N}(v)} \text{softmax}_u(e(h_u, h_v)) h_u \tag{4}$$

$$\alpha_{u,v} = \text{softmax}_u(e(h_u, h_v)) = \frac{exp(e(h_u, h_v))}{\sum_{k \in \mathcal{N}_i} exp(e(h_u, h_v))} \tag{5}$$

The attention score $e : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}$ calculates a score for every edge (u, v), where F is the number of features in each node.

$$e(h_u, h_v) = \text{LeakyReLU}(a^T.[Wh_u \parallel Wh_v]) \tag{6}$$

where $a \in \mathbb{R}^{2F'}$ is a learnable vector, $W \in \mathbb{R}^{F' \times F}$ is learnable linear projection matrix, and $\parallel$ denotes vector concatenation. The normalized attention score or the weight for every edge (u, v) is defined as:

$$\alpha_{u,v} = \frac{exp\left(\text{LeakyReLU}\left(a^T.[Wh_u \parallel Wh_v]\right)\right)}{\sum_{k \in \mathcal{N}_u} \text{LeakyReLU}(a^T.[Wh_u \parallel Wh_k])} \tag{7}$$

As shown in Figure 3.1.1, the target node takes the messages, i.e, features, from its neighbors and combines them with the linear projection matrix W and normalized attention score. As shown in Figure 3.1.1 (a), we consider 4 hops of connections with the node of interest. In Figure 3.1.1 (b), we then have 4 different aggregators (aggregator 1, aggregator 2, aggregator 3, and aggregator 4) for message passing and aggregation functionality. So, the normalized updated representation of a target node can be written as below;

$$h_v = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{u,v} W h_u\right) \tag{8}$$

where σ is a non-linearity activation.

We then use a multi-head attention mechanism, as shown in Figure 3.1.1 (c) with heads = 2, for stabilizing the learning process of the attention mechanism; it creates multiple attention scores – each replica with a different set of parameters in the following way:

$$h_v[1] = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha^1_{u,v} W h_u\right) \tag{9}$$

$$h_v[2] = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha^2_{u,v} W h_u\right) \tag{10}$$

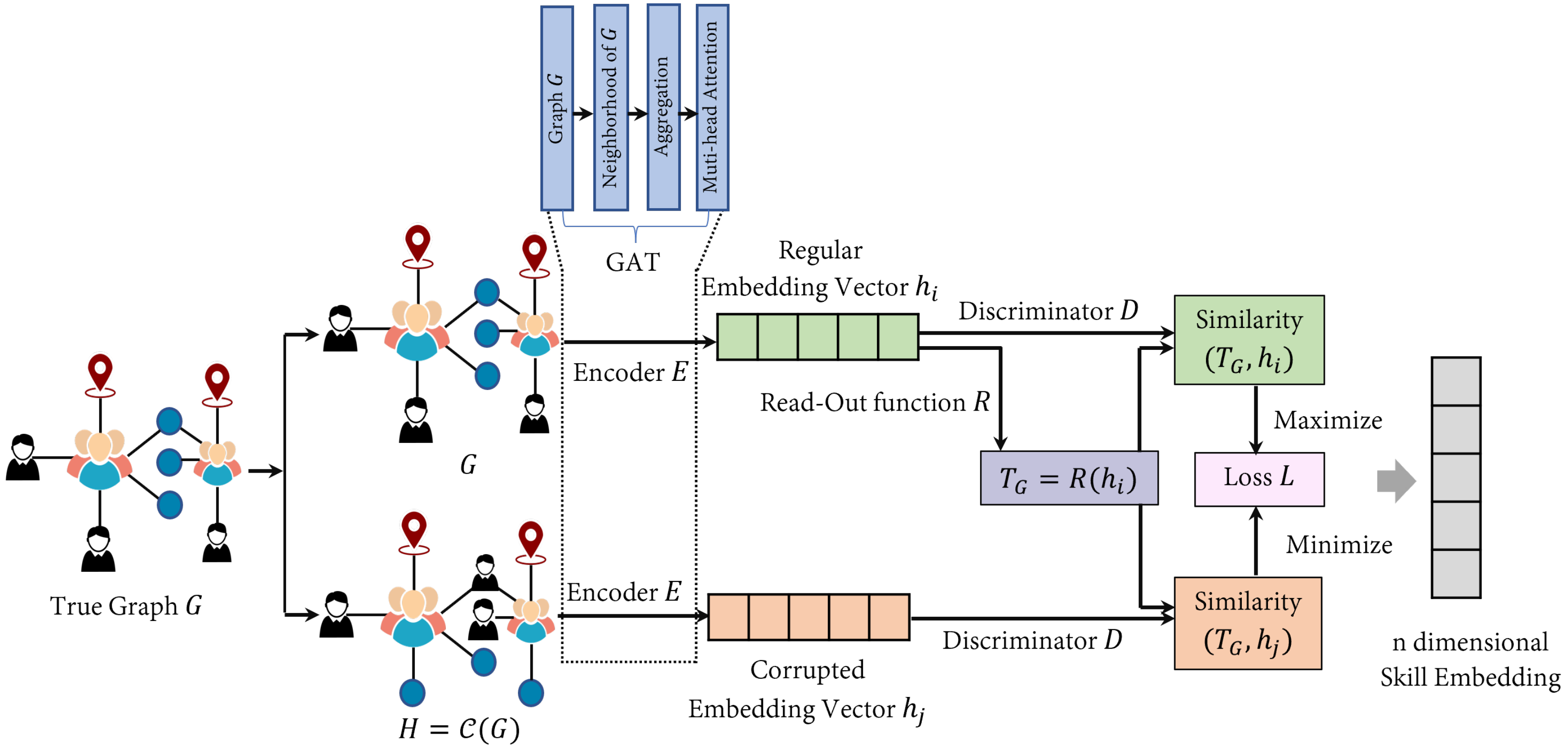


Figure 3.2.1: Proposed Architecture for Transfer Learning with GAT and DGI

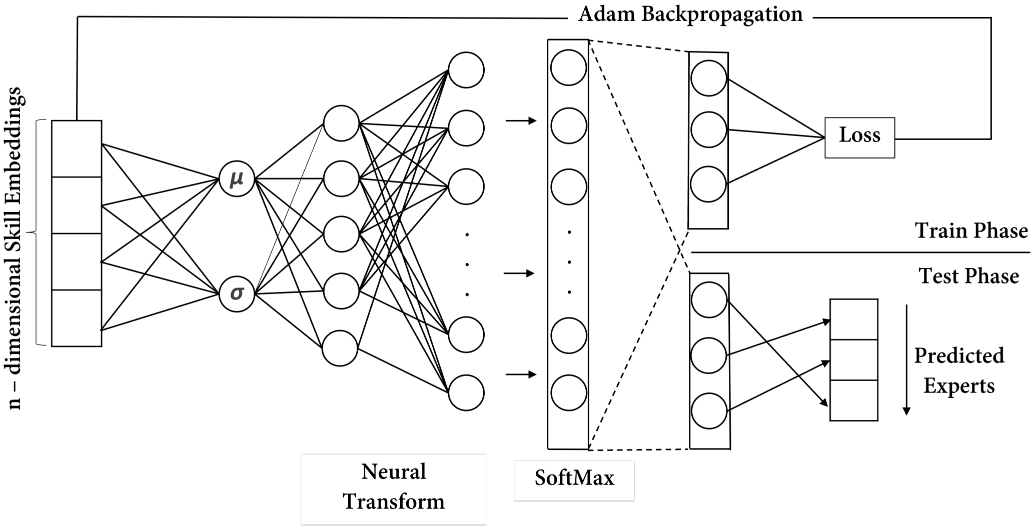


Figure 3.2.2: VBNN for experts recommendation – adapted from Rad et al. [2]

The outputs from the multi-head attention mechanism in equations 9 and 10 are then aggregated (AGG) by concatenation or summation in the following way:

$$h_v = AGG(h_v[1] + h_v[2]) \tag{11}$$

3.2.2 Neural Team Recommender

In the second phase, we adapt the fine-tuning methodology by Rad et al. [2] to construct LANT's team recommender system. The aim is to recommend a team of experts $e \subseteq \mathcal{E}$ for a given skill subset $s \subseteq \mathcal{S}$ using the transform function $f(s; \theta)$. We need to learn the transform function f from the skill space to the experts' space; therefore, we build variational neural networks as shown in Figure 3.2.2. We initialize the input skill embedding vector with the embedding representations $\vec{\mathcal{S}_n}$ from the transfer learning and train the transform function with parameter θ of the conditional probability $p(e|s, \theta)$.

