# An Intelligent Mining System for Prediction and Recommendation based on Link Prediction in Twitter

ZAIRAN WANG [†1] YILANG WU[†1] JUNBO WANG[†2] ZIXUE CHENG[†2]

***Abstract:*** Social Network is nowadays the main media for people communicating with each other. The big data in Social Network often indicates the right trends of the world. However, it is still big challenge to retrieve the demanded data. In this work, we consider link prediction in online social networks, specifically in the popular micro blog platform – Twitter. We aim to develop an intelligent mining system, and then provide recommendation service about strangers to friend based on the Link Prediction approach. We propose a software architecture which uses Twitter application program interfaces (APIs) to collect the tweets that were sent to specific users, and then we extract the user ID and the exact time-stamps of the tweets for user information logging, and create social graph based on specific user and calculate the similarity of non-neighbor users based on analysis to social graph. In future work, we want to analyze the information streams based on Twitter, such as extract URLs which are embedded in tweets, and the final goal is to generate adaptive community based on link prediction for providing personalized and useful recommendation information to users.

***Keyword:*** Internet, Social Networking Service, Twitter analysis, Link Prediction, Recommendation Service

## 1. Introduction

Social Networking Services (SNS) generate huge size of data every day. For example, Twitter, Facebook, Flickr, LinkedIn and Sina Weibo are usually used by mobile users in smartphones or tablet devices. So social networks have been studied extensively in the context of analyzing interactions or structural people relationship graph called social graph. Link prediction is an important approach analyzing the structure of the social graph and the attribute information at different people to determine or predict future links.

In this paper, we focus on Twitter, a popular social networking service. It is a micro blog platform based SNS that launched in 2006, and it has surpassed 200 million active users, handles 340 million tweets, the twitter messages sent by user every day [1]. With the tremendous amount of activities, there come out a lot of new social communities in social work. In Twitter, Twitter communities are groups of interconnected users that follow their interests depending on the streams of information posted in the Twitter social network. These posted information streams are called as tweets that contain the keywords of social interests, user mentions, URLs and hashtags. The following relationship retrieves the followers' tweets to followees', including many tweets are not interesting. Twitter does not efficiently provide support for automatic content recommendations about useful and interesting tweets.

Here we propose an intelligent mining system for Twitter to extract useful information from streams of tweets to create social graph based on specific user and calculate the similarity of non-neighbor users based on analysis to social graph. In future work, the final of goal is to generate adaptive community based on link prediction for providing recommendation of personalized and useful information to users.

In this paper we focus on recommendation based on links that connect the nodes in social graph. As we known, as the link prediction is a critical problem in network modeling and it has been recently studied in social network [2], it can be used to make recommendation based on a user-item interaction graph representation [3]. Second, social networks are sources of big data that allow us to investigate the connections between people and the contents in which such people are interested. Third, framework of social networks can be extended by the link prediction methods. According to the aforementioned points, our main effort is to formulate a solution that allows us to understand the facts of newly established connections in social graph, and analyze the reason that induce connections. In such a purpose, we extract a real social graph from Twitter and the central part of current work is to calculate the similarity of user interests based on link prediction oriented approaches.

Some issues are involved to calculate the similarity of user interests:

- How to extract useful information such as social graph and tweets published by followees.
- How to calculate similarity of user interests between user and followee by link prediction algorithm.
- How to construct social communities for summary the social interests based on the similarity of user interests.

We have developed the system of data collection from Twitter as shown in Figure 1. We can use it to extract social graph, user profiles. The process of data collection from Twitter is shown in three steps below:

1. Extract basic profiles, which include list of friend's id, from Twitter via API, and then saved in format of pair-list to CSV file.
2. Generate social graph in format of an adjacency list from the CSV file.
3. Calculate the transitive similarity, as we defined in Section 3.2, between any pair of nodes based on the connectivity in the graph; we use this transitive similarity to represent the similarity of users' interests in social network.

