Multi-Aspect + Transitivity + Bias: An Integral Trust Inference Model

Yuan Yao, Hanghang Tong, Xifeng Yan, *Member, IEEE,* Feng Xu, and Jian Lu

Abstract—Inferring the pair-wise trust relationship is a core building block for many real applications. State-of-the-art approaches for such trust inference mainly employ the *transitivity* property of trust by propagating trust along connected users, but largely ignore other important properties such as trust bias, multi-aspect, etc. In this paper, we propose a new trust inference model to integrate all these important properties. To apply the model to both binary and continuous inference scenarios, we further propose a family of effective and efficient algorithms. Extensive experimental evaluations on real data sets show that our method achieves *significant* improvement over several existing benchmark approaches, for both quantifying numerical trustworthiness scores and predicting binary trust/distrust signs. In addition, it enjoys linear scalability in both time and space.

Index Terms—Trust inference, trust prediction, transitivity property, multi-aspect property, latent factors, trust bias

1 Introduction

Trust inference aims to build new pair-wise trust relationship based on the existing observations. It is a core building block in many high-impact application domains, such as social networks [1], e-commerce [2], peer-to-peer networks [3], and semantic Web [4], etc.

Generally speaking, there are three important properties behind the existing trust inference models, namely, (1) *transitivity*, (2) *multi-aspect* and (3) *trust bias*. First, rooting in the social structural balance theory [5], the *transitivity* property of trust [6] captures the intuition that trust can be propagated along existing trust relationships. For example, if *Alice* trusts *Bob* and *Bob* trusts *Carol*, *Alice* might also trust *Carol* to some extent. Second, the basic assumption behind the *multi-aspect* property is that trust is the composition of multiple factors, and different users may have different preferences for these factors [7], [8]. For example, in e-commerce, some users might care more about the factor of delivering time, whereas others give a higher weight to the factor of product price. Finally, it was discovered in sociology a long time ago that *trust bias* is an integral part in the final trust decision [9]. For instance, some users tend to generously give higher trust ratings than others, and some users have relatively higher capability in terms of being trusted than others.

- *Y. Yao, F. Xu, and J. Lu are with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210046, China. E-mail: yyao@smail.nju.edu.cn; {xf, lj}@nju.edu.cn.*
- *H. Tong is with the School of Computing, Informatics, Decision Systems Engineering, Arizona State University, Phoenix, AZ 85281 USA. E-mail: htong6@asu.edu.*
- *X. Yan is with the University of California at Santa Barbara, Santa Barbara, CA 93106-5110 USA. E-mail: xyan@cs.ucsb.edu.*

Manuscript received 28 Apr. 2013; revised 25 July 2013; accepted 17 Aug. 2013. Date of publication 22 Aug. 2013; date of current version 9 July 2014.
Recommended for acceptance by F. Bonchi.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier 10.1109/TKDE.2013.147

Despite the extensive existing work (see Section 6 for a review), these important properties (i.e., transitivity, multi-aspect and bias) have been largely examined in isolation. The vast majority of existing work, referred to as *trust propagation* models as a whole, is based on the trust transitivity property. However, many trust propagation models suffer from the scalability issue [10]. On the other hand, the few existing multi-aspect trust inference methods (e.g., [11], [12]) require as its input some side information (e.g., the delivering time as well as user's preference for it, etc) in addition to the locally-generated trust ratings, and therefore become infeasible in many trust networks where such side information may not be available. Finally, it was not until the very recent years did the computer science community begin to incorporate the trust bias into the inference process [13]. How can we integrate all these three important properties into a single model that is solely based on the existing trust ratings, so as to maximally boost the inference performance as well as its applicability? This is the central question that we aim to answer in this paper.

