EXPLOITING NETWORK AND DYNAMICS FOR

TALENT MANAGEMENT

by

Mingfei Teng

A Dissertation submitted to the

Graduate School-Newark

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Management

written under the direction of

Dr. Hui Xiong

and approved by

Newark, New Jersey

October 2021
ABSTRACT OF THE DISSERTATION

EXPLOITING NETWORK AND DYNAMICS FOR TALENT MANAGEMENT

By MINGFEI TENG

Dissertation Director: Dr. Hui Xiong

Network and dynamics based analysis addresses various problems in talent management, emphasizing the connection, the influence and temporal factors in the working social network between different units (employees, departments, and companies). Nowadays, most contemporary commercial activities such as production, sales are organized by companies, which connect employees with various roles and levels into an intact network and involves broadly and deeply corporations; the environment composed of colleagues, departments, and companies inevitably generate positive or negative influences on the individuals, shaping their working behaviors, essentially impact the daily operations of the company; with the digitalization in industries, the abundant data of peoples’ working make it feasible to analyze the role of social connections & influences in working and provide an insight into the patterns in communications between individual units in the working social network, and facilitate to develop data-driven applications to assist decision-making process in the management.

Our main contribution is to incorporate the social influence & network connections to provide intelligent solutions for different tasks in management such as turnover prediction and job position hierarchy extraction. Specifically, we first propose a contagious effect heterogeneous neural network (CEHNN) for individual turnover predic-
tion. The network encodes the connections between employees based on the hierarchy of companies and formulates it as a sequential classification problem with the series of prior-connected-employee turnover events and other environment and profile factors. Further, we design a Turnover Influence-based Neural Network (TINN), which explicitly integrates the company’s internal social networks like email and instant messages, models the diffusion of the turnover influences on these networks, and fuses them to forecast the turnover rate of departments. In addition, at the job level, we show that based on the aggregated job-hopping records, a framework is built to learn the vector representations of job positions across many companies, which is further utilized to reconstruct the hierarchy of job positions, identifying the potential links between them and predicting their rankings. The above approaches are validated for extensive experiments on real-world datasets, demonstrating their effectiveness, and the case studies reveal some interpretable findings which provide actionable insight into intelligent management.
I want to express my deep appreciation to all the people who provided tremendous support and help during my Ph.D. study.

First, I would like to express my greatest gratitude to my advisor, Prof. Hui Xiong, for his continuous support, guidance, and encouragement. Professor Xiong has offered me precious opportunities to be a research assistant, guided me on my research work with invaluable suggestions, provided me a platform with great opportunities to communicate and collaborate with other colleagues, visiting scholars, and experts in data mining. More importantly, he has inspired me with his positive, optimistic, diligent attitude toward life.

I also sincerely thank my other committee members: Professor Jian Yang, Professor Spiros Papadimitriou, and Professor Xiao Fang. They all provide constructive suggestions and comments on my work and offer numerous support and help in my career choice, and I am very grateful for them.

Special thanks are due to my colleagues, visiting scholars, and friends in the Center of Data Mining and Business Analytics (CDMBA) in Rutgers. It is a great opportunity and pleasure to work with them. Thank you, Chuanren Liu, Jingyuan Yang, Bin Liu, Yanjie Fu, Meng Qu, Zijun Yao, Yanchi Liu, Junming Liu, Qingxin Meng, Farid Razzak, Xinjiang Lu, Leilei Sun, Huang Xu, Renjun Hu, etc.
I would also thank Hengshu Zhu and the Talent Intelligent Center (TIC) team of Baidu for providing me the great opportunity and support to research data mining in talent management.

I want to acknowledge the Department of Management Science and Information Systems (MSIS) to supply me with the best imaginable equipment and facilities that helped me accomplish much of this work.

Finally, I would like to thank my family, mother, and father for their love, support, and understanding. Without their encouragement and help, this thesis would be impossible.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ABSTRACT</th>
<th>ii</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
<tr>
<td>CHAPTER 1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Research Motivation</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Contributions</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Overview</td>
<td>6</td>
</tr>
<tr>
<td>CHAPTER 2. EXPLOITING CONTAGION EFFECT FOR TURNOVER PREDICTION</td>
<td>8</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Related Work</td>
<td>12</td>
</tr>
<tr>
<td>2.3 Data Description</td>
<td>14</td>
</tr>
<tr>
<td>2.4 Technical Details of CEHNN</td>
<td>18</td>
</tr>
<tr>
<td>2.4.1 Problem Formulation</td>
<td>18</td>
</tr>
<tr>
<td>2.4.2 Heterogeneous Neural Network</td>
<td>22</td>
</tr>
<tr>
<td>2.4.3 Global Attention Mechanism</td>
<td>23</td>
</tr>
<tr>
<td>2.5 Experiment</td>
<td>24</td>
</tr>
<tr>
<td>2.5.1 Experimental Setup</td>
<td>25</td>
</tr>
<tr>
<td>2.5.2 Baseline Methods and Evaluation Metrics</td>
<td>25</td>
</tr>
<tr>
<td>2.5.3 Performance Comparison</td>
<td>27</td>
</tr>
<tr>
<td>2.5.4 Robustness Analysis</td>
<td>28</td>
</tr>
<tr>
<td>2.5.5 Case Study</td>
<td>30</td>
</tr>
<tr>
<td>2.6 Conclusion</td>
<td>31</td>
</tr>
</tbody>
</table>
LIST OF TABLES

2.1 A toy example of our processed pairwise turnover dataset .................. 12
2.2 The list of input features ......................................................... 20
2.3 The statistics of experimental data ............................................. 23
2.4 The network configuration ....................................................... 25
2.5 The overall performance of different approaches for turnover prediction 26

3.1 A toy example of turnover & profile dataset .................................. 42
3.2 A toy example of network dataset .............................................. 42
3.3 The statistics of major features and labels ..................................... 43
3.4 95% confidence interval of symmetric Kullback-Leibler divergence of estimated distributions of organizational turnovers and the mean of neighborhood turnovers .............................................................. 43
3.5 The overall performance of different approaches for turnover prediction 57
3.6 The performances for normalized turnover rate ................................. 59

4.1 Example of career profile ........................................................ 73
4.2 Example of job salary ................................................................ 74
4.3 Statistics of career profile & salary data ....................................... 74
4.4 Examples of normalization result .................................................. 79
4.5 Statistics of normalization result .................................................. 79
4.6 Performances for job link prediction .............................................. 91
4.7 Performances for job ranking prediction ....................................... 93
# LIST OF FIGURES

2.1 An example of contagious effect for employee turnover, where each node is an employee (i.e., blue is normal, red is a turnover, and yellow is an influenced one), the link between nodes represents the connection (i.e., stronger connection has bolder line weight). ........................................ 11

2.2 The comparison of the sizes of turnover grouped by whether they are under prior turnover influence. ................................................................. 15

2.3 The density histogram of the mean of days between current turnover and prior turnover. ................................................................. 16

2.4 The density histogram of the relative levels of influential managers in our data. ................................................................. 17

2.5 The overview of framework CEHNN for turnover prediction. .......... 21

2.6 The robustness analysis of different methods. ................................ 29

2.7 The visualization of weights from attention. .................................... 32

3.1 Topological & temporal analysis of data. ........................................ 45

3.2 Overall framework of TINN-FNS. ........................................ 48

3.3 Overall framework of TINN-FTI. ........................................ 51

3.4 Topological & temporal analysis of data. ........................................ 60

3.5 The robustness analysis of TINN. ........................................ 61

3.6 Sample attention of TINN-FTI on test data. .................................... 64

4.1 Motivation of job position hierarchy extraction. .............................. 66

4.2 Framework for job position hierarchy extraction. .............................. 80

4.3 Robustness analysis. ................................................................. 95

4.4 Visualization of learned embeddings. ............................................ 97

4.5 Case study: internal connections within companies. .......................... 98

4.6 Case Study: interconnections of jobs from two companies. .............. 99
CHAPTER 1
INTRODUCTION

In the knowledge economy era, human capital becomes a more and more important asset of many organizations. The innovations, operations, productions, and sales rely heavily on the talents that constitute the organization. Therefore, managing, allocating, motivating, and engaging the talents to achieve the company’s goal has always been an important topic in talent management. Nowadays, with the popularization of digitalization in industries, abundant data of peoples’ working have been accumulated, based on which quantitative analysis is conducted, and some research work has been done to exploit data mining & AI techniques to address various problems in management. Different from traditional research in this area which relies on the questionnaire responses from a small set of samples, analyze with statistical tools for the hypothesis test, recent research is more based on a large amount of employees’ working behavior data, which extends the complexity and capability of the model to learn the patterns indiscernible or not revealed with traditional ways, make predictions. With these characteristics, data-driven research is more adaptable to complex problems and can provide actionable insight for management in the real world.

Among these research efforts, our focus is the data-driven network connection and dynamics-based analysis in talent management. The ”network connection” includes concrete ones like email communication, instant message communication, and also
abstract ones like hierarchy relationships, aggregated job hoppings, which builds connections between jobs and companies; meanwhile, the units in the networks are not limited to employees, can be extended to departments, job positions, and companies. The network and dynamics-based analysis is to characterize the influences of each unit, which are context and problem-dependent, to model its dynamic propagation through network connections, to predict the future behavior of the units under the influences. For example, for the individual turnover prediction problem, it is interested in whether colleagues’ or supervisors’ quit decisions will affect the target employee to quit his/her job in the future. The network-based analysis will not treat each unit independently with its own attributes; instead, it assumes that all the units form an interactive environment. Therefore, the behavior of a unit is the consequence of its own characteristics and the impact of influential units around it. This is more close to the nature of the problem in management in the real world. Meanwhile, the analysis becomes more valuable when the data of unit-specific attributes are missing due to privacy protection, but the data of interaction of units are revealed. As a result, the network and dynamics-based analysis builds bridges between the connections, the influences, and the behaviors, which can be further exploited for prediction tasks in talent management.

1.1 Research Motivation

In this dissertation, we aim to address unique challenges of network and dynamics-based analysis in talent management. The first challenge roots in the complexity of the network influence. Specifically, it is impacted by many factors such as the intrinsic
character of the unit, the connection type and strength between units, etc; meanwhile, it is also time-variant due to its broadcasting and decaying mechanisms in the diffusion process; further, it may produce a cumulative effect in the observed and interesting time window. Taking turnover influence as an example, in the literature, most prior research work (March & Simon, 1958; Mobley, Horner, & Hollingsworth, 1978; Jackofskey & Peters, 1983; Mcevoy & Cascio, 1987; Trevor, Gerhart, & Boudreau, 1997; H. Li, Ge, Zhu, Xiong, & Zhao, 2017) focus on the analysis of the relationship between units’ own attributes and their behaviors, a few of them (Felps et al., 2009) have identified the impact of the influence of other units, but still, it lacks efforts to model the influence and exploit it for behavior prediction formally. Next, since there exist multiple types of networks (email, IM and hierarchical relationships, and so on), each of them may impose a unique impact on the propagation of the influence, it is non-trivial to investigate how to fuse the influence diffused through different network connections, how to evaluate and compare the impact of these networks, further how to deal with units with similar behaviors but not revealed in the observed network connections, which is commonly seen in social network data due to the data scarcity and sparsity. Prior connection or network related literature (Y. Li, Yu, Shahabi, & Yan, 2017; Cui, Henrickson, Ke, & Wang, 2018; Qiu, Jian, Hao, Dong, & Jie, 2018) focus on single type network connections, and few efforts have been identified for the fusion of influences from multiple networks and address the similarity in units behaviors not reflected in their connections. Last, network-based modeling often requires learning vector representations of the network structure, existing popular frameworks like DeepWalk (Perozzi, Al-Rfou, & Skiena, 2014), LINE (Tang et al.
2015), Node2Vec (Grover & Leskovec, 2016) more focus on the undirected structure which may downgrade their performances when applied to directed network; Meanwhile, the representations learned through unsupervised learning are not effective enough for supervised tasks. Learning effective representations that encode the directionality and high-order proximities and well-tuned for supervised tasks is to be investigated.

1.2 Contributions

Firstly, to address the complexities of network influence modeling and the relationship between influence and behavior, we build an influence-based framework for employee turnover behavior prediction. Specifically, for each employee, we construct a co-worker turnover events series ordered chronologically and collect the following features for each event: the relative level of the co-worker and target employee, their joint workdays, a year-based communication statistics, and the number of days between their relationships end and prediction time. These features characterize the connection between the co-worker and the target employee based on which the Long Short Term Memory (LSTM) is used to model the temporal dynamics of turnover influence and to predict the future turnover of the target employee. To robustize the framework, employee’s profile and environment factors like department monthly turnover statistics within one year are also included in the input, so the whole problem is formulated as a multi-sequence classification problem. Moreover, a global attention mechanism is designed to evaluate the heterogeneous impact on potential turnover behaviors, which also improves the interpretability of turnover prediction and provides
actionable insights for talent retention.

Secondly, to fully take advantage of the network structure formed by connections and deal with the influence on multiple networks, we propose a temporal networks fusion-based framework. Specifically, we integrate graph convolution network (GCN) and Long Short Term Memory (LSTM), the prior models the influence diffusion on each network, while the latter captures temporal dependency; further, we propose two approaches for multi-network fusion. One is static network structure fusion, which aggregates multiple networks based on the additive Laplacian matrices adjusted by static weights; the other is cross-sequence attention-based, which aggregates the hidden state on each network with dynamically calculated attention weights. It gives the framework more flexibility at the cost of more parameters to be trained. As stated in the last section, due to data scarcity and sparsity in social networks, it is commonly seen that units with similar behaviors but are not revealed in the observed connections. As a solution, we propose to construct a similarity network. The weight of connection is defined as the cross-correlation of historical observations of two units (nodes) and integrates the similarity as a supplement network in our framework.

Last, to learn more effective vector representations of directed network structure, we propose a learning framework that is able to capture both the directionality and the high-order neighborhoods. Specifically, for each node, two embeddings are to be learned, out-flow embedding & in-flow embedding, the prior characterize the nodes as the start of connections, while the other functions in an opposite manner. We design a cross-entropy-like objective function based on the two embeddings. Moreover, the objective function not only depends on the direct connection but also forces to learn
multi-step connections, allowing the embedding to encode high-order neighborhoods. Meanwhile, we integrate a supervised node ranking task with unsupervised learning to further tune the representations. We show the integration can help to train embeddings of nodes which is more effective in identifying potential connections and estimating the rankings of the nodes. As an example, we show the application on a job position graph, the connection is based on peoples’ job-hopping records, and the ranking is based on the average salary data. With the framework, the hierarchical structure of job positions is extracted, which is essential for many tasks in human resource management, such as person-job fit, salary benchmarking, and organizational structure optimization.

1.3 Overview

Chapter 2 address the complexities of network influence modeling and the relationship between influence and turnover behavior. Firstly, we discuss the motivation of the problem, introduce and analyze the data, and investigate the influenced turnover regarding roles of employees, relative levels, and departments. Next, we formulate the problem as a sequential classification problem and propose a contagious effect heterogeneous neural network (CEHNN) for turnover prediction by integrating the employee profiles, the environmental factors, and, more importantly, the influence of turnover behaviors of co-workers.

Chapter 3 presents a networks fusion based framework for the organizational turnover prediction. First, we show it is essential to take advantage of network structure for turnover influence propagation and dynamic prediction; then we analyze the
topological & temporal characteristics in the dataset; a Graph Convolution Network
(GCN) and Long and Short Term (LSTM) based framework is proposed, which shows
two ways to incorporate multiple networks: the fusion of network structure (FNS)
and the fusion of turnover influence (FTI) on different networks. Experimental results
with real-world data validate the effectiveness of the framework in terms of enhancing
turnover prediction. Also, our case studies reveal some interpretable findings, such
as the importance of each network or hidden state, which potentially impacts future
turnovers.

Chapter 4 presents an embedding learning framework to extract the job position
hierarchy of companies. Specifically, we use two types of embeddings, in-flow and
out-flow. We design a cross-entropy-like objective function based on the embeddings
and direct/indirect connections between nodes. The embeddings learned encode both
the directionality and high order proximity. Joint with a supervised regression task,
the embeddings are further tuned to encode the ranking information, which is helpful
to evaluate the relative levels of job positions. The extensive experiments on real-
world datasets validate the effectiveness and robustness and show that a joint learning
structure can benefit both tasks of the link and ranking prediction.
CHAPTER 2
EXPLOITING CONTAGION EFFECT FOR TURNOVER PREDICTION

In this chapter, we are aiming to address the network connection and influence-based modeling and analyze the relationship between influence and behavior. Specifically, we focus on the turnover influence of previous co-workers on the target employee and its effect, whether the target employees will quit their job in the future. We propose a contagious effect heterogeneous neural network (CEHNN) for turnover prediction by integrating the employee profiles, the environmental factors, and, more importantly, the influence of turnover behaviors of co-workers. Moreover, a global attention mechanism is designed to evaluate the heterogeneous impact on potential turnover behaviors. This attention mechanism can improve the interpretability of turnover prediction and provide actionable insights for talent retention. Finally, we conduct extensive experiments and case studies on a real-world dataset from a large company to validate the effectiveness of the contagious effect for turnover prediction.