[†1] Graduate School of Computer Science and Engineers, University of Aizu, Aizuwakamatsu, Fukushima, 965--8580, Japan, m5171121@u-aizu.ac.jp
[†2] School of Computer Science and Engineers, University of Aizu, Aizuwakamatsu, Fukushima, 965--8580, Japan

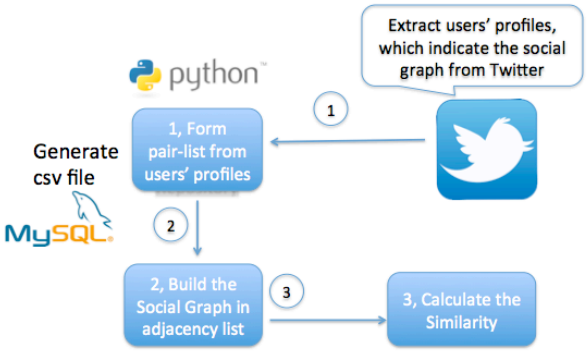
python
Extract users' profiles, which indicate the social graph from Twitter
1
1, Form pair-list from users' profiles
Generate csv file
MySQL
2
2, Build the Social Graph in adjacency list
3
3, Calculate the Similarity

Figure 1. Twitter data collection architecture

## 2. Related Work

The link prediction is an important research field in data mining, it can be used for recommendation systems, social networks, and many other fields.

There are a variety of local similarity measures for analyzing the similarity of nodes in a network, such as Common Neighbors [4], Jaccard's coefficient [5]. Common Neighbor index is adopted by many popular online social networks, like facebook.com for the friend recommendation task. It known as Friend of a Friend algorithm [6], this algorithm is based on the common sense that two node $v_a$ and node $v_b$ are more likely to form a link in the future, if they have many common neighbors. Jaccard's coefficient is a commonly used similarity metric in information retrieval, measures how strongly relationship between two strings. These similarity measures are based on the sum or product of nodes degree. Based on node neighborhoods, the similarity scores between all non-neighbor nodes in a Graph G are zero.

There are a variety of overall approaches, such as Random Walk with Restart (RWR) algorithm, SimRank algorithm, Shortest Path algorithm etc. The computation of the shortest path between two nodes can be used any well known shortest path algorithm. RWR [7] considers a random walker that starts from node $v_a$ who chooses randomly among the available edges every time, except that before he makes a choice, with probability he goes back to node $v_a$ (restart). Thus, the relevance scores of node $v_a$ to node $v_b$ is defined as the steady-state probability.

## 3. Design of an Intelligent Mining System

In order to build an intelligent mining system for prediction and recommendation, we first design a system of data collection, which can be used to extract social graph, user profiles (which include list of friend's id) from Twitter via API. And then we saved in format of pair-list to CSV file. After then we generate social graph in format of an adjacency list from the CSV file. Finally, we calculate the similarity between any pair of nodes based on the connectivity in the graph; we use this similarity to represent the similarity of users' interests in social network.

### 3.1 Notation

In this section, we present the most important notations and the corresponding definitions used throughout the rest of the paper. Let G = <V, E> with a set of nodes V and a set of edges E. Every edge is defined by a specific pair of graph nodes ($v_a$, $v_b$) ∈E, where $v_a$, $v_b$ ∈ V. We assume that the graph G is undirected and un-weighted. Therefore, $sim(p(v_a, v_b))$ and $sim(p(v_b, v_a))$ are equal, can be denoted the same edge on G. We also assume that the graph G has no multiple edges, thus if two nodes $v_a$, $v_b$ are connected with an edge of E, then there is no other edge in E also connecting them. Finally, we assume that there are no loop edges on G, thus no node can be connected to itself. The graph can be seen in Figure 2 based on extracted data from Twitter, will be used as our running example throughout the rest of paper. The defined notations can be seen in Table 1.

