Friendlink: Link Prediction in Social Networks via Bounded Local Path Traversal

Alexis Papadimitriou
Computer Science Department
Aristotle University
Thessaloniki, Greece
Email: apapadi@csd.auth.gr

Panagiotis Symeonidis
Computer Science Department
Aristotle University
Thessaloniki, Greece
Email: symeon@csd.auth.gr

Yannis Manolopoulos
Computer Science Department
Aristotle University
Thessaloniki, Greece
Email: manolopo@csd.auth.gr

***Abstract*—Online social networks (OSNs) like Facebook, Myspace, and Hi5 have become popular, because they allow users to easily share content or expand their social circle. OSNs recommend new friends to registered users based on local graph features (i.e. based on the number of common friends that two users share). However, OSNs do not exploit all different length paths of the network. Instead, they consider only pathways of maximum length 2 between a user and his candidate friends. On the other hand, there are global approaches, which detect the overall path structure in a network, being computationally prohibitive for huge-size social networks. In this paper, we provide friend recommendations, also known as the *link prediction* problem, by traversing all paths of a bounded length, based on the "algorithmic small world hypothesis". As a result, we are able to provide more accurate and faster friend recommendations. We perform an extensive experimental comparison of the proposed method against existing link prediction algorithms, using two real data sets (Hi5 and Epinions). Our experimental results show that our FriendLink algorithm outperforms other approaches in terms of effectiveness and efficiency in both real data sets.**

***Keywords*-Social Networks; Link Prediction;**

I. Introduction

Online social networks (OSNs) such as Facebook.com[1], Myspace[2], Hi5.com[3], etc. contain gigabytes of data that can be mined to make predictions about who is a friend of whom. OSNs recommend other people to users based on their common friends. The reason is that there is a significant possibility that two users are friends, if they share a large number of common friends.

In this paper, we focus on recommendations based on links that connect the nodes of an OSN, known as the *Link Prediction* problem, where there are two main approaches [5] that handle it. The first one is based on local features of a network, focusing mainly on the nodes structure; the second one is based on global features, detecting the overall path structure in a network. For instance, as an example of a local approach, as shown in Figure 1, Facebook.com or Hi5.com use the following style of recommendation for recommending new friends to a target user U_1: "People you may know : (i) user U_7 because you have two common friends (user U_5 and user U_6) (ii) user U_9 because you have one common friend (user U_8) $\ldots$". The list of recommended friends is ranked based on the number of common friends each candidate friend has with the target user.

[1]`http://www.facebook.com`
[2]`http://www.myspace.com`
[3]`http://www.hi5.com`

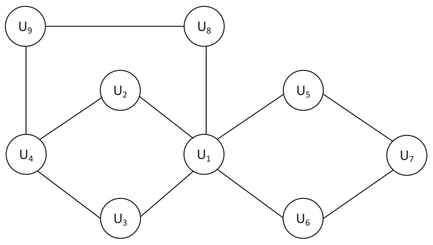

Figure 1. Social Network Example.

A. Motivation

Compared to approaches which are based on local features of a network (i.e. Friend of a Friend (FOAF) algorithm or Common Neighbors, Adamic/Adar index, Jaccard Coefficient, etc. - for more details see Section II), we provide friend recommendations, exploiting paths of greater length. In contrast, they consider only pathways of maximum length 2 between a target user and his candidate friends. In Figure 1, which will be used as our running example, according to existing OSNs, U_1 would get as friend recommendation with equal probability U_4 or U_7. However, if we take into account also paths of length 3, then U_4 should have a higher probability to be recommended as a friend to U_1. In our approach, we assume that a person can be connected to another with many paths of different length (through human chains). Thus, two persons that are connected with many unique pathways of different length have a high probability to know each other, proportionally to the length of the pathways they are connected with.

Compared to global approaches (i.e Katz status index, RWR algorithm, SimRank algorithm etc.), which detect the overall path structure in a network, our method is more efficient. This means, that our method, which is based on a bounded path traversal, requires less time and space complexity than the global based algorithms. The reason is that we traverse only paths of length l in a network based on the "algorithmic small world hypothesis", whereas global approaches detect the overall path structure. (for more details see Section Related Work).