2.1 Introduction

Talent turnover will affect business performance of companies. When an unexpected turnover request is raised in a company, significant effort is required to search for a replacement, and there is a risk that the operation of the company will be disrupted if a suitable replacement is not found. The situation becomes even worse due to
the contagious effect of external talent turnover when a group of employees influence each other and quit their jobs collectively (Felps et al., 2009). Indeed, talent plays an important role in daily business operation, not only because of its abilities and expertise, but also related to its collective influence on each other and the whole company. The quit of a talent often costs a large amount of business time, money, and performance, and the contagious effect of talent turnover will deconstruct the organizational structure and cause a dysfunction in the company. To alleviate the negative impact of talent turnover, it is critical for an employer to proactively predict potential turnovers, which in turn allows effective talent retention or successful talent replacement.

Previous approaches on turnover analysis focused on the ease and the desirability of job movement (March & Simon, 1958). Specifically, the ease of movement is related to factors such as job market availability, unemployment rate, and personal skill level. Factors related to desirability of movement include job satisfaction, salary growth, promotions, organization’s commitment, etc. A series of work (Mobley et al., 1978; Jackofsky & Peters, 1983; McEvoy & Cascio, 1987; Trevor et al., 1997) analyzed the relationships of talent turnover and these various factors. Later, different psychological paths that employees would follow when they quit their jobs were analyzed for modeling turnover behaviors (Lee & Mitchell, 1994; Lee, Mitchell, Wise, & Fireman, 1996; Lee, Mitchell, Holtom, McDaniel, & Hill, 1999). These analyses were mainly based on linear statistical models for testing theoretical hypothesis. Recently, survival analysis has been extended to predict the timing of turnovers with multiple sources of information (H. Li et al., 2017).
All the above research efforts focus on the variables associated with the turnover employees and their working environments, while the social influence among them, specifically the contagious influence of prior turnovers on the following ones, as shown in Figure 2.1, is rarely explored and exploited. The contagious influence effect, or simply contagious effect, has a complex nature in several ways:

- The contagious effect varies from people to people and is shaped by factors such as the positions in the organization and the connection strength between employees;

- The contagious effect is time variant due to its broadcasting and decaying mechanisms on the social network of employees in a company;

- A series of prior turnovers may produce a cumulative effect in the time window for prediction. This effect is even stronger for larger teams.

Given these challenges, it is a non-trivial endeavor to incorporate dynamic contagious effect with comprehensive factors for effective talent turnover prediction. Indeed, the profiling of employees and their working environments is both heterogeneous and dynamic in nature. Therefore, the modeling framework should be able to process multiple sources of sequential information with different length, granularity, and format to effectively predict turnover behaviors and support proactive talent management.

In addition to providing accurate turnover predictions, it is also valuable to investigate the importance of various factors in each turnover’s decision-making process. Quantifying and identifying important decision factors will reveal actionable insights such as motivations or reasons for the employees to quit their jobs. Such information
Figure 2.1. An example of contagious effect for employee turnover, where each node is an employee (i.e., blue is normal, red is a turnover, and yellow is an influenced one), the link between nodes represents the connection (i.e., stronger connection has bolder line weight).

is critical to facilitate employers in identifying unsteady points in their organizational structures and preventing talent turnovers for proactive talent retention and management.

To overcome the above difficulties, inspired by the application of RNN (recurrent neural network) on classification problem (Santos &Gattit, 2014) and the attention mechanism (Bahdanau, Cho, & Bengio, 2014; Yang et al., 2017), we build a contagious heterogeneous neural network (CEHNN) to integrate the peers’ turnover sequence, the environmental change and static profile information, and add a global attention across the multiple chains. The attention mechanism helps to evaluate the contribution of each change (event) in all the sequences, so we can identify the most influential factor for the final turnover decision.

Our contribution can be summarized as below:

- We formulate the turnover prediction from a new angle, developing a new framework CEHNN to capture the contagious effect in the sequential employee
<table>
<thead>
<tr>
<th>Employee</th>
<th>Co-Worker</th>
<th>Role</th>
<th>Level</th>
<th>Intimacy</th>
<th>Start Date</th>
<th>End Date</th>
<th>Co-Worker Leave Date</th>
<th>Leave Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>Bob</td>
<td>Manager</td>
<td>1</td>
<td>0.9</td>
<td>2017-01-01</td>
<td>2018-02-01</td>
<td>2018-02-05</td>
<td>2018-04-15</td>
</tr>
<tr>
<td>Alice</td>
<td>Carl</td>
<td>Peer</td>
<td>0</td>
<td>0.6</td>
<td>2017-04-10</td>
<td>2018-03-01</td>
<td>2018-03-10</td>
<td>2018-04-15</td>
</tr>
<tr>
<td>Alice</td>
<td>David</td>
<td>Manager</td>
<td>2</td>
<td>0.2</td>
<td>2017-02-03</td>
<td>2018-04-9</td>
<td>N/A</td>
<td>2018-04-15</td>
</tr>
<tr>
<td>Ellen</td>
<td>Frank</td>
<td>Peer</td>
<td>0</td>
<td>0.4</td>
<td>2017-07-05</td>
<td>2018-01-07</td>
<td>2018-04-01</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 2.1. A toy example of our processed pairwise turnover dataset.

- We use the framework CEHNN to process and integrate the sequential data from various sources with various formats, and design an attention mechanism across multiple sequences to evaluate the impact of different sequential factors on employee turnovers.

- We conduct extensive experiments on real-world data to validate the effectiveness of our framework and validate the contributing factors with case study.

2.2 Related Work

Recent years, there is a rising trend of applying advanced AI technologies to address talent related business problems (Zhu et al., 2018b; Qin et al., 2018; Xu, Yu, Yang, Xiong, & Zhu, 2018; Chen, Xiao, Sun, & Wu, 2017). Regarding the problem of turnover prediction, it is a hot topic that has been studied for years in human resource management. Previous research efforts generally fall into two categories. One focuses on analyzing the relationship of various factors and turnover, with tools of hypothesis test and linear models, while the other is trying to formulate the problem as survival analysis problems, and aims to predict the time to the occurrence of the turnover.

The early research of first category is mainly based on March and Simon’s work
Generally they proposed the turnover is determined by the desirability of movement and the ease of movement. The desirability of movement can be characterized by job satisfaction, salary growth, promotions, and organization’s commitment, while the ease of movement can be characterized by the job market availability, unemployment rate and personal skill levels etc. The contributions of these factors have been extensively studied in the following work. Mobley (Mobley et al., 1978) quantitatively analyzed the correlations between job satisfaction, age-tenure, intention to quit and turnover, and made predictions based on regression analysis. Glenn (Mcevoy & Cascio, 1987) found the turnover is lower among good performers, moderated by the turnover type, time span and level of unemployment, while Jackofsky (Jackofsky, Ferris, & Breckenridge, 1986) and Trevor (Trevor et al., 1997) identified a curvilinear relationship between job performance and turnover, stating the turnover is higher for low and high performers. Trevor also identified the moderating influence of salary growth and promotions on the curvilinearity. Despite of the analysis of the interested factors and turnover, there also exist some research (Lee & Mitchell, 1994; Lee et al., 1996, 1999) focusing on developing an unfolding model to describe and compare the psychological paths that employees take when they quit the jobs, the process of quitting from the initiation to the final decision is divided into different stages for discussion. One notable work is done by Felps, who studied the relation between job embeddedness and quitting, and mentioned there was a negative relationship between co-workers’ job embeddedness and focal employee turnover (Felps et al., 2009). It indicates a contagious effect when talent turnover happens.
For the research on turnover survival analysis model, some existing approaches can be directly applied like the classic Cox proportional hazards model, which (Cox, 1992) defined the hazard function of time and sample covariates. However it has a strong restriction that the time and covariates are independent and the weights of covariates are shared by all samples. In Li’s work (H. Li et al., 2017), he adopted and extended the multi-task framework of Yan Li (Y. Li, Wang, Ye, & Reddy, 2016), which treats the prediction of presence of employee at each time interval as a task, so the weights of the sample covariates varied from time to time.

2.3 Data Description

In this section, we explore the data to give a brief introduction, and more importantly to observe the contagious effect which inspires us of the way to formulate the research problem. The original dataset is provided by a high-tech company in China. It contains all of turnover records from 2016 to 2018, along with the profiles of employee. Both have been anonymized for privacy protection.

- **Profile Data**: The profile dataset includes the information such as anonymized employee ID, entry date, department, organization level and a year-based metric which characterizes the intimacy between each pair of employee in the company, generated according to the daily interactions.

- **Turnover Data**: This dataset includes the anonymized employee ID and leave date.

For our convenience, we integrated the profile data and turnover records in a pair-
wise way. Each integrated record would be a pair of co-workers (leader/subordinate or peers), for a period of time. Indirect managers and subordinates are also counted. For each record, the features include the anonymized employee’s ID and co-worker’s ID, the role of co-worker, their relative level, their relation start date and end date, and leave dates. Table 2.1 is an example dataset of the organized turnover records.

Figure 2.2. The comparison of the sizes of turnover grouped by whether they are under prior turnover influence.

Based on the intermediate pairwise turnover dataset, we conduct further exploration to see how many turnovers happens under the influence of peers’ or managers’ turnover. Figure 2.2 shows the comparison of the size of turnover grouped by whether they are under a manager’s or peer’s turnovers influence within one year. We can see about 91% turnovers are under the influence of prior turnover (about 2% from managers only, 27% from peers only and 62% from both). Further we analyze the
Figure 2.3. The density histogram of the mean of days between current turnover and prior turnover. The distribution of the mean of days between the pairwise turnovers. Figure 2.3 shows the distribution of the mean of days between employees’ turnover and their closest prior manager/peer’s turnover. We can see there is a decay effect, which indicates the influence would decrease with time as a trend. The interesting thing is in the density histogram related to peer, there is an apparent peak, indicating the influence of prior turnover may need to take certain amount of time to completely spread out and take effect, while in the density histogram related to manager, the distribution is almost decaying from the very beginning. Besides, in the group of turnover influenced by latest managers’ turnover, the distribution of relative level of managers are plotted in Figure 2.4 (a). It can be found that more employees’ turnovers are under the influence of their direct supervisors’ turnovers. The number decays as the relative levels arise.
Figure 2.4. The density histogram of the relative levels of influential managers in our data.
However at a finer granularity, this is not the case, in Figure 2.4 (b), we can see the shape of distributions varies by department, in some of them, the prior managers’ turnovers are followed by more subsequent turnovers in indirect subordinates than in direct ones.

2.4 Technical Details of CEHNN

In this section, we formulate a heterogeneous sequence classification problem, and propose the CEHNN (contagious effect heterogeneous neural network) framework as a solution, followed by discussion of the global attention mechanism in the heterogeneous neural network.

2.4.1 Problem Formulation

Our primary goal is to predict talent turnovers based on the sequential or time-variant information from different sources. Specifically, for the $i$-th sample, we want to make binary classifications $\hat{y}_i$ at a decision time $t_i$ based on static information $A_i$ and a collection of $M$ information series $\{s_{i1}^1, s_{i2}^2, \cdots, s_{iM}^M\}$. All the information $s_{im}^i$ are collected during an observation period $[t_i - \Delta t_{\text{observe}}, t_i]$, and each information series $s_{im}^i$ is a sequence $\{x_{i1}^m, x_{i2}^m, x_{i3}^m, \cdots\}$. Since theses sequences are from different sources, they may vary in terms of lengths and dimensions.

In the context of turnover prediction, $\hat{y}_i = 1$ if the $i$-th employee will quit his/her job in the future prediction period $[t_i, t_i + \Delta t_{\text{predict}}]$. Otherwise $\hat{y}_i = 0$. Here we focus on $M = 2$ sequential information sources together with static employee profiles. One sequential information source is the co-workers’ turnover events ordered by timestamp and denoted by $s_{i}^{\text{turnover}}$. The co-workers includes prior managers and peers, etc. The
other sequential information source is the dynamic environmental statistics of the employee and denoted by $s_{i}^{env}$. The profile of employees is treated as the static information in $A_i$. Our task is to estimate the probability $P(y_i|s_i^{\text{turnover}}, s_i^{env}, A_i)$, so we can make a prediction based on:

$$\hat{y}_i = \begin{cases} 
1, & \text{Pr}(y_i = 1|s_i^{\text{turnover}}, s_i^{env}, A_i) > \phi, \\
0, & \text{otherwise.}
\end{cases}$$

(2.1)

where $\phi$ is a discrimination threshold of employee turnover, and is set to 0.5 in our experiments.
<table>
<thead>
<tr>
<th>Category</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile Input A</td>
<td>Department</td>
</tr>
<tr>
<td></td>
<td>Organization level</td>
</tr>
<tr>
<td></td>
<td>Key Staff</td>
</tr>
<tr>
<td></td>
<td>Job Category</td>
</tr>
<tr>
<td>Sequence $S_{\text{turnover}}$</td>
<td>Relative level of employe $i$-$j$ pair</td>
</tr>
<tr>
<td></td>
<td>Days between the $i$-$j$ relation end and observation end</td>
</tr>
<tr>
<td></td>
<td>Prior turnover $j$’s profile</td>
</tr>
<tr>
<td></td>
<td>Relation type of employe $i$-$j$ pair (peer/manager)</td>
</tr>
<tr>
<td></td>
<td>Common work days of employe $i$-$j$ pair</td>
</tr>
<tr>
<td></td>
<td>Communication statistics of employe $i$-$j$ pair</td>
</tr>
<tr>
<td>Sequence $S_{\text{env}}$</td>
<td>Monthly total employee in the department</td>
</tr>
<tr>
<td></td>
<td>Monthly total turnover in the department</td>
</tr>
<tr>
<td></td>
<td>Monthly total employee at the level</td>
</tr>
<tr>
<td></td>
<td>Monthly total turnover at the level</td>
</tr>
<tr>
<td></td>
<td>Monthly turnover rate at the level</td>
</tr>
</tbody>
</table>

Table 2.2. The list of input features.
Figure 2.5. The overview of framework CEHNN for turnover prediction.
2.4.2 **Heterogeneous Neural Network**

As discussed above, the three input sources are varied in terms of their length and format, so each needs to be taken care of before integration for the prediction. As a whole, a heterogeneous neural network is designed to address this.

For the prior two which are sequential data, we choose the long short-term memory (LSTM) \(\text{Hochreiter & Schmidhuber 1997}\). As a variant of recurrent neural network, LSTM is powerful to process sequential data, with the ability to capture the long and short term dependency and overcome the exploding and vanishing gradients problems. A single LSTM cell is composed of an input gate \(i\), a forget gate \(f\), and an output gate \(o\), which can be formulated as:

\[
\begin{align*}
    f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f), \\
    i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i), \\
    o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o), \\
    c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_c x_t + U_c h_{t-1} + b_c), \\
    h_t &= o_t \odot \tanh(c_t),
\end{align*}
\]

where \(\{x_t\}_t\) is a series of input for the LSTM cell, \(W_f, W_i, W_o, W_c, U_f, U_i, U_o, U_c, b_f, b_i, b_o, b_c\) are parameters to be trained, \(c_t\) is the cell state, \(h_t\) is the output of the cell, \(\sigma\) is the sigmoid function and \(\odot\) is element-wise product.

We use two LSTM cells to process \(s^\text{turnover}_i\) and \(s^\text{env}_i\) respectively. Each element in \(s^\text{turnover}_i\) is a prior co-worker \(j\)'s turnover, including features of employee \(j\)'s profile and the intimacy of the pair \(i\)-\(j\)'s relation; each element in \(s^\text{env}\) is a monthly turnover statistics of specific department and organization level. Table 2.2 lists the detailed
features for each part. Dropout layers are appended to both LSTM cells to avoid the overfitting.

For the static profile information $A_i$, we use a fully connected layer to process it.

