Understanding and Prediction of User Behavior in Online Social Networks: the Case of GitHub

Tong Yang

Delft University of Technology

Abstract

Along with the continuous development of Internet technology, Online Social Networks (OSNs) have gradually become the most popular platforms for content creation, information sharing and communications between users on the Internet. Understanding and prediction of user behavior in OSNs are essentially valuable. In the past few decades, machine learning is widely used and has become incredibly powerful in user behavior prediction. However, few researchers have considered the combination of machine learning and network analysis, especially using the hidden network information as features for prediction problem. In this thesis, we propose a novel method for user activity prediction by using machine learning algorithms as well as network properties from Github. The prediction is based on previous activities of not only the user him/herself but also his/her neighbors on GitHub. The results of prediction and performance evaluation demonstrate that the neighbor activity information from both one-layer and two-dimension networks of GitHub indeed can help to improve the performance of predicting a user's active level. Additionally, the massive analysis of how our methods can help to improve the prediction accuracy is given from both the network and time series analysis perspective.

TU Delft Delft University of Technology

MMC Multimedia Computing Group

Understanding and Prediction of User Behavior in Online Social Networks: the Case of GitHub

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

ELECTRICAL ENGINEERING

by

Tong Yang
born on 19 April 1992 in China

Multimedia Computing Group
Electrical Engineering (track Telecommunications & Sensing Systems)
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Understanding and Prediction of User Behavior in Online Social Networks: the Case of GitHub

by Tong Yang

Abstract

Along with the continuous development of Internet technology, Online Social Networks (OSNs) have gradually become the most popular platforms for content creation, information sharing and communications between users on the Internet. Understanding and prediction of user behavior in OSNs are essentially valuable. In the past few decades, machine learning is widely used and has become incredibly powerful in user behavior prediction. However, few researchers have considered the combination of machine learning and network analysis, especially using the hidden network information as features for prediction problem. In this thesis, we propose a novel method for user activity prediction by using machine learning algorithms as well as network properties from Github. The prediction is based on previous activities of not only the user him/herself but also his/her neighbors on GitHub. The results of prediction and performance evaluation demonstrate that the neighbor activity information from both one-layer and two-dimension networks of GitHub indeed can help to improve the performance of predicting a user's active level. Additionally, the massive analysis of how our methods can help to improve the prediction accuracy is given from both the network and time series analysis perspective.

Name	:	Tong Yang
Student Number	:	4472128
Group	:	Multimedia Computing Group
Master Programme	:	Electrical Engineering
Specialisation	:	Telecommunications & Sensing Systems
Date of Thesis Defense	:	22-08-2017

Committee Members :

Advisor:	Huijuan Wang, MMC, TU Delft
Chairperson:	Alan Hanjalic, MMC, TU Delft
Member:	Alessandro Bozzon, WIS, TU Delft
Member:	Xiuxiu Zhan, MMC, TU Delft

Dedicated to my family and friends

Contents

List of Figures

List of Tables

List of Acronyms

OSN Online Social Network

DT Decision Tree

RF Random Forest

KNN K-Nearest Neighbor

MDP Markov Decision Process

MLP Multi-Layer Perceptron

TP True Positive

TN True Negative

FP False Positive

FN False Negative

PPV Positive Predictive Value

TPR True Positive Rate

ACC Accuracy

ACC(0) Accuracy for active level of Label 0: Inactive

ACC(1) Accuracy for active level of Label 1: Slightly Active

ACC(2) Accuracy for active level of Label 2: Moderate Active

ACC(3) Accuracy for active level of Label 3: Highly Active

OSS Open Source Software

PCC Pearson Correlation Coefficient

Acknowledgements

First of all, I would like to express sincere gratitude to my thesis advisor, Huijuan Wang of Multimedia Computing Group at TU Delft. Without her assistance and dedicated involvement in every step throughout the process, this thesis would have never been accomplished. During my thesis, Dr. ir. Wang always inspired and encouraged me, which really helped me make great progress. She consistently allowed this thesis to be my own work, but steered me in the right direction whenever she thought I needed it.

Furthermore, I would like to gratefully acknowledge the help of my second daily supervisor, Xiuxiu Zhan, who has offered me valuable suggestions during the thesis project and writing. I really learned a lot from her, including her rigorous research attitude and nice personality.

What's more, I also owe a special debt of gratitude to all the experts who were involved in my thesis: Jie Yang, Qin Lin and Yang Ma. I have benefited a lot from their valuable comments on this thesis. I would like to thank Prof. dr. Alan Hanjalic and Dr. Alessandro Bozzon, who are willing to be the members of my thesis committee. I am also very grateful to my friends and my fellow classmates who gave me their time in listening to me, encouraging me and helping me during my tough time.

Finally, I must express my very profound gratitude to my beloved parents, who give me life and bring me up. They have made every effort to offer me the best everything unreservedly, providing me the opportunity to study here. They support me at all times though they may even not know about my work. I owe my parents everything.

Thank myself for not giving up. Life is not smooth sailing. May there be enough clouds in my life to make a beautiful sunset.

Tong Yang
Delft, The Netherlands
August 13, 2017

1 Introduction

The idea of Online Social Network (OSN) has become ubiquitous due to the rapid development of social networking services [4]. OSNs like Facebook, Twitter, GitHub facilitate users to build different relationships on the platforms according to their personal preferences as well as career similarities. Simultaneously, users can also carry out different activities, such as making friends, posting contents, commenting or forwarding their followees' posts. These activities can be considered as different user behavior on OSNs. Besides, data mining of OSNs can help to improve user experience and loyalty to OSNs. Recently, many different types of data analysis have been conducted on OSNs, including user activity prediction, social recommendation, viral marketing, etc [5][6][7][8][9]. In this thesis, we mainly focus on the issue of user activity prediction.

1.1 Motivation

Development of data mining and machine learning technology makes the analysis of massive online data become more easily. Predicting user activities, such as the number of messages a user will post at a given day or when a user will access to the website or mobile application, is essentially crucial for online platforms [10][11]. If we have better understanding and prediction of user behavior on OSNs, we can estimate the usage or the popularity of the online platform in the long run [12]. Besides, the data of a user's clicks, browsing history and other behavior reflects his/her interests and needs, which can be utilized to greatly enhance the effectiveness of online advertising, marketing and so on [13][14]. We can also design targeted marketing strategies based on the data, e.g. to determine at what time or to whom to promote advertisements [15][16][17][18]. In addition, we are able to better arrange the allocation and integration of platform's resources if we know when users will connect to the OSNs, in order to save more resources and provide more stable services to users. What's more, users can save a lot of time in searching for the people or news. Thus, prediction of user activity is an essential and inevitable issue for OSNs.

In the past few decades, machine learning is widely used and has become incredibly powerful in user activity prediction. For example, Althoff et al. proposed a method of exploring the influence of OSNs upon user behavior [19]. The study is carried out in an application which can record and track user's physical activity [19]. In [20], Burke studied user behavior prediction by using the activity data of users and characteristics of a user him/herself, such as the gender, age and country. Another method for prediction is using complex network theory. For instance, Lee and Lim achieved the link prediction by using network information from multiple social networks [21]. However, few researchers have considered the combination of these two methods, especially using the hidden network information as features for prediction problem. As it is studied in [19], social networks

can influence user activity from both online and offline. It is reasonable to use network properties as features in machine learning for prediction problem.

The multidimensional network is a kind of multilayer network in which users may have different relationships in various layers [22][23][24]. Taking Github as an example, there are two kinds of relationships in this network, i.e. follower-followee and collaboration relationship. However, how to mine network properties in different layers and use them as features for prediction issues is still a challenging problem. Therefore, we are planning to gain insight into using network information from multidimensional social networks into prediction of user activity and explore how network information can help to improve the prediction accuracy in this thesis.

1.2 Thesis Goal

In this thesis, we will propose a new method for user activity prediction on Github by combining machine learning methods with network theory. The main purpose of this thesis is to explore how network information can help to improve the prediction accuracy. Additionally, we will further elaborate the improvement of our method by using network and time series analysis. It can be formulated by several subgoals as follows:

First of all, we are about to define the baseline of prediction by using the activity data to predict active level of a user in the next moment. Meanwhile, we would like to investigate the selection of the time interval of activity data and figure out how many previous time intervals should be taken into account when predicting a user's behavior in baseline.

Secondly, we are planning to use multidimensional network information from OSNs to explore what kind of network information can be used as features for predicting user behavior, in order to open new possibilities for prediction and further improve the performance. And if possible, we would like to find out how to use the network information more effectively.

Finally, we are going to use network and time series analysis to provide further explanations for experiment results. We would like to explore more possibilities of combining machine learning with network analysis.

1.3 Contributions

Taking GitHub as a case, the thesis achieved the following contributions:

- We found out when investigating user behavior, the most suitable time interval is 1 week. We can obtain better performance when using the information of previous 4 weeks to predict the active level of a user in the following week, which is set to be the baseline of our experiment.
- When using network properties as features for user behavior prediction, we found that the neighbor activity information from both one-layer networks and two-dimension network can help to improve the prediction accuracy compared to the baseline. What's more, we found out that the common neighbors of a user from

the two layers, those who belong to both the most active neighbors from Follower-Followee Network and the closest collaboration neighbors from Collaboration Network, will have larger impact upon improving the performance of user behavior prediction compared with regular neighbors.

- We provided further explanations for the results of experiment by using network and time series analysis. The results of this thesis successfully give new insights of using the network information from OSNs as features to improve the performance of user behavior prediction.

1.4 Thesis Outline

The outline of this thesis is strongly related to the structure of the subgoals listed above. In Chapter 2, the relevant background knowledge is introduced, including related work, machine learning algorithms, evaluation criteria and statistical methods for evaluation. In Chapter 3, the GitHub dataset we used in this thesis is explained in detail, followed by description of basic characteristics of GitHub user activities. In Chapter 4, the construction of networks is described, along with fundamental network analysis. In Chapter 5, the main experiment of user behavior prediction is expounded step by step, including the prediction results, performance evaluation and analysis of the results. Finally, in Chapter 6, conclusions, discussion, and recommendations for further work are presented.

2 Background

In this chapter, essential background knowledge will be introduced. Section 2.1 will present the related work. Section 2.2 will briefly introduce machine learning algorithms, especially Random Forest algorithm. Afterwards, the criteria to evaluate the precision of user activity prediction used in this thesis will be given in Section 2.3. Lastly, Section 2.4 will describe the statistical method for significance test.

2.1 Related Work

At the beginning of this thesis, we will give some related works to user behavior analysis and prediction on OSNs. Jin et al. studied user behavior in OSNs from 4 various aspects [25], which can give us an overview of this problem. In most cases, researchers usually extract features from massive data and use machine learning methods to predict user activity in OSNs. For instance, Liu et al. proposed a convolutional click prediction model based on convolution neural network [26]. The model can extract local and global key features from the input with variable number of elements of online user activity data and it can be effectively applied to both single and sequential advertisements [26]. The experimental results in practice show that the convolution click prediction model has achieved good performance in click prediction of user activity [26]. Another important topic in this field is recommendation system. Yu et al. proposed a dynamic recurrent model for next basket recommendation [27]. They use the dynamic periodic neural network to extract the features of time series of purchase history and shopping interests of users [27]. The model can learn important features of the dynamic shopping interests of users from the data of user activity, and then provide reasonable commodity recommendation based on these features [27].

In addition to the data and characteristics of users, the network information can be also used for prediction. For instance, in [28], Lü and Zhou reviewed several current link prediction algorithms and emphasized the importance of physical perspective and network information. Besides, Lü et al. also proposed a network model to predict the possibility of the existence of links between users by using local network information [29]. Recently, Lee and Lim carried out a research about link prediction in multiple social networks [21]. User behavior and how the connections are built and maintained in multiple social networks are studied in this thesis. They found out that most users prefer to keep different connections in different OSNs, while retaining only a small group of common friends. They also achieved the prediction of future links by using the network information from multiple social networks. Though their contributions are in link prediction, rather than user activity prediction, their idea of using network analysis of social connections to help prediction inspired us impressively.

Since OSNs contain a large amount of information of social relationship which can

represent the user behavior in another aspect, it is likely that the neighbor information of OSNs will help to predict a user's activity. From the related work discussed above, we know that few researchers have considered the combination of machine learning methods and network theory, especially using the hidden network information as features for predicting user activity. Although current researchers have achieved outstanding successes in the direction of user activity study and prediction in OSNs, there is still room for improvement and further research.

2.2 Machine Learning Algorithms

2.2.1 Basic Description of Machine Learning

Due to the availability of data in diverse domains, many machine learning algorithms have been studied and designed for prediction on data. For instance, by learning users' purchase data, the recommendation algorithms can help users to find more suitable products in Amazon [30]. We can use machine learning algorithms to predict what kind of music the users may be interested in based on the music they have listened frequently or added into their favorite music lists [31]. Machine learning algorithms can be applied to estimate the sales of a company in the future based on its sales and global economic performance in the past [32].

According to different problems, there are three types of algorithms in machine learning: Supervised Learning, Unsupervised Learning and Reinforcement Learning [33]:

- Supervised learning is widely used to predict future events based on previous data. It has a function from the input mapping to the output. It can provide us predicted output based on given input data. Some examples of supervised learning algorithm are: Regression, Decision Tree, Random Forest, KNN Algorithm, Logistic Regression, Naive Bayes, etc [34].

- The goal of unsupervised learning is to explore the data and find internal structure. For example, it can identify customers with the same attributes or find the main attributes of customer groups to distinguish each other. Examples of unsupervised learning algorithm are: K-means Clustering Algorithm and Apriori Algorithm [34].

- Reinforcement learning is widely used for robots, games and navigation systems. The algorithm is designed for the problem that must learn from trial-and-error interactions within a given dynamic environment [35]. The famous Markov Decision Process is a classic example of reinforcement learning algorithm [34].

In this thesis, we aim to understand whether we could enhance the prediction of active level (classified by the number of activities contributed to software development) of a user on GitHub based on not only users' but also their neighbors' previous activities in GitHub social networks. This task can be considered as a classification problem, since we are more interested in predicting the active level of a user on GitHub, rather than focus on the specific number of user activities. Thus, we determined to apply Random Forest algorithm to predict the active level of a user on GitHub.