We start by proposing a multi-aspect trust inference model. The heart of our method is to view the problem as a recommendation problem, and hence opens the door to the rich methodologies in the field of collaborative filtering. The proposed multi-aspect model directly characterizes multiple *latent* factors for each trustor and trustee from the locally-generated trust relationships. Based on that, we propose to incorporate the trust bias as *specified* aspects and automatically learn the relative weights between latent and specified factors. Furthermore, we extend this model to incorporate trust propagation to further improve inference accuracy. Finally, we elaborate the difference between continuous trust inference (i.e., quantifying numerical trustworthiness scores) and binary trust inference (i.e., predicting binary trust/distrust signs), and propose a family of algorithms to solve them, respectively.

In summary, the main contributions of this work are as follows:

TABLE 1
Symbols

Symbol	Definition and Description
$\mathbf{T}$	the partially observed trust matrix
$\mathbf{F}, \mathbf{G}$	the characterized trustor and trustee matrices
$\mathbf{F}_0, \mathbf{G}_0$	the sub-matrix of $\mathbf{F}$ and $\mathbf{G}$
$\mathbf{T}'$	the transpose of matrix $\mathbf{T}$
$\mathbf{T}(i, j)$	the element at the i^{th} row and j^{th} column of $\mathbf{T}$
$\mathbf{T}(i, :)$	the i^{th} row of matrix $\mathbf{T}$
$\mathcal{K}$	the set of observed trustor-trustee pairs in $\mathbf{T}$
μ	the global bias
$\mathbf{x}, \mathbf{y}$	the vector of trustor bias and trustee bias
$\mathbf{x}(i)$	the i^{th} element of vector $\mathbf{x}$
$\mathbf{z}_{ij}$	the vector of propagation elements for trustor-trustee pair (i, j)
n	the number of users
p, r	the number of bias and latent factors
s	total number of factors, $s = p + r$
t	the maximum propagation step
α_i, β_j	the weights/coefficients for bias and propagation
u, v	the trustor and the trustee
m_1, m_2, m_3	the maximum iteration number
ξ_1, ξ_2, ξ_3	the threshold to terminate the iteration

1) *Trust Models.* We proposed a new trust model. It (a) integrates transitivity, multi-aspect and trust bias into one single trust inference model; and (b) directly characterizes the multi-aspect trust relationship solely based on locally-generated trust ratings. It can admit the rich methodologies from collaborative filtering, and it is flexible to model the trust bias as specified factors and further learn their relative weights.
2) *Inference Algorithms.* We proposed a family of inference algorithms. It finds a local optimal solution with linear complexity. It applies to both binary and continuous trust inference scenarios.
3) *Performance Improvements.* We conducted extensive experimental evaluations on four widely used benchmark data sets, and empirically observed significant performance improvements in both effectiveness and efficiency. In the continuous trust inference scenario, for example, our MATRI outperforms the best known existing methods by 26.7% - 40.7% in terms of prediction accuracy; and by pre-computation, our MATRI is much faster in terms of on-line response, achieving up to 7 orders of magnitude speedup. Finally, the pre-computation stage itself of the proposed MATRI scales *linearly* wrt the size of the input data set, indicating that it is suitable for large data sets.

The rest of the paper is organized as follows. Section 2 presents the problem definition. Section 3 describes our optimization formulation to incorporate trust bias and propagation. Section 4 presents the inference algorithm. Section 5 presents experimental results. Section 6 reviews related work. Section 7 concludes the paper.

2 PROBLEM DEFINITION

In this section, we formally define our trust inference problem. Table 1 lists the main symbols we use throughout the paper.

Following conventions, we use bold capital letters for matrices, and bold lower case letters for vectors. For example, we use a partially observed matrix $\mathbf{T}$ to model the locally-generated trust relationships, where the existing/observed trust relationships are represented as non-zero trust ratings and non-existing/unobserved relationships are represented as '?'. As for the observed trust rating, we represent it as a real number between 0 and 1 (a higher rating means more trustworthiness) in the continuous case, and +1/-1 (+1 means trust and -1 means distrust) in the binary case. We use calligraphic font $\mathcal{K}$ to denote the set of observed trustor-trustee indices in $\mathbf{T}$. Similar to MATLAB, we also denote the i^{th} row of matrix $\mathbf{T}$ as $\mathbf{T}(i, :)$, and the transpose of a matrix with a prime. In addition, we denote the number of users as n and the number of characterized factors as s. Without loss of generality, we assume that the goal of our trust model is to infer the unseen trust relationship from the user u to another user v, where u is the trustor and v is the unknown trustee to u.