Table 1. Notations and Definitions

| Symbol | Description |
|---|---|
| G | undirected and unweighted graph |
| $v_i$ | node of graph G |
| V | set of nodes in graph G |
| E | set of edges in graph G |
| \|V\| | number of nodes in graph G |
| \|E\| | Number of edges in graph G |
| $sim(p(v_a, v_b))$ | similarity between $v_a$ and $v_b$ |
| $(v_a, v_b) \in E$ | $v_a$ and $v_b$ are adjacent |
| $(v_a, v_b) \notin E$ | $v_a$ and $v_b$ are not adjacent |
| $c(v_a)$ | connectivity of $v_a$ |

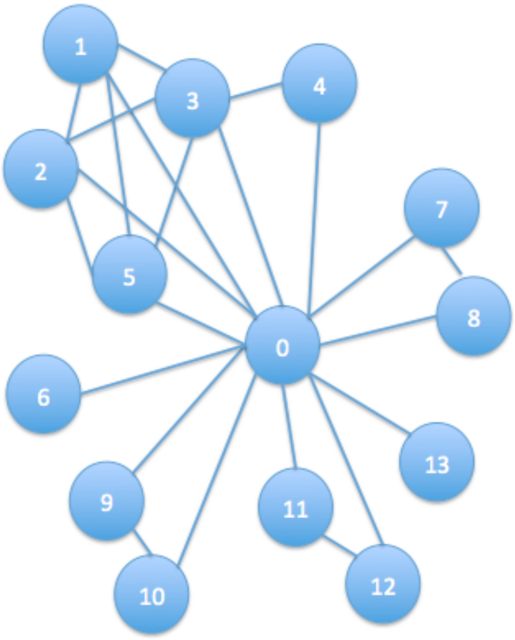

Figure 2: Example of Social Graph based on the extracted data from Twitter

### 3.2 Social Graph Analysis

In this section, we present the definition of similarity between any pair of nodes in a graph G = <V, E>. First, graph distance exists between any people pair in the graph. Some pairs of people, although without existing relations for now, but may have potential relation in the future. The basic idea of algorithm is follow three steps like:

1. Count the number of interactions
2. Assign degrees to every paths
3. Build friendship index

We define the sim(p($v_a$,$v_b$)) is the adjacent_similarity between $v_a$ and $v_b$ based on the connectivity of nodes c($v_a$) and c($v_b$), like Figure 3.

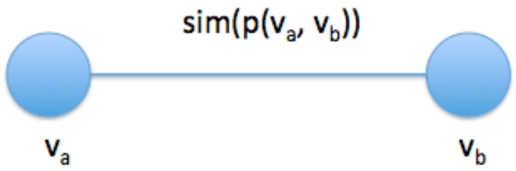

Figure 3. $v_a$ and $v_b$ are adjacent

Formula 1.

$$sim(p(v_a,v_b)) = \begin{cases} 0,(v_a,v_b) \in E \\ 1, v_a = v_b \\ 1/(c[v_a]+c[v_b]-2),(v_a,v_b) \notin E \end{cases}$$

We define the formula 1 to show that if $v_a$ and $v_b$ are not adjacent, the sim(p($v_a$,$v_b$)) = 0, and if $v_a$ = $v_b$, the sim(p($v_a$,$v_b$)) = 1.

If $v_a$ and $v_b$ are adjacent, we have:

$$sim(p(v_a,v_b)) = 1/(c[v_a]+c[v_b]-2)$$

We also define p($v_a$,$v_b$) is a finite sequence of edges which connects a sequence of nodes from the source $v_a$ to the end $v_b$. Given $v_k$ ∈ p($v_a$,$v_b$) = <$v_a$,…$v_k$,…,$v_b$>, $v_k$ is adjacent to $v_{k+1}$, as shown Figure 4.

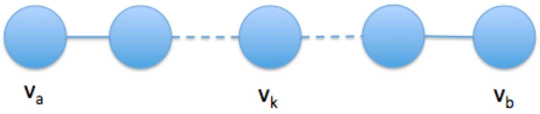

Figure 4. $v_a$ and $v_b$ are not adjacent

Formula 2.

$$sim(p_t(v_a,v_b)) = \prod_{k=a}^{k=b-1} sim(p(v_k,v_{k+1})), v_k \in p_t(\mathrm{v}_a,\mathrm{v}_b)$$

There are a variety of paths from the source $v_a$ to the end $v_b$. We define the critical similarity $p_{cs}$($v_a$,$v_b$), which satisfies the maximum similarity among all the paths between $v_a$ and $v_b$, sim($p_{cs}$($v_a$,$v_b$)) = max(sim(p($v_a$,$v_b$))).

