IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 9, 2020, accepted February 5, 2020, date of publication February 24, 2020, date of current version March 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2975782

Trust Computation in Online Social Networks Using Co-Citation and Transpose Trust Propagation

SAFI ULLAH NASIR AND TAE-HYUNG KIM
Department of Computer Science and Engineering, Hanyang University at ERICA, Ansan 15588, South Korea
Corresponding author: Tae-Hyung Kim (taehyung@hanyang.ac.kr)

ABSTRACT Evaluating trust and distrust between users in online social networks is an important research problem. To address this problem, we provide a method for estimating continuous trust /distrust value between unconnected users. Our method is based on co-citation and transpose trust propagation. We determine on average how differently two users trust or are trusted by other users, and how differently a user trusts another user from how it is trusted by that user. Using these differences, we estimate four partial trust estimates and compute the final trust value from trustor to trustee as the weighted average of these partial estimates. We propose a basic framework that maximizes accuracy, robustness and coverage and show how we can further improve the accuracy at a lower coverage. We perform experiments on real world trust related networks that show that our proposed method outperforms recent state of the art trust computation methods in terms of accuracy and robustness on commonly used datasets.

INDEX TERMS Trust computation, distrust, co-citation, reciprocity, weighted signed networks.

I. INTRODUCTION

In online social networks (OSNs) trust plays an important role in user activities. It allows users to distinguish between reliable and malicious users and content produced by them. Knowing the level of trust or distrust between users is also valuable for OSN service providers, as it can be used for tasks such as suggesting friends, detecting malicious or spam users and community detection. Several e-commerce platforms like Epinions, eBay and Amazon too have a social network component where trust helps users make decisions regarding reliability of reviews and product purchases [1]. Trust has also been used to improve performance of recommender systems [2] and collaborative filtering algorithms [3]. For users that are directly connected by friendship or like/dislike links trust can be estimated by analyzing their past interactions, profile similarity, or by obtaining explicit trust ratings given by the users. However, since most users do not have a direct link or previous interaction with each other, estimating trust between these unconnected users is an important research problem.

To evaluate trust between unconnected users, most earlier algorithms use transitive trust propagation, i.e. if user A trusts user B and user B trusts user C then A should have some degree of trust on C. By using chain of trust links or paths in the trust graph, the level of trust is computed. These algorithms need to extract all, or a very large number of paths and are therefore not efficient on large networks. They can also suffer from lack of robustness i.e. the accuracy may decrease if some parts of network are not visible since missing edges makes it harder to find paths. Moreover, since distrust is not transitive, it's not straightforward to use it, hence most trust computation algorithms choose to ignore distrust.

Recently with availability of more datasets with both trust and distrust information, a few algorithms [15]–[17] have been developed that estimate both trust and distrust as a continuous value. However, some of these algorithms are not efficient to be applied on very large networks or are not based on trust propagation. Their accuracy and robustness can be also further improved. For this purpose, we use co-citation and transpose trust propagation and develop a method for estimating continuous trust/distrust value that is more accurate and robust then other recent existing algorithms and is also efficient to be applied on large networks. Furthermore, our method demonstrates the use of co-citation and

The associate editor coordinating the review of this manuscript and approving it for publication was Xiao Liu.

transpose trust propagation operations (originally proposed for binary trust) for continuous trust/distrust estimation. To estimate trust/distrust value we combine information from neighboring users of trustor (evaluating user), trustee (user being evaluated) and the trust from the trustee to trustor (if available). More specifically, we use information from four sources:

1. **Trustee's in-neighbors**: We find how differently trustee's in-neighbors and the trustor trust some of the other users and use this difference along with trustee's in-neighbors trust of trustee to find a partial trust estimate based on trustee's in-neighbors.
2. **Trustor's out-neighbors**: We find how the trustor's out-neighbors and the trustee differ in being trusted by some other users. This difference and the trustor's trust in its out-neighbors is used to estimate a partial trust estimate based on trustor's out-neighbors.
3. **Trustor's reciprocal neighbors**: We find how differently the trustor trusts some of its reciprocal neighbors from the trust it receives from them. We then use this difference and the trust from trustee to trustor to get another partial estimate.
4. **Trustee's reciprocal neighbors**: We find how much trustee's received trust from some of the reciprocal neighbors differ from the trust it assigns to them. We use this difference and trust from trustee to trustor to find a partial trust estimate.

The final trust is the aggregation of the four partial trust estimates. To verify our method, we perform a series of experiments on multiple real-world trust networks datasets which show that our framework performs better in terms of accuracy and robustness then other recent trust computation methods.

The rest of the paper is organized as follows. Section 2 reviews some related works regarding trust computation in OSNs. Section 3 defines the problem and notations used. In Section 4 we describe the details of our trust estimation framework and algorithms. The experimental results are presented in Section 5. Section 6 concludes the paper.

II. RELATED WORKS

Considering the importance of trust in online social networks a large amount of research has been done in evaluating trust between users. References [4], [5] survey different trust evaluation methods for online social networks. Trust computation methods can be categorized based on whether they include distrust or not. Some of the early works that do not include distrust include Tidal trust [6], Mole Trust [7], Eigen Trust [8]. Tidal trust propagates trust recursively from trustee to trustor along strongest shortest path. Mole trust first removes cycles from graph and then calculates trust of nodes at distance 1,2 and so on up to a maximum depth. Eigen Trust works in a similar way to page rank algorithm and assigns a global trust value to each node. In SW trust [9] authors use adjustable breath first search and small world characteristics of social networks to extract a small trusted graph and make existing trust evaluation methods more efficient. Kim and Song [10] evaluate minmax and weighted mean strategies using reinforcement learning for predicting trust. In [11] we use land mark-based method to make minmax trust computation strategies efficient for large graphs. Jiang *et al.* [12] convert the trust computation problem into a generalized network flow problem and present a modified flow-based algorithm.