The rest of this paper is organized as follows. Section II summarizes the related work. Section III defines a new node similarity measure in OSNs. The proposed approach, its complexity analysis, and its possible extensions to other networks, are described in Section IV. Experimental results are given in Section V. Finally, Section VI concludes this paper.

II. Related work

There is a variety of local similarity measures [5] (i.e. FOAF algorithm, Adamic/Adar index, Jaccard Coefficient, etc.) for analyzing the "proximity" of nodes in a network. FOAF [2] is adopted by many popular OSNs, such as facebook.com and hi5.com for the friend recommendation task. FOAF is based on the common sense that two nodes v_x, v_y are more likely to form a link in the future, if they have many common neighbors. In addition to FOAF algorithm, there are also other local-based measures such as Jaccard Coefficient [5] and Adamic/Adar index [1]. Adamic and Adar proposed a distance measure to decide when two personal home pages are strongly "related". In particular, they computed features of the pages and defined the similarity between two pages x, y as follows: $\sum_z \frac{1}{\log(frequency(z))}$, where z is a feature shared by pages x, y. This refines the simple counting of common features by weighting rarer features more heavily.

There is a variety of global approaches [5]. In this paper, as comparison partners of global approaches, we consider Katz status index [4], and Random Walk with Restart algorithm [9], [7] (RWR) algorithm. Katz defines a measure that directly sums over all paths between any pair of nodes in graph $\mathcal{G}$, exponentially damped by length to count short paths more heavily. RWR considers a random walker that starts from node v_x, and chooses randomly among the available edges every time, except that, before he makes a choice, with probability α, he goes back to node v_x (restart). The similarity matrix (i.e. Kernel) between nodes of a graph, can be computed by Equation 1:

$$Kernel_{RWR} = (I - \alpha P)^{-1} \tag{1}$$

where I is the identity matrix and P is the transition-probability matrix.

III. Defining a Node Similarity Measure

In this section, we define a new similarity measure to determine a way of expressing the proximity among graph nodes. Let v_i and v_j be two graph nodes and $sim(v_i, v_j)$ a function that expresses their similarity in the range [0,1]. If the two nodes are similar, we expect the value $sim(v_i, v_j)$ to be close to 1. On the other hand, if the two nodes are dissimilar, we expect the value $sim(v_i, v_j)$ to be close to 0.

Our method assumes that persons in an OSN can use all the pathways connecting them, proportionally to the pathway lengths. Thus, two persons who are connected with many unique pathways have a high possibility to know each, proportionally to the length of the pathways they are connected with. For example, referring back to Figure 1, if we consider only length-2 paths, then U_1 would get as friend recommendation with equal probability U_4 or U_7. However, if we consider also length-3 paths, then U_4 should have a higher probability to be recommended as a friend to U_1.

Definition 1. The similarity $sim(v_x, v_y)$ between two graph nodes v_x and v_y is defined as the counts of paths of varying length ℓ from v_x to v_y:

$$sim(v_x, v_y) = \sum_{i=2}^{\ell} \frac{1}{i-1} \cdot \frac{\left|paths^{i}_{v_x,v_y}\right|}{\prod_{j=2}^{i}(n-j)} \tag{2}$$

where

- n is the number of vertices in a graph G,
- ℓ is the length of a path between the graph nodes v_x and v_y (excluding paths with cycles). By the term "paths with cycles" we mean that a path can not be closed (cyclic). Thus, a node can exist only one time in a path (e.g. path $v_1 \to v_2 \to v_3 \to v_1 \to v_5$ is not acceptable because v_1 is traversed twice),
- $\frac{1}{i-1}$ is an "attenuation" factor that weights paths according to their length ℓ. Thus, a 2-step path measures the non-attenuation of a link with value equals to 1 ($\frac{1}{2-1}$ = 1). A 3-step path measures the attenuation of a link with value equals to $\frac{1}{2}$ ($\frac{1}{3-1} = \frac{1}{2}$) etc. In this sense, we use appropriate weights to allow the lower effectiveness of longer path chains. Notice that we have also tested experimentally other possible attenuation factors such as Katz's original exponential β^{ℓ}, the logarithmic $\frac{1}{log(i)}$, etc. and, as will be shown later, the attenuation factor $\frac{1}{i-1}$ attains the best accuracy results.
- $\left|paths^{\ell}_{v_x,v_y}\right|$ is the set of all length-ℓ paths from v_x to v_y,
- $\prod_{j=2}^{i}(n-j)$ is the set of all possible length-ℓ paths from v_x to v_y, if each vertex in graph G was linked with all other vertices. By using the fraction $\frac{\left|paths^{\ell}_{v_x,v_y}\right|}{\prod_{j=2}^{i}(n-j)}$, our similarity measure is normalized and takes values in [0,1]. If two nodes are similar we expect the value $sim(v_x, v_y)$ to be close to 1. On the other hand, if

the two nodes are dissimilar, we expect the value $sim(v_i, v_j)$ to be close to 0.