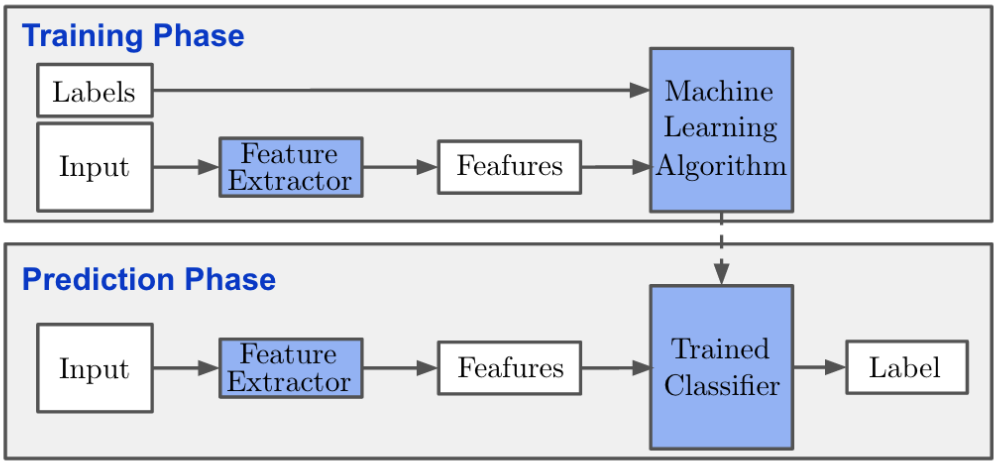


Figure 2.1: Two phases of classifiers using machine learning algorithms.

Figure 2.1 gives a general outline of how we use the machine learning algorithm in our work [36]. The first phase is training phase, where we input the features extracted from training dataset and their corresponding labels to the given machine learning algorithm. The second part is prediction, in which we use the trained classifier to predict the label of new input.

2.2.2 Random Forest

The machine learning algorithm used in this thesis is Random Forest algorithm. Decision tree is the basic element of Random Forest algorithm. Decision tree model is used for the process of classification or regression. According to the features, the data is divided into several sub-regions (sub-trees). Then, the sub-regions are divided recursively until a certain condition is satisfied, which is treated as a leaf node. Otherwise, continue to recursively divide the sub-regions. A major advantage of Decision Tree is that it is easy to explain. It can deal with the interaction between features without pressure, which is nonparametric. Thus, we do not have to worry about the outliers or whether the data is linearly separable or not [37][38].

Random Forest is a collection of decision trees which are formed by random method, and thus there is no correlation between the trees in Random Forest [37][38]. In order to classify new test data, each decision tree gives a classification according to its characteristics. After forming the forest, for a new input sample, each decision tree in the forest will judge separately which label the sample should belong to. And then, the Random Forest algorithm selects the most voted classification as the result of the overall classification (the weight of each decision tree should be also considered). Random Forest algorithm has a lot of advantages, such as better performance in the dataset, high training speed and simply to apply. It can handle very high-dimension data, which means there are a

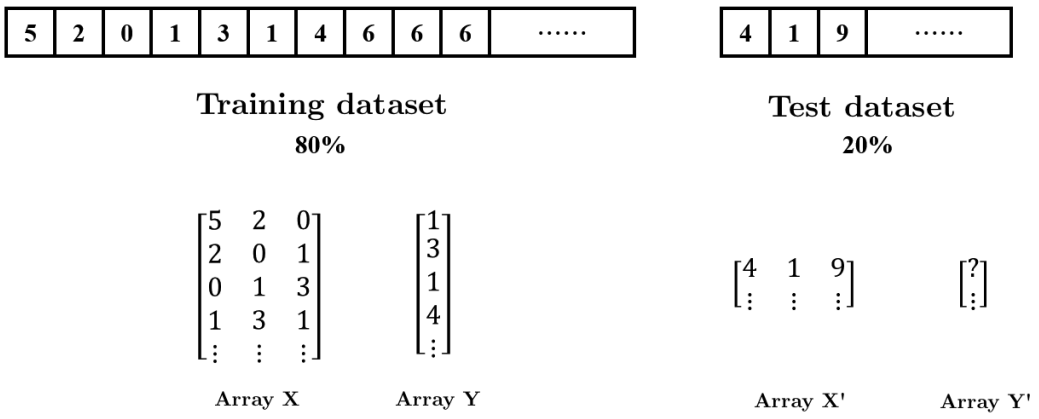


Figure 2.2: An example of prediction using machine learning algorithms.

large amount of features. Compared with Decision Tree algorithm, Random Forest can greatly reduce the over-fitting during the classification.

Figure 2.2 shows an example of prediction. In this example, we use the previous 3 bits to predict the current 1 bit. For training dataset, [5, 2, 0] in Array X are used as features to train the prediction of the corresponding result [1] in Array Y. After training, test dataset is used to test the accuracy of prediction. For example, to predict [?] in Array Y' using [4, 1, 9] in Array X'.

As with other classifiers, Random Forest takes two arrays as input: an array X (of size [n_samples, n_features]), holding the training samples, and an array Y, (of size [n_samples]), holding the training results of samples [39][40]. As to our data of time series of user activity, we divide the overall data into two parts: the first 80% of the data is assigned to be training dataset, while the other 20% of the data is chosen to be test dataset. One reason is that this division is classic and widely used in today's research. Besides, through this kind of division, we can make sure to have sufficient test data, meanwhile as much training data as possible. For the parameters of Random Forest algorithm, we apply the default preset of sklearn.ensemble.RandomForestClassifier in Python 3.5 [39][40][41], for example, the number of trees in the forest is 10.

2.3 Evaluation Criteria

In order to evaluate the performance of our prediction, some evaluation methods are given in this part, i.e. Accuracy, Precision, Recall and F1 score [42]. Before giving the definition of these 4 concepts, we first introduce the confusion matrix of binary classification in Table 2.1. In Table 2.1, we divide the prediction results into 4 types: TP, FN, FP and TN [43]. TP refers to the actual positive samples which are predicted as positive; FN means the actual positive samples which are predicted as negative; FP represents the actual negative samples which are predicted as positive; TN denotes the

Table 2.1: Confusion Matrix of Binary Classification [3]

	Predicted as Positive	Predicted as Negative
Actually is Positive	True Positive (TP)	False Negative (FN)
Actually is Negative	False Positive (FP)	True Negative (TN)

actual negative samples which are predicted as negative [43].

Accuracy indicates the proportion of the number of correctly predicted samples to the total number of samples [43]. It is defined as:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.1}$$

Precision, which is also called Positive Predictive Value, represents the proportion of the number of correctly predicted positive samples to the total number of samples which are predicted as positive [43]. It is defined as:

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Recall, which is also called True Positive Rate, refers to the proportion of the number of correctly predicted positive samples to the total number of samples which are actually positive [43]. It is defined as:

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

F1 score can be explained as the harmonic mean of precision and recall. It considers both the precision and the recall above, where the F1 score reaches its best at 1 and worst at 0 [43]. F1 score is defined as:

$$F1score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{2.4}$$

2.4 Statistical Significance Test

Statistical significance test is necessary for evaluating different prediction results in different cases. In this thesis, we have 10K targeted users. As discussion in Section 2.3, for each prediction, the evaluation criteria of all the 10K users will be reported, forming a set of prediction results. For example, aiming to analyze the prediction performance of case A and B, we can not just simply compare the average evaluation criteria in the two cases. In this case, a highly reliable method for statistical significance testing is introduced, which is so called Z-Test.

As to the statistical significance test, the first step is to make a certain hypothesis on the overall distribution. And then based on the calculation results, make a positive or negative decision for the hypothesis. For example, if we aim to test the difference of the mean number of experimental groups and controlled groups (μ_1 and μ_2), we should make a null hypothesis H_0 that there is no difference between μ_1 and μ_2. Afterwards, we should find out the probability p value of H_0 through statistical calculations. According

Table 2.2: Relationship between p value and H_0

p value	Probability H_0 is True	Degree of Difference
$p \leq 0.01$	Extremely Small	Extremely Significant
$p \leq 0.05$	Very Small	Significant
$p > 0.05$	Big	Insignificant

to the value of p, we can determine whether the null hypothesis H_0 is true or not. Table 2.2 shows the relationship between p value and H_0.

Z-Test is a widely used statistical significance test method, which is applicable if the data follow an approximate normal distribution. It is used to test the significance of difference between the average of two large groups, thereby to determine whether the two groups are significantly different from each other, which is totally suitable for our thesis. Z-Test is operated by comparing the Z value calculated by the distribution of two groups with the prescribed theoretical Z value to make the conclusion whether they are significantly different. The main steps of Z-Test are as follows [44]:

1. Set a null hypothesis H_0, which assume that there is no significant difference between the average of two sample groups.

2. Statistical calculations for Z value. Test the difference between the average of two groups in order to further determine whether the two groups are significantly different from each other. The Z value can be calculated as:

$$Z = \frac{\bar{X_1} - \bar{X_2}}{\sqrt{\frac{{S_1}^2}{n_1} + \frac{{S_2}^2}{n_2}}} \tag{2.5}$$

   where X_1 and X_2 denote the average of two groups, S_1 and S_2 denote the standard deviation of two groups, n_1 and n_2 denote the number of samples of two groups.

Table 2.3: Relationship of Z value, p value and Significance Test result

$\|Z\|$	p value (significance level)	Degree of Difference
≥ 2.58	≤ 0.01	Extremely Significant
≥ 1.96	≤ 0.05	Significant
< 1.96	> 0.05	Insignificant

3. Compare the calculated Z value with the prescribed theoretical Z value to infer the probability of occurrence. Look up on the probabilities form of standard normal distribution to find the corresponding p value. It is easy to make judgments based on the relationship of Z value, p value and significance test result, which is listed in Table 2.3. The critical value in the table is extracted from the standard normal probabilities form, which is attached in Appendix A. It is more intuitive for understanding and application. For instance, when the significance level is 0.05,

the critical Z value of two-tailed hypothesis is 1.96. If the Z value is larger than 1.96, that means the two groups are significantly different from each other.

3 GitHub

In this chapter, the GitHub dataset we used in this thesis will be described in detail. Section 3.1 will present the data we used in this thesis. Afterwards, Section 3.2 will elaborate basic characteristics of GitHub user activity and how we deal with the user activity.

3.1 Data Description

Recently, GitHub has grown to the world's leading social programming and code hosting platform, which has more than 20 million users, 57 million repositories, and 100 million pull requests [45][46]. Besides, GitHub is also a huge social network, a kind of online Open Source Software (OSS) community, where all the connections, activities, timestamp and content are recorded. It is an outstanding pioneering success for GitHub to introduce social networking into projects hosting platform. On GitHub, users can not only follow projects, but also follow other users to know more updated news about the projects and developers. This remarkably broaden the horizon of OSS.

For the data collection of GitHub, we used the database of GHTorrent. It is worth mentioning that GitHub provides a tool called REST API, which can access to the full dataset and retrieve all the interconnection data [47]. GHTorrent is considered to be a scalable, queriable, offline mirror of the data offered by GitHub REST API [45][48]. It contains all the raw data of GitHub and we can extract, archive and share queriable metadata through it [47]. Using this data, developers can easily get insight into the projects and the full process of OSS development on GitHub [47].

In this thesis, we focus on the deeper interconnection and understanding of user activity on GitHub. We collected various information during the past 3 years (from Oct 1st, 2013 to Oct 1st, 2016) of 10K GitHub users by using GHTorrent, and the data of approximately 4K projects related to these 10K developers. The data includes all the data of following relationship, the information of projects and their participants, as well as the whole data of pull request history of all the 10K users in the targeted 3 years.

At the beginning of the thesis, in order to obtain a general overview of GitHub data, we captured all the general information (before April 1st, 2016) of projects and developers on GitHub from GitHub API and existing database. Thereafter, we built two simple networks as shown in Table 3.1 and Table 3.2, the project network regarding how many developers connected to a common project, and the user network regarding how many projects a certain developer has participate in. Interestingly, an underlying fact regarding projects and developers is found: by April 2016, there are more than 34 million projects and about 13 million developers on GitHub, however, among these projects and developers, there are 12 million empty projects without any content and 5 million users never upload their code to any projects.

Table 3.1: Number of contributors per project in GitHub project network

Number of contributors per project	Number of projects	Percentage of projects
0	12,562,060	36.23
1	17,154,424	49.48
2	3,368,206	9.71
3-4	1,092,013	3.15
5-19	452,028	1.30
20-99	39,507	0.11
100-499	4,038	0.01
≥ 500	368	0.00
SUM	34,672,644	

Table 3.2: Number of projects per user in GitHub user network

Number of projects per user	Number of users	Percentage of users
0	5,251,582	39.77
1	4,138,212	31.34
2-4	2,359,851	17.87
5-9	814,726	6.17
10-49	579,912	4.30
50-499	58,504	0.44
500-5000	868	0.01
≥ 5000	41	0.00
SUM	13,203,696	

When collecting data from GHTorrent, first we randomly selected 10K users who is not "*inactive*"in the past 3 years. The word "*inactive*"here refers to the users that have no pull request during the targeted time. Hence, we selected 10K users who have at least one pull request in the past 3 years. Afterwards, all the projects that have connections with the 10K users are picked out, which includes the projects any one of the 10K users have participated in and contributed to. The information of more than 4K projects have been collected. This is because the empty projects and the users who have no activity nearly have no contribution for this project. Perhaps these empty projects and users are just mistakes or misuses, which will not have any impact on GitHub social network. What's more, all the pull request history of the 10K users during the targeted 3 years are stored, especially the timestamp of pull request. For each user, we acquired a time series after choosing a suitable time interval. In this way, all the data of user activity on GitHub we will use in next steps have been collected.