Methods that include distrust include [13] where Mishra *et al.* propose global matrices bias and deserve. Bias of a trustor represents truthfulness or propensity to trust/distrust and deserve or prestige represents how much a trustee would receive trust/distrust form an unbiased trustor. Yao *et al.* [14] propose a trust inference model that integrates transitivity, trust bias and multi aspect property of trust for inferring binary and continuous trust score and trust/distrust signs. More recently Kumar *et al.* [15] present Fairness and Goodness global matrices in weighted signed networks that can be used to compute local trust by multiplying Fairness of trustor and Goodness of trustee. In [16] authors present semiring based trust aggregation method that handles both trust and distrust to infer trust for trust-based recommender systems. Akilal *et al.* [17] present a fast and robust trust computation method based on controversy, eclecticism, and reciprocity that handles both trust and distrust. They use trusting and being trusted pattern of trustor and trustee to compute trust from trustor to trustee. Our method improves the accuracy and robustness of these recent methods and is also based on trust propagation operations i.e. co-citation and transpose trust propagation.

III. PROBLEM DEFINITION AND NOTATIONS

We represent the trust network as a weighted directed graph G (V, E, W). V is the set of nodes representing users in the trust network. E is the set of directed edges representing trust relationships between users. W is the weight function that assigns a weight to each edge denoting the level of trust. The edge weights for networks having both trust and distrust links are from the interval [-1,1] with positive values representing trust and negative values representing distrust. For trust, larger values (closer to 1) represent stronger trust. For distrust, smaller values (closer to -1) represent stronger distrust. Networks having only trust links have edge weights from the interval [0,1] where larger values represent stronger trust. Throughout the paper when we refer to trust estimation it refers to trust/distrust i.e. when the estimated value is positive it is trust, when it is negative it is distrust. Given a trust network and two users having no direct edge, a trustor user S and trustee user T, the trust computation problem is to find the most accurate level of trust from S to T. In graph theory terms we have to predict the weight and sign of edge from trustor node to trustee node i.e. w(S,T). This is continuous trust prediction problem and is different from binary trust prediction where objective is to determine if S should trust T or not.

To compute trust between unconnected users, trust computation methods use trust propagation operations. These operations are transitive/direct propagation, transpose trust,

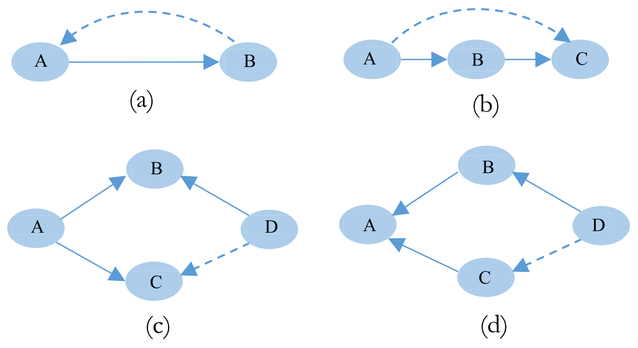


FIGURE 1. Trust propagation operations: (a) Transpose (b) Direct or Transitive (c) Co-citation (d) Coupling [18]. Solid arrows indicate existing trust relations while the dashed line indicate propagated trust.

TABLE 1. Summary of notations.

Notation	Description
S or trustor	User that evaluates for trustworthiness.
T or trustee	User that is being evaluated for trustworthiness.
| |	Absolute value or number of elements in set.
PTE_1, PTE_2 PTE_3, PTE_4	Partial trust estimates that use trustee's in-neighbors, trustor's out-neighbor, trustor and trustee's reciprocal neighbors.
in(x)	In-neighbors of x or users that have direct trust link to x.
out(x)	Out-neighbors of x or users to which x has a direct trust link.
$C_1(x)$, $C_2(x)$	Set of nodes for finding difference between two users about trusting others or being trusted by others respectively.
$C_3(S)$, $C_4(T)$	Set of reciprocal neighbors of for finding difference in trusting from being trusted or vice versa respectively.
$D_1(x)$, $D_2(x)$	Difference between two users about trusting others or being trusted by others respectively.
$D_3(S)$, $D_3(T)$	Difference in trusting from being trusted by reciprocal neighbors or vice versa respectively.
$E_1(x)$, $E_2(x)$	Individual estimates based on trustee in-neighbors or trustors out-neighbors.
w_i	weight for PTE_i.

co-citation and trust coupling [18]. Figure 1 shows these propagation operations. Most methods rely on transitive propagation and very few methods are based other three propagation operations. Our method is based on transpose and co-citation trust propagation.

Table 1 describes some of the notations used in the paper.

IV. TRUST ESTIMATION FRAMEWORK

To compute the trust our basic idea is that for two users A and B, we can find on how differently they trust, or are trusted by other users. We can also find how differently a user trusts its reciprocal neighbors from trust it receives from them. This information about differences can be used to estimate unknown trust values.

To estimate trust from trustor to trustee we use four sources of information. From each source we compute a partial trust estimate (PTE) and then use these PTEs to compute final trust. Our trust estimation framework has two versions a basic version which maximizes the coverage i.e. pairs of users for which trust can be estimated. This version is robust to missing edges and has maximum aggregate accuracy as shown in the experimental results section. The second version is the increased accuracy version that further improves accuracy at a lower coverage by applying some restrictions.

A. BASIC FRAMEWORK

1) PTE_1: TRUSTEE'S IN-NEIGHBORS

To compute the trust level from trustor to trustee, most of the important information can come from in-neighbors of the trustee as they have direct knowledge about trustworthiness of trustee. But since trust is subjective i.e. different users may have different level of trust in the same user, we will not get an accurate estimate if we directly use the trustee's in-neighbors trust ratings of trustor. However, if we knew how much each in-neighbor of trustee and the trustor differ about trusting the trustee, we could use that trustee's in-neighbor's trust in a more accurate way. We can find an approximate estimate of this difference by using trustee's in-neighbors and trustor's trust ratings of some other common users.