Based on these notations, we first define the basic trust inference problem as follows:

Problem 1. *The Basic Trust Inference Problem*

Given: *an $n \times n$ partially observed trust matrix $\mathbf{T}$, a trustor u, and a trustee v, where $1 \leqslant u, v \leqslant n$ $(u \neq v)$ and $\mathbf{T}(u, v) = '?'$;*
Find: *the estimated trustworthiness score/sign $\hat{\mathbf{T}}(u, v)$.*

In the above problem definition, given a trustor-trustee pair, the only information we need as input is the locally-generated trust ratings (i.e., the partially observed matrix $\mathbf{T}$). The goal of trust inference is to infer the new trust ratings (i.e., unseen/unobserved trustworthiness scores in the partially observed matrix $\mathbf{T}$) by collecting the knowledge from existing trust relationships.

As mentioned before, one of our goals is to capture the multi-aspect property of trust. In this paper, we propose a multi-aspect model for such trust inference in Problem 1. That is, we want to infer an *$n \times s$ trustor matrix* $\mathbf{F}$ whose element indicates to what extent the corresponding person trusts others wrt a specific aspect/factor. Similarly, we want to infer another *$n \times s$ trustee matrix* $\mathbf{G}$ whose element indicates to what extent the corresponding person is trusted by others wrt a specific aspect/factor. Such trustor and trustee matrices are in turn used to infer the unseen trustworthiness scores. Based on the basic trust inference problem, we define the multi-aspect trust inference problem as follows:

Problem 2. *The Multi-Aspect Trust Inference Problem*

Given: an $n \times n$ partially observed trust matrix $\mathbf{T}$, the number of factors s, a trustor u, and a trustee v, where $1 \leqslant u, v \leqslant n$ $(u \neq v)$ and $\mathbf{T}(u, v) = '?'$;
Find: *(1) an $n \times s$ trustor matrix $\mathbf{F}$ and an $n \times s$ trustee matrix $\mathbf{G}$; (2) the estimated trustworthiness score/sign $\hat{\mathbf{T}}(u, v)$.*

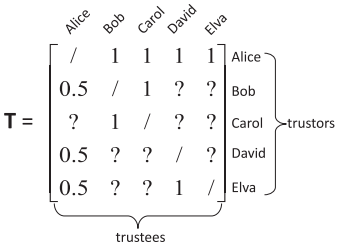
2.1 An Illustrative Example

To further illustrate our multi-aspect trust inference problem (Problem 2), we give an intuitive example for inferring the continuous trustworthiness scores as shown in Fig. 1.

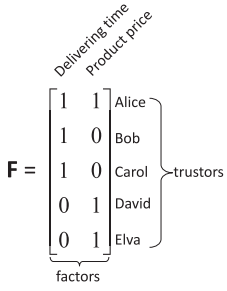
In this example, we observe several locally-generated pair-wise trust relationships between five users (e.g., '*Alice*',

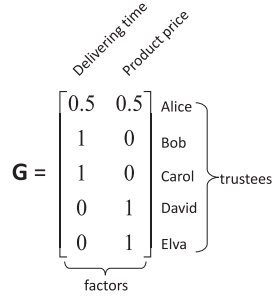
Trustor	Trustee	Rating
Alice	Bob	1
Alice	Carol	1
Alice	David	1
Alice	Elva	1
Bob	Alice	0.5
Bob	Carol	1
Carol	Bob	1
David	Alice	0.5
Elva	Alice	0.5
Elva	David	1

(a)



(b)





(c)

Fig. 1. Illustrative example for multi-aspect trust inference problem. (a) Observed locally-generated pair-wise trust relationships. (b) Partially observed trust matrix **T**. (c) Inferred trustor matrix **F** and trustee matrix **G**.