Here, the Bayes rule is used to infer the posterior distribution after evaluating the prior distribution of θ. Evaluating the posterior distribution $p(\theta|\mathcal{T}) = p(\mathcal{T}|\theta)p(\theta)$, however, $p(\theta)$ is hard to calculate. Therefore, we approximate it by a solvable distribution $q(\theta|\mu, \sigma)$ with multivariate Gaussian distribution $N(\mu, \sigma^2)$, where μ is means and σ is Variance. To estimate the actual posterior $p(\theta)$ by $q(\theta|\mu,\sigma)$, Kullback-Leibler divergence is used to minimize the difference between q and p.

$$KL(q(\theta|\mu,\sigma) \parallel p(\theta)|\mathcal{T})) = \int q(\theta|\mu,\sigma)\ log\left[\frac{q(\theta|\mu,\sigma)}{p(\theta)|\mathcal{T})}\right] d\theta = \mathbb{E}_{q(\theta|\mu,\sigma)} log\left[\frac{q(\theta|\mu,\sigma)}{p(\theta)|\mathcal{T})}\right] d\mathcal{T})$$

$$= KL(q(\theta|\mu,\sigma) \parallel p(\theta))) - \mathbb{E}_{q(\theta|\mu,\sigma)} log\ p(\theta|\mathcal{T}) + log\ p(\mathcal{T})) \quad (12)$$

If we assume l_0 is our input layer, we can define $l_0 = \rho(\vec{\mathcal{S}}_n)$, where ρ calculates the $N(\mu,\sigma^2)$ for the input layer. Then it passes through hidden layers with their activation function, ReLu to the output layer with the activation function, softmax. To fine-tune the embeddings, we use Adam backpropagation. We train the neural network with the loss function in equation 12.

Chapter 4

Experimental Setup

This chapter describes our experimental setup and environment, including tools and libraries used to implement our framework (LANT), system configuration, hyper-parameters for training, dataset details, details of evaluation metrics and baseline models that are used to evaluate our model.

4.1 Tools and Libraries

Our proposed framework, LANT, has been implemented using a combination of the Graph Attention Network (GAT) and Deep Graph Infomax (DGI) algorithms, both of which are implemented in the Stellargraph[54] library. The StellarGraph[54] library offers state-of-the-art algorithms for graph machine learning, making it easy to discover patterns and answer questions about graph-structured data. The up-to-date version of our code for the proposed framework, LANT, is available on GitHub [1]. We have implemented our code in Python 3.8.0 language. The details of the used libraries are listed below:

- Stellargraph 1.2.1
- TensorFlow 2.9.1
- Keras 2.9.0
- Scikit-learn 0.24.2
- NumPy 1.20.3

[1] https://github.com/LANT

- Pandas 1.4.3
- Matplotlib 3.4.2
- Pytrec-Eval 0.5

4.2 System Configuration

To evaluate the efficiency and effectiveness of our model, we build the proposed architecture and compared it against the state-of-the-art models. All the experiments are conducted on Macbook-Air 2017 with device specifications of a 1.8 GHz Dual-Core Intel Core i5 Processor, 8 GB 1600 MHz DDR3 of RAM, and a 64-bit based Mac OS Monterey version 12.6.3.

4.3 Datasets

The proposed model is a generalized model which works well on several datasets. We have evaluated our model (LANT) on two benchmark datasets - DBLP and IMDB with several other state-of-the-art models. Data within both datasets are divided into two groups - Transductive and Inductive:

- Transductive: Here we use the Transductive DBLP and Transductive IMDB with the statistics shown in Table 4.3.1 for the entire process. In this step, the nodes and edges in the heterogeneous collaboration network used in the training phase remain the same in the testing phase.
- Inductive: We use the same Transductive DBLP and Transductive IMDB as above with the statistics shown in Table 4.3.1 for training. Here we modify the networks with the Inductive DBLP and Inductive IMDB correspondingly with the statistics as shown in Table 4.3.1 and directly use the trained LANT model on Transductive DBLP and Transductive IMDB for testing on the modified network.

Table 4.3.1 summarizes the basic statistics of each benchmark dataset. The datasets contain the data in the form of a Comma-Separated Values (CSV) file.

- DBLP[2] dataset is a bibliographic database that contains information about publications in the fields of computer science and related disciplines. It contains information about papers, experts, venues and skills, among other details.
- IMDB[3] dataset is a large and widely-used dataset of movie information. It contains information about movies, directors, release years, and genres, among other details. In IMDB, genres are considered as skills and directors are considered as experts who successfully collaborated to release a movie.

Table 4.3.1: Dataset Statistics

DBLP 2	Papers	Experts	Skills	Venues	Nodes	Edges
Transductive DBLP	4678	2076	1985	21	8760	51228
Inductive DBLP	33002	2470	2000	21	37493	301369

IMDB 3	Movies	Directors	Genres	Years	Nodes	Edges
Transductive IMDB	22392	1928	28	81	24429	194243
Inductive IMDB	44785	2000	28	81	46894	385910

In our analysis of the datasets, we noticed that the experts in DBLP and the directors in IMDB suffer from the long-tailed distribution as shown in figure 4.3.1 and figure 4.3.2, wherein few experts collaborated extensively whereas the majority collaborated sparingly.

[2] *https://originalstatic.aminer.cn/misc/dblp.v12.7z /*

[3] *https://datasets.imdbws.com/*

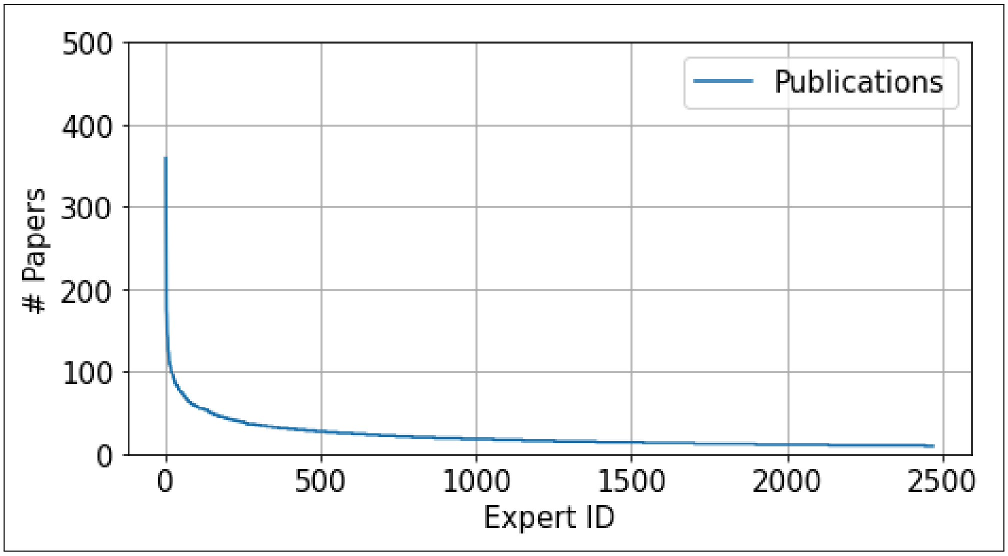


Figure 4.3.1: Distribution of Papers per Expert - DBLP

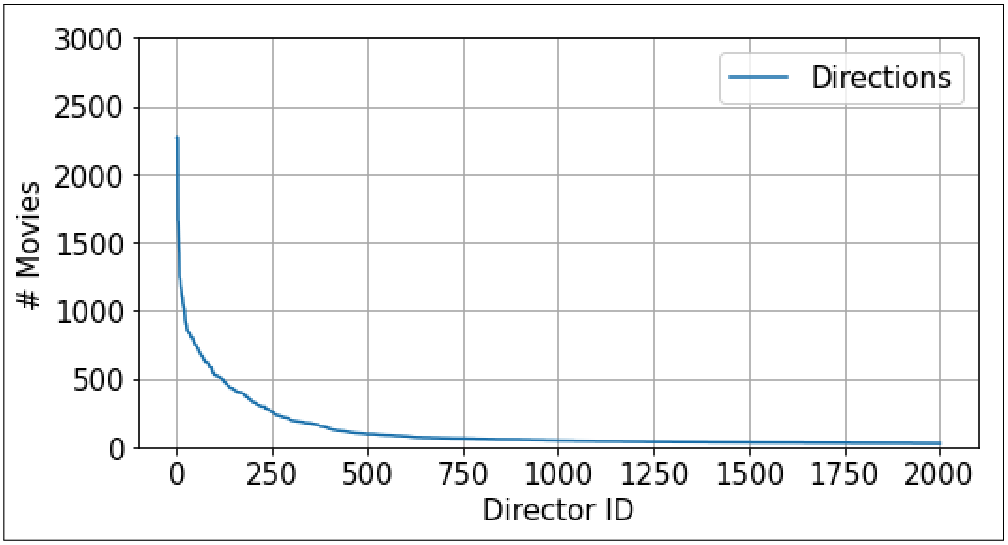


Figure 4.3.2: Distribution of Movies per Director - IMDB

4.4 Evaluation Metrics

We consider publications in the DBLP dataset and movies in the IMDB dataset as the ones having optimal teams. We evaluated the performance of our framework using Quantitative Evaluation and Qualitative Evaluation against the baselines:

- Quantitative Evaluation: The quantitative metrics aim to assess the computational complexity of the proposed framework against baselines. Additionally, it also aims to evaluate the recommendations in order of their relevance by comparing it to a pre-defined ground truth. Here we used the following metrics for comparisons against baselines.

  a) Computational Efficiency

  b) Ranking Metrics

  c) AUCROC

- Qualitative Evaluation: The qualitative metrics aim to assess the recommended experts' quality relative to the ground truth experts. Here we used the following metrcis for our analysis:

  a) Skill Association

  b) Skill Expertise Ratio

  c) Collaboration Score

4.4.1 Quantitative Evaluation

4.4.1.1 Computational Efficiency

In the evaluation of the proposed deep learning framework for team recommendation in this thesis, computational efficiency is considered a key evaluation metric. To compare the proposed model's performance with that of the baseline models, training time per epoch is used as a measure of efficiency. By analyzing the training time per epoch of the proposed model and comparing it with the baseline models, the research aims to determine if the proposed model is able to achieve similar or better results while requiring less training time per epoch. This evaluation metric provides valuable insights into the model's efficiency and scalability for large-scale datasets. The results of this analysis are beneficial for future research in developing more efficient deep learning models that can be trained more quickly and at lower computational costs.

4.4.1.2 Ranking Metrics

The state-of-the-art in neural team recommendation [13] [2] considers ranking metrics to analyze the effectiveness of the models. The ranking metrics used are Recall, Mean Reciprocal Rank (MRR), Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG). We use the same metrics for our evaluation. Below is a detailed explanation of the metrics:

- **Recall:** Recall measures the ability of the model to identify relevant instances or experts from a larger set of potential candidates. It measures the proportion of relevant experts that the model is able to identify out of all the experts in the dataset.

  To calculate recall, the model is evaluated on a test set that includes examples of teams with known expert compositions. The model predicts a set of experts for each team, and the predicted set is compared to the true set of experts using the following formula:

  $$Recall = \frac{true\ positives}{\#\ of\ relevant\ experts} \tag{1}$$

  true positives represent the number of correctly identified relevant experts, while *# of relevant experts* represent the the true set of experts.

  A high recall score indicates that the model is effective at identifying relevant experts, while a low score indicates that the model is missing important experts. By optimizing for recall, the model can be trained to prioritize the identification of relevant experts, which can lead to more effective team compositions.

- **Mean Reciprocal Rank (MRR):** MRR measures the average of the reciprocal ranks of the relevant experts in the model's predictions. It measures the average rank of the first relevant expert in the model's predictions. For example, if a team has five experts, and the model correctly identifies the top expert, the reciprocal rank for this expert would be 1/1=1. If the model correctly identifies the second expert, the reciprocal rank would be 1/2=0.5, and so on. The MRR is calculated as the average of these reciprocal ranks over all the teams in the test set. The formula for calculating MRR is as follows:

$$MRR = \frac{1}{|n|} \sum \frac{1}{rank_i} \tag{2}$$

where n represents the total number of teams in the test set and $rank_i$ is the rank of the first relevant expert in the model's predictions for the i^{th} team and $\sum$ is the sum of all reciprocal ranks.

A high MRR score indicates that the model is effective at identifying the most relevant experts, while a low score indicates that the model is less accurate. By optimizing for MRR, the model can be trained to prioritize the identification of the most relevant experts, which can lead to more effective team compositions.

- **Mean Average Precision (MAP):** MAP is a measure of the average precision of the model's predictions for a set of teams with known expert compositions. It measures the average precision of the model's predictions for a set of teams.

  Precision is defined as the proportion of relevant experts in the model's predictions, while *Recall* is the proportion of relevant experts in the test set that the model identifies. Precision and recall are related by the trade-off between identifying all relevant experts and minimizing the number of false positives.

  The *Average Precision* is calculated by computing the precision for each team in the test set and then taking the average of these values. The *MAP* is the average of the average precision across all teams in the test set.

  The formula for calculating MAP is as follows:

$$MAP = \frac{1}{n}\sum AP_i \tag{3}$$

where n is the number of teams in the test set, AP_i is the average precision for the i^{th} team, and $\sum$ is the sum of all average precisions.

The formula for calculating the average precision (AP) for a given team is as follows:

$$AP_i = \frac{1}{m}\sum P@k * rel(k) \tag{4}$$

where m is the total number of relevant experts in the team, P@k is the precision at the k^{th} position in the model's predictions, and rel(k) is an indicator function that is 1 if the expert at the k^{th} position is relevant, and 0 otherwise.

To calculate precision at position k, we use the formula:

$$P@k = \frac{\#\ of\ relevant\ experts\ in\ top\ k\ predictions}{k} \tag{5}$$

where k is the current position in the ranking.

Overall, the MAP metric aggregates the average precision values across all teams in the test set to give a comprehensive measure of the model's performance in identifying relevant experts.

- **Normalized Discounted Cumulative Gain (NDCG):** NDCG is a measure of the quality of the model's predicted rankings for a set of experts. It measures the quality of the model's predicted rankings by assigning a score to each predicted ranking based on the relevance and order of the experts. The relevance of each expert is assigned a value between 0 and 1, based on their importance to the team. The order of the experts is assigned a discount factor that decreases with position, reflecting the fact that experts who are ranked lower are less important. The formula for calculating NDCG is as follows:

$$NDCG = \frac{DCG}{iDCG} \tag{6}$$

where DCG is the discounted cumulative gain of the predicted ranking, and iDCG is the ideal discounted cumulative gain, which represents the maximum possible score for the ranking.

The discounted cumulative gain (DCG) is calculated as follows:

$$DCG = \sum \left(\frac{rel(i)}{log_2(i{+}1)} \right) \tag{7}$$

where *rel(i)* is the relevance score of the expert at position i in the predicted ranking.

The ideal discounted cumulative gain *iDCG* is calculated by sorting the true set of experts in descending order of relevance and applying the same formula for DCG.

The relevance score for each expert can be defined in various ways, depending on the context of the team formation problem. For example, the relevance score could be based on the expert's domain expertise, their past performance on similar projects, or their fit with the existing team members.

By normalizing the DCG score with the iDCG score, NDCG provides a value between 0 and 1 that indicates the quality of the predicted ranking relative to the ideal ranking. A higher NDCG score indicates a better-predicted ranking. By optimizing for NDCG, the model can be trained to prioritize the identification of the most relevant experts and their appropriate order in the predicted rankings, which can lead to more effective team compositions.

4.4.1.3 AUCROC

AUCROC (Area under the ROC curve) is a measure of the model's ability to distinguish between relevant and irrelevant experts based on their predicted scores. The ROC(Receiver Operating Characteristic) curve is a plot of the true positive rate (TPR) against the false positive rate (FPR) for different threshold values of the model's predicted scores. The TPR represents the proportion of relevant experts that are correctly identified by the model, while the FPR represents the proportion of irrelevant experts that are incorrectly identified by the model.

The AUCROC is the area under the ROC curve, which provides a single value between 0 and 1 that represents the overall performance of the model in distinguishing between relevant and irrelevant experts. A higher AUCROC score indicates a better performance.

To calculate the AUCROC score, the model's predicted scores for a set of experts are first sorted in descending order. Then, for each threshold value, the TPR and FPR are calculated as follows:

$$TPR = \frac{\#\ of\ true\ positives}{\#\ of\ relevant\ experts} \tag{8}$$

$$FPR = \frac{\#\ of\ false\ positives}{\#\ of\ irrelevant\ experts} \tag{9}$$

where a true positive is a relevant expert that is correctly identified by the model, a false positive is an irrelevant expert that is incorrectly identified by the model, and the number of relevant and irrelevant experts are known from the dataset.