### 2.4.3 Global Attention Mechanism

As mentioned before, we aim to evaluate the contributions of different factors into the final decision. At this point, we decide to introduce the attention mechanism. Attention mechanism for RNN ([Bahdanau et al., 2014](#)) is proposed to be added into the sequence-to-sequence model, to achieve a word alignment effect, which reflects the importance of each word in source sentence for each word in the output sentence. By default, it is feasible to add one step attention in our framework, since we only have one step output, to each LSTM separately, however such implementation only helps to evaluate the importance of each event within a sequence. Alternatively, we decide to extend it to a global one across multiple sequences. Suppose for each sample we have $M$ series of output from $M$ LSTM cells, then the global attention can be

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total population</td>
<td>2,935</td>
</tr>
<tr>
<td>Positive samples</td>
<td>1,304</td>
</tr>
<tr>
<td>Negative samples</td>
<td>1,631</td>
</tr>
<tr>
<td>Dimension of Profile</td>
<td>400</td>
</tr>
<tr>
<td>Dimension of Turnover Sequence</td>
<td>419</td>
</tr>
<tr>
<td>Dimension of Environmental Sequence</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 2.3. The statistics of experimental data.
formulated as:

\[ u^m_t = \tanh(W^m h^m_t + b^m) \],
\[ \alpha^m_t = \frac{(\exp(u^m_t)\top u_c)}{\sum_{t \in \cup_m T_m} \exp(u^m_t)\top u_c}, \]
\[ v = \sum_{t \in \cup_m T_m} \alpha^m_t (W^m h^m_t + b^m), \]

where \( h^m_t \) is the output of LSTM cell for \( m \)-th sequence at step \( t \), \( u^m_t \) is the hidden representation of \( h^m_t \), \( \alpha^m_t \) is the normalized importance for \( t \)-th event in \( m \)-th sequence, \( v \) is the representation of the sequence as the aggregated weighted sum of hidden representation, \( T_m \) is the set of steps for \( m \)-th sequence, \( \{W^m\}_m, \{b^m\}_m, u_c \) are parameters to be estimated through training. Based on these, the conditional probability to be estimated become:

\[
Pr(y_i = 1|s^{\text{turnover}}_i, s^{\text{env}}_i, c_i) = \text{softmax}(Wv + b). \quad (2.2)
\]

Figure 2.5 shows the overall structure of our framework.

### 2.5 Experiment

In this section, we will evaluate the effectiveness of our framework. Specifically we trained and tested the framework CEHNN on a real-world dataset, comparing its performance with several state-of-the-art baselines. Meanwhile we will discuss the experiment result, as well as to analyze the attention weights generated through our framework by case study.
2.5.1 Experimental Setup

Data Pre-processing.

We set the observation time span $\Delta_{observe}$ for the sequence to be one year and the turnover prediction period $\Delta_{predict}$ to be three months. An employee who quits the job in the prediction period would be counted as a positive sample otherwise counted as a negative one. To isolate the influence in the prediction period, the decision time $t$ for each sample was chosen with constraint to make sure there was no prior manager/peer turnover in the prediction period. The statistics of the data is listed in Table 2.3. We used 64% for training, 16% for validation, 20% for test.

Model Configuration.

The dimension of our input and network configuration can be found in Table 2.4. We used the Adam optimizer (Kingma & Ba, 2014) for training. The objective function is the cross entropy loss function.

2.5.2 Baseline Methods and Evaluation Metrics

<table>
<thead>
<tr>
<th>Name</th>
<th>Value/Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension of $S^\text{turnover}$ LSTM</td>
<td>90</td>
</tr>
<tr>
<td>Dimension of $S^\text{env}$ LSTM</td>
<td>20</td>
</tr>
<tr>
<td>Dimension of Profile Dense Layer</td>
<td>20</td>
</tr>
<tr>
<td>Dimension of Dense Layer after Attention</td>
<td>35</td>
</tr>
<tr>
<td>Dropout Probability</td>
<td>0.5</td>
</tr>
<tr>
<td>Attention Size</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 2.4. The network configuration.
<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>AUC-ROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression</td>
<td>0.814±0.027</td>
<td>0.808±0.036</td>
<td>0.764±0.055</td>
<td>0.785±0.035</td>
<td>0.869±0.029</td>
</tr>
<tr>
<td>SVM</td>
<td>0.709±0.038</td>
<td>0.816±0.064</td>
<td>0.448±0.091</td>
<td>0.577±0.079</td>
<td>0.818±0.029</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.770±0.026</td>
<td>0.812±0.050</td>
<td>0.629±0.050</td>
<td>0.708±0.036</td>
<td>0.860±0.028</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>0.833±0.030</td>
<td>0.861±0.038</td>
<td>0.744±0.050</td>
<td>0.798±0.039</td>
<td>0.910±0.026</td>
</tr>
<tr>
<td>Turnover sequence HMM</td>
<td>0.752±0.101</td>
<td>0.862±0.229</td>
<td>0.568±0.190</td>
<td>0.671±0.046</td>
<td>0.550±0.088</td>
</tr>
<tr>
<td>Environmental change HMM</td>
<td>0.656±0.115</td>
<td>0.619±0.178</td>
<td>0.684±0.206</td>
<td>0.638±0.077</td>
<td>0.577±0.032</td>
</tr>
<tr>
<td>Turnover sequence RNN</td>
<td>0.853±0.022</td>
<td>0.856±0.025</td>
<td>0.806±0.044</td>
<td>0.830±0.028</td>
<td>0.905±0.025</td>
</tr>
<tr>
<td>Environmental change RNN</td>
<td>0.761±0.034</td>
<td>0.790±0.061</td>
<td>0.633±0.070</td>
<td>0.702±0.049</td>
<td>0.804±0.045</td>
</tr>
<tr>
<td>CEHNN</td>
<td><strong>0.864±0.018</strong></td>
<td><strong>0.871±0.029</strong></td>
<td><strong>0.816±0.049</strong></td>
<td><strong>0.842±0.024</strong></td>
<td><strong>0.914±0.020</strong></td>
</tr>
</tbody>
</table>

Table 2.5. The overall performance of different approaches for turnover prediction.
We compared our framework with baselines to demonstrate its effectiveness comprehensively. The baselines selected fell into three categories, a) popular classification algorithms without ability to process sequential data, such as Logistic Regression, SVM, Random Forest and Gradient Boosting; b) algorithms able to deal with sequential data natively such as HMM c) LSTM for turnover sequence only and environmental change only. For algorithms not designed for sequential data we concatenated and padded the sequential data in preprocessing. HMM is unable to process multiple sequences with varied length, so we trained two HMM models, one was fed with the turnover sequence data concatenated with profile data, the other was fed with the environmental sequential data concatenated with profile data. The hyper-parameters of all above models were found by grid search within a predefined range based on experience and suggested best practice.

Since the problem is a binary classification problem, we chose accuracy, precision, recall, F-measure and the area under the curve of receiver operating characteristic (AUC-ROC) to measure the performance of the framework.

2.5.3 Performance Comparison

The baselines and the framework CEHNN were evaluated on the turnover dataset, the experimental result is listed in Table 2.5. We conducted each algorithm 10 times, each time we generated a random training/validation/test dataset with the same proportion. The values listed in the table are the means with 95% confidence interval. We use bold font to emphasize the top 1 for each metric.

From the table, we have following observations: first, overall our framework per-
forms better than others on all metrics; second, the turnover sequence RNN also achieves a competent result (the runner-up in accuracy, recall and F-measure), which demonstrates exploiting the prior turnover sequence will benefit the future turnover prediction, in contrast, the environmental change plays a less important role; third, the high dimension of concatenated data limits the performance of classical classifiers, but Gradient Boosting still achieves a competent performance; lastly, HMM performs poorly on the dataset, we think the reason may be there is a long dependency which cannot be captured.

2.5.4 Robustness Analysis

To validate the robustness of our framework, we conducted three experiments: a) we tested it under the values of $\Delta_{predict}$ set to 30, 45, 60, 75, 90 days, as the prediction period is getting smaller, some positive (turnover) samples may turn to negative (non-turnover) ones; b) we tested it under the values of $\Delta_{observe}$ set to 3, 6, 9, 12 months; c) we tested it under the values of ratio = $\frac{\text{# of positive}}{\text{# of negative}}$ set to 0.5, 0.6, 0.7, 0.8, 0.9.

Figure 2.6 (a) compares the AUC-ROC values when we applied all the models to predict turnover for 30, 45, 60, 75, 90 days. In general, the Gradient Boosting, Random forest, Turnover Sequence RNN and CEHNN outperforms the others. The Random Forest and Gradient Boosting performs better when the prediction period is short, while the Turnover Sequence RNN and CEHNN are good at predicting for a longer period, which demonstrates they can exploit the sequential data better and make a longer prediction than traditional classifiers.

Figure 2.6 (b) compares the AUC-ROC values of all the models on the datasets
Figure 2.6. The robustness analysis of different methods.

(a) AUC under different prediction period

(b) AUC under different pos./neg. ratio
with different ratios of positive and negative samples. Similar to Figure 2.6 (a), the Gradient Boosting, the Turnover Sequence RNN and CEHNN outperform the others. The Gradient Boosting and the Turnover Sequence RNN is slightly better than CEHNN. We consider this is due to the size of the dataset. In our original dataset, the ratio of positive and negative samples is close to 0.8, when we adjust the ratio to be 0.5, 0.6, 0.7, we actually reduce the size of available data, which makes the framework CEHNN lack of training.

Figure 2.6 (c) compares the AUC-ROC values of all the models on dataset with different lengths of observation period. In general, it can be observed that as the observation period increases, the framework CEHNN performs consistently well, which demonstrates its ability to process and exploit long sequential data. In contrast, the performances of traditional classifier Gradient Boosting and Random Forest drop explicitly as observation period increases, for which the reason might be the increasing high dimension of concatenated data make the prediction more challengeable.

2.5.5 Case Study

We conducted two case studies with our framework on the dataset. One is on the individual level, the other is on the organizational level. On the individual level, a sample (employee) was chosen in the dataset and the global weights for it was learned by the CEHNN and plotted in Figure 2.7 (a). The weights are composed of three parts, the leftmost is weights of the environmental change, the middle is the weights of turnover sequence, and the rightmost one is the weights of profile. It is found for this sample, the largest weight located in the turnover sequence, so the framework evaluate
the 4th prior turnover as the largest contributing factor. On the organizational level, we evaluated the CEHNN on a test set of 587 samples, the weights are summed within environment change, turnover sequence and profile, as shown in Figure 2.7 (b). It is found that for the turnover prediction, the contributions from the prior turnover sequence, the environmental change and profile are approximately 61%, 30% and 9%.

2.6 Conclusion

We propose a contagious effect heterogeneous neural network (CEHNN) for turnover prediction within a period of future in this paper. The heterogeneous structure endows the framework the ability to process the sequential/non-sequential data from different sources together. Specifically it integrates the static profile information and the environmental change. Moreover, it exploits the contagious effect in the turnover sequences, formulating and solving the problem from a new angle. Further a global attention mechanism is implemented in the framework to detect a global importance of all involved factors, which gives more interpretability and actionable insight to our problem. Our experiment validate the effectiveness and robustness of the framework, and shows the value of global attention by case study.
Figure 2.7. The visualization of weights from attention.

(a) Weights for selected sample

(b) Boxplot of aggregated contribution
CHAPTER 3
EXPLOITING NETWORK FUSION FOR ORGANIZATIONAL TURNOVER PREDICTION

This chapter aims to exploit further the connection to model the diffusion of turnover influence and its effect. Primarily, we are focusing on the fusion of multiple networks and how to capture the behavior similarity of nodes when it is not reflected in the network connection. We develop a Turnover Influence-based Neural Network (TINN) for enhancing organizational turnover prediction. Specifically, a unique perspective of TINN is the construction of the turnover similarity network and fusing it with multiple organizational social networks either through learning a hidden turnover influence network or through integrating the turnover influence on various networks. Taking advantage of Graph Convolutional Network (GCN) and Long Short-Term Memory (LSTM) network, TINN can dynamically model the impact of social influence on talent turnover. Meanwhile, the utilization of the attention mechanism improves the interpretability, providing an insight of the impact of different networks along time on the future turnover change. Finally, we conduct extensive experiments in real-world settings to evaluate TINN. The results validate the effectiveness of the framework in terms of enhancing turnover prediction. Also, our case studies reveal some interpretable findings, such as the importance of each network or hidden state, which potentially impacts future turnovers.
3.1 Introduction

In the knowledge economy era, human capital becomes a more and more important asset of many organizations. Therefore talent turnover always has a huge business impact in terms of time, money and performance. Recently, the turnover prediction has become an emerging measure of proactive talent management, which is critically important for employers to retain and engage talents, and avoid the loss of intellectual capital.

Rather than using traditional linear statistical models with hypothesis tests (Mobley et al., 1978; Jackofsky & Peters, 1983; Mcevoy & Cascio, 1987; Trevor et al., 1997), people start exploiting advanced data mining technologies for enhancing turnover prediction (H. Li et al., 2017; Teng, Zhu, Liu, Zhu, & Xiong, 2019). While significant efforts have been made in this direction, it is not clear how to model the influence of employees’ turnover within the organizational social networks. Indeed, the situation of turnover prediction becomes even more complicated when the influence of employee turnover begins to spread out through the organizational social networks, which usually results in the so-called turnover contagion with unexpected consequences in management (Teng et al., 2019). Meanwhile, instead of forecasting the turnover of individual talent, a prediction at the organization or department level will provide a more macro vision of the operation status of a company, allowing the employer to monitor and adjust the strategies on talent management in a timely manner. There are three main challenges to be resolved: 1) how to fuse the turnover influence propagated on different organizational social networks (like email and In-
stant Message (IM)) since the way of message exchanging in different networks are highly related to the nature of the network itself and the closeness of nodes in it; 2) purely relying on the organizational social network will make the turnover forecasting inaccurate, since both the unobserved external network connection (like personal social media) and the disconnected nodes with similar turnover behavior cannot be thoroughly reflected by the organizational social network; 3) in addition to the fusion of multiple networks, it is also important to evaluate the potential impact from different networks at different time steps on the prediction of future turnovers, which will reveal more details of the decision process of the framework in an interpretable way. Therefore, it is a non-trivial task for researchers to study the impact of contagion effect with multiple networks on future organizational turnover.

To this end, in this work, we propose to explore the influence of employee turnovers within social networks and introduce a novel approach, namely Turnover Influence-based Neural Network (TINN), for enhancing organizational turnover prediction. Specifically, a unique perspective of TINN is that it constructs a turnover similarity network, and fuse it with multiple organizational social networks either through the fusion of network structures or through integrating the turnover influence from various networks. Taking advantage of Graph Convolutional Network (GCN) and Long Short-Term Memory (LSTM) network, TINN can dynamically model the impact of social influence on talent turnover. Meanwhile, the utilization of attention mechanism improves the interpretability, enabling us to evaluate the contribution of different networks in the spread of turnover social influence. Finally, we conduct extensive experiments in real-world settings to evaluate TINN. The results validate
the effectiveness of TINN in terms of enhancing turnover prediction. Also, our case studies reveal some interpretable findings, such as the importance of each network or hidden state which potentially impacts the future turnovers rate series.

Our main contributions can be summarized as follows:

• We propose a new research perspective of turnover prediction, through modeling the influence of employee turnovers in multiple social networks.

• We design a novel neural network-based approach TINN for enhancing turnover prediction, which incorporates both the topological and temporal characteristics of social influence by exploiting GCN and LSTM.

• We validate the effectiveness, study the interpretability and insights of our approach for turnover prediction, with extensive experiments conducted on a real-world dataset collected from a major high-tech company.

Overview. The remainder of this chapter is organized as follows. First, we review the literature on turnover prediction and their corresponding methods. Then we present the details, characteristics and analysis of the real-world talent data. Next, we formulate the problem and introduce the technical details of TINN. Further, we show the experimental results, and our case studies related to the interpretability of the framework. Finally we conclude the work in this chapter.

3.2 Related Work

In recent years, there is a rising trend of applying advanced AI technologies to address talent-related business problems [Zhu et al., 2018b] [Qin et al., 2020] [Xu, Yu, Yang].
et al., 2018; Chen et al., 2017; L. Zhang et al., 2020; Lin et al. 2020). Regarding the problem of turnover prediction, it is a hot topic that has been studied for years in human resource management. Our literature review covers both the work related to the topic of turnover analysis and the methodologies involved.

3.2.1 Turnover Analysis

Regarding the topic, we identified numerous literature focusing on analyzing the relationship between various factors and turnover. The early research is mainly based on March and Simon’s work (March & Simon, 1958). Generally, they proposed the turnover is dependent on the desirability of movement and the ease of movement. The desirability of movement can be characterized by job satisfaction, salary growth, promotions, and organization’s commitment, while the ease of movement can be characterized by the job market availability, unemployment rate, and personal skill levels. The contributions of these factors have been extensively studied in the following work. Mobley (Mobley et al., 1978) quantitatively analyzed the correlations between job satisfaction, age-tenure, intention to quit and turnover, and made predictions based on regression analysis. Glenn (Mcevoy & Cascio, 1987) found the turnover is lower among good performers, moderated by the turnover type, time span and level of unemployment, while Jackofsky (Jackofsky et al., 1986) and Trevor (Trevor et al., 1997) identified a curvilinear relationship between job performance and turnover, stating the turnover is higher for low and high performers. Trevor also identified the moderating influence of salary growth and promotions on the curvilinearity. Despite of the analysis of the interested factors and turnover, there also exist some research
& Mitchell 1994; Lee et al. 1996, 1999 focusing on developing an unfolding model to describe and compare the psychological paths that employees take when they quit the jobs, the process of quitting from the initiation to the final decision is divided into different stages for discussion. One notable work is done by Felps, who studied the relationship between job embeddedness and quitting, and mentioned there was a negative relationship between co-workers’ job embeddedness and focal employee turnover (Felps et al., 2009). It indicates a social influence when talent turnover happens. The above research efforts mainly focus on the individual level and are based on the profiling variables associated with the target employee and his/her working environments. Moreover, the connections of employees and the social influence of turnovers are rarely explored and utilized.