'*Bob*', '*Carol*', '*David*', and '*Elva*') as shown in Fig. 1(a). Each observation contains a trustor, a trustee, and a numerical trust rating from the trustor to the trustee. We then model these observations as a 5×5 partially observed matrix $\mathbf{T}$ (see Fig. 1(b)) where $\mathbf{T}(i,j)$ is the trust rating from the i^{th} user to the j^{th} user if the rating is observed and $\mathbf{T}(i,j)=$'?' otherwise. Notice that we do not consider self-ratings and thus represent the diagonal elements of $\mathbf{T}$ as '/'. By setting the number of factors $s=2$, our goal is to infer two 5×2 matrices $\mathbf{F}$ and $\mathbf{G}$ (see Fig. 1(c)) from the input matrix $\mathbf{T}$. Each row of the two matrices is for the corresponding user, and each column of the matrices represents a certain aspect/factor in trust inference (e.g., 'delivering time', 'product price', etc). For example, we can see that *Alice* trusts others strongly wrt both 'delivering time' and 'product price' (based on $\mathbf{F}$), and she is in turn moderately trusted by others wrt these two factors (based on $\mathbf{G}$). On the other hand, both *Bob* and *Carol* put more emphasis on the delivering time, while *David* and *Elva* care more about the product price.

Once $\mathbf{F}$ and $\mathbf{G}$ are inferred, we can use these two matrices to estimate the unseen trustworthiness scores (i.e., the '?' elements in $\mathbf{T}$). For instance, the trustworthiness from *Carol* to *Alice* can be estimated as $\hat{\mathbf{T}}(3,1)=\mathbf{F}(3,:)\mathbf{G}(1,:)'=0.5$. This estimation is reasonable because *Carol* has the same preference as *Bob* and the trustworthiness score from *Bob* to *Alice* is also 0.5.

In the next two sections, we will mainly focus on (1) how to infer $\mathbf{F}$ and $\mathbf{G}$; and (2) how to incorporate trust bias and trust transitivity (i.e., trust propagation).

3 The Optimization Formulation

In this section, we present our optimization formulation to integrate all the three important properties in trust inference, including multi-aspect, trust bias and trust transitivity (i.e., trust propagation). We start with the continuous trust inference for inferring numerical trustworthiness scores, and then present some necessary adaptations for the binary case of predicting trust/distrust signs. Finally, we discuss some generalizations of our formulation.

3.1 Proposed Formulation for Continuous Case

For the continuous case, we first present the basic form to capture the multi-aspect of trust, and then show how to incorporate three types of trust bias and four groups of trust propagation.

3.1.1 The Basic Formulation

Formally, the continuous case of Problem 2 can be formulated as the following optimization problem:

$$\min_{\mathbf{F},\mathbf{G}} \sum_{(i,j)\in\mathcal{K}} (\mathbf{T}(i,j)-\mathbf{F}(i,:)\mathbf{G}(j,:)')^2+\lambda||\mathbf{F}||_{fro}^2+\lambda||\mathbf{G}||_{fro}^2, \quad (1)$$

where λ is a regularization parameter; $||\mathbf{F}||_{fro}$ and $||\mathbf{G}||_{fro}$ are the Frobenius norm of the trustor and trustee matrices, respectively.

By this formulation, it aims to minimize the squared error on the set of observed trust ratings. Notice that in Eq. (1), we have two additional regularization terms ($||\mathbf{F}||_{fro}^2$ and $||\mathbf{G}||_{fro}^2$) to improve the solution stability. The parameter $\lambda\geq0$ controls the amount of such regularization.

A Collaborative Filtering Metaphor: As mentioned in introduction, we view the trust inference problem as a recommendation problem. To be specific, in the trust matrix $\mathbf{T}$, if we treat its rows (i.e., trustors) as 'users'; its columns (i.e., trustees) as 'items'; and its entries (i.e., trustworthiness scores) as 'item ratings', the optimization problem in Eq. (1) resembles the same form as that of so-called factorization-based collaborative filtering [14]. This viewpoint opens the door to the rich methodologies in collaborative filtering to capture the multi-aspect of trust.