Notice that the similarity is computed for nodes that are connected with paths of length $\ell \geq 2$. This is because when there is a path between two nodes of length 1 they are already friends.

IV. The Proposed Approach

In this section we first provide a detailed explanation of our approach, named FriendLink. Then, we perform a complexity analysis comparison of FriendLink with other approaches and finally we extend Friendlink for other types of networks.

A. The FriendLink Algorithm

Friendlink computes node similarity between any two nodes in a graph $\mathcal{G}$. The initial input of Friendlink is the number n of nodes of $\mathcal{G}$, the adjacency matrix A, and the length ℓ of paths that will be explored in $\mathcal{G}$. To enumerate all simple paths in $\mathcal{G}$, Rubin's algorithm [8] can be employed. However, Rubin's algorithm uses O(n^3) matrix operations to find all paths of different length between any pair of nodes, where n is the number of nodes in $\mathcal{G}$. In the following, we customize Rubin's algorithm to create only paths of length up to ℓ for our purpose.

As shown in Figure 2, FriendLink consists of a main program and two functions. In the main program, we modify the adjacency matrix so instead of holding 0/1 values, the (i, j) entry of the matrix A is a list of paths from i to j. Then, in the function Combine Paths(), we perform the matrix multiplication algorithm. However, instead of multiplying and adding entries, we concatenate pairs of paths together. Finally, in the function Compute Similarity(), we update the similarity between nodes i and j, for each length-ℓ path we find, where i is the start node and j is the destination node (i.e all paths of length [2..ℓ]). For the update of the similarity value between nodes i and j we use Equation 2. Notice that, we do not take into account cyclic paths in our similarity measure.

For our running example, in Figure 3, we present the node similarity matrix of graph $\mathcal{G}$ calculated by FriendLink algorithm. New friends can be recommended according to their total weight, which is computed by aggregating all paths connecting them with the target user, and by weighting them proportionally to the length of each path. As shown in Figure 3, user U_1 would receive user U_4 as friend recommendation. The resulting recommendation is reasonable, because U_1 is connected with more paths to user U_4 than those that connect U_1 and U_7. That is, the FriendLink approach is able to capture the associations among the graph data objects.

B. Complexity Analysis

RWR [7] and Katz index [4] as representatives of the global approaches, are computationally prohibitive for large graphs, because they require the inversion of a matrix. For instance, the time complexity of Katz index is mainly determined by the matrix inversion operator, which is $O(n^3)$. RWR algorithm also requires a matrix inversion, which

```
Algorithm FriendLink ($\mathcal{G}$, $A$, $n$, $\ell$ )
Input
    $\mathcal{G}$: an undirected and unweighted graph
    $A$: adjacency matrix of graph $\mathcal{G}$,
    $n$: number of nodes of graph $\mathcal{G}$,
    $\ell$: maximum length of paths explored in $\mathcal{G}$,
    $m$: the length of a path
Output
    $sim(i, j)$: similarity between node $i$ and node $j$ in $\mathcal{G}$
---------------------------------------------------------------
1. Main Program
2.     for $i$ = 1 to $n$
3.         for $j$ = 1 to $n$
4.             if $A(i, j)$ = 1 then
5.                 $A(i, j)$ = $j$
6.             else
7.                 $A(i, j)$= 0
8.             end if
9.         end for $j$
10.    end for $i$
11. for $m$ = 2 to $\ell$
12.    Combine Paths()
13.    Compute Similarity($m$)
14. end for $m$
15. End Main Program
---------------------------------------------------------------
16. Function Combine Paths()
17. for $i$ = 1 to $n$
18.    for $j$ = 1 to $n$
19.        for $k$ = 1 to $n$
20.            if $A(i, k) <> 0$ and $A(k, j) <> 0$ then
21.                $A(i, j) = concatenate(A(i, k), A(k, j))$
22.            end if
23.        end for $k$
24.    end for $j$
25. end for $i$
26. return $A(i, j)$
27. End Function
---------------------------------------------------------------
28. Function Compute Similarity()
29. for $i$ = 1 to $n$
30.    for $j$ = 1 to $n$
31.        $denominator$ = 1
32.            for $k$ = 2 to $m$
33.                $denominator$ = $denominator$ * ($n$ - $k$)
34.            end for $k$
35.    $sim(i, j)$ = $sim(i, j)$ + $\frac{1}{m-1} \cdot \frac{\left|paths^{m}_{i,j}\right|}{denominator}$
36.    end for $j$
37. end for $i$
38. return $sim(i, j)$
39. End Function
```