### 3.2.2 Methodologies

Regarding the methodologies, most above research apply tools of hypothesis test and linear models. In contrast, some researchers are trying to formulate the problem as survival analysis, and aim to predict the time to the occurrence of turnover. Existing approaches can be directly applied like the classic Cox proportional hazards model, which (Cox, 1992) defined the hazard function of time and sample covariates. However, it has a substantial restriction that the time and covariates are independent. In Li’s work (H. Li et al., 2017), he adopted and extended the multi-task framework of Yan Li (Y. Li et al., 2016), which treats the prediction of the presence of the employee at each time interval as a task, so the weights of the sample covariates varied from time to time.
Meanwhile, considering the periodical organizational turnover as a time series, many classical methodologies for sequential data, like LSTM, Gated Recurrent Unit (GRU), would fit into the scenario. Most recent neural network framework for sequence modeling are originally targeting the tasks in the natural language processing (NLP) domain, such as machine translation, named entity recognition and text summarization. Recently, some of these state-of-art models have also been applied to time series forecasting. Sequence to sequence learning (Sutskever, Vinyals, & Le, 2014), a encoder-decoder framework based on multi-layered LSTM, have been proposed for cryptocurrency price prediction (Rebane & Karlsson, 2018), weather prediction (Akram & El, 2016). Transformers (Vaswani et al., 2017), another superior benchmark for machine translation, which replaced recurrent layers commonly used in encoder-decoder framework with multi-head self-attention, have also been applied to forecast time series data (Song, Rajan, Thiagarajan, & Spanias, 2018; Wu, Green, Ben, & O’Banion, 2020).

All the above research efforts do not involve the social influence of turnovers through the network, to fill the research gap, we surveyed the literature related to graph convolution networks which generalize the classic Convolutional Neural Networks (CNN) to irregular topological structure, and are powerful to model the message propagation through the network. The initial work is done by Shuman (Shuman, Narang, Frossard, Ortega, & Vandergheynst, 2012). The authors defined a graph Fourier transform in terms of the eigenvectors of graph Laplacian and generalized the operators like filtering and convolution to graph setting. In (Defferrard, Bresson, & Vandergheynst, 2016), Defferrard proposed to approximate the smooth filters in the
spectral domain with a $K$-th order Chebyshev polynomials of the Laplacian which are $K$-localized, and the learning complexity is reduced to $K$. It is further simplified to a layer-wise model with one hop for each layer in (Kipf & Welling, 2016a). Due to the adaptivity of GCN to irregular shapes, it has mainly been applied to address problems in the social network and traffic domain. Qiu (Qiu et al., 2018) built a framework based on the graph attention and GCN for social influence prediction, while Li (Y. Li et al., 2017) and Cui (Cui et al., 2018) combine the recurrent neural network and GCN for a traffic flow forecasting. Meanwhile, few researches have been identified on graph convolution when multiple networks are involved. This is not a trivial problem since the influence in different networks will take various propagation paths; moreover, interactions may exist between networks, which could either strengthen or weaken the influence spread out and lead to divergent outcomes.

### 3.3 Data Exploration

In this section, we will introduce the data and explore its topological and temporal characteristics, which inspire us of the way to formulate and address the turnover prediction problem.

Our data is provided by a high-tech company in China. This company consists of many organizations, for which we want to predict future talent turnovers. The data contains essential statistics of employee turnovers, along with the profiles of the organizations and the network data spanning from 2016 to 2018. All the data are anonymized for privacy protection. Table 3.1 is an example of the profile and turnover dataset, Table 3.2 is an example of the network dataset, and Table 3.3 lists
all the three kinds of features, the labels (future turnover) and related statistics. To be noted, for the data alignment, all the features are collected in one month before the target month, for which the turnovers are predicted. Due to the privacy policies and regulations of the company, we are unable to provide detailed information and statistics related to job categories and levels, this is why some statistics in Table 3.3 is unavailable.

**Turnover Data:** The turnover data include monthly organizational turnover and summary statistics of star-employee attribute, job categories and hierarchical levels of the employees quitting their jobs in each organization.

**Profile Data:** The organization profile data include the monthly summary statistics of different job titles, star-employee attribute, job categories and hierarchical levels of all the employees in each organization.

**Network Data:** The communication data of different organizations are collected in two networks: the internal *email network* and the internal *Instant Message (IM) network*. As shown in table 3.2, the raw values in column # Email are the total number of emails between the specified two organizations in the target month, similar for # IM. These data characterize the closeness between organizations and are updated each month. Based on these raw values, at each time step $t$, for each network $i$ (email or IM), the adjacency matrix $A_t^i$ can be constructed, where the value in cell $(m, n)$ is the total frequency of communications between organization $m$ and $n$ in the network $i$. As preprocessing, we normalize the adjacency matrix $A_t^i$ through $D_t^{-\frac{1}{2}} A_t^i D_t^{-\frac{1}{2}}$ for future use, where $D_t$ is the degree
matrix. With no confusion, in the following content, we use $A^t_i$ to denote the normalized adjacency matrix.

<table>
<thead>
<tr>
<th>Month</th>
<th>Org. 1</th>
<th>Org. 2</th>
<th># Email</th>
<th># IM</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-01</td>
<td>A</td>
<td>B</td>
<td>70</td>
<td>200</td>
</tr>
<tr>
<td>2017-01</td>
<td>A</td>
<td>C</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td>2017-01</td>
<td>B</td>
<td>C</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### 3.3.1 Topological Analysis

From the topological perspective, in each network, we want to investigate if the distribution of the organizational turnover is associated with the neighborhoods. Thus, we define random variables like the turnover of organization, the mean of turnover of neighbor organizations, then we estimate the distributions of the random variables, and use the symmetric Kullback-Leibler (KL) divergences to measure the ”distance” between the distributions of these random variables.

Specifically, for each normalized adjacency matrix $A^t_i$, we select the quantile of $p$ percentile of all non-zero values of $A^t_i$ as a threshold and then convert $A^t_i$ into a
Table 3.3. The statistics of major features and labels.

<table>
<thead>
<tr>
<th>Category</th>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Features</td>
<td>Number of Organizations</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>Number of Observed Months</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>Number of Job Categories</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Number of Job Subcategories</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Number of Tracks</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Number of Level</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>Profile Features</td>
<td>Number of Unique Job Titles for Each Organization</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Percent of Star Employees in Each Organization</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Percent of Technical Star Employees in Each Organization</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Percent of Employees with Different Job Categories in Each Organization</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Percent of Employees in Different Levels</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>Turnover Features</td>
<td>Percent of Start Employees Quit in Each Organization</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Percent of Employees Quit with Different Job Categories in Each Organization</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Percent of Employees Quit in Different Levels</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>Label</td>
<td>Turnover</td>
<td>0</td>
</tr>
</tbody>
</table>

0-1 adjacency matrix based on the threshold. In practice, different values of \( p \) (0.05, 0.1, 0.2, ..., 0.5) are tested, all of them produced similar results, and we use \( p = 0.5 \) for illustration in the following. Once we have all the 0-1 adjacency matrices, the equivalent graphs can be constructed in which each node is an organization, if cell \((i, j)\) has value 1 in the adjacency matrix, then there is an edge connecting node \(i\)

Table 3.4. 95% confidence interval of symmetric Kullback-Leibler divergence of estimated distributions of organizational turnovers and the mean of neighborhood turnovers.

<table>
<thead>
<tr>
<th>Network</th>
<th>1st Neighborhood</th>
<th>2nd Neighborhood</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td>0.1662± 0.0140</td>
<td>0.7883± 0.2972</td>
</tr>
<tr>
<td>IM</td>
<td>0.1795± 0.0301</td>
<td>0.7682± 0.3581</td>
</tr>
</tbody>
</table>
and $j$ in the graph, otherwise they are not directly connected. In the graph, for each node (organization), we divide the nodes connected (directly or indirectly) to it into different groups according to the lengths of the shortest paths from these nodes to the target one. As illustrated by example in Figure 3.1(a), the first order neighborhood would be the nodes within two hops, and the second order neighborhood would be the nodes with more than two hops. Next, we compute the average turnover within each neighborhood for each node (organization). As a result, for each node (organization) at each time step, three values are collected: the turnover, the mean turnover of 1st-order-neighborhood, the mean turnover of 2nd-order-neighborhood, treated as observations for random variables. Then we use repeated sampling to estimate the distributions of the three random variables. Further, we calculate the symmetric KL divergences as a measure of the ”distance” between the estimated organizational turnover distribution and the estimated mean turnover distribution of neighborhoods.

The symmetric KL divergence, along with 95% confidence interval is listed in Table 3.4. We can see how close is the distribution of organizational turnover to that of the neighborhood mean turnovers. For all the organizational social networks, we found that the closer neighborhood has a smaller symmetric KL divergence, which means exploiting the turnover information of neighbors in the networks will benefit the turnover prediction for the target organization.

3.3.2 Temporal Analysis

To explore the temporal patterns in the data, we decompose the series of organizational turnover into the trend and seasonality, compute the strength proposed by
(Felps et al., 2009):

\[
\text{Trend Strength} = \max(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(R_t + T_t)}) \quad (3.1)
\]

\[
\text{Seasonality Strength} = \max(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(R_t + S_t)}) \quad (3.2)
\]

where $R_t$ is the residual series, $T_t$ is the trend series and $S_t$ is the seasonality series.

The decompositions are produced by a moving average over one year. Figure 3.1 is the box plot of trend strength and seasonality strength of all the organizations. It shows both the trend (with mean of 0.82) and seasonality (with mean of 0.68) are strong in the turnover series since the mean value is far from 0 and close to 1, which suggests taking the advantage of the temporal pattern will facilitate the turnover prediction.

3.4 Methodology

In this section, we first introduce the turnover similarity network and formulate the organizational turnover prediction problem, then we propose two approaches to in-
corporate multiple networks for turnover prediction: the fusion of network structure (FNS) and the fusion of turnover influence (FTI) on different networks. Both solutions take advantage of the graph convolution and LSTM, meanwhile, the differences are reflected in the fusion mechanism, the number of parameters, and the interpretability.

3.4.1 Turnover Similarity Network Construction

Unlike prior research on social connection prediction (Qiu et al., 2018), traffic prediction (Y. Li et al., 2017; Cui et al., 2018) in which scenarios the structure of the network is well defined and straightly observed, in the turnover prediction problem, there is not a clear definition of what a turnover influence network is. Even it is observed the turnover is related to the topological structure of social networks of email and IM in section 3.3.1, it is still uncertain how much the connection strength of two nodes in organizational social networks can reflect the similarity of their turnovers, and how close is the observable social networks (email and IM) to the ”true” turnover influence network. Besides, in reality, two organizations without social connections may still have similar turnover series. At this point, utilizing the observable social networks like email and IM may be inadequate. Therefore, the construction of a network directly reflecting the turnover similarity will fill the gap. Given $N$ organizations and their historical turnovers $\{y_i^t\}_{i,t}$. The adjacency matrix $A_{\text{similarity}}$ is defined as:

$$A_{i,j} = c(y_i^{1:T}, y_j^{1:T}) = \sum_{t=1}^{T} y_i^t y_j^t$$  \hspace{1cm} (3.3)$$

where $c$ is the cross-correlation at lag 0 which is a measure of similarity of two time series and $T$ is the observed period. In this turnover similarity network, organizations with similar turnover series are considered to be closer neighbors compared to others.
Similar to the email and IM networks, we assume the similarity network provides a third channel which allows the propagation of the turnover influence. Therefore we include the turnover similarity network as a supplement to existing observable networks (Email & IM) in our framework.

### 3.4.2 Problem Formulation

Our aim is to find the optimal estimator $f(\cdot)$ to predict the turnover $y$ of next period given historical organization profiles $X$ and network adjacency matrices $A$ formulated as:

$$y^{T+1} = f(X^{1:T}, y^{1:T}, A_{Email}^{1:T}, A_{IM}^{1:T}, A_{Similarity}, \Theta)$$

(3.4)

where $T$ is the observed period, $X^{1:T}$ is the series of input features, $A_{Email}^{1:T}$ is the series of normalized adjacency matrices of email network, $A_{IM}^{1:T}$ is the series of normalized adjacency matrices of IM network, and $A_{Similarity}$ is the normalized adjacency matrices of turnover similarity network.

### 3.4.3 Fusion of Network Structure (FNS)

In this fusion approach, we consider there exists a hidden network with adjacency matrix $A_{hidden}^{t}$, which directly determines the propagation of turnover influence, each existing network (email, IM, and turnover similarity network) only reveals a partial view of it. When the turnover influence is propagated, it will still utilize the link (edge) in the existing network; however, it will suffer an external impact which is named "preference". Organizations will have their preferences of choosing which network to spread out the turnover influence. It is based on the fact that people will choose different channels to distribute different messages. Such a choice will rely
on the message type, the message content, the connection (network) type, and the connection strength, etc.

The preferences for all the networks can be defined with different granularities. Specifically, they can be defined as global ones which are shared by all the organizations, or locally ones defined for each organization. Considering the size of our dataset, we will only proceed at the global level. Therefore, for each network, all the organizations share a static preference denoted by coefficient $\alpha$. It can be formulated as:

$$\hat{A}_i = \alpha_i A_i, \quad i = 1, 2, 3$$ (3.5)

$$A_{\text{hidden}} = \text{Agg}(\hat{A}_i)$$ (3.6)

where $i$ is the index of the network, $\alpha_i$ is a scalar preference for network $i$ to be trained, $\hat{A}_i$ is the adjacency matrix of observed network $i$ adjusted by the preference $\alpha_i$. Agg is the aggregation function which could either be average or max pooling.

Since the adjacency matrix $A_{\text{hidden}}$ will be used further in graph convolution, there are some constraints on $A_{\text{hidden}}$ such as non-negativity and symmetry, some may not
hold after being processed with above transformations. Some extra normalization operations are required, for example we apply the Relu or sigmoid activation functions to ensure the non-negativity, and apply $f(X) = (X + X^\top)/2$ to ensure the symmetry.

**Graph Convolution**

Previous turnover prediction rarely exploits the topological structure of networks for the turnover influence. To fill this research gap, inspired by the recent advance of the graph convolution network, we propose to integrate the GCN into our framework. We adopt the design from [Kipf & Welling, 2016a], feed a single-layer or multi-layer GCN with input features $X$ and the adjacency matrix $A_{\text{hidden}}$. Each layer of GCN can be formulated as:

$$\tilde{A} = A_{\text{hidden}} + I_N$$  \hspace{1cm} (3.7)

$$Z = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta)$$ \hspace{1cm} (3.8)

where $D \in R^{N \times N}$ is the degree matrix which is diagonal with elements $D_{ii} = \sum_j \tilde{A}_{ij}$, $X \in R^{N \times C}$ is the input from last layer ($C$ is the input dimension), $\Theta \in R^{C \times F}$ is the parameters to be trained ($F$ is the output dimension) and $\sigma$ is the activation function.

**Recurrent Layer**

The recurrent neural network has been largely applied to process sequential data and achieves state-of-art performance in many tasks such as translation, text classification, etc. Indeed, as a variant of RNN, LSTM is a robust model with the capability to capture the long term dependency and overcome the exploding and vanishing gradients problems. In our problem, both the profiles and the networks are dynamic,
changing by month, and as we see in section 3.3.2 the turnovers have strong seasonality and trend. Therefore LSTM would be a reasonable solution to capture the temporal pattern in the time series.

$$f_t = \sigma(W_f z_t + U_f h_{t-1} + b_f), \quad (3.9)$$

$$i_t = \sigma(W_i z_t + U_i h_{t-1} + b_i), \quad (3.10)$$

$$o_t = \sigma(W_o z_t + U_o h_{t-1} + b_o), \quad (3.11)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_c z_t + U_c h_{t-1} + b_c), \quad (3.12)$$

$$h_t = o_t \circ \tanh(c_t), \quad (3.13)$$

where \(\{z_t\}_t\) is a series of output of the graph convolution layers, \(W_*, U_*, b_*\) are parameters to be trained, \(c_t\) is the cell state, \(h_t\) is the output of the cell, \(\sigma\) is the sigmoid function and \(\circ\) is element-wise product. Figure 3.2 shows the overall structure of TINN-FNS.

**Fully Connected Layer**

The output of LSTM layer are further fed into the fully connected layer to compute the estimated future turnover:

$$y^{T+1} = Wh_T + b \quad (3.14)$$

where \(y^{T+1}\) is the estimated future turnovers at time \(T + 1\), \(W\) and \(b\) are parameters to be trained.

**3.4.4 Fusion of Turnover Influence (FTI)**

Different from the prior approach (FNS) which introduces a hidden turnover influence network, here our general idea is the turnover influence will be propagated directly
Figure 3.3. Overall framework of TINN-FTI.

through all the existing networks (email, IM, and turnover similarity network), and
further it will be aggregated to predict future. This framework is still composed of
graph convolution and LSTM, but the fusion does not happen prior to the graph
convolution (like in FNS); instead, it occurs after the turnover influence propagated
through different networks.