3.1.2 Incorporating Trust Bias

The formulation in Eq. (1) can naturally incorporate some prior knowledge such as trust bias into the inference procedure. In this paper, we explicitly consider the following three types of trust bias (i.e., $p=3$ where p is the number of bias factors): *global bias*, *trustor bias*, and *trustee bias*, although other types of bias can be incorporated in a similar way.

Global bias: The global bias represents the average level of trust in the community. The intuition behind this is that users tend to rate optimistically in some reciprocal environments (e.g., e-commerce) while they are more conservative in others (e.g., security-related applications). As a result, it might be useful to take such global bias into account and we model it as a scalar μ.

Trustor bias: The trustor bias is based on the observation that some trustors tend to generously give higher trust ratings than others. This bias reflects the propensity of a given trustor to trust others, and it may vary a lot among different trustors. Accordingly, we can model the trustor bias as

vector $\mathbf{x}$ with $\mathbf{x}(i)$ indicating the trust propensity of the i^{th} trustor.

Trustee bias: The third type of bias aims to characterize the fact that some trustees might have relatively higher capability in terms of being trusted than others. Similar to the second type of bias, we model this type of bias as vector $\mathbf{y}$, where $\mathbf{y}(j)$ indicates the overall capability of the j^{th} trustee compared to the average.

Each of these three types of bias can be represented as a *specified* factor for our model, respectively. By incorporating such bias into Eq. (1), we have the following formulation:

$$\begin{aligned}\min_{\mathbf{F},\mathbf{G}} \sum_{(i,j)\in\mathcal{K}} (\mathbf{T}(i,j) - \mathbf{F}(i,:)\mathbf{G}(j,:)')^2 + \lambda||\mathbf{F}||_{fro}^2 + \lambda||\mathbf{G}||_{fro}^2 \\ \text{subject to: } \mathbf{F}(:,1) = \mu\mathbf{1},\ \mathbf{G}(:,1) = \alpha_1\mathbf{1}/\sqrt{n} \quad \text{(global bias)} \\ \mathbf{F}(:,2) = \mathbf{x},\ \mathbf{G}(:,2) = \alpha_2\mathbf{1}/\sqrt{n} \quad \text{(trustor bias)} \\ \mathbf{F}(:,3) = \alpha_3\mathbf{1}/\sqrt{n},\ \mathbf{G}(:,3) = \mathbf{y} \quad \text{(trustee bias)}\end{aligned} \tag{2}$$

where α_1, α_2, and α_3 are the weights of bias that we need to estimate based on the existing trust ratings.

In addition to these three specified factors, we refer to the remaining factors in the trustor and trustee matrices as *latent* factors. Let us define two $n \times r$ sub-matrices of $\mathbf{F}$ and $\mathbf{G}$ for the latent factors. That is, we define $\mathbf{F}_0 = \mathbf{F}(:, 4{:}s)$ and $\mathbf{G}_0 = \mathbf{G}(:, 4{:}s)$, where each column of $\mathbf{F}_0$ and $\mathbf{G}_0$ corresponds to one latent factor and r is the number of latent factors. With this notation, we have the following equivalent form of Eq. (2):

$$\begin{aligned}\min_{\mathbf{F}_0,\mathbf{G}_0,\boldsymbol{\alpha}} \sum_{(i,j)\in\mathcal{K}} (\mathbf{T}(i,j) - (\boldsymbol{\alpha}'[\mu, \mathbf{x}(i), \mathbf{y}(j)]' + \mathbf{F}_0(i,:)\mathbf{G}_0(j,:)'))^2 \\ +\lambda\|\mathbf{F}_0\|_{fro}^2 + \lambda\|\mathbf{G}_0\|_{fro}^2 + \lambda\|\boldsymbol{\alpha}\|^2,\end{aligned} \tag{3}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3]'$.