Consider Figure 2. We show a section of network containing trustor S and trustee T and one in-neighbor of T i.e. B. We show one in-neighbor only for clarity, generally a trustee would have several in-neighbors. To use B's trust of T in our framework, we find difference between B and S about trusting other users. For this we observe the trust of B and S for common users whose trust rating from B is close to B's trust rating of T, i.e. the absolute difference between their trust from B and B's trust in T is less than or equal to a constant ε. The reason of choosing these nodes is that they can be considered similar to T from B's perspective, and the difference of trust in these similar nodes is more relevant to estimating difference of trust about T. Let the set of common nodes between S and B for estimating S's trust of T using B be denoted as $C_1(B)$

$$C_1(B) = out(S) \cap \{I \in out(B) \text{ and } |w(B, I) - w(B, T)| \le \varepsilon\} \tag{1}$$

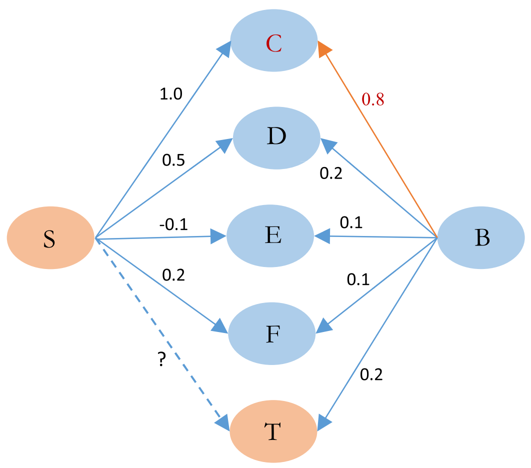


FIGURE 2. Finding difference between Trustor S and Trustee's in-neighbor B about trusting other users.

We use ε as 0.5 for all our datasets. In Figure 2 the set $C_1(B)$ includes D, E and F but not C since absolute difference between B's trust in C and B's trust in T is greater than 0.5. This scenario is similar to collaborating filtering problem, however rather than using some correlation matric to find similarity between S and B, we find the average difference in S and B's trust value for nodes in $C_1(B)$ which we denote as $D_1(B)$.

$$D_1(B) = \begin{cases} \frac{\sum_{I \in C_1(B)} w(S, I) - w(B, I)}{|C_1(B)|}, & |C_1(B)| > 0 \\ 0, & |C_1(B)| = 0 \end{cases} \tag{2}$$

$D_1(B)$ represents on average how much higher or lower than B, user S trusts users that similar to T, and therefore can be considered an approximate estimate of how much higher or lower than B, S should trust T. We add $D_1(B)$ to B's trust rating of T to get the estimate of S's trust for T that uses B.

$$E_1(B) = w(B, T) + D_1(B) \tag{3}$$

Some of the in-neighbors of the trustee will not have any similar common user with the trustor i.e. for some in-neighbors x, $C_1(x)$ is empty. We could ignore these in-neighbors and only use those where $|C_1(x)| > 0$, however this will increase the number of users x for which no in-neighbor with $|C_1(x)| > 0$ exists and in turn increase pairs of users for which trust cannot be estimated. This effect will be stronger when more edges are missing, so robustness will decrease. Therefore, in our basic framework we include in-neighbors x where $|C_1(x)| = 0$, but since we don't have information about their difference with S about trusting other users, we let $D_1(x) = 0$.

In the Figure 2

$$D_1(B) = ((0.5 - 0.2) + (-0.1 - 0.1) + (0.2 - 0.1))/3 = 0.067$$
$$E_1(B) = 0.2 + 0.067 = 0.267.$$

From the trust propagation perspective this estimate is can be considered a form of co-citation trust propagation. As shown in Figure 2, according to co-citation propagation if A trusts B and C, and D trusts B, then D may also trust C because both A and D have similar views on B. In our method for computing $E_1(B)$, trustee's in-neighbor B trusts D, E, F and T whereas trustor S trusts D, E, F and so S may also trust T.

The partial trust estimate based on in-neighbors of trustee is the average of trust values estimated from each in-neighbor.

$$PTE_1 = \begin{cases} \frac{\sum_{I \in in(T)} E_1(I)}{|in(T)|}, & |in(T)| > 0 \\ undefined, & |in(T)| = 0 \end{cases} \tag{4}$$

2) PTE_2: TRUSTOR'S OUT-NEIGHBORS

Apart from the in-neighbors of trustee, another important source of information are the direct out-neighbors of the trustor node as they indicate the trusting behavior of trustor i.e. how much trust/distrust trustor assigns to different types of users. In order to use trustor's trust for its out-neighbors to estimate trustor's trust for the trustee, we need to know how differently the out-neighbors and the trustee are trusted by the trustor. We estimate this difference by observing the trust assigned to the out-neighbor of trustor and trustee by some other common nodes.

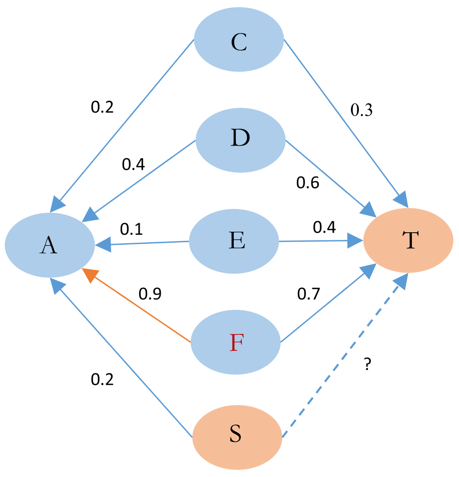


FIGURE 3. Finding difference between Trustee T and Trustor's out-neighbor A about being trusted by other users.

Consider Figure 3. We show a section of network showing one out-neighbor of trustor i.e. A. To use the trustor's trust of A in our framework we find the difference in trust received by A and T from common users i.e. C, D, E and F. Like the trustee's in-neighbors case, we use only those users that show similar trusting behavior as S, i.e. those who's trust in A is close to S's trust in A. So, for estimating S's trust in T using trustor's out-neighbor A, the set of common nodes is given as

$$C_2(A) = in(T) \cap \{i \in in(A) \text{ and } |w(i, A) - w(S, A)| \leq \varepsilon\} \tag{5}$$

In Figure 3, $C_2(A)$ includes C, D and E but no F since difference between F's trust in A and S's trust in A is greater than 0.5.

To find the difference between A and T about being trusted by other users for estimating S's trust in T, we find the average difference in trust rating of A and T by users in $C_2(A)$.

$$D_2(A) = \begin{cases} \frac{\sum_{I \in C_2(A)} w(I, T) - w(I, A)}{|C_2(A)|}, & |C_2(A)| > 0 \\ 0, & |C_2(A)| = 0 \end{cases} \tag{6}$$

$D_2(A)$ estimates on average how much higher or lower, nodes in $C_2(A)$ trust T than A. This can be considered as an approximate estimate of how much higher or lower S should trust T compared to A. The estimate based on trustors out-neighbor A denoted by $E_2(A)$ is given as

$$E_2(A) = w(S, A) + D_2(A) \tag{7}$$

To keep our framework robust, we also include those out-neighbors x of trustor for which $C_2(x)$ is empty. So, if difference cannot be obtained, we define $D_2(x) = 0$.