Figure 2. The FriendLink algorithm.

	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9
U_1	0	0	0	**0.2975**	0	0	**0.2856**	0	0.167
U_2	0	0	0.286	0	0146	0.146	0.024	0.156	0.156
U_3	0	0.286	0	0	0.146	0.146	0.024	0.156	0.156
U_4	0.298	0	0	0	0.025	0.025	0	0.167	0
U_5	0	0.146	0.146	0.025	0	0.286	0	0.144	0.015
U_6	0	0.146	0.146	0.025	0.286	0	0	0.144	0.015
U_7	0.286	0.024	0.024	0	0	0	0	0.024	0
U_8	0	0.156	0.156	0.167	0.144	0.144	0.024	0	0
U_9	0.167	0.156	0.156	0	0.015	0.015	0	0	0

Figure 3. Node Similarity Matrix. It presents the possibility of two users being friends.

can be, however, pre-computed [9] and stored for faster on-line operations. In the same direction, there is also a faster version [3] of Katz status index that reduces the computational complexity from time $O(n^3)$ to $O(n+m)$, where m is the number of edges.

FOAF as representative of the local-based methods, considers very small paths (only paths of length 2) between any pair of nodes in $\mathcal{G}$. In particular, for each v_x node, FOAF traverses all its neighbors and then traverses the neighbors of each of v_x's neighbor. Since the time complexity to traverse the neighborhood of a node is simply h (h is the average nodes degree in a network) and our graph $\mathcal{G}$ is sparse, it holds that $h << n$. Thus, the time complexity of FOAF is O($n \times h^2$). The space complexity for FOAF is O($n \times h$).

In contrast to FOAF algorithm which traverses only paths of length 2, our FriendLink algorithm considers also paths with higher length (l-length paths). Based on Milgram's [6] "small-world hypothesis", l can take integer values in the interval [2,6], where for l=2 our FriendLink equals to the FOAF algorithm. Thus, FriendLink's time complexity is O($n \times h^l$). The space complexity for FriendLink is also O($n \times h$). Notice that in our code we store adjacent nodes using adjacency lists and not a matrix structure. However, for simplicity reasons, in Figure 2 we present our algorithm using a matrix structure.

C. *Extending FriendLink for different types of Networks*

Applying FriendLink to directed graphs can be achieved (i) by simply disregarding the edge directions [10], (ii) or by replacing the original adjacency matrix A with an asymmetric one [10]. For weighted networks, if edges weights are all positive, FriendLink applies trivially.

V. Experimental Evaluation

In this section, we compare experimentally FriendLink with RWR, Katz and FOAF algorithms, respectively. Our experiments were performed on a 3 GHz Pentium IV, with 2 GB of memory, running Windows XP. All algorithms were implemented in Matlab.

A. *Real Data Sets*

To evaluate the examined algorithms, we have used two real data sets from the Hi5 and Epinions web sites. We crawled the graph data from the Hi5 web site at two different time periods. In particular, we crawled the Hi5 web site on the 15th of April, 2010 and on the 20th of June, 2010. Our data crawling method was the following: For each user u, we traverse all his friends and then traverse the friends of each of u's friends etc. ¿From the first crawl of Hi5 web site we created a training data set with 63329 users and 88261 edges among them, denoted as Hi5 63K[4], where the initial starting node of our crawling was a random user in the US. From the second crawl of Hi5 web site we created the probe data set with the same users by only preserving 16512 new emerged edges connecting them. The graph data from the first crawl are used to predict the new links emerging in the second crawl.