3.2 Basic Characteristics of GitHub User Activity

The user activity on GitHub that interests us most is the pull request. As an indispensable part of GitHub, the pull request starts discussion about various versions of submitted code. Through it, a GitHub user can make changes and suggestions to the

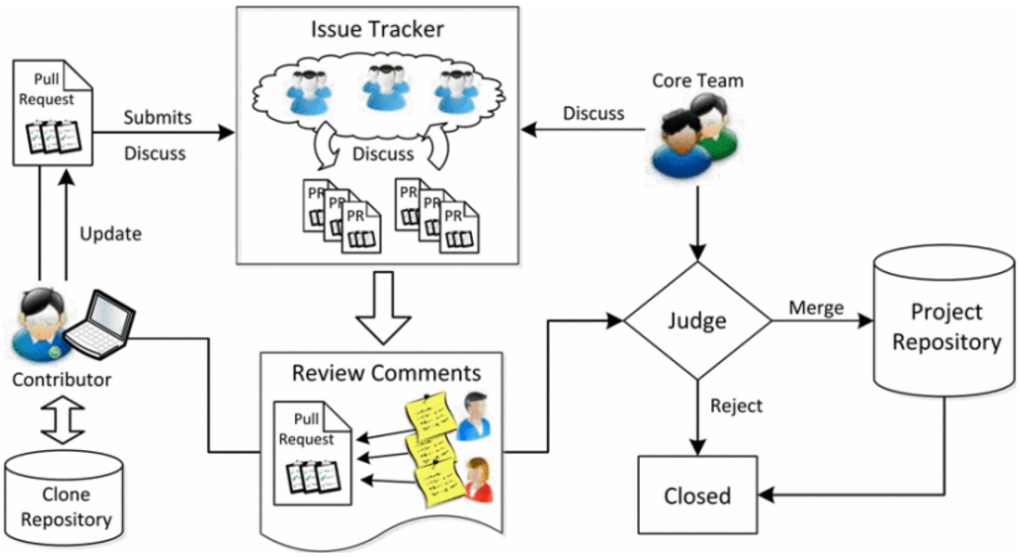


Figure 3.1: An overview of pull request [1].

source code of others' project, the owner of the project can easily see these changes and review them. If a pull request is accepted and merged into master branch, each users of GitHub can see the changes of new version. The pull request is a way that allows developers to collaborate more easily. It provides a user friendly web interface to discuss the changes before the proposed changes are merged into the formal project. The pull request is not just a simple notification, but rather a specialized forum for discussing the submitted functions. Figure 3.1 shows us a brief overview of pull request [1]. If there are any questions about the changes, the development team will feedback in the pull request, or even push new commits for the feedback.

As discussed in Section 2.2, we focus more on predicting the active level of a user's pull requests, which are classified by the number of activities contributed to software development, rather than focus on the specific number of pull requests in the future. Thus, first of all, we need to classify the number of user's pull requests into 4 labels representing the active level of each user in each given time slot according to its distribution. As shown in Table 3.3, the 4 active levels are: Inactive, Slightly Active, Moderate Active and Highly Active. When classifying the 4 labels, we should make sure the proportion of each label be as balanced as possible, though the large proportion of Inactive State is unchangeable.

As discussed in Section 2.3, the evaluation criteria we used in the thesis are accuracy and F1 score, which can be easily extended to the case of multiclass classification. For instance, the F1 score of multiclass classification can be represented by the weighted average of each label's F1 score. Furthermore, in the main text, we show the results by using accuracy to evaluate the performance of prediction. The prediction results when

Table 3.3: Classification of 4 active levels of user's pull request (time interval = 1 week)

		Number of pull requests	Proportion
Label 0:	Inactive	0	82.28 %
Label 1:	Slightly Active	1-2	6.05 %
Label 2:	Moderate Active	3-6	5.51 %
Label 3:	Highly Active	≥ 7	6.16 %

using F1 score as the evaluation criterion will be shown in Appendix B.

Additionally, the classified data is not as balanced as we expected. Though the amount of Label 1 (Slightly Active), Label 2 (Moderate Active) and Label 3 (Highly Active) are equally matched, the proportion of Label 0 (Inactive) is far larger. Due to the unbalanced data, the label of larger amount will dominate the calculation of accuracy and F1 score. Thus, with the aim of solving this issue, for each user, we decide to calculate the accuracy and F1 score for each label, in order to gain insight into the relationship between the overall performance and the performance of each active level. What's more, we would like to pay more attention to the prediction accuracy when a user is highly active, which makes more sense to our real life in OSNs.

To sum up, after the prediction of user's active level on GitHub, for each user, we report the following evaluation criteria:

Accuracy	: ACC
Accuracy for each active level	: ACC(0), ACC(1), ACC(2), ACC(3)
F1 score (weighted-average)	: F1score
F1 score for each active level	: F1score(0), F1score(1), F1score(2), F1score(3)

4 Network Construction

In this chapter, the construction of networks is described, along with some fundamental network analysis. In order to gain deeper insight into users' active level on GitHub, we will build 3 kinds of relationship networks of GitHub, i.e. Follower-Followee Network, Collaboration Network and Similarity Network.

Section 4.1 will introduce Follower-Followee Network which denotes the following relationship on GitHub. Then, in Section 4.2, Collaboration Network representing collaboration relationship on GitHub will be described. Finally, Section 4.3 will present the Similarity Network we built in this thesis, which indicates the similarity of user activity patterns on GitHub.

4.1 Follower-Followee Network

The following connections on GitHub social network, makes up a basic and essential part of the relationship of GitHub users. Users can choose to keep track of some specific developers to stay updated in time. Using the data collected from GHTorrent, we constructed network G_1 which we called Follower-Followee Network, representing the following relationship on GitHub. As we have mentioned before, a user follows other developers for the purpose of receiving timely updates about events of the followees. Regarding Follower-Followee Network G_1, each node denotes a user on GitHub and the directed links indicate the following relationship. Statistical characteristics of G_1 is shown in Table 4.1. G_1 is a directed, unweighted network. The low link density and clustering coefficient indicate that the Follower-Followee network is exceedingly sparse. Figure 4.1 illuminates the distribution of in-degree, out-degree and total degree of Follower-Followee Network on GitHub. All of them follow power-law distribution with almost the same exponent.

In Figure 4.2, we show the relationship between the number of followers (degree in G_1) and the number of pull requests (the number of user activities) for each user. We observe a positive relationship between the two variables, especially for the degree larger than 10. This has been validated in [2], in which user behavior may have influence on their neighbors. Therefore, Figure 4.2 inspires us that using the information of Follower-

Table 4.1: Statistical characteristics of networks

	Nodes	Links	Attributes	Clustering Coefficient	Link Density
G_1	10,000	53,814	directed, unweighted	0.1380	5.3819e-04
G_2	10,000	5,441,052	undirected, weighted	0.4511	1.0883e-01
G_3	10,000	49,995,000	undirected, weighted	1.0000	1.0000

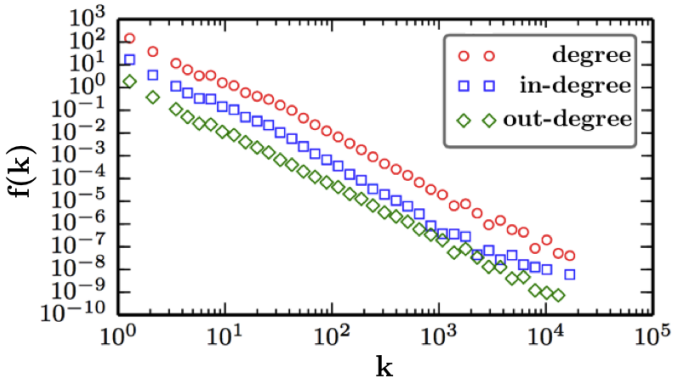


Figure 4.1: Degree distribution of Follower-Followee Network [2].

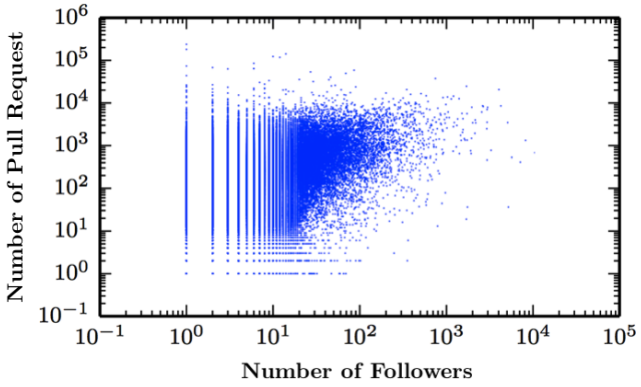


Figure 4.2: Number of pull requests for each user against the number of followers in Follower-Followee Network [2].

Followee Network may have a positive impact upon user behavior prediction on GitHub. Therefore, it is an exceedingly critical branch to go deeper and find out whether the network information of Follower-Followee Network will have any help on the prediction of user active level on GitHub. And if so, what kind of help will the Follower-Followee Network offer.

4.2 Collaboration Network

Another important relationship on Github is collaboration, which implies two users work on a common repository and contribute codes to a same project. We constructed the Collaboration Network G_2 by using the data from GHTorrent. Regarding Collaboration Network G_2, each node denotes a user of GitHub and the undirected links indicate the collaboration relations. Different from the Follower-Followee Network G_1, the links of G_2 are weighted, denoting the number of common projects the two users may contribute to on Github in the past 3 years.

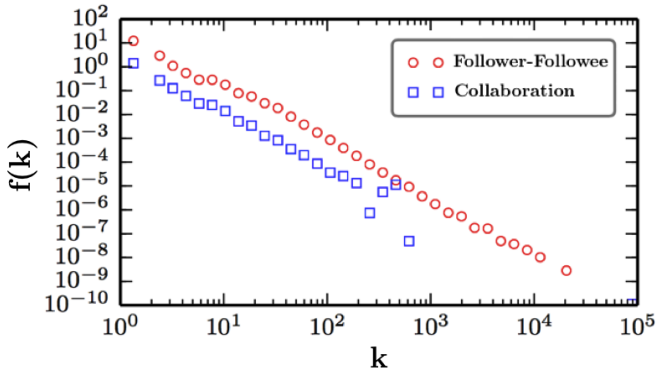


Figure 4.3: Degree distribution of Collaboration Network [2].

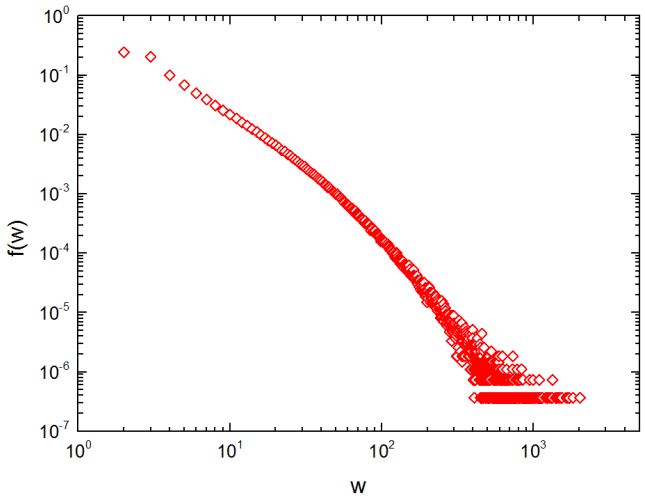


Figure 4.4: Weight distribution of Collaboration Network.

Statistical characteristics of G_2 is shown in Table 4.1. We noticed that the clustering coefficient of G_2 is considerably larger than that of G_1, revealing that there is more clustering in Collaboration Network on GitHub as compared with Follower-Followee Network. This can be explained by the formation of this network, as people in the same group are more inclined to know each other. Figure 4.3 gives the degree distribution of G_1 and G_2. Interestingly, both of these two networks show power-law distribution with almost the same exponent. In addition, we plot the weight distribution of Collaboration Network in Figure 4.4, representing the number of common projects the two users have collaborated. As shown in Figure 4.4, the weight of Collaboration Network follows an approximate power-law distribution.

Figure 4.4 shows that the collaboration relationship between users is heterogeneous, indicating there are strong and weak ties in G_2. As it is studied before, the strength of the relationship may show different impact in user behavior [49][50]. For example,

if two users have experienced more common projects, they may have closer connections even more similar user behavior on social networks. Besides following connection, the collaboration relationship should also be taken into account for user activity prediction [51].

4.3 Similarity Network

In order to gain deeper insight into the prediction of user activity on GitHub, the similarity of user activity is studied in this thesis. To analyze the similarity of user activity over time, firstly, we selected a suitable time interval, 1 week (7 days), to achieve the time series of user activity (pull request) of each user. The time series of each user can also be deemed as an m-dimension vector $v(\vec{t})$, which represents the user activity over time. Each coordinate of $v(\vec{t})$ denotes the number of pull requests in each time slot divided by the 1-week time interval. Inspired by [52], to measure the similarity of users' activity patterns, we introduced one of the common measures "*CosineSimilarity*". The definition of cosine similarity is shown as follows:

$$Similarity = cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{\left\|\vec{A}\right\|_2 \left\|\vec{B}\right\|_2} = \frac{\sum_{i=1}^{n} \vec{A}_i \vec{B}_i}{\sqrt{\sum_{i=1}^{n} \vec{A}_i^2} \sqrt{\sum_{i=1}^{n} \vec{B}_i^2}} \tag{4.1}$$

where $\vec{A}_i$ and $\vec{B}_i$ are elements of vector $\vec{A}$ and $\vec{B}$ respectively, $\left\|\vec{A}\right\|_2$ and $\left\|\vec{B}\right\|_2$ denote the Euclidean norm of $\vec{A}$ and $\vec{B}$ respectively.

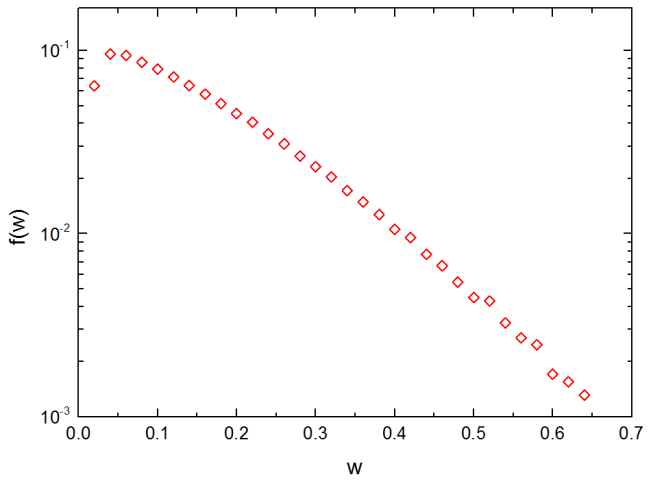


Figure 4.5: Weight distribution of Similarity Network.

After the definition of similarity measure, we use the data collected from GHTorrent to construct network G_3 which is called Similarity Network, representing the similarity of user activity on GitHub. As mentioned before, the similarity of two users' activity

patterns can be reflected by the cosine similarity metric. The similar individuals may be inclined to have following connection in G_1 or collaboration relationship in G_2.

Statistical characteristics of Similarity Network G_3 is shown in Table 4.1. Each node of G_3 denotes a user of GitHub and the weighted links indicate the level of activity similarity between the two users. Different from the other two networks G_1 and G_2, Similarity Network is a complete network. The weight distribution of Similarity Network is shown in Figure 4.5. For the weight of links in G_3, it ranges from 0 to 1, since all the elements of time series vector of each user are non-negative. The larger weight means the two connected users are more similar to each other in users' activity patterns.