On top of this design, we add the attention mechanism\cite{Zhou2016} for
the output of multiple LSTM, which is formulated as below:

\[
M = \tanh(H) \quad (3.15)
\]

\[
\alpha = \text{softmax}(w^\top M) \quad (3.16)
\]

\[
r = H\alpha^\top \quad (3.17)
\]

where \(H\) is the concatenated output of every the LSTM layer for every existing
network, \(w\) is the parameters to train and \(r\) is the output of attention which will
further be fed to a fully connected layer for final prediction. The overall structure is
displayed in Figure\textsuperscript{3.3}

The attention brings two benefits into the framework: (1) the impact of turnover

influence from different network formulated with attention is dynamic and adequate to
capture complex temporal dependencies, unlike FNS in which the impact is defined as
preference and invariant to time; (2) it improves the interpretability of the framework,
revealing the importance of hidden states from different networks at different time
steps which the framework evaluates in the prediction process.

3.5 Experiment

In this section, we will evaluate the effectiveness of our framework. Specifically, we
trained and tested the framework on a real-world dataset, comparing its performance
with several state-of-the-art baselines. Meanwhile, we will discuss the experiment
result, as well as analyze the hidden network and attention weights generated through
our framework by the case study. To summarize, we intend to address the following
questions:

1. Is it beneficial to exploit the topological structure of the networks for turnover
   prediction?

2. Is it beneficial to utilize the network fusion approach for turnover prediction?

3. Is it beneficial to provide extra interpretability in addition to precision?

3.5.1 Experimental Setting

Data Pre-processing

We filtered out the organizations with few observations, which is more like outliers
compared to others with consistent observations. In total, there are 70 organizations
with observation period spanning 44 months, the features are constructed by month, for each month and each organization, it includes the profile (general statistics of the organization) and some statistics of turnover in this month, the labels are the turnovers of next month, as shown in Table 3.3. After selection, 20 features are kept and standardized by removing the mean and scaling to unit variance. To avoid information leaking, we split the data into training, validation, and test data according to the chronological order. Specifically, we used 70% for training (31 months), 10% for validation (4 months), and 20% for test (9 months). The turnover similarity network is constructed based on the turnover series in training data. The observation sliding window $T$ is 9 months.

Model Configuration

The loss function is composed of two parts, the mean squared error and the regularization for the parameters trained. We adopt the random searching (Bergstra & Bengio, 2012) for hyper-parameters optimization. Specifically, both the dimensions of graph convolution and LSTM output are drawn geometrically from 4 to 32; the number of graph convolution layers, LSTM layers and fully connected layers are drawn uniformly between 1 and 2. If there are 2 fully connected layers, the dimension of the first one is drawn geometrically from 2 to 16. For initializers, both the network preference parameters in FNS and the attention parameters in FTI use the uniform initializer on $[-0.05, 0.05]$; the graph convolution and LSTM use the Xavier normal initializer. For activation functions, the graph convolution layer and fully connected layer use ReLU while LSTM uses the tanh function. The Adam optimizer is applied, with the
learning rate set to 0.001. The identified optimal configuration for TINN-FNS is 2 graph convolution layers with output dimensions to be 12 and 6, 1 LSTM layer with output dimension to be 5, and 1 fully connected layer; for TINN-FTI the optimal configuration is 1 graph convolution layer with output dimension to be 11, 2 LSTM layers with output dimensions to be 23 and 8, and 1 fully connected layer.

3.5.2 Baseline Methods

To see the effectiveness of our framework, specifically the benefit exploitation of hidden network, the GCN and LSTM structure, we choose the following baselines to compare with.

- **Classic Regression.** It includes linear regression, support vector regression, random forest regression, and gradient boosting regression. Since these models are not adaptive to the sequential data, we concatenate the features and the row vector of the adjacency matrix of the networks in the observable period.

- **Sequential Regression.** As a state-of-art model for time series modeling and forecasting, the ARIMA, LSTM, Sequence to Sequence Learning (Seq2Seq) \cite{Sutskever2014} and transformers \cite{Vaswani2017} are selected as the baselines.

- **CEHNN.** We also include the framework CEHNN presented in last chapter, for which we aggregate the individual prediction into organization level and convert it to turnover rate.

- **GCN+LSTM.** To show the effectiveness of the network fusion in our frame-
work, we replace the fusion part with the adjacency matrices of the existing
organizational social networks (email, IM) and the turnover similarity network
respectively, which has a similar structure to Traffic Graph Convolutional LSTM
(TGC-LSTM) (Cui et al., 2018).

- **TINN on Directed Graph.** To check the impact of directed graph on our
  framework, the TINN-FNS and TINN-FTI on direct graphs are also selected
  as baselines. \( D_{out}^{-\frac{1}{2}} A D_{in}^{-\frac{1}{2}} \) is used as normalization for the raw adjacency matrix
  where \( D_{out} \) is the out-degree matrix and \( D_{in} \) is the in-degree matrix.

  The hyper-parameters of all the above models were found by grid search within a
  predefined range based on experience and suggested best practice.

### 3.5.3 Performance Comparison

For each algorithm, we repeat the test experiment 10 times. The mean of mean
absolute errors (MAE) are listed in Table 3.5 along with the 95% confidence interval.
We use bold font to emphasize the top 4 records.

From the table, we have the following observations: (a) all our approaches TINN-
FNS and TINN-FTI consistently outperform other methods; (b) the GCN (Similar-
ity)+LSTM performs significantly better than GCN+LSTM on organizational social
networks (Email, IM), which indicate the turnover similarity network is much more ef-
fective than the social networks; (c) the GCN (Similarity)+LSTM surpasses ARIMA,
LSTM, Seq2Seq and transformer which are better than GCN(Email)+LSTM and
GCN(IM)+LSTM, showing exploiting the topological structure of an appropriate
network will improve the sequential regression; (d) both TINN-FNS and TINN-FTI
outperforms the GCN(Similarity)+LSTM, which proves the effectiveness of fusion approaches; we consider the reason is, compared to the turnover similarity network which is static in the observation period, organizational social networks are dynamic, reflecting the variant nature of the turnover influence network, making them good supplements to the similarity one. As a whole, the fusion strategy allows all the networks to interact and complete each other and achieve the optimal result. (e) Both TINN-FNS and TINN-FTI on directed graph achieve similar performance to the ones on undirected graph, which shows the frameworks are robust to the directed graph on our dataset; (f) for the classic regressions, linear regression and SVR achieve the worst MAEs among all the approaches, in contrast, the ensemble learning methods like random forest and gradient boosting are still robust, even better than the state-of-art sequential regressors like LSTM and transformer; (g) for CEHNN, it only shows a performance close to sequential methods, since it is originally designed for individual prediction, the error for single prediction are aggregated when applied to the organizational scenario, and also the differences in the dataset may also downgrade its performance here.

Since the turnover rates range from 0 to 0.28 with a mean 0.015, which is close to 0, which may not put enough penalty on non-zero rates prediction, to better view and compare the performances of these baselines, we normalize the turnover rate by rescaling:

\[
y_{\text{normalized}} = \frac{y - \min y}{\max y - \min y},
\]

and repeat the experiment. The result is shown in Table 3.6. Similar to the results
Table 3.5. The overall performance of different approaches for turnover prediction.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Regression</td>
<td>0.0436</td>
<td>0.00550</td>
</tr>
<tr>
<td>SVR</td>
<td>0.0825</td>
<td>0.00709</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.0124 ± 0.0002433</td>
<td>0.0004635 ± 0.0000245</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>0.0126 ± 0.0000449</td>
<td>0.0004390 ± 0.0000035</td>
</tr>
<tr>
<td>ARIMA</td>
<td>0.0131</td>
<td>0.00051</td>
</tr>
<tr>
<td>LSTM</td>
<td>0.0130 ± 0.0008997</td>
<td>0.0004550 ± 0.0000350</td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>0.0123 ± 0.0006508</td>
<td>0.0004364 ± 0.0000265</td>
</tr>
<tr>
<td>Transformer</td>
<td>0.0134 ± 0.0018260</td>
<td>0.0005294 ± 0.0003044</td>
</tr>
<tr>
<td>CEHNN</td>
<td>0.0132 ± 0.0009421</td>
<td>0.0004680 ± 0.0000460</td>
</tr>
<tr>
<td>GCN (Email) +LSTM</td>
<td>0.0150 ± 0.0014565</td>
<td>0.0006981 ± 0.0002203</td>
</tr>
<tr>
<td>GCN (IM)+LSTM</td>
<td>0.0138 ± 0.0013138</td>
<td>0.0005254 ± 0.0001114</td>
</tr>
<tr>
<td>GCN (Similarity) +LSTM</td>
<td>0.0120 ± 0.0005140</td>
<td>0.0004196 ± 0.0000549</td>
</tr>
<tr>
<td>TINN-FNS (Undirected)</td>
<td>0.0116 ± 0.0001493</td>
<td>0.000412 ± 0.0000181</td>
</tr>
<tr>
<td>TINN-FNS (Directed)</td>
<td>0.0119 ± 0.0005384</td>
<td>0.000417 ± 0.0000181</td>
</tr>
<tr>
<td>TINN-FTI (Undirected)</td>
<td>0.0118 ± 0.0003529</td>
<td>0.000402 ± 0.0000271</td>
</tr>
<tr>
<td>TINN-FTI (Directed)</td>
<td>0.0118 ± 0.0002353</td>
<td>0.000408 ± 0.0000226</td>
</tr>
</tbody>
</table>
displayed in Table 3.5, our framework TINN-FNS and TINN-FTI outperforms other baselines, validating their effectiveness on the turnover prediction task.

### 3.5.4 Cost Analysis

In this section, we are trying to analyze both the time and space costs of our proposed framework. Specifically, we construct TINN-FNS and TIFF-FTI with 1 layer graph convolution, 1 layer LSTM and collecting the total number of parameters, which is proportional to the memory cost, and the total training time (in seconds) under different parameters like the units of graph convolution and LSTM layers.

Figure 3.4(a) and Figure 3.4(c) shows the number of parameters and training time of TINN-FNS and TIFF-FTI with the LSTM units set to 16 and the graph convolution units varying from 8 to 128. Figure 3.4(b) and Figure 3.4(d) shows the number of parameters and training time of TINN-FNS and TIFF-FTI with the graph convolution units set to 16 and the LSTM units varying from 8 to 128. Due to the graph convolution and LSTM are defined for each network respectively in TINN-FTI, plus the parameters for attention, TINN-FTI have far more parameters than TINN-FNS, and the difference is increasing as the numbers of GCN & LSTM units grow. As a result, TINN-FTI with large numbers of GCN & LSTM units requires a large amount of training data otherwise the model will lack training. Regarding the time cost, the increase in training time of TINN-FNS is not that significant as we increase the units of GCN & LSTM. Similarly a slight time increase can be seen for TINN-FTI when the number of GCN units grows from 8 to 128. In contrast, the training time increases more significantly for TINN-FTI when the number of LSTM units grows
Table 3.6. The performances for normalized turnover rate.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>MSE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Regression</td>
<td>0.1698</td>
<td>0.0761</td>
<td>0.2759</td>
</tr>
<tr>
<td>SVR</td>
<td>0.0737</td>
<td>0.0086</td>
<td>0.0931</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.0437±0.000388</td>
<td>0.0058±0.000104</td>
<td>0.0759±0.000687</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>0.0443±0.000038</td>
<td>0.0054±0.000008</td>
<td>0.0738±0.000053</td>
</tr>
<tr>
<td>ARIMA</td>
<td>0.0470</td>
<td>0.0062</td>
<td>0.0787</td>
</tr>
<tr>
<td>LSTM</td>
<td>0.0463±0.001141</td>
<td>0.0060±0.000202</td>
<td>0.0778±0.001306</td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>0.0457±0.000777</td>
<td>0.0059±0.000123</td>
<td>0.0768±0.000802</td>
</tr>
<tr>
<td>Transformer</td>
<td>0.0470±0.001823</td>
<td>0.0054±0.000169</td>
<td>0.0735±0.001143</td>
</tr>
<tr>
<td>CEHNN</td>
<td>0.0468±0.001210</td>
<td>0.0061±0.000317</td>
<td>0.0782±0.001412</td>
</tr>
<tr>
<td>GCN(Email) + LSTM</td>
<td>0.0476±0.000906</td>
<td>0.0062±0.000450</td>
<td>0.0790±0.002839</td>
</tr>
<tr>
<td>GCN(IM) + LSTM</td>
<td>0.0586±0.006770</td>
<td>0.0096±0.002524</td>
<td>0.0978±0.012294</td>
</tr>
<tr>
<td>GCN(Similarity) + LSTM</td>
<td>0.0436±0.000992</td>
<td>0.0056±0.000699</td>
<td>0.0747±0.004629</td>
</tr>
<tr>
<td>TINN-FNS (Undirected)</td>
<td>0.0426±0.000634</td>
<td>0.0051±0.000229</td>
<td>0.0713±0.001575</td>
</tr>
<tr>
<td>TINN-FNS (Directed)</td>
<td>0.0427±0.000446</td>
<td>0.0051±0.000141</td>
<td>0.0715±0.000982</td>
</tr>
<tr>
<td>TINN-FTI (Undirected)</td>
<td>0.0429±0.000682</td>
<td>0.0053±0.000241</td>
<td>0.0725±0.001213</td>
</tr>
<tr>
<td>TINN-FTI (Directed)</td>
<td>0.0428±0.000417</td>
<td>0.0052±0.000133</td>
<td>0.0722±0.000820</td>
</tr>
</tbody>
</table>
since this also increases the number of attention parameters significantly which need more time to be trained.

3.5.5 Robustness Analysis

In this section, to evaluate the robustness of our framework, we conducted two experiments: a) we test it with observation window ranging from 3 to 9; b) we test it with the portion of training data ranging from 50% to 70%, 10% for validation and the rest for test.

Figure 3.5 (a) shows the MAE(s) of our approaches for different observation window. Overall, both TINN-FNS and TINN-FTI are steady with low variances, and
the MAEs slightly decrease as the observation window being enlarged. The optimal observation window located at 8 months. Figure 3.5 (b) shows the MAE(s) of our approaches for different ratios of training data. Similar to (a), both TINN-FNS and TINN-FTI are steady. Further, it is found both TINN-FNS and TINN-FTI achieve the best performance when the portion of training data is around 55%.

3.5.6 Evaluation of Business Value

Overall, the turnover prediction is critically important for companies to retain and engage talents, and helpful to prevent the loss of intellectual. Specifically, the losses caused by turnover includes the cost of hiring like advertising, interviewing, the cost of onboarding like training and management, and the cost of productivity, since the new employee may take months or even years to reach the productivity of the existing employees. Meanwhile, there are also some negative impact that essentially affects the business performance but not easy to be quantified like lowering the morale of the team, increase the workload and pressures of other employees.
To evaluate the business value brought by the improvement introduced by our framework, we estimate the economic cost incurred by imprecise prediction, especially for the cases of underestimated turnover rate, in which some future turnovers are not reflected in the predicted turnover rate. Specifically we use the means of MAE in Table 3.3 as the prediction errors for the baselines, then $N_{\text{employee}} \cdot \text{MAE}$ can be considered the mispredicted number of turnover employees, where $N_{\text{employee}}$ is the number of employee for the whole company; next, based on our calculation, the portion of underestimation error in the total MAE is around 0.53 (with mean 0.53, std 0.06 for all baselines), which means the underpredicted number of turnover employee is $0.53 \cdot N_{\text{employee}} \cdot \text{MAE}$; according to prior related report (Boushey & Glynn, 2012) and articles (Agovino, 2019; Heinz, 2020), the typical cost for employee turnover ranges from 5.8% to 213% of the employee’s annual salary, here we use 21%, which is the median value, as a raw estimation of employee turnover cost, then the total cost of underprediction based on MAE is given by: $0.53 \cdot 0.21 \cdot N_{\text{employee}} \cdot \text{MAE} \cdot \text{Annual Salary}$. Suppose the company has 10 000 employees, the average annual salary for each employee is 180 000 Chinese Yuan, then every 0.001 of MAE leads to a cost of 200 340 Chinese Yuan. In fact, the actual cost will be larger for two reasons: (1) as we mentioned, there are also some unquantifiable negative impact of turnovers on business performance; (2) the total cost we calculated is only for one month, and it will be accumulated with time. To summarize, as presented in Section 3.5.3, our framework outperforms other baselines in the turnover rate prediction, based on the above analysis, the small improvement in MAE of turnover rate actually reflects significant savings on the cost incurred by undetected turnovers in the real world, which can grow even larger considering a big
company size and time cumulative effect.

3.5.7 Case Studies

In addition to the improvement of turnover prediction, we are also interested in what the framework can learn through training. Specifically, it’s interesting to see how the framework evaluates the contributions of different networks in an interpretable way.