[4]`http://delab.csd.auth.gr/∼symeon`

We also use in our experiments the Epinions[5] data set, which is a who-trusts-whom social network. In particular, users of Epinions.com express their Web of Trust, i.e. reviewers whose reviews and ratings they have found to be valuable. The Epinions data set is a directed network and, thus, we treat it by simply disregarding the directions of links [10]. It contains 49K users and 487K edges among pairs of users.

We calculated several topological properties of the real data sets which are presented in Figure 4. As shown, Epinions 49K presents (i) a large clustering coefficient (LCC) equal to 0.26, and (ii) a small average shortest path length (ASD) equal to 4.01. These topological features can be mainly discovered in small-worlds networks. Small-world networks have sub-networks that are characterized by the presence of connections between almost any two nodes within them (i.e.high LCC). Moreover, most pairs of nodes are connected by at least one short path (i.e. small ASD).

In contrast, as also shown in Figure 4, Hi5 63K has a very small LLC (0.02) and a quite big ASD (7.18). In other words, Hi5 data set can not be considered as a small-world network, since (i) most of its nodes can not be reached from every other by a small number of hops or steps and (ii) does not have sub-networks that can be considered as cliques.

TOPOLOGICAL PROPERTIES
N = total number of nodes
E = total number of edges
ASD = average shortest path distance between pair nodes
ADEG = average node degree
LCC = average local clustering coefficient
GD = graph diameter (maximum shortest path distance)

Data-Set	N	E	ASD	ADEG	LCC	GD
Hi5 63K	63329	88261	7.18	2.78	0.02	19
Epinions 49K	49288	487183	4.01	19.76	0.26	14

Figure 4. Topological properties of the real data sets.

B. *Experimental Protocol and Evaluation Metrics*

As already described in previous Section, in our evaluation we consider the division of Hi5 63K data set into two sets, according to the exact time stamp of the links downloaded: (i) the training set $\mathcal{E}^T$ is treated as known information and, (ii) the probe set $\mathcal{E}^P$ is used for testing. No information in the probe set is allowed to be used for prediction. It is obvious that $\mathcal{E}^T \cap \mathcal{E}^P = \oslash$. For each user that has at least one new friend in $\mathcal{E}^P$ we generate friend recommendations based on his friends in $\mathcal{E}^T$. Then, we average the results for each user and compute the final

[5]`http://www.trustlet.org/wiki/`

performance of each algorithm. Epinions data sets does not have time stamps of the edges. The performance of the algorithms is evaluated by applying double cross-validation (internal and external). Each data set was divided into 10 subsets. Each subset ($\mathcal{E}^P$) was in turn used for performance estimation in the external cross-validation. The 9 remaining subsets ($\mathcal{E}^T$) were used for the internal cross-validation. In particular, we performed an internal 9-fold cross-validation to determine the best values of the algorithms' needed parameters. In particular, for RWR we set parameter α= 0.001, whereas for Katz we set parameter β=0.005. We chose as values for the parameters those providing the best performance on the internal 9-fold cross-validation. Then, their performance is averaged on the external 10-fold cross-validation. The presented results, based on two-tailed t-test, are statistically significant at the 0.05 level.

We use the classic precision/recall metric as performance measure for friend recommendations. For a test user receiving a list of k recommended friends (top-k list), precision and recall are defined as follows: **Precision** is the ratio of the number of relevant users in the top-k list (i.e., those in the top-k list that belong in the probe set $\mathcal{E}^P$ of friends of the target user) to k. **Recall** is the ratio of the number of relevant users in the top-k list to the total number of relevant users (all friends in the probe set $\mathcal{E}^P$ of the target user).

C. *Sensitivity Analysis for the FriendLink Algorithm*

In this Section, we study the sensitivity of FriendLink accuracy performance in real networks (i) with different possible attenuation factors, (ii) with different controllable sparsity, (iii) with different ℓ values for path traversal.