**Description of algorithm:**

This algorithm is based on Dijkstra's algorithm.

Input: non-weighted and undirected social network graph G = <V, E>, source node $v_a$, end node $v_{b,}$ and connectivity list c[].

Output:

- Find out the critical similarity path $p_{cd}$($v_a$,$v_b$) between $v_a$ and $v_b$.
- Compute the critical similarity between $v_a$ and $v_b$ through the critical similarity path $p_{cs}$($v_a$,$v_b$).

```
for each vertex v in G:
        sim[v]    := 0 ;
        previous[v]    := undefined ;
end for
sim[source] := 1 ;
Q := the set of all nodes in G;
while Q is not empty:
        u := vertex in Q with critical similarity in sim[]; // Source node in first case
        remove u from Q ;
        if sim[u] == 0:
            break ;
        end if
        for each neighbor v of u:
            $alt := sim[u] \times (1/(c[u]+c[v]-2))$
            if alt > sim[v]:
                    sim[v] := alt ;
                    previous[v] := u ;
                    decrease-key v in Q;
            end if
        end for
end while
        return struct{sim[], previous[]};
```

**Example based on Figure 2. :**

We run an example for the social agent measure based on Figure 2. Table 2 shows the connectivity list for all nodes in G.

Table 2. The connectivity list

| **Index of node** | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| **Connectivity** | 13 | 4 | 4 | 5 | 2 | 4 | 1 |
| **The Columns Continue as below** | | | | | | | |
| **Index of node** | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| **Connectivity** | 2 | 2 | 2 | 2 | 2 | 2 | 1 |

Table 3 shows critical similarity of all nodes in G, includes non-neighbor nodes.

Table 3. The critical similarity of node

| $v_a$<br>$v_b$ | **0** | **1** | **2** | **3** | **4** | **5** | **6** |
|---|---|---|---|---|---|---|---|
| **0** | 1 | 0.067 | 0.067 | 0.0625 | 0.077 | 0.067 | 0.083 |
| **1** | 0.067 | 1 | 0.167 | 0.143 | 0.029 | 0.167 | 0.006 |
| **2** | 0.067 | 0.167 | 1 | 0.143 | 0.029 | 0.167 | 0.006 |
| **3** | 0.0625 | 0.143 | 0.143 | 1 | 0.2 | 0.143 | 0.005 |
| **4** | 0.078 | 0.029 | 0.029 | 0.2 | 1 | 0.029 | 0.006 |
| **5** | 0.067 | 0.167 | 0.167 | 0.143 | 0.029 | 1 | 0.006 |
| **6** | 0.083 | 0.006 | 0.006 | 0.005 | 0.006 | 0.006 | 1 |

The Columns Continue as below

| $v_a$ $v_b$ | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|
| **0** | 0.077 | 0.077 | 0.077 | 0.077 | 0.077 | 0.077 | 0.083 |
| **1** | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.006 |
| **2** | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.006 |
| **3** | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 |
| **4** | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 |
| **5** | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.006 |
| **6** | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 | 0.007 |

| $v_a$ $v_b$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| **7** | 0.077 | 0.005 | 0.005 | 0.005 | 0.006 | 0.005 | 0.006 |
| **8** | 0.077 | 0.005 | 0.005 | 0.005 | 0.006 | 0.005 | 0.006 |
| **9** | 0.077 | 0.005 | 0.005 | 0.005 | 0.006 | 0.005 | 0.006 |
| **10** | 0.077 | 0.005 | 0.005 | 0.005 | 0.006 | 0.005 | 0.006 |
| **11** | 0.077 | 0.005 | 0.005 | 0.005 | 0.006 | 0.005 | 0.006 |
| **12** | 0.077 | 0.005 | 0.005 | 0.005 | 0.006 | 0.005 | 0.006 |
| **13** | 0.083 | 0.006 | 0.006 | 0.005 | 0.006 | 0.006 | 0.007 |