5 Methods and Results

This chapter will give the experiments and results of our method, including the baseline which only use the users' previous activities as features and other conditions where using the neighbor information as features for prediction. Section 5.1 will present the baseline of this thesis, where only the past user activity data is considered. In Section 5.2, the neighbor information from Follower-Followee Network will be taken into consideration for the prediction, including the past user activity data itself. Similarly, the neighbor information from Collaboration Network will be used as features to predict user activity in Section 5.3. Afterwards, as to Section 5.4, we will use neighbor information from two-dimension network consisting of Follower-Followee Network and Collaboration Network. Finally, Section 5.5 will present the discussion of prediction results.

5.1 Baseline

In this section, the baseline of this thesis is presented, where only the previous activities of a user are considered. We aim to investigate how many previous weeks should be taken into account when predicting the active level of a user. Define n denotes the number of previous weeks should be considered. What's more, the reason that we choose the time interval of 1 week is also discussed in this section.

5.1.1 Results

The results of using user previous time series of activities are given in Figure 5.1, which show how the prediction accuracy changes with various time period of user activity information considered in the features. We find that all the accuracy values increase with the increase of number of weeks considered before $n = 4$. After that, the accuracy remains almost stable. Taking all the accuracy values into consideration (i.e. ACC, ACC(0), ACC(1), ACC(2), ACC(3)), it can be seen that considering last 4 weeks of user activity can give a better prediction results. In addition, we also use F1 score to evaluate the prediction accuracy, the results are given in Appendix B, Figure B.1, which show similar trend as ACC value. Regarding the performance of $n = 4$, the overall accuracy (ACC) is 76.25%. Besides, prediction accuracy when a user is highly active (ACC(3)) equals to 65.26%, which can be considered as an impressive outcome. Therefore, in the following study, 4 weeks of user activity information will be used as the baseline for the other prediction methods.

5.1.2 Evaluation and Explanation

In Section 5.1.1, we get a conclusion that considering 4 weeks of user activity can give a better prediction of user activity. In this part, we will give further explanation of this

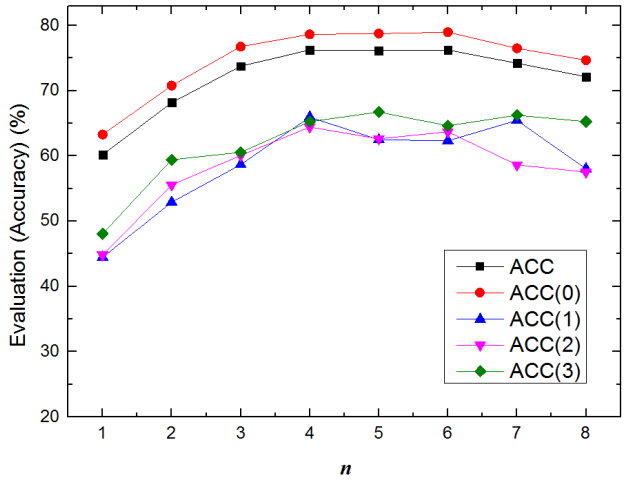


Figure 5.1: Results of user activity prediction by using the previous activities of a user.

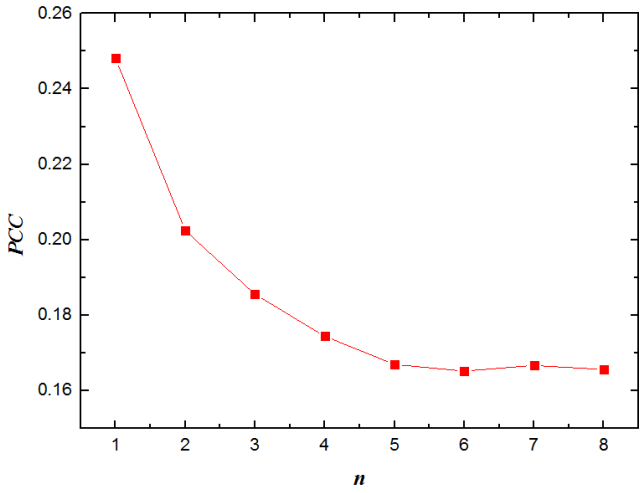


Figure 5.2: The average PCC between users' original time series and the n-position-shifted time series to the number of weeks shifted.

results based on time series analysis.

For a user, we shifted the time series for n positions along the timeline and analyze the Pearson Correlation Coefficient (PCC) between the original time series and the n-position-shifted time series. The PCC indicates how strength the linear relationship between the information of previous n weeks and that of the current moment to a certain degree. We plot the curve of the average PCC to the number of weeks shifted, as shown in Figure 5.2. It can be seen that, when considering less than 4 weeks, there is a dramatically declined reduction in PCC. While, when the number of shifted weeks is

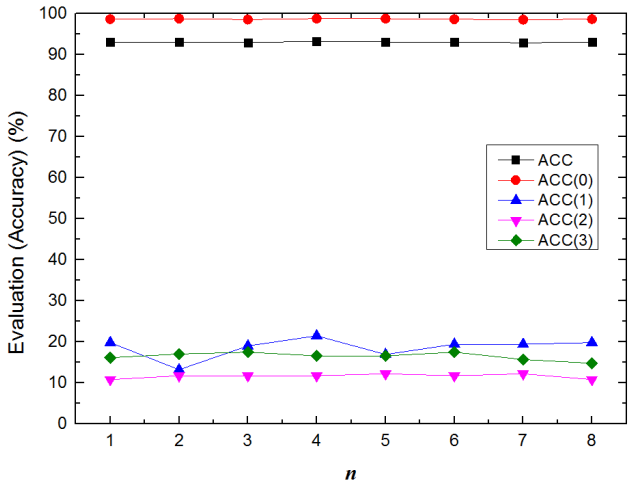


Figure 5.3: Results of user activity prediction by only using the previous activities of a user when choosing 1 day as time interval.

larger than 4, the PCC changes not too much and tend to be stable, reaching a figure of 0.17. Figure 5.2 could be a good explanation for the issue that why previous 4 weeks (n = 4) can give a better prediction in Figure 5.1.

In our work, we choose 1 week as the interval of a user's activity time series. To select a suitable time interval is a crucial task. On the one hand, if the time slot is set too small, there could be no activity in the vast majority of the time slots, which may lead to serious overfitting during the prediction. On the other hand, if the time slot is set too large, there could be less data of time series, which may result in severe inaccuracy of the prediction. What's more, we should focus more on the time interval which is more intuitive and makes more sense to our real life in OSNs, such as 1 day, 1 week or 1 month.

As for the case of choosing 1 day as the time interval, we noticed that about 93.22% of all the time interval has no user activity happened. It indicates that the dataset in this situation is exceedingly unbalanced. To demonstrate this, the results of prediction in this condition are shown in Figure 5.3. Although the overall accuracy is about 93.04%, the prediction accuracy for the other active levels is very low. It is because the proportion of inactive label in overwhelmingly larger than the other 3 labels, making the inactive label dominate the prediction result.

We also carried out similar time series analysis and plot the curve of the average PCC to the number of days shifted when time interval is chosen as 1 day in Figure 5.4. It can be seen that in this condition, the PCC sustains intensely low and fluctuated slightly up and down at around 0.07. This result also justifies that 1 day can not be a suitable time interval for predicting a user's active level in this thesis.

As for the case of choosing 1 month as time interval, the prediction accuracy of user activity will become intensely low. This is because that if we set 1 month as the time interval to obtain time series during the targeted 3 years, for each user the time series

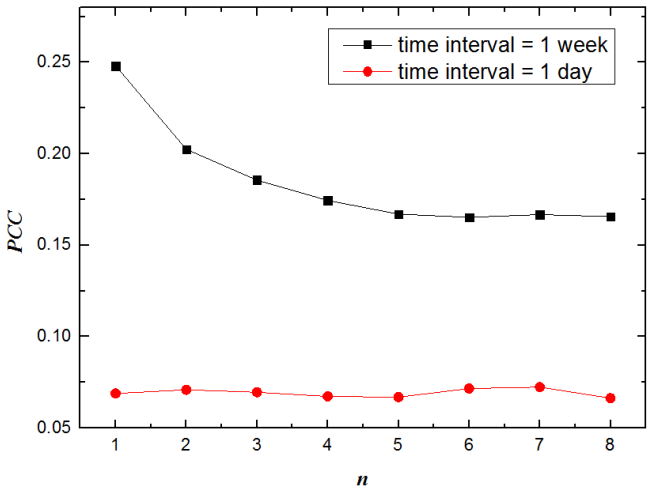


Figure 5.4: The average PCC between users' original time series and the n-position-shifted time series to the number of days shifted.

will only contain 36 elements, resulting in exceedingly inaccuracy.

Besides, since GitHub is widely used by the people who are programmers, researchers, engineers and students, the working period of these group of users might play a vital role in selecting the interval of time series. They work with a period of 1 week. Furthermore, in our real life, as to the projects, most of the workload is arranged by working weeks in the plan, which lists the tasks and schedule of progress in each week. Additionally, the prediction performance is very impressive when choosing 1 week as the time interval. Therefore, choosing 1 week as the time interval of a user's activity time series is a suitable decision. This is also the reason why we choose 1 week as the time slot to build up the similarity network in Section 4.3.

5.2 Using Neighbor Information from Follower-Followee Network

In this section, the neighbor activity information from Follower-Followee Network will be take into consideration during the prediction, besides the user activity data. Figure 5.5 gives a schematic diagram of an ego network in Follower-Followee network. First of all, we use the user activity during the last 1 week of all the neighbors of a user in Follower-Followee Network the prediction of user activity based on the baseline. Then, we optionally use the same information of the most active m_1 neighbors, rather than all the neighbors this time. In this way, we can find out the most suitable number of neighbors we should consider when using the neighbor activity information from Follower-Followee Network into prediction. What's more, we carry out the prediction to figure out how many weeks should be taken into account when using the number of activities of neighbors.

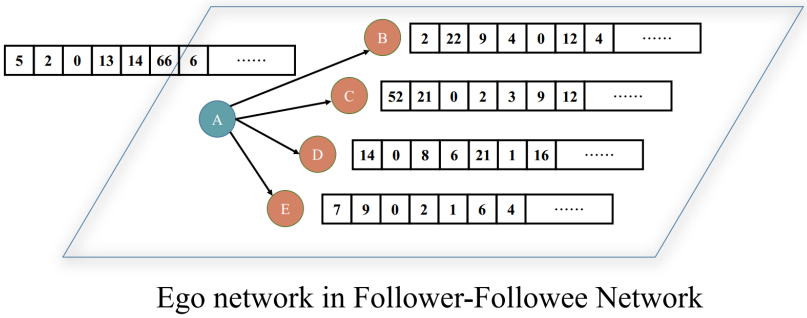


Figure 5.5: An example of ego network in Follower-Followee Network on GitHub. Each number in the time series denotes the number of activities of a user in 1 week. The directed links represent the following relationship in GitHub social network.

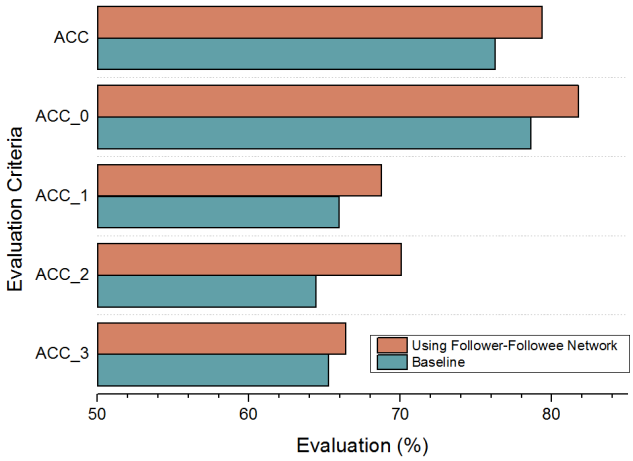


Figure 5.6: The comparison of prediction accuracy by using neighbor activity information from Follower-Followee Network and the baseline.

5.2.1 Results

First of all, we use the number of activities during the last 1 week of all the neighbors of a user in Follower-Followee Network as features for the prediction of user activity, based on the baseline. The results are shown in Figure 5.6, which gives a comparison of this method and the baseline. Compared with the baseline, the prediction performance is improved greatly when using Follower-Followee Network: the overall accuracy has reached 79.37%, indicating a rise of 4.10%. It is also demonstrated by Z-Test that the result is significant at $p < 0.01$, which means the information of the number of activities during the last 1 week of all the neighbors of a user in Follower-Followee Network can significantly improve the prediction of a user's active level on GitHub.

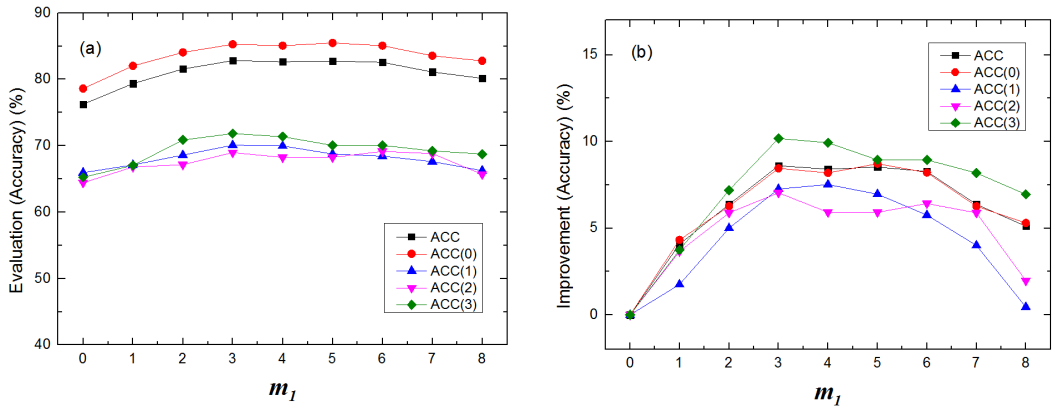


Figure 5.7: Results of user activity prediction by using the number of activities during the last 1 week of the most active m_1 neighbors of the user in Follower-Followee Network based on the baseline. (a) Accuracy; (b) Improvement.