The ROC curve is then plotted by connecting the TPR and FPR values for each threshold value. The AUCROC score is calculated as the area under the ROC curve. By optimizing for AUCROC, the model can be trained to prioritize the identification of the most relevant experts while minimizing the identification of irrelevant experts, which can lead to more effective team compositions.

4.4.2 Qualitative Evaluation

In qualitative analysis, we maintained the number of recommended experts equal to the number of experts in the ground truth. While the ranking metrics used in the quantitative evaluation section 4.4.1.2 examined whether the original experts were retrieved by the team recommendation methods, the quality metrics are focused on analyzing the quality of the recommended experts in comparison to the ones in the ground truth using the following metrics:

4.4.2.1 Skill Association

In Skill Association based qualitative analysis, we need to identify experts who can provide valuable insights on specific topics or skills. To do this, we compare the recommendations of our system with the list of experts who are already known to have the necessary expertise (the ground truth). To evaluate the performance of our recommended experts, we count how many times each expert has worked on the specific topics or skills in question (i.e. input). We then compare the total number of times each recommended expert has worked with the total number of times the experts in the ground truth have worked on the same topics or skills. If the recommended expert has worked on the topics or skills an equal or greater number of times than the experts in the ground truth, we consider the recommended expert to be as good as the experts in the ground truth.

For example, suppose we have two input skills (S1 and S2) and two recommended experts (A and B), and two experts in the ground truth (C and D). We count how many times A and B have worked on S1 and S2, and similarly, for C and D. We then compare the total number of times A and B have worked on S1 and S2 with the total number of times C and D have worked on those skills. If A and B have worked on S1 and S2 an equal or greater number of times than C and D, we consider A and B to be as good as C and D.

We then report the average percentage of the recommended experts who had equal or better skill associations in comparison to the experts in the ground truth.

4.4.2.2 Skill Expertise Ratio

In Skill Expertise Ratio based qualitative analysis, we first calculated the skill expertise ratio of individual expert on each input skill. This ratio is obtained by dividing the number of times an expert has worked on a specific input skill by the total number of skills the expert has worked on. Once we have obtained the skill expertise ratios for each expert and input skill, we calculate the combined skill expertise ratio of each expert by summing up their skill expertise ratios across all input skills. This provides us with a comprehensive measure of an expert's overall expertise level across all input skills.

For example, let's say we have two input skills (S1 and S2) and two recommended experts (A and B) and two experts in the ground truth (C and D). Let's assume that in the dataset experts A, B, C, and D each worked on four different skills. We calculate the skill-expertise ratio of A on S1 by dividing the number of times A has worked on S1 by 4. We do the same for S2. We then sum up the skill-expertise ratios of A across all input skills to get A's combined skill expertise ratio. We do the same for B, C, and D. Next, we compare the combined skill expertise ratio of A with those of C and D. If A has a combined skill expertise ratio that is equal to or greater than both C and D, we consider A to be as good as C and D for those input skills. Similarly, we compare the combined skill expertise ratio of B with those of C and D. If B has a combined skill expertise ratio that is equal to or greater than both C and D, we consider B to be as good as C and D for those input skills.

In summary, our evaluation method involves comparing the combined skill expertise ratios of each recommended expert with those of the experts in the ground truth to determine whether the recommended experts are as good as the experts in the ground truth for the given input skills.

We then report the average percentage of skill expertise ratio of the recommended experts which were equal to or better than the experts in the ground truth.

4.4.2.3 Collaboration Score

For Collaboration Score-based qualitative analysis, we measured the number of times the recommended experts collaborated with each other on the given set of input skills. We then compared this number with the collaboration frequency of the experts in the ground truth.

For example, if we have three recommended experts named A, B, C and three experts in the ground truth named D, E, F, we would create all possible combinations of teams using the recommended experts (AB, BC, AC) and do the same for the experts in the ground truth (DE, EF, DF). Then, we would compare the sum of the number of times each team in the recommended expert set collaborated to the sum of the number of times each team in the ground truth expert set collaborated. If a recommended team had an equal or higher collaboration score compared to the teams in the ground truth, we considered the recommended team to be as good as the ground truth team.

Finally, we reported the percentage of recommended teams that were equal to or better than the teams in the ground truth.

4.5 Baselines

The following state-of-the-art are compared against our proposed model, LANT:

a) ParagraphVectors'20[13]: In this work, they incorporated Document to Vectors method Paragraph Vectors[55] for low dimensional dense representation of sparse skill vectors, which are used as inputs by Variational Bayesian Neural Network for ranking experts.

b) Metapath2Vec'21[2]: In this work, they incorporated the Heterogeneous Graph Representation Learning method Metapath2Vec[24] for low dimensional dense representation of sparse skill vectors, which are used as inputs by Variational Bayesian Neural Network model for ranking experts.

Additionally, we compared the baselines and our proposed framework against randomly selecting experts based on the input set of skills. It helps to establish the minimum level of performance that the system must surpass to be considered useful.

Chapter 5

Discussions, Comparisons and Analysis

In this chapter, we compare our results with the state-of-the-art deep neural network-based models *ParagraphVectors'20* [13] and *Metapath2Vec'21* [2] used in Team formation problems. We conducted our experiments on two different datasets. The details of the experimental setup are explained in Chapter 4. Herein, we have not compared our results with deep neural frameworks for TFP in [46] and [20] because although they performed better than [13] but we did not notice any significant improvements in evaluation metrics with their changes when compared with [2]. Therefore, we used [2] as the baseline for our comparative analysis.

To determine the efficiency of our proposed framework, we measured and compared the computational time for graph representation learning of our framework against Metapath2vec used in [2] and the document-to-vector or paragraph-vector representation learning used in [13]. Our comparative results indicate that our model converges faster than both the previous representation learning methods.

5.1 Result Analysis

5.1.1 Quantitative Evaluation - DBLP

5.1.1.1 Computational Efficiency - DBLP

Here, we used Transductive DBLP with the attributes as shown in Table 4.3.1.

MetaPath2Vec'21 [2] and our proposed framework LANT uses graph representation learning algorithms for generating low-dimensional real-valued vectors for nodes in a network. In terms of computational efficiency, as shown in Figure 5.1.1, MetaPath2Vec takes approximately 600 seconds for 100 epochs to generate embeddings for the DBLP network as it trains the embeddings for each node individually. On the other hand, LANT uses Graph Attention Networks (GAT) to train an aggregation function using an attentive mechanism, enabling it to generate embeddings in a much shorter time of approximately 60 seconds for the DBLP network.

ParagraphVectors'20 [13], on the other hand, uses the document to vectors technique for generating fixed-length feature representations of variable-length pieces of text such as paragraphs, documents, or sentences. To generate the document embeddings for the DBLP network, Paragraph Vectors [55] takes approximately 155 seconds for 100 epochs. It considers teams as documents and skills as document words to map them into a real-valued embedding space. Overall, we observed a decrease of **90%** and **61%** in the time taken when LANT is used instead of MetaPath2Vec'21 and ParagraphVectors'20 correspondingly in transductive settings.

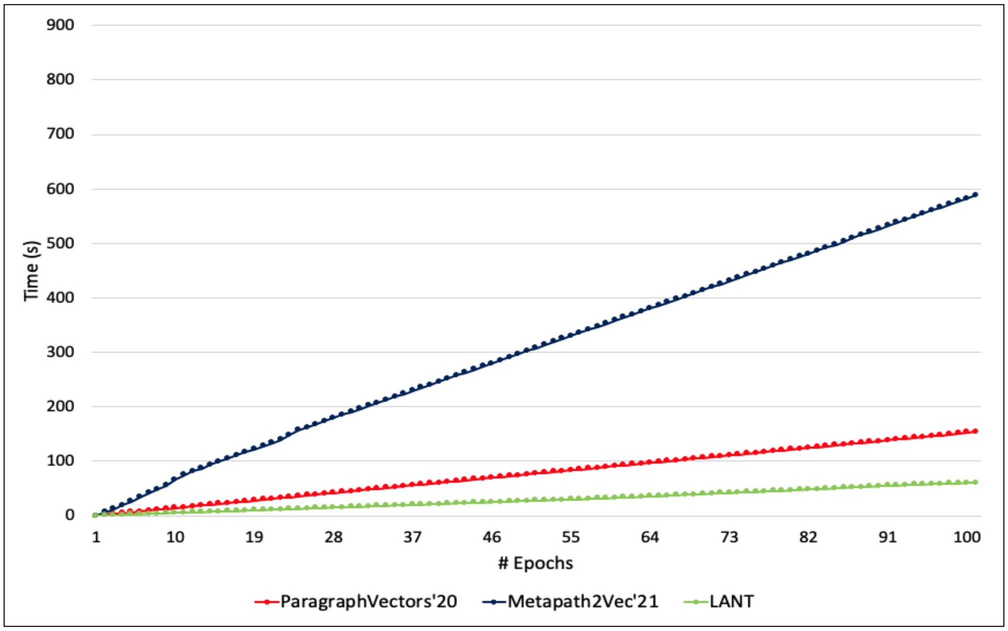


Figure 5.1.1: ParagraphVectors'20 vs Metapath2Vec'21 vs LANT - Computation Time in Transductive DBLP

5.1.1.2 Ranking Metrics - DBLP

Table 5.1.1 shows the performance of our proposed model against baselines in Transductive DBLP.