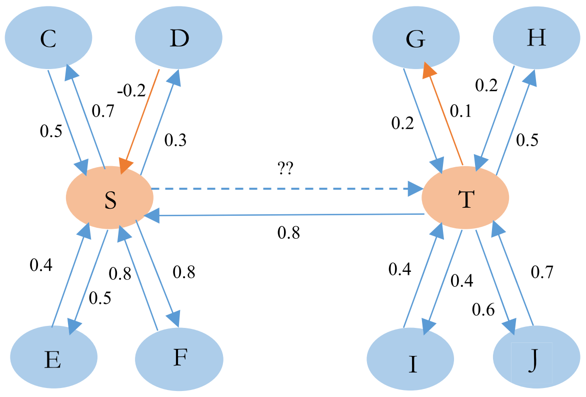


FIGURE 4. Finding how differently S and T trust their reciporcal neigbors from being trusted by them.

In Figure 3

$$D_2(A) = ((0.3 - 0.2) + (0.6 - 0.4) + (0.4 - 0.1))/3 = 0.2$$
$$E_2(A) = 0.2 + 0.067 = 0.4$$

This estimate $E_2(x)$ is also a form of co-citation trust propagation since users in C_2 (x) i.e. C, D, E each trust both A and T whereas S only trusts A, so according to co-citation propagation S may also trust T.

The partial trust estimate based on out-neighbors of trustor is the average of trust values estimated using each out-neighbor.

$$PTE_2 = \begin{cases} \frac{\sum_{i \in out(T)} E_2(i)}{|out(T)|}, & |out(T)| > 0 \\ \text{undefined}, & |out(T)| = 0 \end{cases} \quad (8)$$

3) PTE$_3$: TRUSTOR'S RECIPROCITY

Although trust is not symmetric i.e. if a user A trust user B, it doesn't mean B also trusts A or the level of trust is same in both directions. However, trust is not random, and many trust related networks exhibit a certain degree of reciprocity. Especially, e-commerce related trust networks have high degree of reciprocity. According to transpose trust propagation if A trust B then B may also trust A. We can use the reciprocity in our framework to improve accuracy and robustness. We can use reciprocity to predict trust from S to T only if there is a reciprocal trust link from T to S.

Consider Figure 4. It shows a section of network containing S, T, edge (T,S) and reciprocal neighbors of S and T. To estimate S's trust of T based on trustor's reciprocity and T trust rating of S, we need to know how differently S would trust T from T's trust of S. We find an approximate difference by using S's trust to and from other reciprocal neighbors. Here also we only use those reciprocal neighbors of S who's trust rating of S is close to T's trust rating of S since those have a more similar trusting behavior to T. The set of such common reciprocal neighbors is denoted by $C_3(S)$

$$C_3(S) = \{I \in \{in(S) \cap out(S)\} \text{ and } |w(T, S) - w(I, S)| \leq \varepsilon\} \quad (9)$$

In Figure 4, $C_3(S)$ includes C, E, F and not D since |w(T,S) – w(D,S)| >0.5.

The difference in trust that S assigns to its reciprocal neighbors from the trust it receives from them is then given as

$$D_3(S) = \begin{cases} \frac{\sum_{I \in C_3(S)} w(S, I) - w(I, S)}{|C_3(S)|}, & |C_3(S)| > 0 \\ 0, & |C_3(S)| = 0 \end{cases} \quad (10)$$

As in case of $D_1(x)$ and $D_2(x)$ if there are no similar reciprocal neighbors, we assign $D_3(x) = 0$ so that we can still use T's trust rating of S in our trust prediction framework. The partial trust estimate using trustor's reciprocity is

$$PTE_3 = \begin{cases} w(T, S) + D_3(S), & \text{edge (t, s) exist} \\ \text{undefined}, & \text{edge (t, s) not exists} \end{cases} \quad (11)$$

In Figure 4

$$D_3(S) = ((0.7 - 0.5) + (0.5 - 0.4) + (0.8 - 0.8))/3 = 0.1$$
$$PTE_3 = 0.8 + 0.1 = 0.9$$

4) PTE$_4$: TRUSTEE'S RECIPROCITY

In a similar way to trustor's reciprocity we can use trustee's reciprocity in our trust estimation framework if edge from T to S exists. We use reciprocal neighbors of trustee that have similar trust rating from T as T's trust rating of S to find the average difference in trust received by T from the trust it assigns to them.

$$C_4(T) = \{I \in \{in(T) \cap out(T)\} \text{ and } |w(T, S) - w(T, I)| \leq \varepsilon\} \quad (12)$$

In Figure 4, $C_4(T)$ includes H, I, J and not G. since |w(T,I) – w(T,S)| > 0.5.

$$D_4(T) = \begin{cases} \frac{\sum_{I \in C_4(T)} w(I, T) - w(T, I)}{|C_4(T)|}, & |C_4(T)| > 0 \\ 0, & |C_4(T)| = 0 \end{cases} \quad (13)$$

If there are no similar reciprocal neighbors we assign $D_4(T) = 0$ so that we can still use the edge (T,S) in our trust prediction framework. The partial trust estimation using trustor's reciprocity is

$$PTE_4 = \begin{cases} w(T, S) + D_4(T), & \text{edge(t, s)exists} \\ \text{undefined}, & \text{edge (t, s) not exists} \end{cases} \quad (14)$$

In Figure 4

$$D_4(T) = ((0.2 - 0.5) + (0.4 - 0.4) + (0.7 - 0.6))/3 = -0.067$$
$$PTE_4 = 0.8 + (-0.67) = 0.734$$

5) FINAL TRUST

The final trust is the weighted average of the partial trust estimates that have defined values. If all partial estimates are undefined final trust assigned is 0. (or any other default value like average trust rating etc.). For our datasets we keep the weights of all defined partial estimates as 1. let w_i be the weight for partial trust estimate PTE_i for i = 1,2,3,4

$$w_i = \begin{cases} 1, & PTE_i \text{ defined} \\ 0, & PTE_i \text{ undefined} \end{cases} \tag{15}$$

$$\text{Trust}(S, T) = \begin{cases} \frac{\sum PTI_i * w_i}{\sum w_i}, & \sum w_i > 0 \\ 0, & \sum w_i = 0 \end{cases} \tag{16}$$

where undefined $* \, 0 = 0$

B. INCREASED ACCURACY

In the basic framework we maximize the accuracy while ensuring trust is computable for maximum pairs of users i.e. the coverage is also maximized. However, we can further increase the accuracy of trust prediction albeit with a lower coverage. We can apply conditions to restrict which of the partial trust estimates are used or not used for final trust computation.