Furthermore, in Follower-Followee Network, what would happen if we do not use the information of all the neighbors of a user, but rather selectively use the neighbor activity information of part of the neighbors, such as the most active neighbors. Thus, in the next step, we use the number of activities during the last 1 week of the most active m_1 neighbors of a user in Follower-Followee Network, based on the baseline. The results are given in Figure 5.7. It can be seen that the most suitable number of neighbors we should consider is $m_1 = 3$. When considering less than 3 neighbors, as the number of considered neighbors increases, the accuracy of prediction also goes up; However, when beyond 3 neighbors, the accuracy remains stable and then drops down very slightly. Moreover, the similar characteristic of F1 score is illustrated in Appendix B, Figure B.4 and Figure B.5.

Regarding the case of $m_1 = 3$, the overall accuracy is 82.81%, improved by 8.60% compared with the baseline. Besides, prediction accuracy equals to 71.83% when a user is highly active, indicating a rise of 10.07%. It is also demonstrated by Z-Test that the result is significant at $p < 0.01$, which means the number of activities during the last 1 week of the most active 3 neighbors of a user in Follower-Followee Network can significantly improve the prediction of a user's active level on GitHub.

What's more, we would like to go even deeper into Follower-Followee Network. Previously, we used the information of the number of activities during the last 1 week of the most active 3 neighbors of a user, however, what would happen if we consider more than 1 week of these neighbors? In order to solve this problem, we use the number of activities during the last t_1 weeks of the most active 3 neighbors of a user in Follower-Followee Network. The results are shown in Figure 5.8. When considering more than 1 week of the neighbors, as the number of considered weeks increases, the accuracy of prediction drops down gradually. Moreover, the similar characteristic of F1 score is illustrated in Appendix B, Figure B.6 and Figure B.7. The results demonstrate that the information of the number of activities during the last 1 week of the most active 3 neighbors of a user in Follower-Followee Network indeed can significantly improve the prediction accuracy

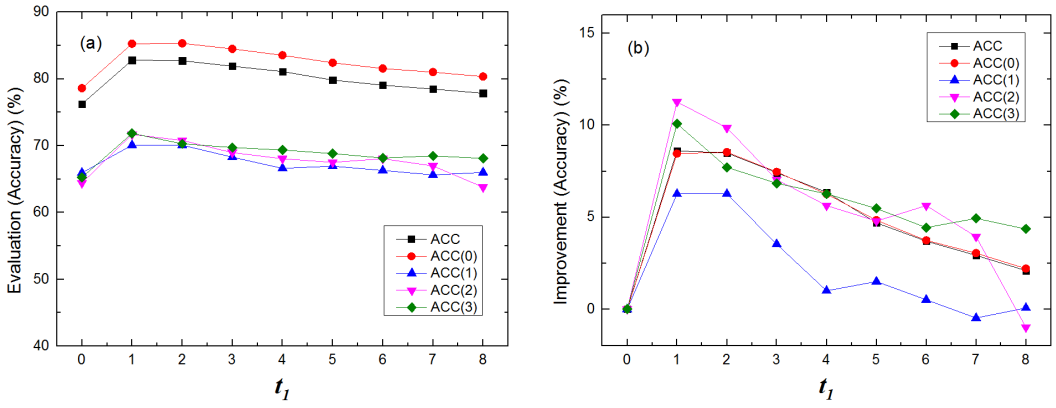


Figure 5.8: Results of user activity prediction by using the number of activities during the last t_1 weeks of the most active 3 neighbors of the user in Follower-Followee Network based on the baseline. (a) Accuracy; (b) Improvement.

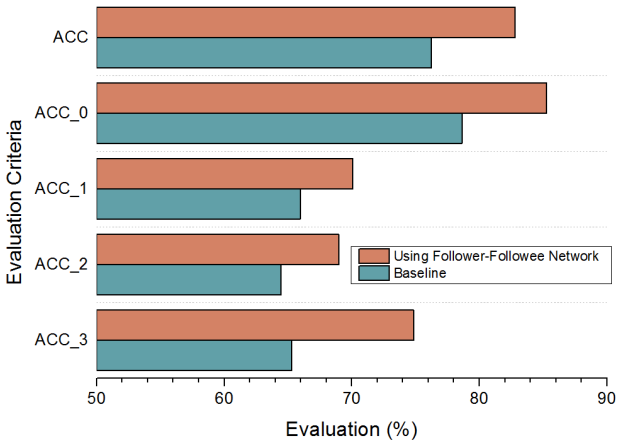


Figure 5.9: The comparison of prediction accuracy by using neighbor information of Follower-Followee Network and the baseline.

of a user's active level on GitHub.

5.2.2 Evaluation and Explanation

As discussed in Section 5.2.1, using the information of the number of activities during the last 1 week of the most active 3 following neighbors as features can gain better prediction of user activity. Figure 5.9 shows the comparison of prediction accuracy by using neighbor information from Follower-Followee Network and the baseline. In this part, we will analyze the time series to gain more insight into the PCC.

First of all, the average PCC of activities between a user and the user's most active m_1 neighbors is calculated, as shown in Figure 5.10(a). It can be seen from the diagram

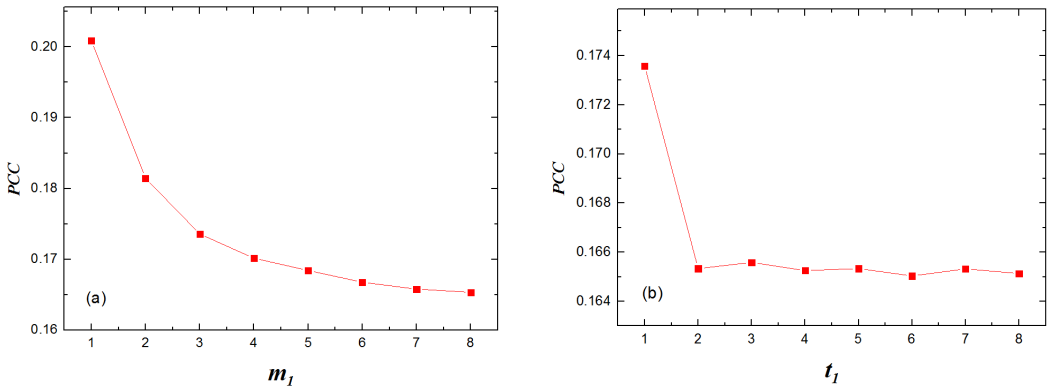


Figure 5.10: Average PCC: (a) of activities between a user and the user's most active m_1 neighbors; (b) between the time series of a user and t_1-position-shifted time series of the user's most active 3 neighbors.

that the PCC decreases gradually with the increase of number of neighbors, who are listed in descending order of the number of activities. Although Figure 5.10(a) may not expound why choosing the most active 3 neighbors can gain better performance clearly, at least it can be a good explanation for the judgment that less active neighbors are more likely to have less impact upon user activity prediction compared with active neighbors.

What's more, for a user, we also shifted the time series for t_1 positions along the timeline and analyze the PCC between the time series of a user and t_1-position-shifted time series of the user's most active 3 neighbors. The average PCC is computed, as shown in Figure 5.10(b). It is obvious that two users will have more PCC when we only consider the previous activity in last 1 week. If more weeks are taken into account, the PCC will drop down speedily and stay stable at a low level. It can be justification of applying the information of the most active 3 neighbors in last 1 week.

5.3 Using Neighbor Information from Collaboration Network

In this section, the neighbor activity information from Collaboration Network will be used as features in the prediction. Figure 5.11 gives a schematic diagram of an ego network in Collaboration Network on GitHub. First of all, we use the information of the number of activities during the last 1 week of all the neighbors in Collaboration Network, according to the baseline. Afterwards, we selected the closest m_2 neighbors of the user who has the most collaboration experience with the user, namely, the neighbors with largest link weight in Collaboration Network. Through this, we can find out the most suitable number of neighbors we should consider when using the neighbor activity information from Collaboration Network to predict a user's active level. Moreover, the prediction will be carried out to point out how many weeks should be taken into account when using the information of the number of activities of collaboration neighbors.

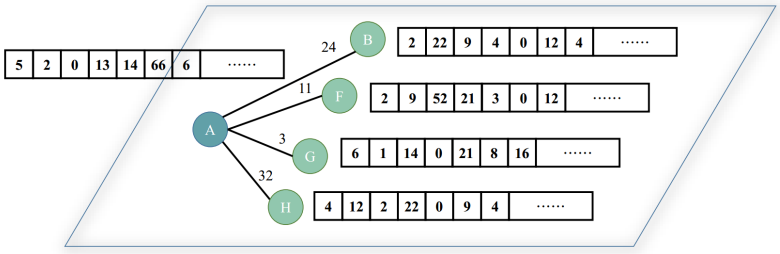


Figure 5.11: An example of ego network in Collaboration Network on GitHub. Each number in the time series denotes the number of activities of a user in 1 week. The undirected links represent the collaboration relationship in GitHub social network. The weight of links indicates number of common projects the two users have participated together.

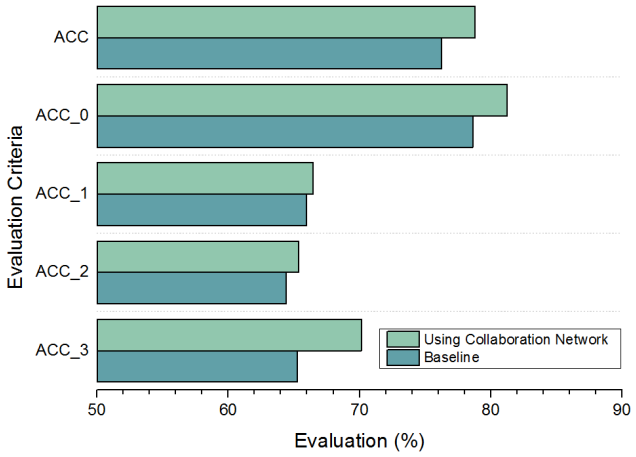


Figure 5.12: The comparison of prediction accuracy by using neighbor activity information from Collaboration Network and the baseline.

5.3.1 Results

First of all, we use the number of activities during the last 1 week of all the neighbors of a user in Collaboration Network as features for the prediction of user activity, based on the baseline. The results are shown in Figure 5.12, which gives a comparison of this method and the baseline. Compared with the baseline, the prediction performance is improved greatly when using Collaboration Network: the overall accuracy has reached 78.79%, indicating a rise of 3.33%. It is also inferred by Z-Test that the result is significant at $p < 0.01$. In other words, the information of the number of activities during the last 1 week of all the neighbors of a user in Collaboration Network can significantly improve

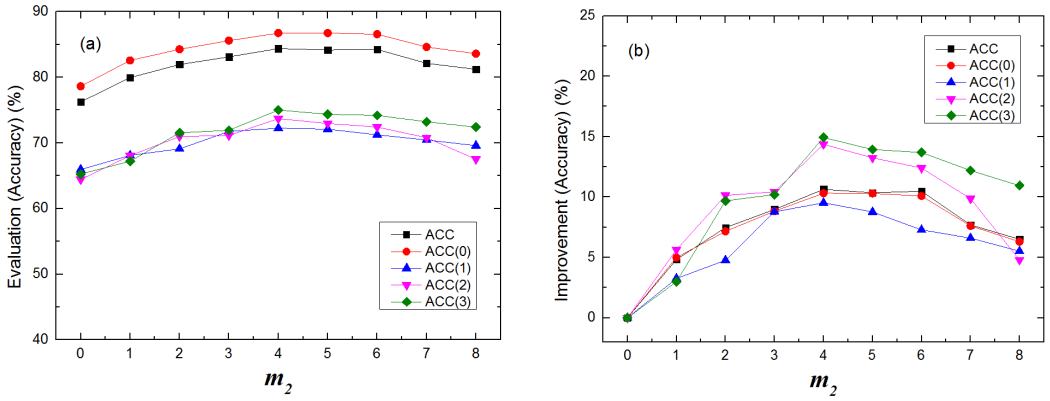


Figure 5.13: Results of user activity prediction by using the number of activities during the last 1 week of the closest m_2 collaboration neighbors of the user in Collaboration Network based on the baseline. (a) Accuracy; (b) Improvement.

the prediction of a user's active level on GitHub.

Similarly, in Collaboration Network, we also plan to explore the performance when we do not use the information of all the neighbors of a user, but rather optionally select the information of part of the neighbors, such as the closest collaboration neighbors who have experienced the most collaboration with the user. Thus, the following step, in order to better predict the active level of a user's behavior, we use the information of the number of activities during the last 1 week of the closest m_2 collaboration neighbors of a user in Collaboration Network, based on the baseline. The prediction results are shown in Figure 5.13. It can be seen that the most suitable number of neighbors we should consider into the prediction is $m_2 = 4$. When considering less than 4 neighbors, as the number of considered neighbors increases, the accuracy of prediction also goes up; However, when more than 4 neighbors are taken into account, the accuracy remains stable and then drops down very slightly. The similar characteristic of F1 score is illustrated in Appendix B, Figure B.10 and Figure B.11.

As for the case of $m_2 = 4$, the overall accuracy is improved by 10.64% compared with the baseline, reaching a figure of 84.36%. In addition, when a user is highly active, the prediction accuracy equals to 75.01%, indicating a rise of 10.07%. It is also demonstrated by Z-Test that the result is significant at $p < 0.01$, which means the information of the number of activities during the last 1 week of the closest 4 collaboration neighbors of the user in Collaboration Network can significantly improve the prediction of a user's active level.

What's more, we would like to go even deeper in Collaboration Network. By now, we have used the information of the number of activities during the last 1 week of the the closest 4 collaboration neighbors of a user, however, what would happen if we consider more than 1 week of these neighbors? In order to solve this problem, we use the information of the number of activities during the last t_2 weeks of the closest 4 collaboration neighbors of a user in Collaboration Network. The results are shown in Figure 5.14. We can conclude the most suitable number of weeks we should consider of a user's neighbors

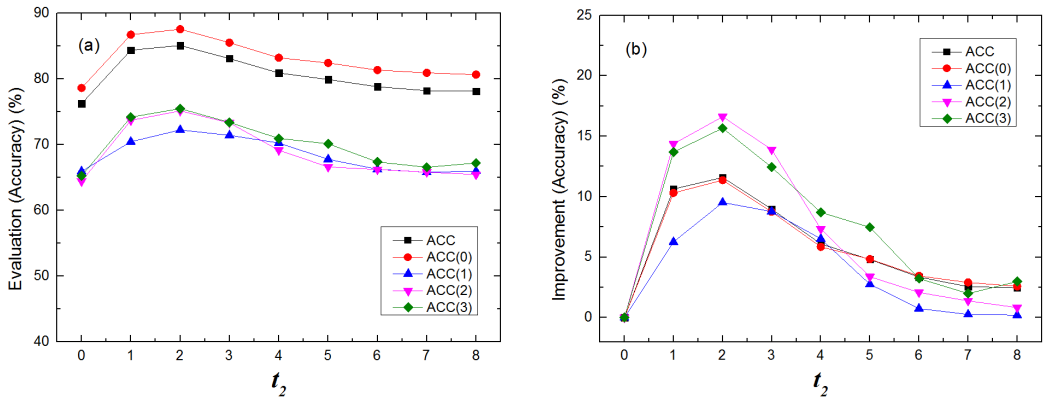


Figure 5.14: Results of user activity prediction by using the number of activities during the last t_2 weeks of the closest 4 collaboration neighbors of a user in Collaboration Network based on the baseline. (a) Accuracy; (b) Improvement.

is 2 weeks ($t_2 = 2$). When considering less than 2 week of the neighbors, as the number of considered weeks increases, the accuracy of prediction goes up. While the accuracy drops down gradually if more than 2 weeks of neighbors' activities is considered. Moreover, the similar characteristic of F1 score is demonstrated in Appendix B, Figure B.12 and Figure B.13.