Recall that in this paper, we aim to perform trust inference only using the partially observed trust matrix $\mathbf{T}$. Therefore, we estimate the parameters (μ, $\mathbf{x}$ and $\mathbf{y}$) of the trust bias as follows:

$$\begin{cases} \mu = \sum_{(i,j)\in\mathcal{K}} \mathbf{T}(i,j)/|\mathcal{K}| \\ \mathbf{x}(i) = \sum_{j,(i,j)\in\mathcal{K}} \mathbf{T}(i,j)/|row_i| - \mu \\ \mathbf{y}(j) = \sum_{i,(i,j)\in\mathcal{K}} \mathbf{T}(i,j)/|col_j| - \mu, \end{cases} \tag{4}$$

where $|row_i|$ is the number of the observed elements in the i^{th} row of $\mathbf{T}$, and $|col_j|$ is the number of the observed elements in the j^{th} column of $\mathbf{T}$.

3.1.3 Incorporating Trust Propagation

We next describe how to incorporate trust propagation into the model. We consider the following four groups of trust propagation operators defined in [15]: *direct propagation*, *transpose trust*, *co-citation*, and *trust coupling*.

Direct propagation: Direct propagation is probably the most intuitive way to propagate trust as shown in Fig. 2(a). The basic operator in the figure presents the two-step propagation and it can be generalized to multiple steps. We define the first group of $(t-1)$ propagation elements in the matrix form as $\mathbf{T}^2, \mathbf{T}^3, \ldots, \mathbf{T}^t$, where t is the largest propagation step.

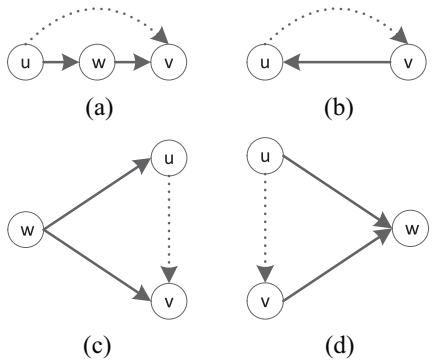


Fig. 2. Four propagation operators. The solid lines indicate existing trust relationships, and the dotted lines indicate propagated trust. (a) Direct propagation. (b) Transpose trust. (c) Co-citation. (d) Trust coupling.

Transpose trust: The second operator is the transpose trust as shown in Fig. 2(b). This operator indicates that user v's trust on user u can cause some level of trust in the opposite direction. This group of t propagation elements can be represented in the matrix form as $\mathbf{T}'$, $(\mathbf{T}')^2$, $(\mathbf{T}')^3, \ldots, (\mathbf{T}')^t$.

Co-citation: Co-citation is found to be very powerful to predict trust and distrust in the Epinions website. As shown in Fig. 2(c), co-citation means that if two users u and v are both trusted by another user w, then u might also trust v to some extent. Based on the transitive closure computation, we can represent this group of propagation elements as: $(\mathbf{T}'\mathbf{T})$, $(\mathbf{T}'\mathbf{T})^2$, $(\mathbf{T}'\mathbf{T})^3, \ldots, (\mathbf{T}'\mathbf{T})^t$.

Trust coupling: Finally, Fig. 2(d) shows the trust coupling operator, which means that if two users both trust another user, they might also trust each other. Similar to co-citation, we represent the fourth group of propagation elements as $(\mathbf{T}\mathbf{T}')$, $(\mathbf{T}\mathbf{T}')^2$, $(\mathbf{T}\mathbf{T}')^3, \ldots, (\mathbf{T}\mathbf{T}')^t$.