As mentioned earlier in basic frame work we define $D_i(x) = 0$ when $|C_i(x)| = 0$ in Eq. (2),(6),(10),(13) to maximize pairs for which trust can be computed. However, now since we want to increase accuracy, we define $D_i(x)$ as undefined if $|C_i(x)| = 0$ and therefore not use the information associated with that neighbor node. (or reciprocal edge for $D_3(x)$ and $D_4(x)$).

For trustee's in-neighbors and trustor's out-neighbors we only use the PTE if it is based on at least a minimum number of estimates (users) and the variation among the estimates is small.

For trustee's in-neighbors case let indef(T) be the set of in-neighbors x for which $|C_1(x)| > 0$ and therefore $D_1(x)$ is defined. We apply condition on |indef(S)| and standard deviation of individual estimates $E_1(x)$.

$$\mu_1 = \frac{\sum_{I \in indef(T)} E_1(I)}{|indef(T)|} \tag{17}$$

$$std_1 = \sqrt{\frac{\sum_{I \in indef(T)} (E_1(I) - \mu_1)^2}{|indef(T)|}} \tag{18}$$

PTE_1 would be defined as

$$PTE_1 = \begin{cases} \mu_1, & |indef(t)| \geq tn \text{ and } std_1 \leq ts \\ \text{undefine}, & \text{otherwise} \end{cases} \tag{19}$$

Similarly, for trustor's out-neighbors case, outdef(S) is the set of out-neighbors x for which $|C_2(x)| > 0$ and $D_2(x)$ is defined. We apply condition on |outdef(S)| and standard deviation of individual estimates $E_2(x)$

$$\mu_2 = \frac{\sum_{I \in outdef(T)} E_2(I)}{|outdef(T)|} \tag{20}$$

$$std_2 = \sqrt{\frac{\sum_{I \in outdef(T)} (E_2(I) - \mu_2)^2}{|outdef(T)|}} \tag{21}$$

$$PTE_2 = \begin{cases} \mu_2, & |outdef(t)| \geq tn \text{ and } std_2 \leq ts \\ \text{undef}, & \text{otherwise} \end{cases} \tag{22}$$

For the trustor reciprocal case we get the partial trust estimate from the difference $D_3(S)$ and the reciprocal trust. The difference $D_3(S)$ will be defined only if $|C_3(S)| > 0$ and we also apply a restriction on variation in individual differences. Since $D_3(S)$ is the mean of individual differences

$$std_3 = \sqrt{\frac{\sum_{I \in C_3(S)} ((w(S, I) - w(I, S)) - D_3(S))^2}{|C_3(S)|}} \tag{23}$$

$$PTE_3 = \begin{cases} w(T, S) + D_3(S), \text{ if } |C_3(S)| > 0, std_3 \leq ts, \\ \quad (t, s) \text{ exists} \\ \text{undefined, otherwise} \end{cases} \tag{24}$$

Similarly, for trustee's reciprocal neighbors $D_4(T)$ will be defined only if $|C_4(T)| > 0$ and we apply condition on variation of individual differences.

$$std_4 = \sqrt{\frac{\sum_{I \in C_4(T)} ((w(I, T) - w(T, I)) - D_4(T))^2}{|C_4(T)|}} \tag{25}$$

$$PTE_4 = \begin{cases} w(T, S) + D_4(T), \text{ if } |C_4(T)| > 0, std_4 \leq ts, \\ \quad (t, s) \text{ exists} \\ \text{undefined, otherwise} \end{cases} \tag{26}$$

The final trust will be computed in same way as in the basic framework using Eq. (15) and (16).

C. ALGORITHM

In Alg.1 to 5 we show the pseudocode of our basic framework. In Alg.5, PTE_1, PTE_2, PTE_3, PTE_4, w_1, w_2, w_3 and w_4 are global variables initialized to 0. In line 3-6 of Alg.5,

Algorithm 1 ComputePTE1(G,S,T)

Input: G (V, E, W), trustor node S, trustee node T.

Output: PTE_1, w_1

```
1.  cnt = temp = 0
2.  for each node B in in(T)
3.    d = c = 0
4.    for each node I in out(B)
5.      if $I \neq T$ and $I \in$ out(S)
6.        if |w(B,I)-w(B,T)| $\leq$ 0.5
7.          d = d+ (w(S,I)-w(B,I))
8.          c = c+1
9.    if c $\geq$ 1
10.     d = d/c
11.   temp = temp+ (w(,B,T)+d)
12.   cnt = cnt+1
13. if cnt $\geq$ 1
14.   $PTE_1$ = temp/cnt
15.   $w_1$ = 1
```

Algorithm 2 ComputePTE2(G,S,T)

Input: G (V, E, W), trustor node S, trustee node T.

Output: PTE_2, w_2

```
1.  cnt = temp = 0
2.  for each node A in out(S)
3.    d = c = 0
4.    for each node I in in(A)
5.      if $I \neq S$ and $I \in in(T)$
6.        if $|w(I,A)-w(S,A)| \leq 0.5$
7.          d = d+ (w(I,T)-w(I,A))
8.          c = c+1
9.    if $c \geq 1$
10.     d = d/c
11.   temp = temp + (w(S,A)+d)
12.   cnt = cnt+1
13. if $cnt \geq 1$
14.   $PTE_2$ = temp/cnt
15.   $w_2 = 1$
```

Algorithm 3 ComputePTE3(G,S,T)

Input: G (V, E, W), trustor node S, trustee node T.

Output: PTE_3, w_3

```
1.  if (T,S) exists
2.    d = c = 0
3.    for each node I in out(S)
4.      if (I,S) exists
5.        if $|w(T,S)-w(I,S)| \leq 0.5$
6.          d = d+ (w(S,I)-w(I,S))
7.          c = c+1
8.    if $c \geq 1$
9.      d = d/c
10.   $PTE_3$ = w(T,S)+d
11.   $w_3 = 1$
```

Algorithm 4 ComputePTE4(G,S,T)

Input: G (V, E, W), trustor node S, trustee node T.