Table 5.1.1: Performance of LANT vs baselines models on Transductive DBLP

Transductive Learning					
Metrics	Models	TOP@1	TOP@3	TOP@5	TOP@10
Recall (%)	Random	0.12	0.77	1.28	2.46
	ParagraphVectors'20	1.17	2.23	3.20	5.34
	Metapath2Vec'21	1.17	2.25	3.23	5.35
	LANT	1.17	**2.27**	**3.24**	**5.40**
MRR (%)	Random	0.43	1.28	1.69	2.14
	ParagraphVectors'20	4.42	6.02	6.67	7.43
	Metapath2Vec'21	4.42	6.07	6.74	7.47
	LANT	4.42	**6.10**	**6.75**	**7.51**
MAP (%)	Random	0.12	0.38	0.5	0.73
	ParagraphVectors'20	1.17	1.63	1.88	2.22
	Metapath2Vec'21	1.17	1.64	1.89	2.25
	LANT	1.17	**1.65**	**1.90**	**2.26**
NDCG (%)	Random	0.43	0.75	1.02	1.62
	ParagraphVectors'20	4.42	3.11	3.32	4.30
	Metapath2Vec'21	4.42	3.14	3.35	4.33
	LANT	4.42	**3.15**	**3.40**	**4.35**

Table 5.1.2 shows the performance of our proposed model in Inductive DBLP. As in inductive settings, the baseline models fail to infer low dimensional vector representation for newly added nodes in DBLP network, therefore we only calculated LANT performance in this case.

Table 5.1.2: Performance of LANT vs baselines models on Inductive DBLP

Inductive Learning					
Metrics	Models	TOP@1	TOP@3	TOP@5	TOP@10
Recall(%)	Random	0.06	0.3	0.43	0.77
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**0.42**	**1.08**	**1.42**	**2.55**
MRR(%)	Random	0.1	0.26	0.31	0.37
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**0.99**	**1.52**	**1.67**	**1.99**
MAP(%)	Random	0.06	0.15	0.18	0.23
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**0.42**	**0.70**	**0.78**	**1.00**
NDCG(%)	Random	0.1	0.22	0.28	0.41
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**0.99**	**1.01**	**1.15**	**1.63**

5.1.1.3 AUCROC - DBLP

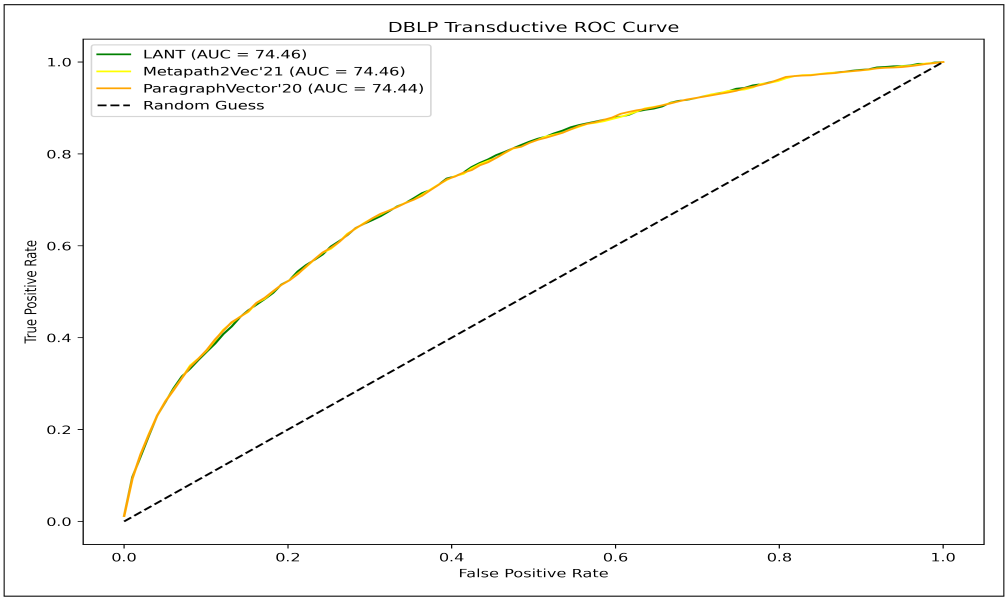


Figure 5.1.2: AUCROC - Transductive DBLP

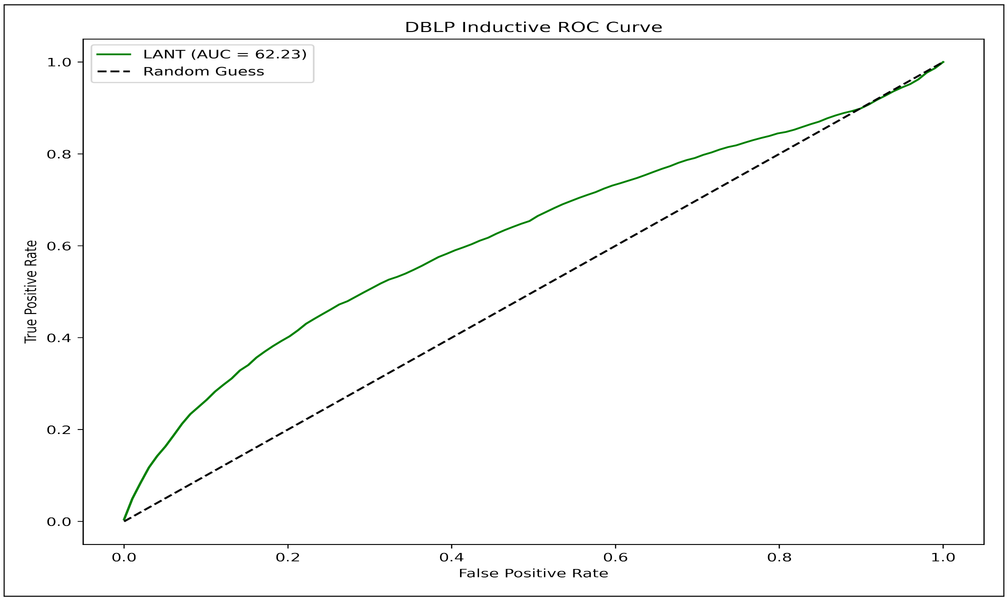


Figure 5.1.3: AUCROC - Inductive DBLP

5.1.2 Qualitative Evaluation - DBLP

5.1.2.1 Skill Association - DBLP

Table 5.1.3: Performance of LANT vs baselines

Models	Transductive Learning	Inductive Learning
Random	4.34%	5.65%
ParagraphVectors'20	67.56%	NA
Metapath2Vec'21	67.58%	NA
LANT	**67.62%**	**91.45%**

5.1.2.2 Skill Expertise Ratio - DBLP

Table 5.1.4: Performance of LANT vs baselines

Models	Transductive Learning	Inductive Learning
Random	0.43%	0.67%
ParagraphVectors'20	3.27%	NA
Metapath2Vec'21	3.33%	NA
LANT	**3.41%**	**16.34%**

5.1.2.3 Collaboration Score - DBLP

Table 5.1.5: Performance of LANT vs baselines

Models	Transductive Learning	Inductive Learning
Random	0.83%	1.42%
ParagraphVectors'20	18.77%	NA
Metapath2Vec'21	18.80%	NA
LANT	**18.85%**	**27.42%**

5.1.3 Quantitative Evaluation - IMDB

5.1.3.1 Computational Efficiency - IMDB

Here, we used Transductive IMDB with the attributes as shown in Table 4.3.1. MetaPath2Vec'21 [2] and our proposed framework LANT uses graph representation learning algorithms for generating low-dimensional real-valued vectors for nodes in a network. In terms of computational efficiency, as shown in Figure 5.1.4, MetaPath2Vec takes approximately 850 seconds for 100 epochs to generate embeddings for the IMDB network as it trains the embeddings for each node individually. On the other hand, LANT uses Graph Attention Networks (GAT) to train an aggregation function using an attentive mechanism, enabling it to generate embeddings in a much shorter time of approximately 99 seconds for the IMDB network.