After training TINN-FNS, we export the network preferences which are learned by the framework through training. It is found the preference of IM network is 0.06, the preference of email network is $-0.048$, and the preference of turnover similarity network is 0.22 which shows the similarity network has a much higher contribution than the rest organization social networks, and turnover influences are more likely to be spread through similarity and IM network than the email network.

For the TINN-FNS, the attentions for a sample calculated by Equation 3.16 are exported and plotted as a heatmap in Figure 3.6. Specifically, for this sample, there are 27 (3 networks × 9 time steps) hidden states in total. With the attention defined in Equation 3.16 the framework evaluates the contributions from the 27 hidden states to the future prediction. The higher the attention value is, the more contribution is from the related hidden state. In Figure 3.6 the color represents the value of attention, and according to the definition in Equation 3.16 all the attention values are non-negative and summed to 1. We can see the prediction is more dependent on the hidden states of similarity network since the attention values of similarity network are larger than the other two, which is consistent with the network preferences learned in TINN-FNS. Meanwhile the hidden states of similarity network closer to the target
prediction period also have more impact on the future turnover prediction. Therefore the attention introduced in TINN-FTI enables us to see how the framework evaluates the importance of hidden states for various networks at different time steps and makes a decision based on that.

3.6 Conclusion

In this paper, we studied the impact of social influence on employee turnover prediction and developed a Turnover Influence-based Neural Network (TINN) for enhancing organizational turnover prediction. Specifically, TINN constructs an effective similarity network and incorporates it with other organizational social networks through either the fusion of network structure (FNS) or the fusion of turnover influence (FTI) on networks. Combined with fusion strategy, TINN exploits graph convolution and long short-term memory (LSTM) for dynamically modeling the impact of social influence on talent turnover. Finally, extensive experiments on real-world data validated the effectiveness of TINN in terms of enhancing turnover prediction. Also, the case studies provided some interpretable findings, such as the importance of each network or hidden state which potentially impacts future turnovers.
CHAPTER 4

RANKING-GUIDED GRAPH EMBEDDING FOR EXTRACTING THE JOB
POSITION HIERARCHY

This chapter aims to address learning effective vector representations of nodes in a network structure. By “effective,” we expect the vectors learned can capture the directionality and the high order proximity, further, it can facilitate supervised learning tasks. As an example, we develop a holistic data-driven approach for job position hierarchy extraction. Specifically, we introduce a joint embedding and ranking framework to learn representations of job titles based on the heterogeneous graphs of job positions and the affiliated companies. Indeed, the representations can effectively capture the directionality of job transitions and the proximities of high-order neighborhoods. Also, being optimized with the ranking loss, the representations can encode the information of hierarchical position levels and thus helps identify the potential links between job titles and predict their rankings as the estimations of the hierarchical levels. Finally, extensive experiments on real-world datasets validate the effectiveness and robustness of our framework. The results show that a joint learning structure can benefit both tasks of the link and ranking prediction. Moreover, the visualization and case studies show the interpretability of the embeddings learned by our approach.
4.1 Introduction

In modern companies, a variety of job titles are used to designate the hierarchy, from executive management to low-ranking employees, within the job structure of an organization. Understanding the job position hierarchy is essential for many tasks in human resource management, such as person-job fit, salary benchmarking and organizational structure optimization. Indeed, this task relies on the extraction of relationships among jobs and their hierarchical levels from massive labor market data, which usually involves large human efforts and financial costs. Therefore, it is appealing to introduce a cost-efficient and automated solution for job position hierarchy extraction. In particular, Figure 4.1 shows a motivating example of job position hierarchy extraction in this work.

Figure 4.1. Motivation of job position hierarchy extraction.
Recently, based on abundant job transition data in the online professional social networks (e.g., LinkedIn), considerable research efforts (Zhu et al., 2018a; Xu, Yu, Guo, Teng, & Xiong, 2018; Qin et al., 2019; Sun et al., 2019; D. Zhang et al., 2019) have been made in relevant areas. Generally, the first thread is to focus on predicting the rankings of job titles (Xu, Yu, Guo, et al., 2018), while the other is to apply graph embedding methods to learn the potential relationships among job titles (D. Zhang et al., 2019). Indeed, both directions are critical and describe different perspectives of job position hierarchy extraction. Specifically, job embeddings provide a way to encode the local structure of the job graph purely constructed on the job transition data, while job ranking estimation enables us to see the levels of different jobs, and further to find whether the job transition is a promotion, demotion, or transfer. Moreover, we consider the job embedding learning and ranking are compliments to each other for job position hierarchy extraction. For example, if two job titles have a significant difference in ranking, like junior developer and chief technical officer, then it is unlikely to identify job transitions between them. On the other hand, if a large amount of bi-directional transitions of two job titles are observed, the two job positions may have similar job functions and hierarchy levels. This motivates us to achieve both tasks under a unified framework for job position hierarchy extraction.

However, there are still challenges to be addressed. First, there is no uniform naming convention and standardized hierarchical levels for similar jobs at different companies. This hinders the efforts of building the job position hierarchy in an accurate and efficient way. Second, many existing graph embedding methods (Perozzi et al., 2014; Tang et al., 2015; Grover & Leskovec, 2016) rely on the input data
which are paths generated through random walks, making it time-consuming for pre-processing and training; meanwhile, the embeddings learned are less effective for directed graphs since the in-flow and out-flow are usually not distinguished and combined into one. Last, in addition to the total number of transitions between jobs, other information like total time costs for job transitions and the connections between jobs and companies may also provide extra insight into the problem, how to integrate these features into the framework is becoming critical.

To this end, in this chapter, we aim to develop a holistic data-driven approach for job position hierarchy extraction. Specifically we introduces a joint embedding and ranking framework to learn representations of job titles based on the heterogeneous graphs of job positions and the corresponding companies transitions. Indeed, the representations can effectively capture the directionality of job transitions and the proximities of high-order neighborhoods. Also, being optimized with the ranking loss, the representations can encode the information of hierarchical job position levels, and thus helps identify the potential links between job titles and predict their rankings as the estimations of hierarchical levels. In particular, the proposed embedding method focuses on the encoding of the local structure of the graph, not relying on the paths generated from random walks, which is more time-efficient. Meanwhile, the decomposition of in and out embeddings and the extension of log-likelihood objective enables the embeddings to effectively capture both the directionality and high order proximities. Finally, extensive experiments on real-world datasets validate the effectiveness and robustness of our framework. The results show that a joint learning structure can bring benefits for both tasks of the link and ranking prediction. Moreover, the
visualization and case studies show the interpretability of the embeddings learned by our approach.

To be noted, even though we propose the framework as a solution to a specific essential problem in talent management, there are many similar scenarios in reality in which we need to build the hierarchical structure of entities and estimate the rankings of them, such as rankings on web page network, rankings on a set of products that are connected based on the taxonomy.

The remainder of this chapter is organized as follows. In section 4.2 we discuss the related work of this paper; in section 4.3 we introduce some preliminaries and formulate our research problem; and in section 4.4 we present the whole framework, including the normalization, the embedding, and the ranking modules. At last, we show the experimental results and conclude the paper in section 4.7.

4.2 Related Work

In this section, we review previous work related to this topic and the methodology. Aiming to extract job position hierarchy, we are mainly focusing on two sub-tasks on the job graph: link prediction and ranking prediction. Link prediction has been studied in many domains, such as online social networks, bioinformatics, and transportation network prediction, etc. Existing methodologies of link prediction fall into the following categories: (1) calculating similarity metrics and predicting the link with a threshold of the similarity metrics, many indexes have been proposed such as Common Neighbors (CN), Jaccard Index(JC) (Jaccard, 1901), Adamic-Adar Index (AA) (Adamic & Adar, 2003), etc; (2) learning low dimensional representations of
graph nodes which encode the topological structure or node features and can be used to reconstruct the graph. Typical examples include matrix factorization, like Singular Value Decomposition (SVD), Laplacian Eigenmap [Belkin & Niyogi, 2001] methods which minimize the loss function summing norms of node vector difference weighted by node pair similarities, inner-product methods aiming to approximate the node pair similarities with inner product of node vectors, like GraRep [Cao, Lu, & Xu, 2015], HOPE [Ou, Cui, Pei, Zhang, & Zhu, 2016]. Another type of learning methods is random walk based, like DeepWalk [Perozzi et al., 2014], LINE [Tang et al., 2015] and Node2Vec [Grover & Leskovec, 2016], these methods generate paths through random walks, and introduce the skip-gram structure in the NLP domain, treating paths as sentences, nodes as words, to learn the representations of nodes, and further train a classifier with these node embeddings as features for link prediction. Unlike matrix factorization approaches, a deep-biased search strategy in random enables the models to capture long dependency for nodes far away from each other more easily at a price of extra cost to generate the random walk paths. Meanwhile, some neural network models specifically designed for graph structure data, with higher efficiency and the ability to integrate node attributes, also gain extensive interests of researchers, like Graph Convolution Network (GCN) [Kipf & Welling, 2019] which defines the efficient layer-wise operation that is based on a first-order approximation of spectral convolutions on graphs, Graph Attention Networks (GATs) [Veličković et al., 2018] which introduces the attention mechanism into the graph neural network, and allows different weights for different nodes within the neighborhood.

Regarding the ranking tasks on the graph nodes, early approaches are based
on metrics like degree centrality, closeness centrality and betweenness centrality. PageRank (Page, Brin, Motwani, & Winograd, 1998) and HITS (Kleinberg, 1999) are also prominent as unsupervised ranking approaches. PageRank assigns weights to each node and iteratively updates it according to the weights of neighbors. HITS works similarly but two scores are calculated: the hub score and the authority score, the prior generally describes how many other nodes are "pointed" by target one, while the latter describes how many different hubs "point" to the target node. Meanwhile, there are also supervised or semi-supervised algorithms for ranking on graphs, meaning the models are trained with labeled nodes. Some are based on regularization theory like (D. Zhou, Bousquet, Lal, Weston, & Schölkopf, 2004; Agarwal, 2006), in (Agarwal, 2006) the objective function is formulated as the sum of the empirical ranking error and smoothness regularization, solved as a convex quadratic program.

Recently there are many studies conducted in talent management with advanced artificial intelligence (Zhu et al., 2018a; Xu, Yu, Guo, et al., 2018; Qin et al., 2019; Sun et al., 2019; D. Zhang et al., 2019; L. Zhang et al., 2019). Specifically, regarding the problems of job link prediction and job ranking, in (Xu, Yu, Guo, et al., 2018), the author quantified the Difficulty of Promotion (DOP) based on the length of tenure and further proposed a Gaussian Bayesian Network (GBN) to model the joint distribution of job ranks and DOPs; in (D. Zhang et al., 2019), the author proposed a multi-view embedding method to learn representations of jobs for different perspectives and further utilized the embeddings to predict links between jobs; in (L. Zhang et al., 2019), a talent flow embedding model is developed to learn the bi-directional talent attractions of each company. Meanwhile, few efforts have been identified in
integrating both tasks in a unified framework, improving the overall performance through joint learning.

4.3 Preliminaries

4.3.1 Data Description

Our dataset includes career profile data and job salary data. The prior is collected from LinkedIn, an Online Professional Network providing employment-oriented services while the latter is collected from Glassdoor, a website displaying the reviews and salary information of jobs and companies submitted by current and former employees anonymously.

- **Career Profile Data.** It includes the job title, company name, and the start and end dates for the job positions of each user.

- **Job Salary Data.** It includes the numerical salary data of job titles and time units (year, month, day, hour). Different from career profile data which is user-specific, the salary data is only recorded at the job level, meaning it is an average salary for each job title. As preprocessing, we convert the raw salaries to the annual level, based on the consideration that 1 year includes 12 months, 260 working days, 2080 (260 × 8) working hours.

Table 4.1 and Table 4.2 are toy examples of datasets. The statistics are summarized in Table 4.3

The career profile data does not include the job level information, even some users put the job levels in the job titles, however, for most records, the job title only
Table 4.1. Example of career profile.

<table>
<thead>
<tr>
<th>Employee</th>
<th>Company</th>
<th>Job</th>
<th>Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>IBM</td>
<td>Software Engineer</td>
<td>2010.5 - 2013.4</td>
</tr>
<tr>
<td>Alice</td>
<td>IBM</td>
<td>Senior Software Engineer</td>
<td>2013.6 - 2015.4</td>
</tr>
<tr>
<td>Bob</td>
<td>BOA</td>
<td>Financial Analyst</td>
<td>2004.8 - 2007.4</td>
</tr>
<tr>
<td>Bob</td>
<td>BOA</td>
<td>Financial Manager</td>
<td>2007.4 - 2013.7</td>
</tr>
<tr>
<td>Bob</td>
<td>Chase</td>
<td>Financial Consultant</td>
<td>2013.9 - 2015.8</td>
</tr>
</tbody>
</table>

... ...

describes the job function and the job level is unavailable. Meanwhile, the existing taxonomies of job levels are only defined within companies, not globally applicable, even for companies in the same industry. Due to these limitations, we are seeking a substitute for job levels. In the literature related, it can be identified that there is a significant positive correlation between job levels and job salaries (Guan et al., 2014; Mainert, Niepel, Murphy, & Greiff, 2018). Moreover, job salaries can function as global ranking scores for job comparison, and are more accessible than job levels. Therefore, we collected the job salary data for job positions presented in the career profile data, to characterize the hierarchical levels of these jobs.

4.3.2 Definitions

Here we give some essential definitions which are used throughout this chapter.

Definition 1. **Job Transition** is the action of changing from one job to another.

*Given a job list* \(< j_1, j_2, ..., j_n >\) *of a talent which is chronologically ordered, all the job transitions are* \(< j_1, j_2 >, < j_2, j_3 >, ..., < j_{n-1}, j_n >\).*
Table 4.2. Example of job salary.

<table>
<thead>
<tr>
<th>Company</th>
<th>Job Title</th>
<th>Salary</th>
<th>Time Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>Technical Services Specialist</td>
<td>$70000</td>
<td>Year</td>
</tr>
<tr>
<td>Microsoft</td>
<td>UX Web developer</td>
<td>$94000</td>
<td>Year</td>
</tr>
<tr>
<td>Amazon</td>
<td>HR Intern</td>
<td>$6000</td>
<td>Month</td>
</tr>
<tr>
<td>Adidas</td>
<td>Sales Associate</td>
<td>$20</td>
<td>Hour</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 4.3. Statistics of career profile & salary data.

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td># of employees</td>
<td>8177</td>
</tr>
<tr>
<td># of Job Titles</td>
<td>9333</td>
</tr>
<tr>
<td># of Companies</td>
<td>1655</td>
</tr>
<tr>
<td># of Job Transitions</td>
<td>40417</td>
</tr>
<tr>
<td>Minimum Salary</td>
<td>18720</td>
</tr>
<tr>
<td>Maximum Salary</td>
<td>220000</td>
</tr>
<tr>
<td>Median Salary</td>
<td>69239</td>
</tr>
<tr>
<td>Mean Salary</td>
<td>76960.91</td>
</tr>
</tbody>
</table>
Definition 2. **Job Graph** is a graph \( G_J(V_J,E_J,W_J) \) where \( V_J \) is the set of job titles, \( E_J \) is the set of edges between job titles, there is an edge between two job titles if and only if there is a transition between these two jobs. Suppose the size of graph \( G_J \) is \( N_J \), \( W_J \in \mathbb{R}^{N_J \times N_J} \) is the weighted adjacency matrix associated with graph \( G_J \), specifically \( w_{ij} \in W_J \) is positive if there is an edge between \( i \) and \( j \) while \( w_{ij} = 0 \) means they are disconnected. According to the type of \( W \), we have two graphs of job transitions:

**Definition 2.1 Job Transition Count Graph** is the job graph \( G_J^{\text{count}}(V_J,E_J,W_J^{\text{count}}) \) where \( w_{ij} \in W_J^{\text{count}} \) is the total number of transitions from job \( i \) to job \( j \). A higher value of \( w_{ij} \) means more transitions between these two jobs, and the connection between them is stronger.

**Definition 2.2 Job Transition Time Graph** is the job graph \( G_J^{\text{time}}(V_J,E_J,W_J^{\text{time}}) \) where \( w_{ij} \in W_J^{\text{time}} \) is dependent on the transition time from job \( i \) to \( j \). A shorter transition time usually means two jobs have similar job functions, while a longer transition time means the latter job position may need more knowledge and specific skills to be qualified. Therefore, we use a monotonic decreasing function of transition time to characterize the connection between two jobs, formulated as:

\[
 w_{ij} = e^{-\frac{\text{avg}(\Delta t_{ij})}{\Delta t_{\text{max}}}}
\]

where \( \text{avg}(\Delta t_{ij}) \) is the average transition time from job \( i \) to \( j \), \( \Delta t_{\text{max}} \) is the maximum transition time for all job pairs.

**Definition 3. Company Graph** is a graph \( G_C(V_C,E_C,W_C) \) where \( V_C \) is the set of
companies, \( E_C \) is the set of edges between companies, there is an edge between two companies if and only if there is a talent flow between these companies. Suppose the size of graph \( G_C \) is \( N_C \), \( W_C \in \mathbb{R}^{N_C \times N_C} \) is the weighted adjacency matrix associated with graph \( G_C \), specifically \( w_{ij} \in W_C \) is the total number of employees flowing from company \( i \) to \( j \).