Output: PTE_4, w_4

```
1.  if (T,S) exists
2.    d = c = 0
3.    for each node I in in(T)
4.      if (T,I) exists
5.        if $|w(T,S)-w(T,I)| \leq 0.5$
6.          d = d+ (w(I,T)-w(T,I))
7.          c = c+1
8.    if $c \geq 1$
9.      d = d/c
10.   $PTE_4$ = w(T,S)+d
11.   $w_4 = 1$
```

Algorithm 5 ComputeTrust (G,S,T)

Input: G (V, E, W), trustor node S, trustee node T.

Output: Trust (S,T)

```
1.  Global variables $PTE_1, PTE_2, PTE_3, PTE_4, w_1, w_2, w_3, w_4$
2.  Initialize: $PTE_1 = PTE2 = PTE_1 = PTE_1 = w_1 = w_2 = w_3 = w_4 = sumw = 0$
3.  computePTE1(G,S,T)
4.  computePTE2(G,S,T)
5.  computePTE3(G,S,T)
6.  computePTE4(G,S,T)
7.  sumw = $w_1+w_2+w_3+w_4$
8.  if sumw = 0
9.    trust(S,T) = 0
10. else
11.   trust(S,T) = $(PTE_1^* w_1+PTE_2^* w_2+PTE_3^* w_3 +PTE_4^* w_4)$/sumw
```

Alg.1-4 are used to compute PTEs and weights. As it can be seen in Alg.1-4 if a PTE is defined the weight will be 1 otherwise it retains its initial value of 0. As described in in Eq. (16), in line 7,8 of Alg.5 if all weights are 0 final trust will be 0 otherwise it is the average of defined PTEs. We only show the pseudocode for basic framework case as it its straightforward to include the restrictions for increased accuracy case.

V. EXPERIMENTAL RESULTS

To verify the accuracy of our framework in predicting trust, we performed a series of experiments on four real world trust related datasets. We compare the performance of our framework with other recent trust prediction methods. Our framework out performs existing methods in terms of accuracy and robustness to missing edges. In experiment 1 and 2 we use our basic framework without any additional restrictions to compare our method with existing methods whereas in experiment 3 we demonstrate increased accuracy version of our method.

A. DATASET DESCRIPTION

We used four real-world trust related network datasets that have been used by various researchers to evaluate their trust computation methods. The Bitcoin and Advogato networks have explicit trust values as edge weights whereas the WikiRFA network has implicit trust values. Table 2 shows the size of each dataset.

TABLE 2. Size of networks.

Datasets	#nodes	#edges
Wiki RFA	9654	104554
Bitcoin Alpha	3783	24186
Bitcoin OTC	5881	35592
Advogato	5417	51327

1) BITCOIN OTC AND BITCOIN ALPHA

These datasets are obtained from two bitcoin exchanges Bitcoin OTC and Bitcoin Alpha. These websites allow users to rate others according to how much they trust them. The scale is from $+10$ to -10. $+10$ indicates maximum trust while -10 is maximum distrust. The data is downloaded and scaled to interval [-1,1] by [15] and available at [19].

2) ADVOGATO

Advogato is an online social network for open source software developers. It allows members to rate the ability of other members at four levels. Observer, Apprentice, Journeyer, and Master. These ratings are taken as a level of trust in another member's ability. We map Observer, Apprentice, Journeyer, and Master to 0.1,0.4,0.7 and 0.9 respectively. This dataset has been widely used in evaluating trust related matrices and algorithms. We use the snapshot of the network taken at 2014-07-07 [20].

3) WIKIPEDIA RFA

This is Wikipedia request for adminship dataset where an edge (u,v) represents a vote of u for v to be become an administrator. The weight is -1 for negative, 0 for neutral and $+1$ for positive. The comments in the votes are analyzed by performing sentiment analysis by [15] using VADER sentiment engine. The difference of positive and negative sentiment score is used as the weight which lies in interval $[-1,1]$. The dataset has been used to evaluate fairness and goodness [15] and TOW [17] and is available at [19].

B. EXISTING METHODS

We compare the performance of our method with following existing methods

1) RECIPROCAL

This is the simplest method in which the estimated trust from S to T is equal to weight of edge (T,S) if it exists or 0 otherwise.

2) MIN-MAX ALL

According to this method the strength of trust path is equal to the weight of its minimum weight edge and strongest path is path with maximum minimum weight edge. The in-neighbor of T having strongest path from S is considered as the most reliable in-neighbor and its direct trust to T is compared with strength of strongest path. The minimum of the two is the estimated trust value. In case of most reliable multiple in-neighbors we use the maximum of estimated trust values from those in-neighbors as the final trust value. This aggregation strategy is used in [9]–[11].

3) MIN-MAX SHORTEST

This method is similar to Min-Max all however only the shortest paths are considered when finding the strongest path from S to in-neighbors of T.

4) FAIRNESS AND GOODNESS

Reference [15] introduce two global matrices fairness and goodness of nodes in weighted signed networks. Fairness is how fairly a user rates other user's likeability or trust whereas goodness indicates how much a user is liked/trusted by others. Trust from S to T is product of fairness of S and goodness of T.

5) TOW(TUG OF WAR)

Reference [17] Describe three characteristics of nodes i.e. controversy, eclecticism, and reciprocity. Trust is treated as three-way tug of war between controversy of trustor, eclecticism of trustee and reciprocity of trustor.