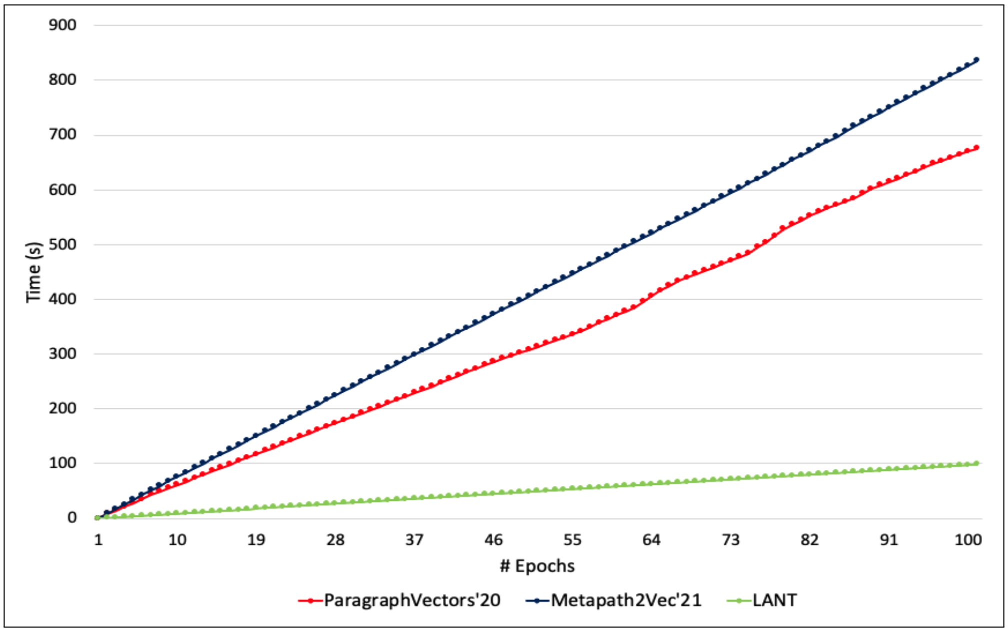


Figure 5.1.4: ParagraphVectors'20 vs Metapath2Vec'21 vs LANT - Computation Time in Transductive IMDB

ParagraphVectors'20 [13], on the other hand, uses the document to vectors technique for generating fixed-length feature representations of variable-length pieces of text such as paragraphs, documents, or sentences. To generate the document embeddings for the IMDB network, Paragraph Vectors [55] takes approximately 675 seconds for 100 epochs. It considers teams as documents and skills as document words to map them into a real-valued embedding space. Overall, we observed a decrease of **88.35%** and **85.33%** in the time taken when LANT is used instead of MetaPath2Vec'21 and ParagraphVectors'20 correspondingly in transductive settings. We also notice that there is not much significant improvement in the top positions for TOP@1, TOP@3, and TOP@5 scores. We believe that it is because of the limitation of the long-tailed distribution of directors (fewer directors collaborated highly, whereas the majority collaborated sparingly) in the IMDB dataset as shown in Figure 4.3.2.

5.1.3.2 Ranking Metrics - IMDB

Table 5.1.6 shows the performance of our proposed model against baselines in Transductive IMDB.

Table 5.1.6: Performance of LANT vs baselines models on Transductive IMDB

Transductive Learning					
Metrics	Models	TOP@1	TOP@3	TOP@5	TOP@10
Recall (%)	Random	0.25	0.57	0.89	1.52
	ParagraphVectors'20	1.21	3.21	4.49	7.75
	Metapath2Vec'21	1.22	3.21	4.53	7.76
	LANT	1.22	**3.22**	4.53	**7.77**
MRR (%)	Random	0.77	1.44	1.72	2.07
	ParagraphVectors'20	6.48	10.73	11.49	11.94
	Metapath2Vec'21	6.49	10.74	11.58	12.09
	LANT	6.49	10.74	**11.60**	**12.10**
MAP (%)	Random	0.25	0.63	0.94	1.63
	ParagraphVectors'20	1.21	2.15	2.61	3.64
	Metapath2Vec'21	1.22	2.17	2.62	3.72
	LANT	1.22	2.17	**2.64**	3.72
NDCG (%)	Random	0.77	1.04	1.16	1.63
	ParagraphVectors'20	6.48	5.88	5.20	6.83
	Metapath2Vec'21	6.49	5.88	5.23	6.85
	LANT	6.49	**5.89**	5.23	**6.86**

Table 5.1.7 shows the performance of our proposed model in Inductive IMDB. As in inductive settings, the baseline models fail to infer low dimensional vector representation for newly added nodes in IMDB network, therefore we only calculated LANT performance in this case.

Table 5.1.7: Performance of LANT vs baselines models on Inductive IMDB

Inductive Learning					
Metrics	Models	TOP@1	TOP@3	TOP@5	TOP@10
Recall (%)	Random	0.13	0.39	0.56	1.02
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**1.02**	**2.61**	**3.84**	**6.55**
MRR (%)	Random	0.51	0.98	1.16	1.45
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**5.34**	**8.41**	**9.60**	**10.38**
MAP (%)	Random	0.13	0.30	0.40	0.61
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**1.02**	**1.82**	**2.16**	**2.77**
MAP (%)	Random	0.51	0.63	0.64	0.90
	ParagraphVectors'20	NA	NA	NA	NA
	Metapath2Vec'21	NA	NA	NA	NA
	LANT	**5.34**	**4.82**	**4.46**	**5.77**