**Definition 4.** **Job Rankings** are jobs ordered by their hierarchy levels in the company. Equivalently we can also use job ranking scores \( < r_1, ..., r_{|V_J|} > \) which also defines the rankings of jobs. A higher ranking score means a higher position in the ranking list.

### 4.3.3 Problem Formulation

In this chapter, we are aiming to extract the hierarchy structure of company job titles, which can be decomposed into two sub-tasks: predict the potential links between job titles and estimate their ranking scores. Suppose we have a job title graph \( G \) with size \( N \), for which \( V \) is the set of nodes (job titles), \( E \) is the set of edges (transitions between job titles) and \( W \in \mathbb{R}^{N \times N} \) is the weighted adjacency matrix associated with the graph. Our goal is to learn sharable representations \( M = [m_1, m_2, ..., m_N] \in \mathbb{R}^{N \times K} \) of the graph nodes, and optimized it with the link prediction function \( f : V \times V \to R \):

\[
f(e_{ij} \in E | m_i, m_j) = \begin{cases} 1 & \text{if } s(m_i, m_j) \geq \delta; \\ 0 & \text{if } s(m_i, m_j) < \delta, \end{cases} \tag{4.1}
\]

where \( s(., .) \) gives a similarity score for any two nodes in \( V \) and \( \delta \) is the threshold, as well as the estimation function \( g : V \to R \) for the ranking scores:

\[
r_i = g(m_i) \quad \forall i \in \{1, 2, ..., N\}. \tag{4.2}
\]
4.4 Methodology

In this section, we propose the whole framework which is composed of three parts: the data normalization, the job and company graph embedding, and the job ranking prediction.

4.4.1 Data Normalization

There are no unified naming conventions for job titles across companies, and the original career profile data are submitted by users themselves respectively. Therefore, one company and one job will have multiple names in the raw dataset. This will tremendously increase the graph size, the training time, and memory cost, meanwhile it will also weaken the patterns of connection strength between jobs, lowering the effectiveness of the framework. To alleviate the problem, we conduct data normalizations for company names and job titles.

**Company Names Normalization**

We treat the series of company names \(<c_1, c_2, ..., c_{|V_C|}>\) as documents and compute the TFIDF (term frequency-inverse document frequency) of each word \(t\) in the company name vocabulary, and use the TFIDF vector \(v_i\) as features for company name \(c_i\). These procedures are formulated by the equations below:
\begin{align}
tf(t, c_i) &= f_{t,c_i}; \\
idf(t) &= \log\left(\frac{|C|}{1 + n_t}\right); \\
\text{TFIDF}(t, c_i) &= tf(t, c_i) \ast idf(t, C);
\end{align}

\begin{equation}
v_i = \langle \text{TFIDF}(t, c_1), ..., \text{TFIDF}(t, c_N) \rangle,
\end{equation}

where \( f_{t,c_i} \) is the frequency of word \( t \) in company name \( c_i \), \(|C|\) is the number of company names and \( N \) is the vocabulary size. Based on the TFIDF features of company names, the DBSCAN (Ester, Kriegel, Sander, & Xu, 1996) clustering is applied to the features, with \( \epsilon \) set to 0.2, min samples set to 2, and cosine is used as the distance metric. Company names within the same cluster are treated as one company, while the rest is reserved. Table 4.4 and Table 4.5 show the results and statistics of the company name normalization.

**Job Titles Normalization**

According to (D. Zhang et al., 2019), the words with low frequency in job titles are filtered out since these words are more like users’ unique information, which are redundant for the general job information. Similarly, we use the DBSCAN (Ester et al., 1996) method to identify clusters of job titles, and job titles within the same cluster are treated as one job title. Different from company names normalization, the job title clustering is not for all job titles from all companies, instead, we only cluster job titles within each company, considering each company may have its own naming convention, and even similar job titles may still be related to different job functions.
Table 4.4. Examples of normalization result.

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Samples in Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Corporation, Intel, Intel Technology, Intel technologies, Intel research,...</td>
<td></td>
</tr>
<tr>
<td>wells fargo, wells fargo bank, wells fargo securities, wells fargo bank NA, wells fargo financial, ...</td>
<td></td>
</tr>
<tr>
<td>SAP labs India pvt ltd, SAP labs LLC,...</td>
<td></td>
</tr>
<tr>
<td>quality assurance analyst, project quality assurance, quality assurance specialist,...</td>
<td></td>
</tr>
<tr>
<td>graduate trainee, sales graduate trainee, graduate sales trainee, graduate trainee sales,...</td>
<td></td>
</tr>
<tr>
<td>credit analyst, LA credit analyst, analyst credit, global credit analyst,...</td>
<td></td>
</tr>
</tbody>
</table>

and require different knowledge and skills in different companies. Table 4.4 and Table 4.5 show the results and statistics of the job title normalization.

4.4.2 Job Transition Count Graph Embedding

In this section, we present how to learn the embeddings of job titles in job transition count graph \( G_{\text{count}}^J(V_J, E_J, W_{\text{count}}^J) \).

Table 4.5. Statistics of normalization result.

<table>
<thead>
<tr>
<th># of Companies</th>
<th># Of Job Titles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before</td>
<td>1655</td>
</tr>
<tr>
<td>After</td>
<td>997</td>
</tr>
</tbody>
</table>
Figure 4.2. Framework for job position hierarchy extraction.
Graph Construction

After the normalization of job titles, the job transition count graph $G_{\text{count}}^J(V_J, E_J, W_{\text{count}}^J)$ is constructed. As stated in definition 2.1 given a set of job titles $V_J$, an edge $e_{ij}$ exists if and only if there is a transition from job $i$ to $j$, and $w_{ij}$ is the total number of such transitions for $\forall i, j \in V_J, i \neq j$. For convenience, we normalize the weight matrix as:

$$W_{\text{normalized count}}^J = D^{-1}W_{\text{count}}^J,$$

(4.7)

where $D$ is the diagonal matrix satisfying $D_{ii} = \sum_k w_{ik}$. Then we can interpret $w_{ij} \in W_{\text{normalized count}}^J$ as the probability of 1 step transition to job $j$ given current job $i$ in the job transition count graph, denoted by:

$$\Pr_{\text{count}}^1(j|i) = w_{ij} \in W_{\text{normalized count}}^J,$$

(4.8)

where the bottom suffix 1 of $\Pr$ means the 1 step transition. For brevity, we use $W_{\text{count}}^J$ to represent the normalized weight matrix in the following content without confusion.

Directed Graph Embedding

In previous graph embedding approaches, for each node, there is only one vector regarded as the representation for it, which is not adequate for a directed graph, especially when our purpose is to predict the link between two nodes. For example, given the embeddings $m_A, m_B$ of two nodes $A, B$, the popular edge functions for link prediction like Hadamard, Average, L1 or L2 will give the same output for $e_{AB}$ and $e_{BA}$, which is not reasonable for directed graphs. To address the problem, we propose
to learn two graph embeddings $M^{\text{in}}, M^{\text{out}}$. $M^{\text{in}}$ characterizes the nodes as ends of edges, while $M^{\text{out}}$ characterizes the nodes as starts of edges. As a result, for prediction of edge $e_{AB}$, the out-embedding $m^{\text{out}}_A$ of node $A$ and the in-embedding $m^{\text{in}}_B$ of node $B$ will be used, while for prediction of edge $e_{BA}$, embeddings $m^{\text{out}}_B$ and $m^{\text{in}}_A$ will be used, therefore, the predictions of edges $e_{AB}$ and $e_{BA}$ will be distinguished from each other.

In order to learn the embeddings of graph nodes, we seek to optimize the following objective function which is the log-likelihood of accessible neighborhood nodes for every given node:

$$\max \sum_{u \in V} \log \Pr(N_1(u) | u), \quad (4.9)$$

where $N_1(u) = \{ v \in V_j | e_{uv} \in E_j \}$, after plugging the embeddings in, the equation can be further written as:

$$\max_{M^{\text{in}}, M^{\text{out}}} \sum_{(u,v) \in E} \log \Pr_1(m^{\text{in}}_v | m^{\text{out}}_u) \quad (4.10)$$

$$= \max_{M^{\text{in}}, M^{\text{out}}} \sum_{(u,v) \in V^2} 1_{e_{uv} \in E} \log \Pr_1(m^{\text{in}}_v | m^{\text{out}}_u). \quad (4.11)$$

Equation (4.11) is valid under the assumption that $\Pr(v_1, v_2 | u) = \Pr(v_1 | u) \Pr(v_2 | u)$ for any given nodes $(v_1, v_2, u)$. For a weighted graph the above Equation (4.11) can be written as:

$$\max_{M^{\text{in}}, M^{\text{out}}} \sum_{(u,v) \in V^2} w_{uv} \log \Pr_1(m^{\text{in}}(v) | m^{\text{out}}(u)). \quad (4.12)$$

Given the definition

$$\Pr_1(m^{\text{in}}_v | m^{\text{out}}_u) = \frac{\exp(m^{\text{in}}_v \cdot m^{\text{out}}_u)}{\sum_{v' \in V} \exp(m^{\text{in}}_{v'} \cdot m^{\text{out}}_u)}, \quad (4.13)$$
High-order Proximity Encoding

The objective function (4.12) relies on the direct connections between nodes, therefore the embedding learned may provide less information about the connections to high-order neighborhoods. To encode the local structure of a wider neighborhood, we extend Equation (4.12) to K-order neighborhoods.

\[
\text{Loss}_{\text{embedding}}^\text{count} = -\sum_{k=1}^{K} \sum_{(u,v) \in V^2} \text{Pr}^{\text{count}}_k(v|u) \log \text{Pr}_k(m_v^\text{in}|m_u^\text{out})
\]

(4.14)

\[
= -\sum_{k=1}^{K} \sum_{(u,v) \in V^2} W^k_{u,v} \log \text{Pr}_k(m_v^\text{in}|m_u^\text{out}),
\]

(4.15)

where \(\text{Pr}^{\text{count}}_k(v|u)\) is the probability that job \(v\) is visited after \(k\) steps given job \(u\), \(W\) is the normalized weight matrix in Equation 4.7, \(W^k\) is the matrix product of \(k\) copies of \(W\), \(\text{Pr}_k(m_v^\text{in}|m_u^\text{out})\) is recursively defined as below:

\[
\text{Pr}_k(m_v^\text{in}|m_u^\text{out}) = \sum_{u^* \in V} \text{Pr}_{k-1}(m_v^\text{in}|m_u^\text{out}^*) \text{Pr}_1(m_v^\text{in}|m_u^\text{out}).
\]

(4.16)

4.4.3 Job Transition Time Graph Embedding

In addition to the job transition count graph, the transition time graph provides another view of the connections between jobs, which may also facilitate the job position hierarchy extraction. For example, suppose there are 3 jobs A, B, and C. If the average transition time from A to B is 12 months, while it is 14 months from B to A. It is likely both jobs require similar skills and background, further they may have similar ranks in the hierarchical structure. Meanwhile, if the average transition time from A to C is more than 24 months, meaning job C may need different or higher-level skills than job A, then we can infer compared to job B, job C has a weaker connection with
job A, probably with a higher rank than A and B. Therefore, we are motivated to construct the job transition time graph $G_{time}^{j}(V_J, E_J, W_{time}^{j})$ as defined in Definition 2.2. The weight $w_{ij}$ is defined as $e^{-\frac{\text{avg}(\Delta t_{ij})}{\Delta t_{\text{max}}}}$ since a longer transition time often means a higher requirement of skills taking large efforts to be acquired which can be viewed as a dissimilarity between jobs. The embedding strategy is similar to job transition, we construct the graph, normalize the weight matrix, and optimize with loss functions. For brevity, we just show the loss function below:

$$\text{Loss}_{\text{time embedding}} = -\sum_{k=1}^{K} \sum_{(u,v) \in V^2} W^k_{u,v} \log \text{Pr}_k(m^\text{in}_v | m^\text{out}_u).$$  \hspace{1cm} (4.17)

### 4.4.4 Company Graph Embedding

Different from the above two graphs, the company transition graph enables us to view the job transitions at a higher level, which may bring extra insights into the solution of job position hierarchy extraction. It is common to see similar companies in the same industry, like Google and Facebook, having a large number of similar job positions. Meanwhile, there are also large talent flows between these job positions. In other words, job positions from two companies with large talent flow are more likely to be connected than the ones from two companies with few transitions. Moreover, it is reasonable to consider companies with larger in-flow talents are more competitive, as a result, job positions in these companies will be paid with higher salaries than similar ones in other companies, indicating the talent flow of companies potentially reflects the ranks of their job positions in the global job position hierarchy. This motivates us to construct the company transition graph as defined in Definition 3. Similar to job transition count graph embedding, we normalize the weight matrix $W_C$.
as we did in Equation 4.8 and formulate the loss function as below:

\[
\text{Loss}^\text{company}_{\text{embedding}} = -\sum_{k=1}^{K} \sum_{(u,v) \in V^2} W_k \log \Pr_k \left( m^{\text{in}}_v \mid m^{\text{out}}_u \right).
\]  (4.18)

### 4.4.5 Ranking Estimation

In order to extract the hierarchy of job titles, it is important to precisely estimate the rankings of the job titles (nodes) in the graph. Specifically, our task is to find the estimator \( g \) which optimize the following objective function:

\[
\min \sum_{v \in V} \| g(m^\text{count}_v \oplus m^\text{time}_v \oplus m^\text{company}_v) - r_v \|^2; \quad (4.19)
\]

\[
m^S_v = m^{\text{in}}_v \oplus m^{\text{out}}_v, \forall S \in \{\text{count}, \text{time}, \text{company}\}, \quad (4.20)
\]

where \( r_v \) is the ranking score for node \( v \), \( \oplus \) is the concatenation operation for vectors.

We implement \( g \) as two fully connected layers neural network with tanh activation function, which is formulated as below:

\[
m_v = m^\text{count}_v \oplus m^\text{time}_v \oplus m^\text{company}_v; \quad (4.21)
\]

\[
h_v = \tanh(W_1 m_v + b_1); \quad (4.22)
\]

\[
g_v := g(m_v) = \tanh(W_2 h_v + b_2). \quad (4.23)
\]

Now we can see the ranking loss is based on the embeddings of job titles and companies, as a result, by optimizing the ranking loss, the information of known rankings is forced to be encoded into these embeddings. In other words, the embeddings learned will function as effective features for the ranking prediction task.
4.4.6 Training

As a whole, the embedding modules for the job transition count graph, job transition time graph, and job transition company graph are integrated with the ranking module. The unified framework can learn embeddings of jobs which not only encode the local structures of three graphs defined but also optimized for ranking task. The total loss is formulated as below:

\[
\text{Loss} = \alpha_1 \text{Loss}_{\text{count}} + \alpha_2 \text{Loss}_{\text{time}} + \alpha_3 \text{Loss}_{\text{company}} + \alpha_4 \text{Loss}_{\text{ranking}},
\]

(4.24)

where \(\alpha_i (i = 1, 2, 3, 4)\) are hyperparameters for losses. We apply the gradient descent to learn the embeddings and all the parameters.

4.4.7 Prediction

Once the training is completed, we can further utilize the framework to predict the link between jobs as well as the ranking scores of the job titles. The link prediction is based following rules:

\[
m_{u}^{\text{out}} = m_{u}^{\text{out,count}} \oplus m_{u}^{\text{out,time}} \oplus m_{u}^{\text{out,company}}; \quad (4.26)
\]

\[
m_{v}^{\text{in}} = m_{v}^{\text{in,count}} \oplus m_{v}^{\text{in,time}} \oplus m_{v}^{\text{in,company}}; \quad (4.27)
\]

\[
\hat{w}_{uv} = m_{u}^{\text{out}}, m_{v}^{\text{in}}; \quad (4.28)
\]

\[
f(e_{uv} \in E| M^{\text{count}}, M^{\text{time}}, M^{\text{company}}) = \begin{cases} 
1 & \text{if } \hat{w}_{uv} \geq \delta; \\ 
0 & \text{if } \hat{w}_{uv} < \delta.
\end{cases} \quad (4.29)
\]
For the ranking score prediction, we use a neural network with two-fully-connected layers which has the same structure as the one defined in Section 4.4.5.