Regarding the case of $m_2 = 4$ and $t_2 = 2$, the overall accuracy is 85.09%, improved by 11.59% compared with the baseline. Besides, prediction accuracy equals to 75.49% when a user is highly active, indicating a rise of 15.67%. It is also demonstrated by Z-Test that the result is significant at $p < 0.01$, which means the information of the number of activities during the last 2 weeks of the closest 4 collaboration neighbors of a user in Collaboration Network can significantly improve the prediction accuracy of a user's active level on GitHub.

5.3.2 Evaluation and Explanation

As discussed in Section 5.3.1, we can gain better prediction of user activity if the information of the number of activities during the last 2 week of the closest 4 collaboration neighbors of a user is used as features. Figure 5.15 shows the comparison of prediction accuracy by using neighbor information of Collaboration Network, using neighbor information of Follower-Followee Network and the baseline. In this part, we will provide further explanation of the results based on network and time series analysis.

Aiming to understand the issue deeper, we would like to gain more insight into the PCC between Collaboration Network and Similarity Network. The PCC between the two networks refers to the PCC between the weight of links in the two networks. As shown in Figure 5.16, there might be varied possibility of similarity when two users have less collaboration experience. While, two users are more likely to behave similarly with each other if they have collaborated more common projects in Collaboration Network. It demonstrated that the collaboration relationship on GitHub might have contribution

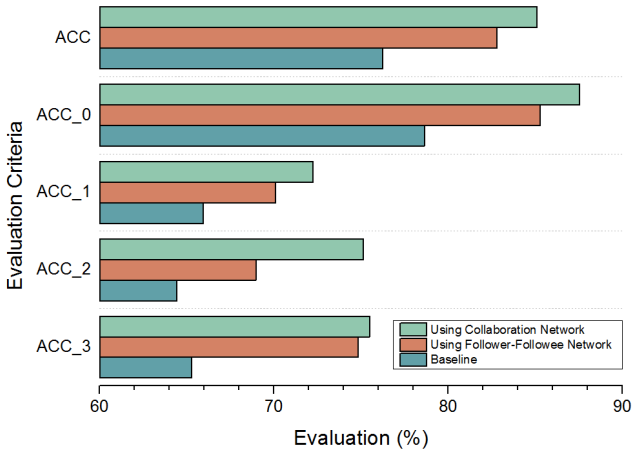


Figure 5.15: The comparison of prediction accuracy by using neighbor information from Collaboration Network, Follower-Followee Network and the baseline.

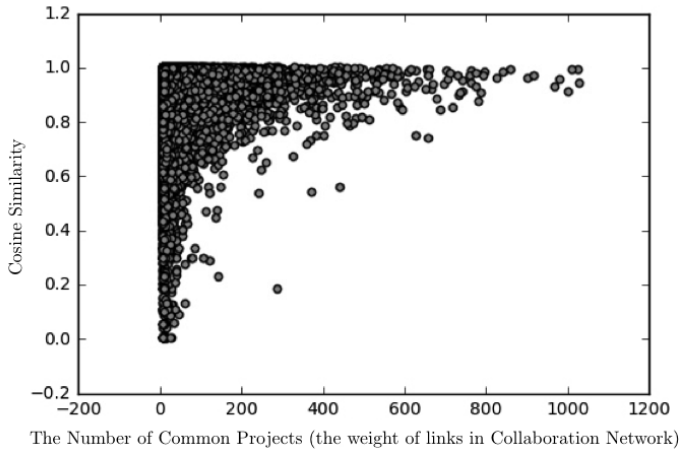


Figure 5.16: The PCC between the weight of links in Collaboration Network and Similarity Network.

to similarity of two users to some extent. In other words, it could be an explanation to expound the judgment that neighbor information of Collaboration Network has great help in predicting a user's active level.

For a user, we also shifted its time series for t_2 positions along the timeline and analyze the PCC between the time series of a user and t_2-position-shifted time series of the user's closest 4 collaboration neighbors. The average PCC is computed, as shown in Figure 5.17. It is evident that two users will have more PCC when we only consider the activities in last 2 weeks. If more weeks are taken into account, the PCC will drop down steeply and stay stable at a low level. It could be a justification that using the information of the closest 4 collaboration neighbors in last 2 weeks will obtain better

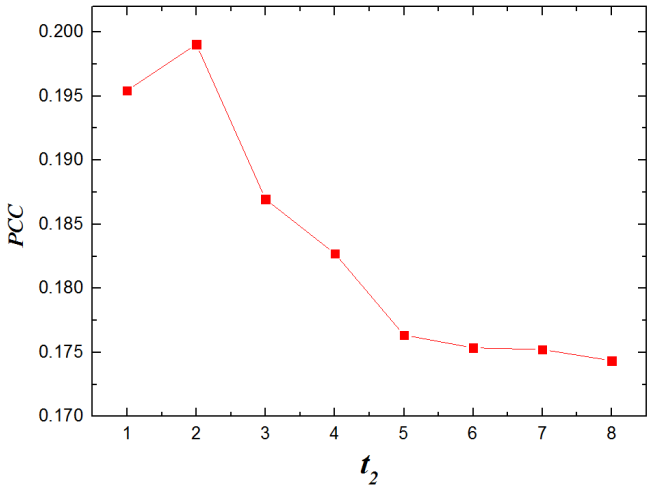


Figure 5.17: Average PCC between the time series of a user and t_2-position-shifted time series of the user's closest 4 collaboration neighbors.

performance.

What's more, the improvement of using neighbor information from Collaboration Network and Follower-Followee Network are compared. It seems that the collaboration relationship of GitHub has more impact upon user activity than the following relationship to some degree, since the improvement of accuracy when using Collaboration Network is relatively larger. We would like to research deeper and try to speculate the reason by using network analysis.

First, we checked the overlap of networks. However, as introduced in Chapter 4, Follower-Followee Network, Collaboration Network and Similarity Network have different link density. Thus, a lower bound of weight of links in Collaboration Network and Similarity Network is required to filter out the links with low weight, in order to reach a condition where the three networks have the same link density. After carrying out overlap checking, we found out that Collaboration Network and Similarity Network share 5926 common links, which is 1.34 times more than the common links between Follower-Followee Network and Similarity Network. This could be an explanation for using the neighbor information from Collaboration Network can help to improve the prediction accuracy more significantly, compared with Follower-Followee Network.

5.4 Using Neighbor Information from Two-Dimension Network

As discussed above, neighbor information from Follower-Followee Network and Collaboration Network can improve the prediction accuracy of a user's active level respectively. So the question is coming, what will happen if we use the neighbor information from both networks simultaneously? Therefore, a two-dimension network consisting of Follower-

Followee Network and Collaboration Network is built. Figure 5.18 gives a schematic diagram of an ego network in two-dimension network. In this section, we aim to find out the difference or improvement of prediction performance when using neighbor information from two-dimension network and one-layer network. And then, along with evaluating the prediction results, try to use network analysis to speculate the reason.

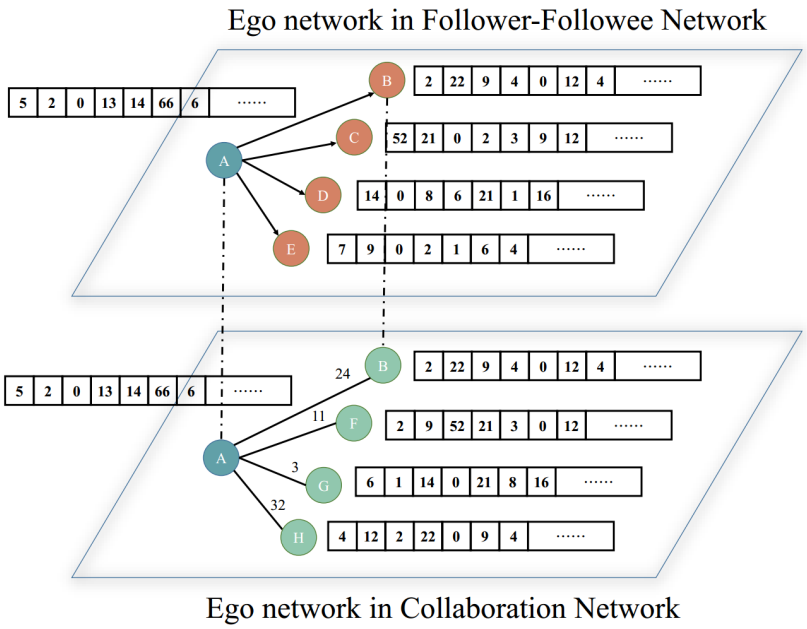


Figure 5.18: An example of ego network in two-dimension network consisting of Follower-Followee Network and Collaboration Network. Each number in the time series denotes the number of activities of a user in 1 week. The directed links represent the following relationship in GitHub social network. The undirected links represent the collaboration relationship. The weight of links indicates number of common projects the two users have participated together. The dotted lines connect the same user in two networks.

The comparison of prediction accuracy by using network information from two-dimension network, two social networks respectively and the baseline is shown in Figure 5.19. In this case, besides the previous activities of a user, the number of activities during the last 1 week of the most active 3 neighbors in Follower-Followee Network and the number of activities during the last 2 weeks of the closest 4 collaboration neighbors in Collaboration Network are taken into consideration. Compared with the baseline condition, the prediction performance is improved greatly when using neighbor information from the two-dimension network: the overall accuracy has reached 87.01%, indicating a rise of 14.11%. It is also demonstrated by Z-Test that the result is significant at $p < 0.01$, which means the improvement of accuracy is significant.

However, we found that the effect of improvement is not simple arithmetic addition. Why we obtain this result? We noticed that there are the common neighbors of a user, such as the User B in Figure 5.18, those who belong to both the most active 3 neighbors

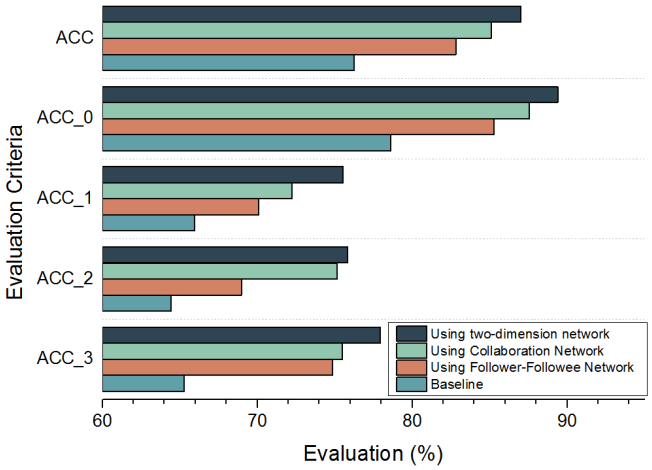


Figure 5.19: The comparison of using network information from two-dimension network, two social networks respectively and the baseline.

in Follower-Followee Network and the closest 4 neighbors who has the most collaboration experience with the user in Collaboration Network. So the related information is taken into account more than once. According to our research, the users who have this kind of common neighbors account for 21.43% of 10k studied users on GitHub.

Now, we are more curious about what kind of role these common neighbors may play. In the next step, all the users who has this kind of common neighbors are picked out. During this process, we ignored the users who have less than 3 following neighbors and 4 collaboration neighbors. The number of activities of a single neighbor of a user from either the most active 3 following neighbors in 1 week or the closest 4 collaboration neighbors in 2 weeks are used as features to prediction a user's active level. Then, we computed the average improvement of common neighbors and regular neighbors. We found out that when using the information of common neighbors, an average improvement of 3.10% is achieved. When using the information of regular neighbors, the average improvement is 2.24%. It indicates that the common neighbors who belong to both the most active 3 following neighbors and the closest 4 collaboration neighbors might have larger impact upon improving the performance of user activity prediction, compared with regular neighbors.

Interestingly, we notice that when using the information of one common neighbor, the effect of improvement is similar with using all the neighbors' activity information in either Follower-Followee Network or Collaboration Network. This consequence also reaffirms the importance of our thesis: using neighbor information from network analysis contributes to the selection of features in machine learning algorithm and can help to improve the prediction of a user's active level.

5.5 Discussion

In this chapter, all the experiments and results of our method, including the baseline which only use the users' activities as features and other conditions where using the neighbor information as features for prediction were presented. We found out that user activity data can be used to predict a user's active level in the next moment. What's more, the neighbor information from Follower-Followee Network and Collaboration Network can be used as features for prediction of user activity, in which the largest improvement can be 8.60% and 11.59% respectively. When using neighbor information from two-dimension network consisting of the two social networks, we can obtain an accuracy of 87.01%, with 14.11% improvement compared with the baseline. Furthermore, we found out that the common neighbors of a user, those who belong to both the most active 3 following neighbors and the closest 4 collaboration neighbors, will have larger impact upon improving the prediction of user activity.

In addition, all the experiments presented in this chapter used Random Forest algorithm during the prediction. In order to avoid special cases and make sure our conclusion is correct enough, we also tried other prediction algorithms, such as Support Vector Machine (SVM) algorithm. The prediction results are in line with the results we showed in this chapter, eliminating the possibility of being caused by some unknown characteristics of specific algorithm.

6 Conclusions

This chapter will conclude the thesis work. Contributions and conclusions will be given in Section 6.1. Then, Section 6.2 will present some new possibilities or directions for future work of this thesis.