5.1.3.3 AUCROC - IMDB

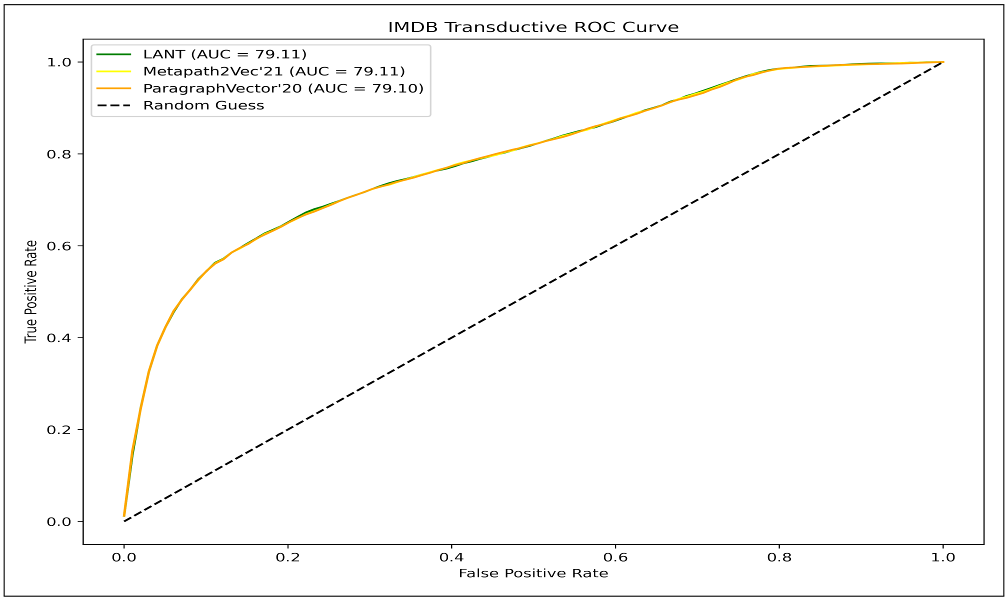


Figure 5.1.5: AUCROC - Transductive IMDB

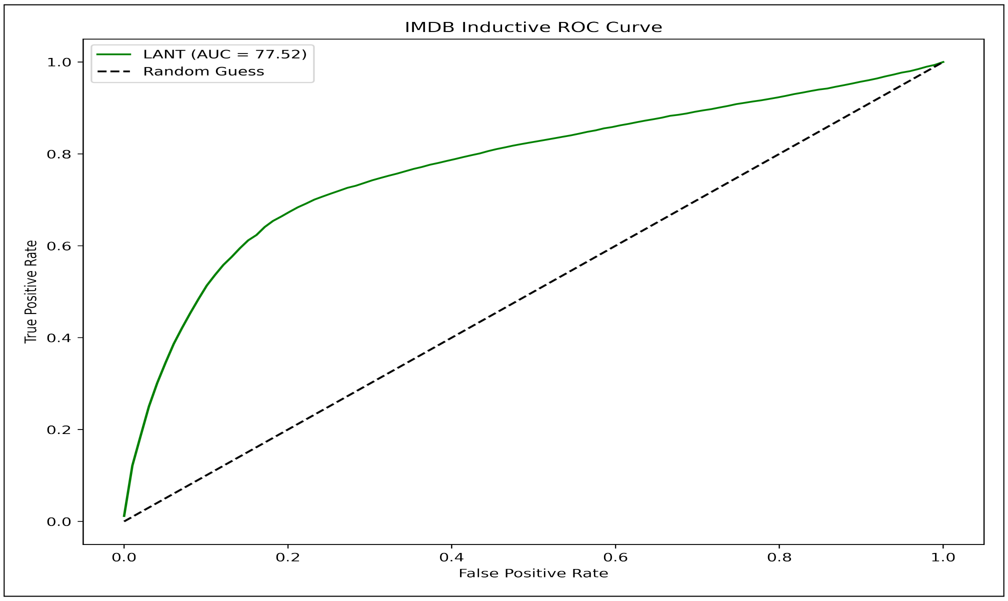


Figure 5.1.6: AUCROC - Inductive IMDB

5.1.3.4 Skill Association - IMDB

Table 5.1.8: Performance of LANT vs baselines

Models	Transductive Learning	Inductive Learning
Random	13.98%	14.45%
ParagraphVectors'20	64.27%	NA
Metapath2Vec'21	64.30%	NA
LANT	**64.32%**	**78.89%**

5.1.3.5 Skill Expertise Ratio - IMDB

Table 5.1.9: Performance of LANT vs baselines

Models	Transductive Learning	Inductive Learning
Random	13.86%	14.63%
ParagraphVectors'20	57.96%	NA
Metapath2Vec'21	58.03%	NA
LANT	58.03%	**65.27%**

5.1.3.6 Collaboration Score - IMDB

Table 5.1.10: Performance of LANT vs baselines

Models	Transductive Learning	Inductive Learning
Random	8.52%	9.02%
ParagraphVectors'20	32.80%	NA
Metapath2Vec'21	32.92%	NA
LANT	**32.95%**	**38.40%**

5.2 Consistency in Results

We ran the experiment 10 times to ensure the consistency of our proposed approach result. Based on the small values of standard deviation and variance of the different experiments, we can conclude that our algorithm can generate consistent results in all measures.

Chapter 6

Conclusions, Limitations and Future Works

6.1 Conclusions

In this research, we proposed a novel deep learning architecture LANT for team recommendation. We overcome the computational complexity of the previously proposed state-of-the-art models for team formation problems. This is a significant improvement as it allows for faster and more efficient team recommendations. Our proposed architecture is more robust. It can recommend teams in both transductive and inductive settings, which is important for real-world applications where data can be incomplete or missing. In transductive settings, the model is trained on a complete set of data. In inductive settings, the model trained in the transductive settings is used to draw inferences on the new set of data, which is not seen during training.

Overall, the proposed architecture demonstrated significant reductions in training time compared to MetaPath2Vec'21 and ParagraphVectors'20 in both the DBLP and IMDB datasets. Specifically, the proposed architecture achieved a 90% and 61% reduction in training time against MetaPath2Vec'21 and ParagraphVectors'20 in the DBLP dataset, and an 88.35% and 85.33% reduction in training time against MetaPath2Vec'21 and ParagraphVectors'20 in the IMDB dataset.

6.2 Limitations and Future Works

We observed that one of the primary limitations is the long-tailed distribution of experts, which results in lower prediction scores in the evaluation metrics for recommending experts. This issue is more commonly observed in real-world data for team formation, and it can lead to approximation errors that result in a poor estimate of the posterior distribution in VBNN. Additionally, the long-tailed distribution can lead to overfitting, where the model becomes too specialized for the training data and performs poorly on unseen data.

To mitigate the effects of the long-tailed distribution in VBNNs for team formation, several techniques can be employed. These techniques include negative sampling, regularization, weight decay, and early stopping. Additionally, more advanced techniques such as hierarchical Bayesian models and mixture density networks can be used to capture the complex relationships between experts and their skills and help improve the accuracy of our recommendations.

To improve the explainability of the model, we plan to experiment with various node features and incorporate saliency maps to analyze their impact on the recommendation metrics. Salience maps can help visualize the contribution of each input feature to the final recommendation score. Furthermore, model-agnostic techniques such as LIME (Local Interpretable Model-Agnostic Explanations) and SHAP (SHapley Additive exPlanations) can be explored to provide users with more interpretable explanations of the model's recommendations.

Another important direction for future research is evaluating the fairness of the team recommendation process. Techniques such as counterfactual fairness and group fairness can be employed to ensure that the recommendations are fair across different groups of users, such as different genders or races. We will also investigate the use of fairness metrics such as equal opportunity and demographic parity to evaluate the fairness of our model.

BIBLIOGRAPHY

[1] K. Selvarajah, P. M. Zadeh, Z. Kobti, Y. Palanichamy, and M. Kargar, “A unified framework for effective team formation in social networks,” *Expert Systems with Applications*, vol. 177, p. 114886, 2021.

[2] R. Hamidi Rad, E. Bagheri, M. Kargar, D. Srivastava, and J. Szlichta, “Retrieving skill-based teams from collaboration networks,” in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 2015–2019. [Online]. Available: https://doi.org/10.1145/3404835.3463105

[3] S. Wasserman and K. Faust, “Social network analysis: Methods and applications,” 1994.

[4] M. Marks, J. Mathieu, and S. Zaccaro, “A temporally based framework and taxonomy of team processes,” *The Academy of Management Review*, vol. 26, p. 356, 07 2001.

[5] E. L. Fitzpatrick and R. G. Askin, “Forming effective worker teams with multi-functional skill requirements,” *Comput. Ind. Eng.*, vol. 48, no. 3, p. 593–608, may 2005.

[6] A. Gajewar and A. D. Sarma, “Multi-skill collaborative teams based on densest subgraphs,” in *SDM*, 2011.

[7] O. Hlaoittinun, E. Bonjour, and M. Dulmet, “A multidisciplinary team building

method based on competency modelling in design project management," *International Journal of Management Science and Engineering Management*, vol. 3, pp. 163 – 175, 2008.

[8] H. M. E. Abdelsalam, *Multi-objective Team Forming Optimization for Integrated Product Development Projects.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 461–478. [Online]. Available: https://doi.org/10.1007/978-3-642-01085-9_15

[9] F. Ahmed, K. Deb, and A. Jindal, "Multi-objective optimization and decision making approaches to cricket team selection," *Appl. Soft Comput.*, vol. 13, pp. 402–414, 2013.

[10] C. Dorn and S. Dustdar, "Composing near-optimal expert teams: A trade-off between skills and connectivity," in *OTM Conferences*, 2010.

[11] J. Juárez and C. A. Brizuela, "A multi-objective formulation of the team formation problem in social networks: Preliminary results," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 261–268. [Online]. Available: https://doi.org/10.1145/3205455.3205634

[12] L. Zhang and X. Zhang, "Multi-objective team formation optimization for new product development," *Comput. Ind. Eng.*, vol. 64, pp. 804–811, 2013.

[13] R. Hamidi Rad, H. Fani, M. Kargar, J. Szlichta, and E. Bagheri, "Learning to form skill-based teams of experts," ser. CIKM '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 2049–2052. [Online]. Available: https://doi.org/10.1145/3340531.3412140

[14] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations.* Springer, 1972, pp. 85–103.

[15] T. Lappas, K. Liu, and E. Terzi, "Finding a team of experts in social networks,"

in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 467–476.

[16] J. Basiri, F. Taghiyareh, and A. Ghorbani, "Collaborative team formation using brain drain optimization: a practical and effective solution," *World Wide Web*, vol. 20, pp. 1385–1407, 2017.