In section III, we presented the definition of our similarity measure (see Equation 2). The attenuation factor that was mentioned, weights paths according to their length ℓ. In this section, we test other possible attenuation factors in order to discover the best precision value that we can attain. In particular, we have tested the following possible attenuation factors: (i) $\frac{1}{m-1}$ (ii) $\frac{1}{2 \cdot m}$ (iii) $\frac{1}{m^2}$ (iv) $\frac{1}{log(m)}$ and (v) the Katz's index attenuation factor β^m, where m is the path length. The attenuation factors performance can be seen in Figure 5a, for both real data set. As shown, the best performance in both data sets is attained by $\frac{1}{m-1}$. In the following, we keep the $\frac{1}{m-1}$ as the default attenuation factor of the FriendLink algorithm.

Next, we measure the accuracy that FriendLink attains, with different controllable sparsity. To examine the accuracy performance of FriendLink in terms of different network sparsity, we have created 5 different sparsity cases, by changing the fraction of observed edges, as shown in Figure 5b. As expected, as the fraction of edges observed increases, the precision increases too. This is reasonable, since every prediction algorithm is expected to give higher accuracy for a denser network.

In Section IV-A, one of the required input values for the FriendLink algorithm is the length ℓ of paths considered in a graph. To improve our recommendations, it is important to fine-tune the ℓ variable. Based on Milgram's [6] "small-world hypothesis", ℓ should take integer values in the interval [2,6]. Figures 5c and 5d illustrate precision for varying ℓ values for the Epinions 49K, and Hi5 63K data sets, respectively. As expected, precision decreases as the number of recommended friends is increased. The best precision is attained by $\ell = 3$ in both data sets. Notice that we omit to show results for $\ell = 6$ because precision follows a degraded performance for $\ell = 4$ and $\ell = 5$, respectively. In the following, we keep the $\ell = 3$ as the default maximum length of paths of the FriendLink algorithm.

D. *Accuracy Comparison of FriendLink with other methods*

We proceed with the comparison of FriendLink with RWR, Katz, and FOAF algorithms, in terms of precision and recall. This reveals the robustness of each algorithm in attaining high recall with minimal losses in terms of precision. We examine the ranked list, which is recommended to a target user, starting from the top friend. In this situation, the recall and precision vary as we increase the number of recommended friends. For the Epinions 49K data set, as shown in Figure 5e, our FriendLink algorithm again attains the best precision value of 55% when we recommend a top-1 list of friends. The precision of FriendLink is impressive in this specific data set. The main reason is the topological characteristics of Epinions 49K data set (i.e. high LCC and small ASD). Thus, Epinions 49K can be considered as a small-world network. This experiment shows that FriendLink is more robust in finding relevant users for the test user. The reason is that FriendLink exploits proportionally the ℓ-length pathways between two nodes in the graph. In contrast, RWR and Katz traverse globally the social network, missing to capture adequately the local characteristics of the graph. Moreover, FOAF fails to provide accurate recommendations because it exploits only length-2 paths.

For the Hi5 63K data set, as shown in Figure 5f, our FriendLink algorithm attains the best results. However, the overall performance of FriendLink, RWR and Katz algorithms is significantly decreased compared with the results in the Epinions data set. The first reason is the high sparsity (i.e. ADEG equal to 2.78) of the Hi5 63K data set. The second reason is the fact that Hi5 cannot be considered as a small-world network.

E. *Time Comparison of FriendLink with other Methods*

In this Section, we compare FriendLink against RWR, Katz and FOAF algorithms in terms of efficiency using the 2 real data sets. We measured the clock time for the off-line parts of all algorithms. The off-line part refers to the building of the similarity matrix between any pair of