The Columns Continue as below

| $v_a$ $v_b$ | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|
| **7** | 1 | 0.5 | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 |
| **8** | 0.5 | 1 | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 |
| **9** | 0.006 | 0.006 | 1 | 0.5 | 0.006 | 0.006 | 0.006 |
| **10** | 0.006 | 0.006 | 0.5 | 1 | 0.006 | 0.006 | 0.006 |
| **11** | 0.006 | 0.006 | 0.006 | 0.006 | 1 | 0.5 | 0.006 |
| **12** | 0.006 | 0.006 | 0.006 | 0.006 | 0.5 | 1 | 0.006 |
| **13** | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 | 0.006 | 1 |

## 4. Toward implementation

In this paper, we did not fully implement the whole system except the critical part. The Figure 5 shows the Implementation model, the environment of implementation is Twitter server and the local server.

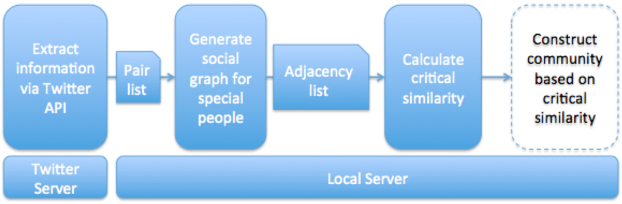
Extract information via Twitter API
Pair list
Generate social graph for special people
Adjacency list
Calculate critical similarity
Construct community based on critical similarity
Twitter Server
Local Server

Figure 5. Implementation model

1. We extract information via Twitter API and generate a pair-list.
2. We save extracted the pair-list in CSV file.
3. We generate social graph based on the CSV file.
4. We have not finished automatically create adjacency list for storing graph.
5. We use real data from Twitter to calculate critical similarity.
6. We have not completed that construct community based on critical similarity.

As shown Figure 6, the social graph can be generated by the work of implementation from 1 to 3.

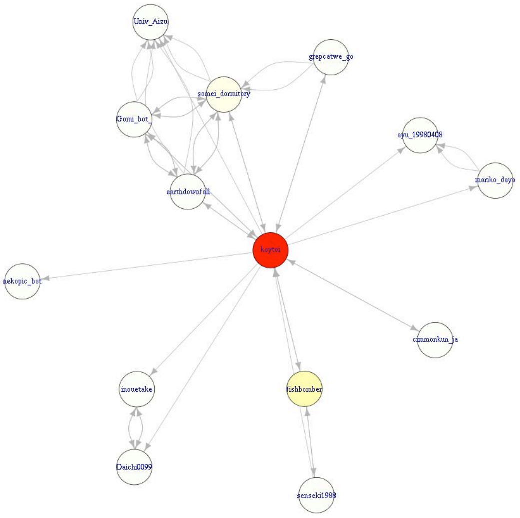

Figure 6. The social graph generated from Twitter data

## 5. Conclusion

We designed an intelligent mining system, it can be used to generating social graph and calculate the similarity between any pair of nodes based on the connectivity in the graph

In the future, we will continue to use the link prediction as a measure for representing relation between people and content in social network, compare with other existing link prediction methods. The final of goal is to generate adaptive community based on link prediction for providing personalized and useful recommendation information to users.

## Reference

[1] Twitter. What is Twitter? Available from http://business.twitter.com/basics/what-is-twitter/

[2] Liben-Nowell, D. and Kleinberg, J. The link prediction problem for social networks. In Proceedings of the Twelfth International Conference on Information and Knowledge Management, 2003, 556-559.

[3] Zan Huang, Xin li, and Hsinchun Chen. Link prediction approach to collaborative filtering. In JCDL '05: Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries, pages 141-142, New York, NY, USA, 2005.ACM Press.

[4]G. Kossinets, "Effects of missing data in social networks," *Social Networks*, vol. 28, no. 3, pp. 247–268, 2006.

[5]Gerard Salton and Michael J. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, 1983.

[6]J. Chen, W. Geyer, C. Dugan, M. Muller, and I. Guy. Make new

friends, but keep the old: recommending people on social networking sites. In Proceedings 27th International Conference on Human Factors in Computing Systems (CHI), pages 201-210, 2009.

[7]J. Pan, H. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-model correlation discovery. In Proceedings 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 653-658, Seattle, WA, 2004.