C. EVALULATION MATRICES

To evaluate our proposed method, we use the following two measures

1) ROOT MEAN SQUARE ERROR

It is the square root of average squared difference between the actual trust values and the predicted or estimated trust values. Smaller value indicates higher accuracy.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (pred_i - act_i)^2}{n}}$$

2) PEARSON CORRELATION COEFFICIENT

It measures correlation or trend between the predicted and the actual trust value. Its value lies between -1 and 1 with value closer to 1 indicating higher correlation.

$$\mathrm{PCC} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{act_i} - \overline{act}_{\mathrm{i}})(\mathrm{pred_i} - \overline{\mathrm{pred_i}})}{\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{act_i} - \overline{act}_{\mathrm{i}})}\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{pred_i} - \overline{\mathrm{pred_i}})}}$$

D. EXPERIMENT 1

The first experiment performed was leave one out. This is experiment is commonly used to evaluate the accuracy of continuous trust prediction and edge weight prediction algorithms [15], [17]. The accuracy of predicted trust value for a given pair of users is compared to the actual weight of edge in the network. For each network we remove an edge, apply our trust estimation framework and other existing methods on the network without that edge, and compare the predicted trust value with the actual trust value i.e. weight of the removed edge. We repeat this process for every edge in the network to find the average RMSE and PCC. Table 3 shows the calculated RMSE and PCC values. On all the datasets our method has the least RMSE and highest PCC. On WikiRFA our results show less improvement and are similar to TOW, this could be because our method relies on trust propagation operations and is more suitable for trust networks with explicit trust values, whereas in WikiRFA doesn't have explicit trust values but trust is inferred from comments using sentiment analysis. In [15] the authors apply various methods like Triadic Balance, Status Theory, Bias and Deserve, Eigen Trust

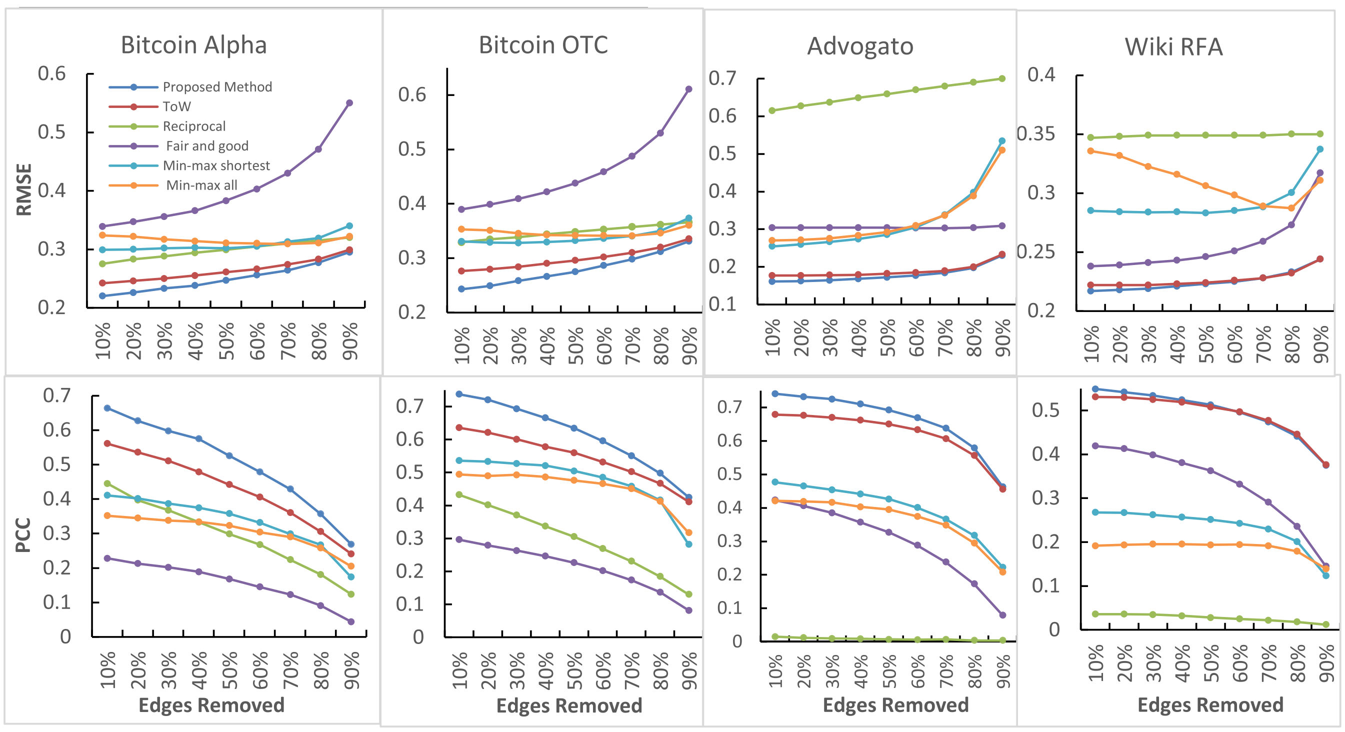


FIGURE 5. RMSE and PCC for leave N out experiment for four networks. The horizontal axis shows percentage of random edges removed. Our proposed method has least RMSE and highest PCC.

TABLE 3. RMSE,PCC on four datasets by existing and propsed methods. Proposed method has the least RMSE and highest PCC.

	B-Alpha	B-OTC	Advagato	WikiRFA
Reciprocal	0.27,0.47	0.33,0.46	0.60,0.01	0.35,0.04
fairness goodness	0.33,0.24	0.38,0.31	0.30,0.44	0.24,0.43
TOW	0.24,0.59	0.27,0.65	0.18,0.68	**0.22**,0.54
minmax short	0.30,0.41	0.33,0.54	0.25,0.48	0.29,0.27
minmax all	0.33,0.35	0.36,0.49	0.27,0.43	0.34,0.19
Proposed method	**0.21,0.68**	**0.24,0.75**	**0.16,0.75**	**0.22,0.55**

etc. on Bitcoin and WikiRFA datasets to compare RMSE and PCC with that of their proposed method i.e. Fairness and Goodness. Our method gives better RMSE and PCC then those methods too.

E. EXPERIMENT 2

There may be situations where parts of network may not be visible due to privacy reasons or nodes and their trust ratings are marked as unreliable or malicious and therefore cannot be used. The trust prediction method should be robust in case of missing edges. To evaluate and compare the robustness of our framework with other methods we performed leave N% out experiment. In this experiment for each dataset we randomly remove 10%, 20%, up to 90% of edges and then use the remaining network to predict the weight of the removed edges. Since the edges removed are random, we repeat each this process 20 times and find the average RMSE and PCC.