6.1 Contributions

In this thesis, aiming to gain deeper understanding and explore more possibilities of using network information from OSNs as features to enhance the prediction of user activity, we used the neighbor activity information from both one-layer networks and two-dimension network of GitHub. Fortunately, at the end of thesis, we accomplished nearly all the established goals. We found out that the neighbor activity information from OSNs can be used as features for user activity prediction. Beside, we also provided further explanations by using network and time series analysis.

The thesis achieved the following contributions:

- We found out that the previous user activity data can be used to predict active level of a user in the next moment. Meanwhile, when investigating prediction, the most suitable time intervals is 1 week. We can obtain better performance when using the user activity data of last 4 weeks to predict the active level of a user, which is set to be the baseline of our experiment.

- The neighbor information from Follower-Followee Network has a positive effect on user activity prediction. We can gain better prediction performance of user activity if the information of the number of activities during the last 1 week of the most active 3 following neighbors of a user is used as features. In this case, the overall prediction accuracy is improved by 8.60% compared with the baseline, reaching a figure of 82.81%.

- As far as Collaboration Network is concerned, it also has a positive effect on user activity prediction. We can gain better prediction performance of user activity if the information of the number of activities during the last 2 weeks of the closest 4 collaboration neighbors of a user is used as features. In this case, the overall prediction accuracy is 85.09%, improved by 11.59% compared with the baseline.

- When using neighbor information from two-dimension network consisting of the two social networks, we can obtain an accuracy of 87.01%, with 14.11% improvement. What's more, we find out that the common neighbors of a user, those who belong to both the most active 3 neighbors from Follower-Followee Network and the closest 4 neighbors who has the most collaboration experience with the user from

Collaboration Network, will have larger impact upon improving the performance of user activity prediction.

- The results of this thesis open new possibilities of using the neighbor information from OSNs as features to improve the performance of user activity prediction and provide further explanations by using network and time series analysis.

6.2 Future Directions

This thesis has demonstrated that using neighbor information from OSNs as features can improve the user activity prediction. However, the future directions related to this topic have many other possibilities, which are necessary or worth studying in the future.

- First of all, the platforms can be chosen from other OSNs, such as Facebook, Twitter or Instagram. At the same time, the targeted user activity can be different, such as posting photos, clicking likes or the access to the website or mobile application.
- Secondly, the network can be constructed by other social relations, rather than following relationship or collaboration relationship. Beside, in this thesis, the structure of our network is fixed, where the nodes, links and weight of links will not change over time. We can also try temporal networks, also known as time-varying networks, where new nodes will be connected to the network and new links will be built over time.
- At last, the neighbor information is not limited to the number of user activities. The neighbor information from networks can be other characteristics of users, such as gender, sex, location or nationality. Besides, some network properties can be also taken into account, such as clustering coefficient, betweenness, closeness or coreness. You may use the information from more dimension to further improve the prediction performance.

Bibliography

[1] Y. Yu, H. Wang, G. Yin, and C. X. Ling, "Reviewer recommender of pull-requests in github," in *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*. IEEE, 2014, pp. 609–612.

[2] A. Lima, L. Rossi, and M. Musolesi, "Coding together at scale: Github as a collaborative social network." in *ICWSM*, 2014.

[3] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[4] J. Scott, *Social network analysis*. Sage, 2017.

[5] Y. Zhang, M. Chen, S. Mao, L. Hu, and V. Leung, "Cap: Community activity prediction based on big data analysis," *Ieee Network*, vol. 28, no. 4, pp. 52–57, 2014.

[6] J. Ye, Z. Zhu, and H. Cheng, "What's your next move: User activity prediction in location-based social networks," in *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 2013, pp. 171–179.

[7] H. Ma, H. Yang, M. R. Lyu, and I. King, "Sorec: social recommendation using probabilistic matrix factorization," in *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 2008, pp. 931–940.

[8] A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke, "Personalized recommendation in social tagging systems using hierarchical clustering," in *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008, pp. 259–266.

[9] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002, pp. 61–70.

[10] L. Hong, O. Dan, and B. D. Davison, "Predicting popular messages in twitter," in *Proceedings of the 20th international conference companion on World wide web*. ACM, 2011, pp. 57–58.

[11] M. Eirinaki and M. Vazirgiannis, "Web mining for web personalization," *ACM Transactions on Internet Technology (TOIT)*, vol. 3, no. 1, pp. 1–27, 2003.

[12] T. Fawcett and F. J. Provost, "Combining data mining and machine learning for effective user profiling." in *KDD*, 1996, pp. 8–13.

[13] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich, "Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 13–20.

[14] C. Perlich, B. Dalessandro, T. Raeder, O. Stitelman, and F. Provost, "Machine learning for targeted display advertising: Transfer learning in action," *Machine learning*, vol. 95, no. 1, pp. 103–127, 2014.

[15] J. E. Aronson, T.-P. Liang, and E. Turban, *Decision support systems and intelligent systems*. Pearson Prentice-Hall, 2005.

[16] Q. Ye, Z. Zhang, and R. Law, "Sentiment classification of online reviews to travel destinations by supervised machine learning approaches," *Expert systems with applications*, vol. 36, no. 3, pp. 6527–6535, 2009.

[17] G. Cui, M. L. Wong, and H.-K. Lui, "Machine learning for direct marketing response models: Bayesian networks with evolutionary programming," *Management Science*, vol. 52, no. 4, pp. 597–612, 2006.

[18] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 57–66.

[19] T. Althoff, P. Jindal, and J. Leskovec, "Online actions with offline impact: How online social networks influence online and offline user behavior," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 2017, pp. 537–546.

[20] M. Burke, C. Marlow, and T. Lento, "Social network activity and social well-being," in *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 2010, pp. 1909–1912.

[21] K.-W. R. Lee and E.-P. Lim, "Friendship maintenance and prediction in multiple social networks," in *Proceedings of the 27th ACM Conference on Hypertext and Social Media*. ACM, 2016, pp. 83–92.

[22] S. Boccaletti, G. Bianconi, R. Criado, C. I. Del Genio, J. Gómez-Gardenes, M. Romance, I. Sendina-Nadal, Z. Wang, and M. Zanin, "The structure and dynamics of multilayer networks," *Physics Reports*, vol. 544, no. 1, pp. 1–122, 2014.

[23] P. Kazienko, K. Musial, E. Kukla, T. Kajdanowicz, and P. Bródka, "Multidimensional social network: model and analysis," *Computational Collective Intelligence. Technologies and Applications*, pp. 378–387, 2011.

[24] P. Kazienko, K. Musial, and T. Kajdanowicz, "Multidimensional social network in the social recommender system," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 41, no. 4, pp. 746–759, 2011.

[25] L. Jin, Y. Chen, T. Wang, P. Hui, and A. V. Vasilakos, "Understanding user behavior in online social networks: A survey," *IEEE Communications Magazine*, vol. 51, no. 9, pp. 144–150, 2013.

[26] Q. Liu, F. Yu, S. Wu, and L. Wang, "A convolutional click prediction model," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 1743–1746.

[27] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, "A dynamic recurrent model for next basket recommendation," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2016, pp. 729–732.

[28] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A: statistical mechanics and its applications*, vol. 390, no. 6, pp. 1150–1170, 2011.

[29] L. Lü, C.-H. Jin, and T. Zhou, "Similarity index based on local paths for link prediction of complex networks," *Physical Review E*, vol. 80, no. 4, p. 046122, 2009.

[30] J. B. Schafer, J. A. Konstan, and J. Riedl, "E-commerce recommendation applications," in *Applications of data mining to electronic commerce*. Springer, 2001, pp. 115–153.

[31] M. I. Mandel, G. E. Poliner, and D. P. Ellis, "Support vector machine active learning for music retrieval," *Multimedia systems*, vol. 12, no. 1, pp. 3–13, 2006.

[32] W. Wong and Z. Guo, "A hybrid intelligent model for medium-term sales forecasting in fashion retail supply chains using extreme learning machine and harmony search algorithm," *International Journal of Production Economics*, vol. 128, no. 2, pp. 614–624, 2010.

[33] N. J. Nilsson, "Artificial intelligence: A modern approach: Stuart russell and peter norvig,(prentice hall, englewood cliffs, nj, 1995); xxviii+ 932 pages," 1996.

[34] K. Jain, "Machine learning basics for a newbie," https://www.analyticsvidhya.com/blog/2015/06/machine-learning-basics/, june 11, 2015.

[35] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[36] A. Moujahid, "A practical introduction to deep learning with caffe and python," http://adilmoujahid.com/posts/2016/06/introduction-deep-learning-python-caffe/, jun 26, 2016.

[37] A. Liaw, M. Wiener *et al.*, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[38] M. Pal, "Random forest classifier for remote sensing classification," *International Journal of Remote Sensing*, vol. 26, no. 1, pp. 217–222, 2005.

[39] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[40] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[43] D. L. Olson and D. Delen, *Advanced data mining techniques*. Springer Science & Business Media, 2008.

[44] R. Paternoster, R. Brame, P. Mazerolle, and A. Piquero, "Using the correct statistical test for the equality of regression coefficients," *Criminology*, vol. 36, no. 4, pp. 859–866, 1998.

[45] G. Gousios, B. Vasilescu, A. Serebrenik, and A. Zaidman, "Lean ghtorrent: Github data on demand," in *Proceedings of the 11th working conference on mining software repositories*. ACM, 2014, pp. 384–387.

[46] Bfire, "Celebrating nine years of github with an anniversary sale," https://github.com/blog/2345-celebrating-nine-years-of-github-with-an-anniversary-sale, april 10, 2017.

[47] G. Gousios, "The ghtorent dataset and tool suite," in *Proceedings of the 10th Working Conference on Mining Software Repositories*. IEEE Press, 2013, pp. 233–236.

[48] G. Gousios and D. Spinellis, "Ghtorrent: Github's data from a firehose," in *Proceedings of the 9th IEEE Working Conference on Mining Software Repositories*. IEEE Press, 2012, pp. 12–21.

[49] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social coding in github: transparency and collaboration in an open software repository," in *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*. ACM, 2012, pp. 1277–1286.

[50] J. Tsay, L. Dabbish, and J. Herbsleb, "Influence of social and technical factors for evaluating contribution in github," in *Proceedings of the 36th international conference on Software engineering*. ACM, 2014, pp. 356–366.

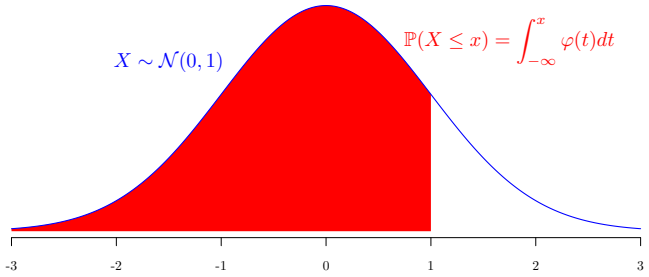
[51] F. Chatziasimidis and I. Stamelos, "Data collection and analysis of github repositories and users," in *Information, Intelligence, Systems and Applications (IISA), 2015 6th International Conference on*. IEEE, 2015, pp. 1–6.

[52] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri, "Feedback effects between similarity and social influence in online communities," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 160–168.

Standard Normal Distribution Table

A

The standard normal distribution table is listed in the following page.



	0	0.001	0.002	0.003	0.004	0.005	0.006	0.007	0.008	0.009
0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990

Prediction Result (F1 score)

B

In this section, the figures of prediction results when using F1 score as evaluation criterion will be presented.

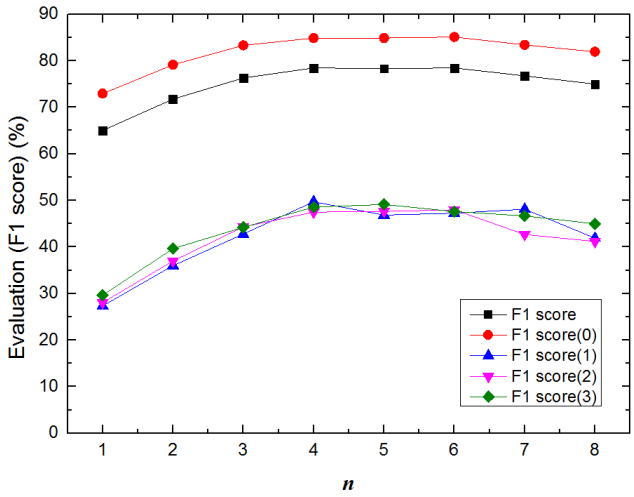


Figure B.1: Results of user activity prediction by using the previous activities of a user.

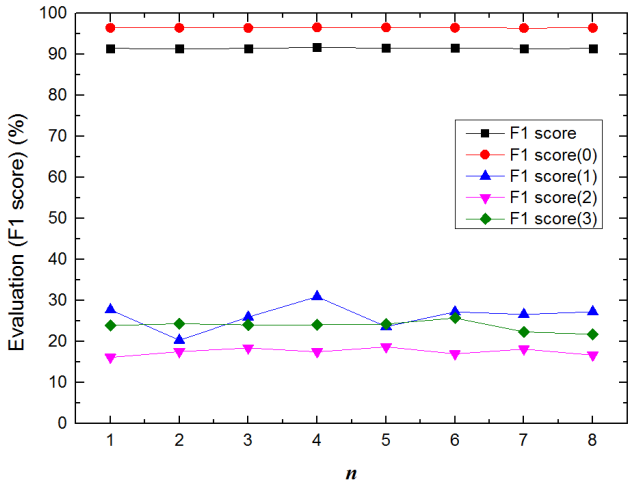


Figure B.2: Results of user activity prediction by only using the previous activities of a user when choosing 1 day as time interval.

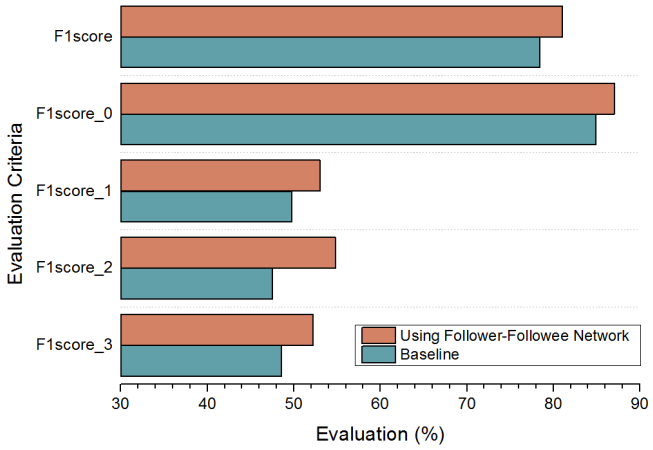


Figure B.3: The comparison of prediction accuracy by using neighbor activity information from Follower-Followee Network and the baseline.