4.5 Experiment

Regarding the effectiveness of the framework, there are a few questions to be answered through experiments:

1. How does the framework perform on the link prediction task compared to existing classification methodologies?

2. How does the framework perform on the ranking task compared to existing advanced ranking methodologies?

3. How is the quality of the embedding of job titles in terms of interpretability compared to other embedding methods?

4.5.1 Experiment Setup

The statistics of the data are listed in Table 4.5. In preprocessing, the salary data are normalized to $[0,1]$ through $\frac{s_i - s_{\text{min}}}{s_{\text{max}} - s_{\text{min}}}$ and are used as ranking scores for the job levels as stated in Section 4.3.1. To set up the experiment, similar to (Grover & Leskovec, 2016; Kipf & Welling, 2016b), for the link prediction task, we randomly selected 65% of edges for training (making sure the graph is still connected), 10% of edges for validation, and 25% of edges for test. For each part of the data, we add the same amount of node pairs without connection as negative samples, so each of them is a balanced dataset. For the ranking prediction task, similarly, we randomly selected...
65% of nodes for training, 10% of nodes for validation, and 25% of nodes for test. In the experiment, we set our framework with the following configuration: \( K \) is set to 8 so the framework can capture the dependency within 8th-order neighborhoods, the embedding dimensions for all three graphs are set to 64, 32 for in embedding and 32 for out embedding, \( \alpha_1, \alpha_2, \alpha_3 \) are set to 0.05 and \( \alpha_4 \) is set to 0.85, the learning rate is set to 0.05.

### 4.5.2 Evaluation on Link Prediction

For the ranking module, we use a neural network with two fully connected layers with hidden units set to 8, 1, and \( tanh \) as the activation function. Following baselines are selected to compare the performance of link prediction:

- **Common Neighbors** is a metric defined as the number of common neighbors for a pair of nodes:

  \[
  s(u, v) = |N(u) \cap N(v)|.
  \]

  where \( N(u) \) is the set of neighbor nodes of node \( u \).

- **Jaccard’s Coefficient** is similar to Common Neighbors, but with normalization by the total number of shared and non-shared neighbors:

  \[
  s(u, v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}.
  \]

- **Adamic-Adar Score** is an index giving more weight to relatively fewer common neighbors:

  \[
  s(u, v) = \sum_{w \in N(u) \cap N(v)} \frac{1}{\log |N(w)|}.
  \]
• **GraphRep** (Cao et al., 2015) is a matrix factorization based method, with a k-step loss function defined on graphs, it integrates rich local structural information and captures the global structural properties of the graph. The highest order is set to 8.

• **HOPE** (Ou et al., 2016) is a matrix factorization based framework, but different from GraphRep, it aims to preserve asymmetric transitivity and high-order proximity.

• **DeepWalk** (Perozzi et al., 2014) is a framework that uses truncated random walks to learn the latent representations of nodes by maximizing the co-occurrence probability among nodes within a window, treating nodes as words and walks as sentences. For each node, the number of walks is set to 10, the walk length is set to 80, the window size is set to 5.

• **Node2vec** (Grover & Leskovec, 2016) designs a flexible neighborhood sampling strategy allowing to smoothly interpolate between Breadth-First Searching and Depth-First Searching. The objective is to maximize the likelihood of preserving network neighborhoods. Parameter $p$ is set to 2, $q$ is set to 0.5, and other parameters are the same as DeepWalk.

• **VGAE** (Kipf & Welling, 2016b) is an unsupervised framework on graph-structured data based on variational auto-encoder. It applies a graph convolution network encoder and an inner product decoder.

• **GAE** (Kipf & Welling, 2016b) is a non-probabilistic variant of the VGAE model.
with an auto-encoder structure.

- **Job2Vec** (D. Zhang et al., 2019) is a state-of-the-art approach for job embedding based on multi-view of job transition graph. Unlike our approach, it does not emphasize the directionality and high order proximity and purely based on job transition graph.

The Common Neighbors, Jaccard’s Coefficient, and Adamic-Adar Score are all similarity metrics based approaches; GraphRep and HOPE are both matrix factorization based methods; DeepWalk, LINE, and Node2vec are random walks based methods; VGAE and GAE are graph auto-encoder frameworks. For all the approaches which generate latent representations of nodes, we set the embedding dimension to 64, which is the same as our approach, and we learn representations for all three graphs: job transition count graph, job transition time graph, and company graph. Once the training phase is complete, we use the inner product of concatenated embeddings of nodes as a similarity score and make link prediction as stated in Equation 4.29. The metric for performance evaluation is Area Under the Receiver Operating Characteristic Curve (ROC AUC), which illustrates the diagnostic ability of binary classifier with varied discrimination score. The value is between 0 and 1, a higher value indicates better performance.

The experiment is repeated 10 times, the means of evaluation metrics are listed in Table 4.6. We have the following findings: firstly our own approach outperforms other baselines, demonstrating the overall effectiveness of the framework; secondly, HOPE is the next best baseline, indicating the encoding of directionality and high
Table 4.6. Performances for job link prediction.

<table>
<thead>
<tr>
<th>Method</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Neighbors</td>
<td>0.7045</td>
</tr>
<tr>
<td>Jaccard's Coefficient</td>
<td>0.6970</td>
</tr>
<tr>
<td>Adamic-Adar Score</td>
<td>0.7069</td>
</tr>
<tr>
<td>GraphRep</td>
<td>0.7435</td>
</tr>
<tr>
<td>HOPE</td>
<td>0.7508</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>0.7374</td>
</tr>
<tr>
<td>Node2vec</td>
<td>0.7388</td>
</tr>
<tr>
<td>VGAE</td>
<td>0.6812</td>
</tr>
<tr>
<td>GAE</td>
<td>0.7337</td>
</tr>
<tr>
<td>Job2Vec</td>
<td>0.7422</td>
</tr>
<tr>
<td>Our Method</td>
<td><strong>0.7782</strong></td>
</tr>
</tbody>
</table>

Order proximities are essential for link prediction task; meanwhile by comparing our own method with HOPE, both models are capable to capture directionality and high order proximities, we can see the joint structure of embedding learning and ranking prediction has improved the performance; in general the embedding methods are better than metric-based ones, meanwhile these baselines are not specifically designed for directionality and not integrating the ranking information leading to inferior performances.
4.5.3 Evaluation on Ranking Prediction Task

For ranking prediction, we selected baselines like Gaussian Bayesian Network (GBN) which is a state-of-the-art unsupervised approach for job title ranking estimation (Xu, Yu, Guo, et al., 2018), and embeddings learning framework GraphRep, HOPE, DeepWalk, LINE, Node2vec, VGAE, and GAE, for which once the representations are learned, we further built and trained a network with two fully connected layers, hidden units setting to 8, 1 as regressor for the ranking score prediction. To be noted, GBN (Xu, Yu, Guo, et al., 2018) is an unsupervised approach, which is why its MAE and MSE are unavailable.

The evaluation metrics for ranking score prediction are Mean Absolute Error (MAE), Mean Squared Error (MSE), and Normalized Discounted Cumulative Gain (NDCG@10) on the test nodes based on ranking scores predicted.

defined as below:

\[
\text{MAE} = \frac{1}{|V_J|} \sum_{v \in V_J} |\hat{r}_v - r_v|; 
\]

\[
\text{MSE} = \frac{1}{|V_J|} \sum_{v \in V_J} ||\hat{r}_v - r_v||; 
\]

\[
\text{DCG}_p = \sum_{i=1}^{p} \frac{r_i}{\log(i+1)}; 
\]

\[
\text{NDCG}_p = \frac{\text{DCG}_p}{\text{IDCG}_p}, 
\]

where \(r_v\) is the true ranking score for job \(v\), \(\hat{r}_v\) is the predicted ranking score, IDCG is the ideal DCG for the list of nodes sorted by ranking scores in descending order.

The experimental results for ranking score prediction are listed in Table 4.7 Overall, our approach achieves a better result than other baselines, demonstrating the
### Table 4.7. Performances for job ranking prediction.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>MSE</th>
<th>NDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>GraphRep</td>
<td>0.2866</td>
<td>0.0991</td>
<td>0.4694</td>
</tr>
<tr>
<td>HOPE</td>
<td>0.2015</td>
<td>0.0730</td>
<td>0.5081</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>0.2390</td>
<td>0.0948</td>
<td>0.4781</td>
</tr>
<tr>
<td>Node2vec</td>
<td>0.2360</td>
<td>0.0928</td>
<td>0.4406</td>
</tr>
<tr>
<td>VGAE</td>
<td>0.2353</td>
<td>0.0967</td>
<td>0.5023</td>
</tr>
<tr>
<td>Job2Vec</td>
<td>0.2241</td>
<td>0.0855</td>
<td>0.4281</td>
</tr>
<tr>
<td>GAE</td>
<td>0.1948</td>
<td>0.0650</td>
<td>0.5320</td>
</tr>
<tr>
<td>GBN</td>
<td>N/A</td>
<td>N/A</td>
<td>0.4699</td>
</tr>
<tr>
<td>Our Method</td>
<td><strong>0.1901</strong></td>
<td><strong>0.0633</strong></td>
<td><strong>0.5395</strong></td>
</tr>
</tbody>
</table>

effectiveness of the joint structure for ranking prediction task, since the ranking loss is integrated with the embedding loss in the training process; meanwhile, VGAE, GAE, and HOPE also perform well, indicating the structure information including the proximities of nodes and directionality can facilitate the ranking prediction task on graphs.

### 4.5.4 Robustness Analysis

In this section, we test the sensitivity of the hyperparameters in our methodology. Specifically, we have the following parameters: the embedding dimensions, $K$ the highest order neighborhood to encode, $d$ the embedding dimension, and $\alpha_i (i = 1, 2, 3, 4)$ weights for different losses. Initially we set $d = 32, K = 4, \alpha_1 = 0.3, \alpha_2 = 0.3, \alpha_3 = 1, \alpha_4 = 4$. Next, we fix all parameters except one of them and evaluate
the framework for various values of the specific parameter. \{\alpha_i\}^4_{i=1} control the relative weight for different losses, instead of tuning the value for each of them, we set \(\alpha_1 = \alpha_2\) and tune the values of \(\alpha_1/\alpha_3\) and \(\alpha_4/\alpha_3\), since the embedding losses for job transition count graph and job transition time graph are usually at the same scale.

The impact of the parameters on performances is tested for both link prediction and ranking prediction tasks. For link prediction, we use AUC as the evaluation metric, while for ranking prediction, we use MAE. The range of values for parameters are set as follows: \(K\) is set with values 1, 2, 4, 6, 8; \(d\) is set with values 16, 32, 64, 128, 256; \(\alpha_1/\alpha_3\) is set with values 0.05, 0.1, 0.3, 0.5, 0.7; \(\alpha_4/\alpha_3\) is set with values 1, 2, 4, 6, 8.

The results are displayed in Figure 4.3. We have the following findings: firstly, as we increase \(K\), AUC is getting larger, indicating encoding high order proximities will improve the performance of link prediction, meanwhile the value of MAE versus \(K\) is high at intermediate orders like 2, 4 but is improved with both lower and higher orders; secondly increasing the embedding dimension will improve the performances for both link and ranking prediction; as we increase the ratio \(\alpha_1/\alpha_3\) for job graphs embedding, the AUC is getting larger, since the link prediction task is mainly based on job graphs, moreover, the MAE is also improved with a higher job graph embedding ratio \(\alpha_1/\alpha_3\), indicating at some level, the performances for both tasks can be improved with better job graph embeddings; compared with other parameters, the curves of AUC and MAE for ranking ratio \(\alpha_4/\alpha_3\) are more complex to interpret, the changes of the curves suggest the existences of more than one optimal values of the ratio, since potentially changing the ratio \(\alpha_4/\alpha_3\) will also change the ratio \(\alpha_4/\alpha_1\) which may lead to complicated results.
Figure 4.3. Robustness analysis.

(a) AUC for $K$

(b) MAE for $K$

(c) AUC for $d$

(d) MAE for $d$

(e) AUC for $\alpha_1/\alpha_3$

(f) AUC for $\alpha_4/\alpha_3$

(g) MAE for $\alpha_1/\alpha_3$

(h) MAE for $\alpha_4/\alpha_3$
4.5.5 Visualization

To evaluate the qualities of embeddings, we visualize and compare the embeddings learned from different approaches including DeepWalk, Node2Vec, GraphRep, HOPE, Node2Vec, GAE, and our method. Specifically, we proportionally and randomly select 2000 engineering job titles from 5 companies: Amazon, Microsoft, Google, IBM, Facebook, and use TSNE to project the embeddings of these job titles (excluding the company embedding) learned by different approaches into a 2D space and make scatter plots in Figure 4.4. Job titles from the same company will be marked with the same color. It can be seen for both DeepWalk and Node2vec, the job titles are close to each other, the boundaries of job titles from different companies are not quite clear. In contrast, for GraphRep, HOPE, and GAE, the job titles are more distributed and form small clusters. Our method is a balance between the prior two, the job titles form small and large clusters and boundaries are also clearer.

4.6 Case Study

As a case study, the job position hierarchies extracted are plotted in Figure 4.5 and Figure 4.6. Specifically, we select a subset of job titles from two companies, Amazon and Microsoft. For both companies, the job titles with similar ranking scores are grouped together. 77% of ranking scores are revealed in the training phase, while the rest 23% are from predictions. The predicted links between job titles are also plotted. Figure 4.5(a) and Figure 4.5(b) show the internal hierarchy of Amazon and Microsoft jobs; Figure 4.6 shows the interconnections of jobs from two companies. These can assist with tasks like person-job fit, salary benchmarking in management.
Figure 4.4. Visualization of learned embeddings.
(a) Hierarchy of Amazon jobs.

(b) Hierarchy of Microsoft jobs.

Figure 4.5. Case study: internal connections within companies.
Figure 4.6. Case Study: interconnections of jobs from two companies.
4.7 Conclusion

In this chapter, we proposed a holistic data-driven approach to address the problem of job position hierarchy extraction. Along this line, we introduced a joint embedding and ranking framework to learn the representations of job titles based on the graphs of job positions and the transition relationships of employees between companies. In fact, the representations can effectively capture the directionality of job transitions as well as the proximities of high-order neighborhoods. Also, once optimized with the ranking loss, the representations can encode the information of hierarchical job position levels, and thus enabling to identify the potential links between job position titles and predict their rankings as the estimations of hierarchical levels. Finally, extensive experiments on real-world datasets validated the effectiveness and robustness of our framework. Indeed, our joint learning structure can bring benefits for both tasks of the link and ranking prediction. Moreover, the visualization and case studies showed the good interpretability of the embeddings learned by our approach.
CHAPTER 5

CONCLUSIONS

In this dissertation, we address some unique challenges in connection and influence-based analysis for talent management, such as influence modeling and behavior prediction, the fusion of multiple networks, and learning on.

Firstly, we build an influence-based framework to address the complexities of influence modeling and the relationship between influence and behavior. We construct a co-worker turnover events series ordered chronologically for each employee and collect features that characterize the connection between the co-worker and the target employee. Further, the Long Short Term Memory (LSTM) is used to model the temporal dynamics of turnover influence and predict the future turnover of target employees. Employee’s profile and environment factors like department monthly turnover statistics within one year are also included in the input, so the whole problem is formulated as a multi-sequence classification problem. Moreover, a global attention mechanism is designed to evaluate the heterogeneous impact on potential turnover behaviors, which also improves the interpretability of turnover prediction and provides actionable insights for talent retention.

Secondly, we propose a temporal networks fusion-based framework to fully take advantage of the network structure formed by connections and deal with the influence on multiple networks. We integrate graph convolution network (GCN) and Long Short
Term Memory (LSTM), the prior models the influence diffusion on each network, while the latter captures temporal dependency; further, we propose two approaches for multi-network fusion. One is static network structure fusion, which aggregates multiple networks based on the additive Laplacian matrices adjusted by static weights; the other is cross-sequence attention based on which we aggregate the hidden state on each network with dynamically calculated attention weights; it gives the framework more flexibility at the cost of more parameters to be trained. To address the units with similar behaviors but are not revealed in the observed connections, which is commonly seen in social network data due to scarcity and sparsity, we construct a similarity network, the weight of connection is defined as the cross-correlation of historical observations of two units (nodes) and integrate the similarity as a supplement network in our framework.

Last, we propose a learning framework that is able to capture both the directionality and the high-order neighborhoods, facilitating us to learn more effective vector representations of directed network structure. For each node, two embeddings are to be learnt, out-flow embedding & in-flow embedding. We design a cross-entropy-like objective function based on the two embeddings. Moreover, the objective function depends on both the direct and indirect (connected through multiple edges) connections, allowing the embedding to encode high-order neighborhoods. Meanwhile, we integrate a supervised node ranking task with unsupervised learning to further tune the representations. We show the integration can help to train embeddings of nodes which is more effective in identifying potential connections and estimating the rankings of the nodes. As an example, we show the application on the construction of
job positions graph, the connection is based on peoples’ job-hopping records, and the ranking is based on the average salary data. With the framework, the hierarchical structure of job positions is extracted, which is essential for many tasks in human resource management, such as person-job fit, salary benchmarking, and organizational structure optimization.
BIBLIOGRAPHY


doi: 10.1145/1143844.1143848

Agovino, T. (2019). To have and to hold. Amid one of the tightest labor markets in the past 50 years, employee retention is more critical than ever.


Boushey, H., & Glynn, S. J. (2012). *There are significant business costs to replacing employees.*


Heinz, K. (2020). Dangers of turnover: Battling hidden costs. revenue loss shouldn’t be your only concern — understand the true costs of turnover.