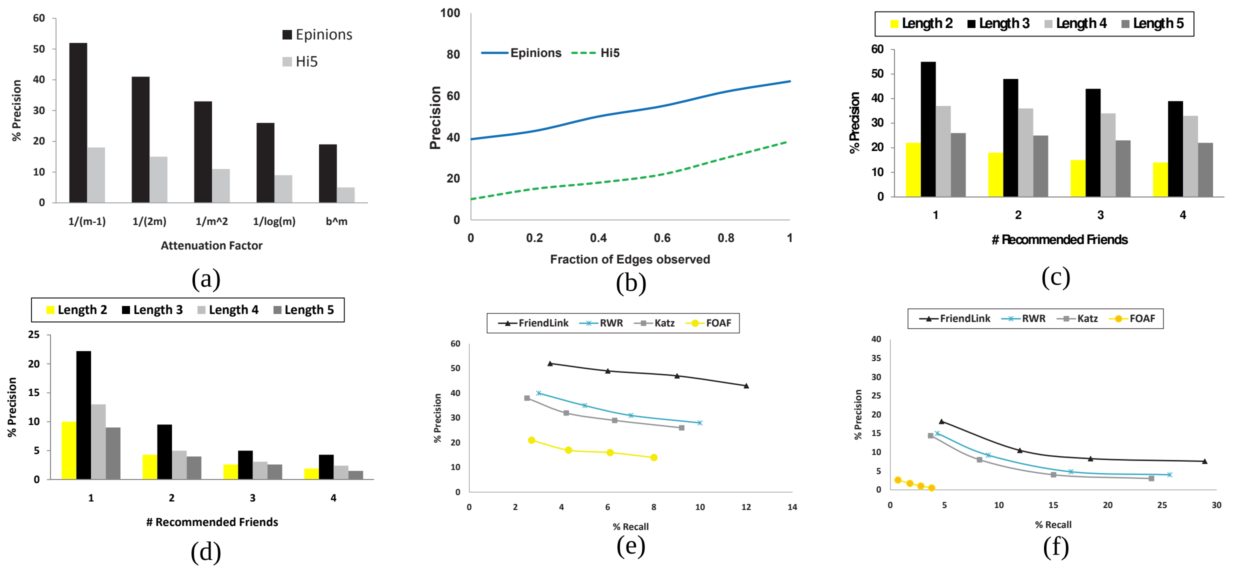

Figure 5. (a) Precision vs. Attenuation factors diagram, (b) Precision vs. fraction of edges observed diagram. (c) Precision vs. number of recommended friends for Epinions 49K data set, (d) Precision vs. number of recommended friends for Hi5 63K data sets. (e) Precision vs. Recall diagram for Epinions 49K data set, (f) Precision vs. Recall diagram for Hi5 63K data set.

nodes in a graph. The results are presented in Table I. As shown, FriendLink outperforms RWR and Katz, since they calculate the inverse of an $n \times n$ matrix. As expected, FOAF algorithm, outperforms the other algorithms due to its simpler complexity. Notice that the results depict the time needed to compute the whole similarity matrix. On the other hand, if we were to calculate the similarity matrix of only one user, then the computation would require only part of a second to produce a recommendation.

Data Sets	FriendLink	RWR	Katz	FOAF
Epinions 49K	245 sec	380 sec	460 sec	**55 sec**
Hi5 63K	340 sec	520 sec	617 sec	**221 sec**

Table I
TIME COMPARISON OF TESTED ALGORITHMS FOR BOTH DATA SETS.

VI. CONCLUSIONS

In this paper, we introduced a framework to provide friend recommendations in OSNs. We define a new node similarity measure that exploits local and global characteristics of a network. We performed extensive experimental comparison of the proposed method against existing link prediction algorithms showing that our FriendLink algorithm provides more efficient and more accurate friend recommendations. In the future, we want to examine other ways of improving friend recommendations based on other features that OSNs offer, such as photo and video tagging, groups and common applications. The combination of such features can provide information on different ways that users are connected and therefore yield to more accurate friend recommendations.

REFERENCES

[1] L. Adamic and E. Adar. How to search a social network. *Social Networks*, 27(3):187–203, 2005.

[2] 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 *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 201–210, 2009.

[3] K. C. Foster, S. Q. Muth, J. J. Potterat, and R. B. Rothenberg. A faster katz status score algorithm. *Comput. Math. Organ. Theory*, 7:275–285, December 2001.

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

[5] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. *Proceedings of the 12th International Conference on Information and Knowledge Management (CIKM)*, 2003.

[6] S. Milgram. The small world problem. *Psychology Today*, 22:61–67, 1967.

[7] J. Pan, H. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *KDD '04: Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 653–658, 2004.

[8] F. Rubin. Enumerating all simple paths in a graph. *IEEE Transactions on Circuits and Systems*, 25(8):641–642, 1978.

[9] H. Tong, C. Faloutsos, and J. Pan. Fast random walk with restart and its applications. In *ICDM '06: Proceedings of the 6th International Conference on Data Mining*, pages 613–622. IEEE Computer Society, 2006.

[10] S. Wasserman and K. Faust. Social network analysis: Methods and applications. 1994.