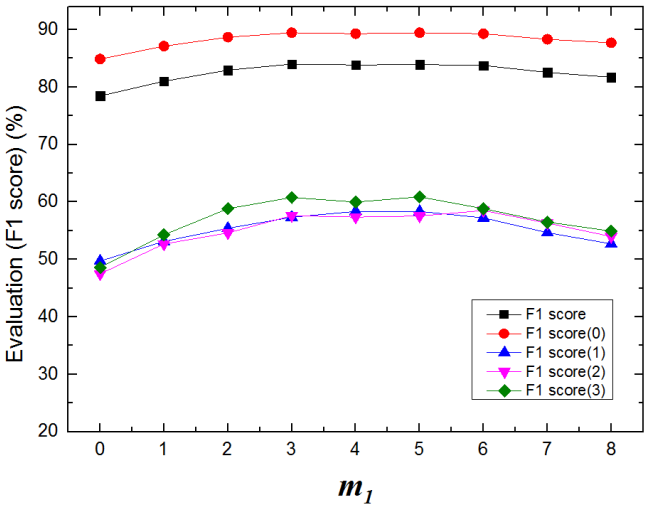


Figure B.4: Results of user activity prediction by using the number of activities during the last 1 week of the most active m_1 neighbors of the user in Follower-Followee Network based on the baseline.

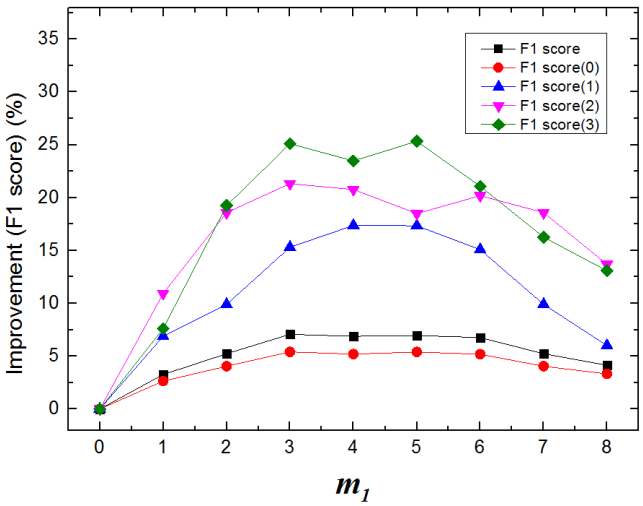


Figure B.5: Improvement of user activity prediction by using the number of activities during the last 1 week of the most active m_1 neighbors of the user in Follower-Followee Network based on the baseline.

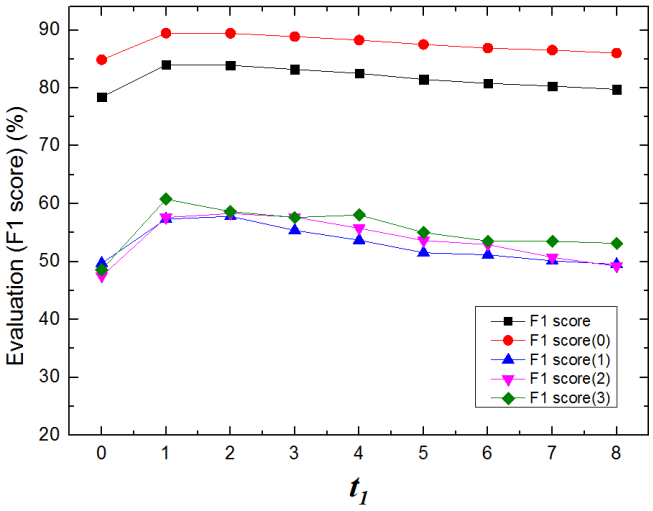


Figure B.6: Results of user activity prediction by using the number of activities during the last t_1 weeks of the most active 3 neighbors of the user in Follower-Followee Network based on the baseline.

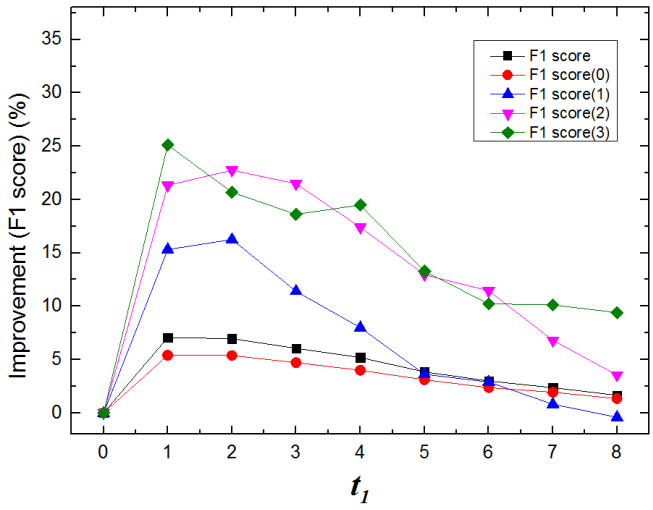


Figure B.7: Improvement of user activity prediction by using the number of activities during the last t_1 weeks of the most active 3 neighbors of the user in Follower-Followee Network based on the baseline.

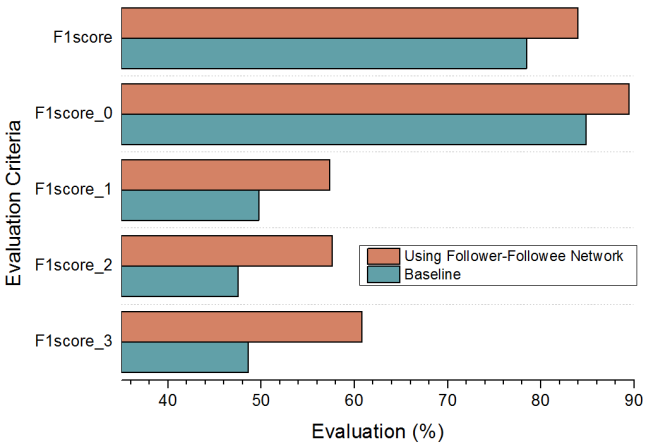


Figure B.8: The comparison of prediction accuracy by using neighbor information of Follower-Followee Network and the baseline.

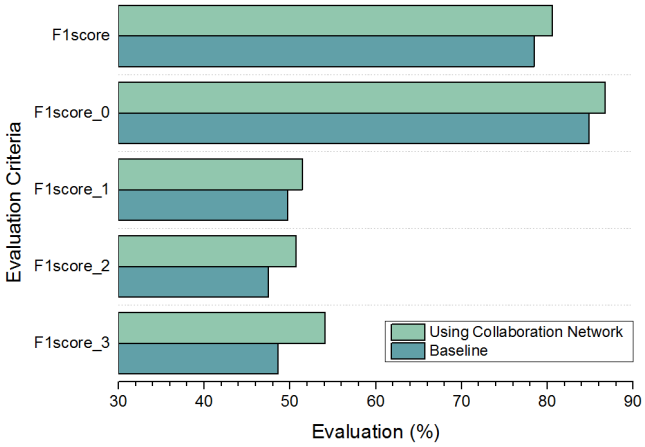


Figure B.9: The comparison of prediction accuracy by using neighbor activity information from Collaboration Network and the baseline.

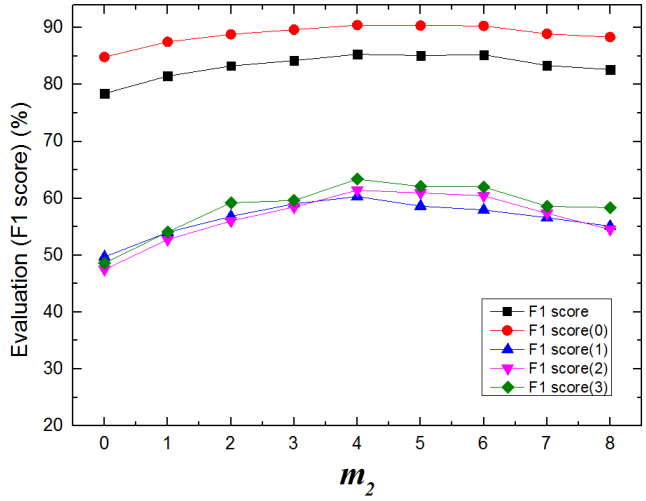


Figure B.10: Results of user activity prediction by using the number of activities during the last 1 week of the closest m_2 collaboration neighbors of the user in Collaboration Network based on the baseline.

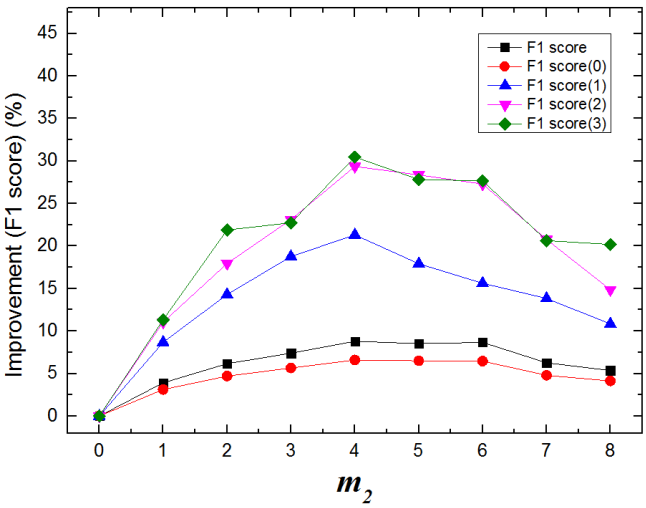


Figure B.11: Improvement of user activity prediction by using the number of activities during the last 1 week of the closest m_2 collaboration neighbors of the user in Collaboration Network based on the baseline.

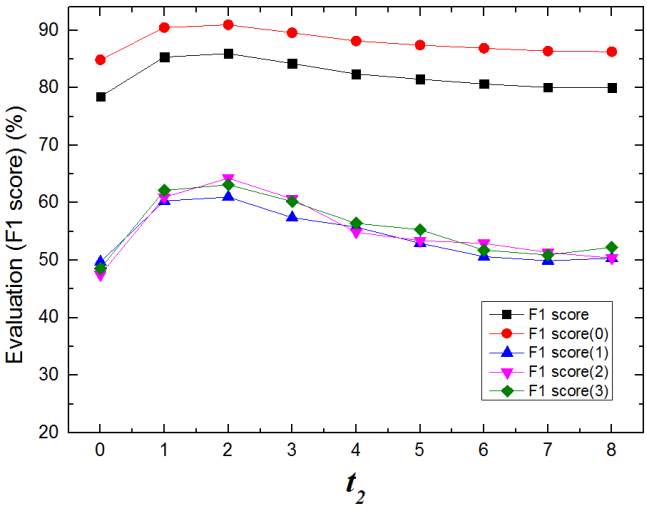


Figure B.12: Results of user activity prediction by using the number of activities during the last t_2 weeks of the closest 4 collaboration neighbors of a user in Collaboration Network based on the baseline.

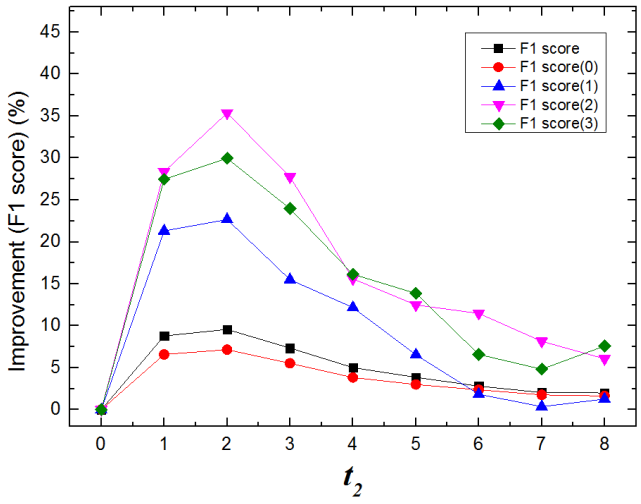


Figure B.13: Improvement of user activity prediction by using the number of activities during the last t_2 weeks of the closest 4 collaboration neighbors of a user in Collaboration Network based on the baseline.

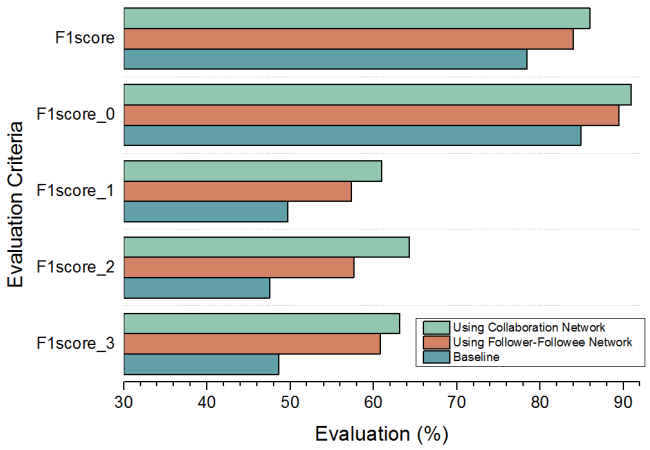


Figure B.14: The comparison of prediction accuracy by using neighbor information from Collaboration Network, Follower-Followee Network and the baseline.

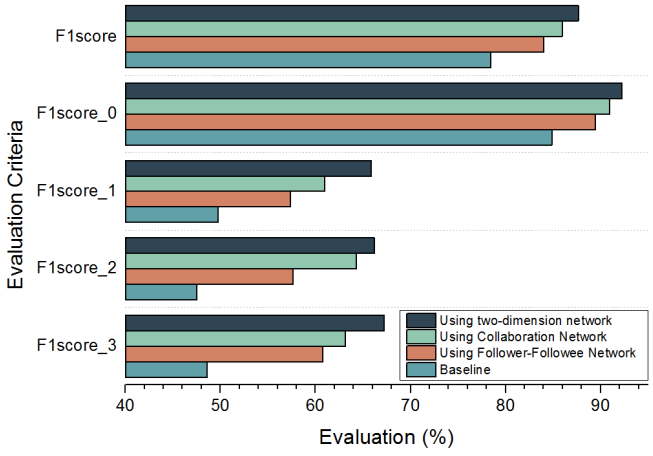


Figure B.15: The comparison of using network information from two-dimension network, two social networks respectively and the baseline.