TABLE 4. RMSE,PCC,Coverage on four datasets by increased accuracy version of propsed method with differnent thresold values tn and ts.

	tn=5	tn=10
B-Alpha: ts=0.2	0.18, 0.68, 87%	0.17, 0.68, 86%
B-OTC: ts=0.2	0.19, 0.80, 85%	0.18, 0.81, 83%
Advagato: ts=0.15 w_3=0, w_4=0	0.12, 0.81, 67%	0.11, 0.82, 57%
WikiRFA: ts=0.2 w_3=0, w_4=0	0.19, 0.56, 65%	0.19, 0.56, 59%

We apply this experiment on other methods and compare the results with our method. Figure 5 shows the results. As it can be seen the performance of our method degrades slowly and is better than then other methods both in terms of RMSE and PCC especially on the bitcoin datasets.

F. EXPERIMENT 3

In this experiment we evaluated increased accuracy version of our framework. As mentioned earlier there is a tradeoff between accuracy and coverage. We tried different values of threshold tn and ts and measured the accuracy i.e. RMSE, PCC and coverage. Table 4 shows the results with $tn = 5$ and $tn = 10$ when $ts = 0.2$ for Bitcoin and WikiRFA datasets, and 0.15 for Advogato dataset. Since WikiRFA and Advogato dataset have low reciprocity, in this experiment we don't use PTE_3 and PTE_4 for these datasets i.e. we set the weights w_3 and w_4 as 0. Comparing RMSE and PCC obtained in this experiment to that of Table 3 for bitcoin datasets, our method gives much better accuracy than any other method at good coverage of more than 80%. For Advagato and WikiRFA datasets, although the coverage gets low when we increase the accuracy it still shows that we can increase accuracy on these datasets too by applying the restrictions.

VI. CONCLUSION

In this paper we propose a method for accurate and efficient estimation of trust and distrust in online social networks. Based on idea of co-citation and transpose trust propagation we show that difference between two users in trusting or being trusted by other users can be used for accurate trust estimation. Our method gives better accuracy in terms of RMSE and PCC then existing methods.

REFERENCES

[1] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for Online service provision," *Decis. Support Syst.*, vol. 43, no. 2, pp. 618–644, Mar. 2007.

[2] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowl.-Based Syst.*, vol. 46, pp. 109–132, Jul. 2013.

[3] B. Yang, Y. Lei, J. Liu, and W. Li, "Social collaborative filtering by trust," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1633–1647, Aug. 2017.

[4] Y. Ruan and A. Durresi, "A survey of trust management systems for Online social communities–trust modeling, trust inference and attacks," *Knowl.-Based Syst.*, vol. 106, pp. 150–163, Aug. 2016.

[5] W. Jiang, G. Wang, M. Z. A. Bhuiyan, and J. Wu, "Understanding graph-based trust evaluation in Online social networks: Methodologies and challenges," *ACM Comput. Surv.*, vol. 49, no. 1, pp. 1–35, May 2016.

[6] J. A. Golbeck, "Computing and applying trust in web-based social networks," Ph.D. dissertation, Univ. Maryland College Park, College Park, MD, USA, 2005.

[7] P. Massa and P. Avesani, "Controversial users demand local trust metrics: An experimental study on epinions. com community," in *Proc. 20th Nat. Conf. AAAI*, 2005, pp. 121–126.

[8] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The Eigen trust algorithm for reputation management in p2p networks," in *Proc. 12th ACM Int. Conf. World Wide Web*, May 2003, pp. 640–651.

[9] W. Jiang and G. Wang, "SWTrust: Generating trusted graph for trust evaluation in Online social networks," in *Proc. IEEE 10th Int. Conf. Trust, Secur. Privacy Comput. Commun.*, Nov. 2011, pp. 320–327.

[10] Y. A. Kim and H. S. Song, "Strategies for predicting local trust based on trust propagation in social networks," *Knowl.-Based Syst.*, vol. 24, no. 8, pp. 1360–1371, Dec. 2011.

[11] S.-U. Nasir and T.-H. Kim, "Fast trust computation in Online social networks," *IEICE Trans. Commun.*, vol. E96.B, no. 11, pp. 2774–2783, 2013.

[12] W. Jiang, J. Wu, F. Li, G. Wang, and H. Zheng, "Trust evaluation in Online social networks using generalized network flow," *IEEE Trans. Comput.*, vol. 65, no. 3, pp. 952–963, Mar. 2016.

[13] A. Mishra and A. Bhattacharya, "Finding the bias and prestige of nodes in networks based on trust scores," in *Proc. 20th ACM Int. Conf. World Wide Web*, 2011, pp. 567–576.

[14] Y. Yao, H. Tong, X. Yan, F. Xu, and J. Lu, "Multi-aspect+ transitivity+ bias: An integral trust inference model," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 7, pp. 1706–1719, Jul. 2014.

[15] S. Kumar, F. Spezzano, V. S. Subrahmanian, and C. Faloutsos, "Edge weight prediction in weighted signed networks," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 221–230.

[16] P. Gao, H. Miao, J. S. Baras, and J. Golbeck, "STAR: Semiring trust inference for trust-aware social recommenders," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 301–308.

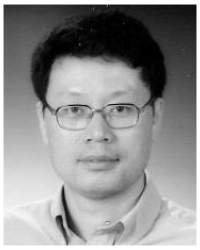
[17] K. Akilal, H. Slimani, and M. Omar, "A very fast and robust trust inference algorithm in weighted signed social networks using controversy, eclecticism, and reciprocity," *Comput. Secur.*, vol. 83, pp. 68–78, Jun. 2019.

[18] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in *Proc. 13th Conf. World Wide Web (WWW)*, 2004, pp. 403–412.

[19] [Online]. Available: https://cs.stanford.edu/~srijan/wsn/

[20] [Online]. Available: http://www.trustlet.org/datasets/advogato/

SAFI ULLAH NASIR received the master's degree in computer science from the Lahore University of Management Sciences (LUMS), Pakistan, in 2006. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Hanyang University, South Korea, funded by the Higher Education Commission, Pakistan. From 2006 to 2011, he worked as a Lecturer with the Department of Computer Science, University of Sargodha, Pakistan. His research interest includes fast and accurate trust computation for online social networks.



TAE-HYUNG KIM received the B.S. and M.S. degrees in computer science engineering from Seoul National University, in 1986 and 1988, respectively, and the Ph.D. degree from the University of Maryland, College Park, in 1996. From 1988 to 1990, he worked with Hyundai Electronics Company Ltd. From 1996 to 1999, he worked with Hyundai Information Technology Company Ltd., South Korea. In 1999, he joined Hanyang University as a Faculty Member.