[17] W. Chen, J. Yang, and Y. Yu, "Analysis on communication cost and team performance in team formation problem," in *Collaborative Computing: Networking, Applications and Worksharing: 13th International Conference, CollaborateCom 2017, Edinburgh, UK, December 11–13, 2017, Proceedings 13*. Springer, 2018, pp. 435–443.

[18] B. Ashenagar, N. F. Eghlidi, A. Afshar, and A. Hamzeh, "Team formation in social networks based on local distance metric," in *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*. IEEE, 2015, pp. 946–952.

[19] Y. Han, Y. Wan, L. Chen, G. Xu, and J. Wu, "Exploiting geographical location for team formation in social coding sites," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2017, pp. 499–510.

[20] R. Hamidi Rad, E. Bagheri, M. Kargar, D. Srivastava, and J. Szlichta, "Subgraph representation learning for team mining," in *14th ACM Web Science Conference 2022*, ser. WebSci '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 148–153. [Online]. Available: https://doi.org/10.1145/3501247.3531578

[21] A. Dashti, S. Samet, and H. Fani, "Effective neural team formation via negative samples," ser. CIKM '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 3908–3912. [Online]. Available: https://doi.org/10.1145/3511808.3557590

[22] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[23] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio', and Y. Bengio, "Graph attention networks," *ArXiv*, vol. abs/1710.10903, 2017.

[24] Y. Dong, N. V. Chawla, and A. Swami, "Metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 135–144. [Online]. Available: https://doi.org/10.1145/3097983.3098036

[25] C.-T. Li and M.-K. Shan, "Team formation for generalized tasks in expertise social networks," in *2010 IEEE second international conference on social computing*. IEEE, 2010, pp. 9–16.

[26] C.-T. Li, M.-K. Shan, and S.-D. Lin, "On team formation with expertise query in collaborative social networks," *Knowledge and Information Systems*, vol. 42, no. 2, pp. 441–463, 2015.

[27] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi, "Online team formation in social networks," in *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 839–848.

[28] A. Majumder, S. Datta, and K. Naidu, "Capacitated team formation problem on social networks," in *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, 2012, pp. 1005–1013.

[29] K. Selvarajah, P. M. Zadeh, M. Kargar, and Z. Kobti, "Identifying a team of experts in social networks using a cultural algorithm," *Procedia Computer Science*, vol. 151, pp. 477–484, 2019, the 10th International Conference on Ambient Systems, Networks and Technologies (ANT 2019) / The 2nd International Conference on Emerging Data and

Industry 4.0 (EDI40 2019) / Affiliated Workshops. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050919305277

[30] M. Kargar and A. An, "Discovering top-k teams of experts with/without a leader in social networks," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 985–994.

[31] ——, "Teamexp: Top-k team formation in social networks," in *2011 IEEE 11th International Conference on Data Mining Workshops.* IEEE, 2011, pp. 1231–1234.

[32] M. Kargar, M. Zihayat, and A. An, "Finding affordable and collaborative teams from a network of experts," in *Proceedings of the 2013 SIAM international conference on data mining.* SIAM, 2013, pp. 587–595.

[33] C.-T. Li, M.-Y. Huang, and R. Yan, "Team formation with influence maximization for influential event organization on social networks," *World Wide Web*, vol. 21, pp. 939–959, 2018.

[34] K. Selvarajah, A. Bhullar, Z. Kobti, and M. Kargar, "Wscan-tfp: weighted scan clustering algorithm for team formation problem in social network," in *The Thirty-First International Flairs Conference*, 2018.

[35] K. Selvarajah, Z. Kobti, and M. Kargar, "A cultural algorithm for determining similarity values between users in recommender systems," in *Applications of Evolutionary Computation: 22nd International Conference, EvoApplications 2019, Held as Part of EvoStar 2019, Leipzig, Germany, April 24–26, 2019, Proceedings 22.* Springer, 2019, pp. 270–283.

[36] J. H. Gutiérrez, C. A. Astudillo, P. Ballesteros-Pérez, D. Mora-Melià, and A. Candia-Véjar, "The multiple team formation problem using sociometry," *Computers & Operations Research*, vol. 75, pp. 150–162, 2016.

[37] M. Kargar, A. An, and M. Zihayat, "Efficient bi-objective team formation in social networks," in *Machine Learning and Knowledge Discovery in Databases:*

European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II 23. Springer, 2012, pp. 483–498.

[38] K. Selvarajah, P. M. Zadeh, Z. Kobti, M. Kargar, M. T. Ishraque, and K. Pfaff, "Team formation in community-based palliative care," in *2018 Innovations in Intelligent Systems and Applications (INISTA).* IEEE, 2018, pp. 1–7.

[39] L. Chen, Y. Ye, A. Zheng, F. Xie, Z. Zheng, and M. R. Lyu, "Incorporating geographical location for team formation in social coding sites," *World Wide Web*, vol. 23, no. 1, pp. 153–174, 2020.

[40] H. Zhu, E. Chen, H. Xiong, H. Cao, and J. Tian, "Ranking user authority with relevant knowledge categories for expert finding," *World Wide Web*, vol. 17, pp. 1081–1107, 2014.

[41] M. Neshati, S. H. Hashemi, and H. Beigy, "Expertise finding in bibliographic network: Topic dominance learning approach," *IEEE transactions on cybernetics*, vol. 44, no. 12, pp. 2646–2657, 2014.

[42] F. Farhadi, M. Sorkhi, S. Hashemi, and A. Hamzeh, "An effective expert team formation in social networks based on skill grading," in *2011 IEEE 11th International Conference on Data Mining Workshops.* IEEE, 2011, pp. 366–372.

[43] A. Sapienza, P. Goyal, and E. Ferrara, "Deep neural networks for optimal team composition," *Frontiers in big Data*, vol. 2, p. 14, 2019.

[44] Y. Sun and J. Han, "Mining heterogeneous information networks: principles and methodologies," *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 3, no. 2, pp. 1–159, 2012.

[45] F. Schliski, J. Schlötterer, and M. Granitzer, "Influence of random walk parametrization on graph embeddings," in *European Conference on Information Retrieval.* Springer, 2020, pp. 58–65.

[46] R. H. Rad, S. Seyedsalehi, M. Kargar, M. Zihayat, and E. Bagheri, "A neural approach to forming coherent teams in collaboration networks." in *EDBT*, 2022, pp. 2–440.

[47] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *IEEE Data Eng. Bull.*, vol. 40, no. 3, pp. 52–74, 2017. [Online]. Available: http://sites.computer.org/debull/A17sept/p52.pdf

[48] Q. Lv, M. Ding, Q. Liu, Y. Chen, W. Feng, S. He, C. Zhou, J. Jiang, Y. Dong, and J. Tang, "Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks," ser. KDD '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 1150–1160. [Online]. Available: https://doi-org.ledproxy2.uwindsor.ca/10.1145/3447548.3467350

[49] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ArXiv*, vol. abs/1609.02907, 2016.

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

[51] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*, 2018.

[52] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," *ICLR (Poster)*, vol. 2, no. 3, p. 4, 2019.

[53] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," *arXiv preprint arXiv:1808.06670*, 2018.

[54] C. Data61, "Stellargraph machine learning library," https://github.com/stellargraph/stellargraph, 2018.

[55] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning.* PMLR, 2014, pp. 1188–1196.

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

[57] J. Zhang, P. S. Yu, and Y. Lv, “Enterprise employee training via project team formation,” in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017, pp. 3–12.

[58] A. Bozzon, M. Brambilla, S. Ceri, M. Silvestri, and G. Vesci, “Choosing the right crowd: expert finding in social networks,” in *Proceedings of the 16th international conference on extending database technology*, 2013, pp. 637–648.

[59] G. K. Awal and K. K. Bharadwaj, “Team formation in social networks based on collective intelligence–an evolutionary approach,” *Applied intelligence*, vol. 41, pp. 627–648, 2014.

[60] T. Largillier and J. Vassileva, “Using collective trust for group formation,” in *Collaboration and Technology: 18th International Conference, CRIWG 2012 Raesfeld, Germany, September 16-19, 2012 Proceedings 18*. Springer, 2012, pp. 137–144.

VITA AUCTORIS

NAME:	Sagar Kaw
PLACE OF BIRTH:	Himachal Pradesh, India
YEAR OF BIRTH:	1995
EDUCATION:	Masters in Computer Science
	University of Windsor, M.Sc in Computer Science, Windsor, Ontario, 2023