Altogether, we have generated $(4t-1)$ trust propagation matrices, with each corresponding entry measuring one specific trust propagation between the two corresponding users, respectively. For example, $\mathbf{T}^t(i,j)$ measures direct propagation from user i to user j after t steps, and $(\mathbf{T}\mathbf{T}')(i,j)$ quantifies the one-step trust coupling effect between user i and user$j$, etc. If we further stack all these $(4t-1)$ entries into a propagation vector $\mathbf{z}_{ij}$ for the given user pair (i,j), we have the following form when we incorporate both trust bias and trust propagation into Eq. (1):

$$\begin{aligned}\min_{\mathbf{F}_0,\mathbf{G}_0,\boldsymbol{\alpha},\boldsymbol{\beta}} \sum_{(i,j)\in\mathcal{K}} \Big(\mathbf{T}(i,j) - \Big(\boldsymbol{\alpha}'\left[\mu, \mathbf{x}(i), \mathbf{y}(j)\right]' + \boldsymbol{\beta}'\mathbf{z}_{ij} \\ + \mathbf{F}_0(i,:)\mathbf{G}_0(j,:)'\Big)\Big)^2 + \lambda\|\mathbf{F}_0\|_{fro}^2 \\ +\lambda\|\mathbf{G}_0\|_{fro}^2 + \lambda\|\boldsymbol{\alpha}\|^2 + \lambda\|\boldsymbol{\beta}\|^2,\end{aligned} \tag{5}$$

where $\mathbf{z}_{ij}$ is the vector of propagation elements for the trustor-trustee pair (i,j), $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3]'$ is the weight vector for bias, and $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_{4t-1}]'$ is the weight vector for trust propagation.

Notice that there is no coefficient before $\mathbf{F}_0(i,:)\mathbf{G}_0(j,:)'$ as it will be automatically absorbed into $\mathbf{F}_0$ and $\mathbf{G}_0$ in our iterative algorithm. Once we have inferred all the parameters (i.e., $\mathbf{F}_0$, $\mathbf{G}_0$, $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$) of Eq. (5), the unseen trustworthiness

score $\hat{\mathbf{T}}(u, v)$ can be immediately estimated as:

$$\hat{\mathbf{T}}(u, v) = \mathbf{F}_0(u, :)\mathbf{G}_0(v, :)' + \boldsymbol{\alpha}' \left[\mu, \mathbf{x}(u), \mathbf{y}(v)\right]' + \boldsymbol{\beta}' \mathbf{z}_{uv}. \quad (6)$$

3.2 Proposed Formulation for Binary Case

In the previous subsection, we have already shown how to characterize the multi-aspect property and how to incorporate trust bias and trust propagation for continuous trust inference. As for the binary case, some proper adaptations are necessary, and we now present these adaptations.

First, in Eq. (5), we use square loss as the loss function to minimize the difference between the real value and the estimated value. However, in the binary setting, the consistency between the real sign and the estimated sign is more important. For example, if $\mathbf{T}(i, j) = -1$ and $\hat{\mathbf{T}}(i, j) = -5$, there should be no loss as the signs are the same. Therefore, instead of square loss, other functions such as sigmoid loss and squared-hinge loss are more suitable in binary setting. In this work, sigmoid loss function is chosen to penalize the inconsistency of the estimated signs. The objective formulation can then be written as follows:

$$\begin{aligned} \min_{\mathbf{F}_0, \mathbf{G}_0, \boldsymbol{\alpha}, \boldsymbol{\beta}} \sum_{(i,j)\in\mathcal{K}} g\Big(\mathbf{T}(i, j), \boldsymbol{\alpha}' \left[\mu, \mathbf{x}(i), \mathbf{y}(j)\right]' + \boldsymbol{\beta}' \mathbf{z}_{ij} \\ + \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)'\Big) + \lambda \|\mathbf{F}_0\|_{fro}^2 \\ + \lambda \|\mathbf{G}_0\|_{fro}^2 + \lambda \|\boldsymbol{\alpha}\|^2 + \lambda \|\boldsymbol{\beta}\|^2. \end{aligned} \quad (7)$$

where $g(x, y) = 1/(1+\exp(xy))$ is the sigmoid loss function.

The second difference between the binary setting and the continuous setting lies in the propagation vector $\mathbf{z}_{ij}$. The basic intuition is that distrust cannot be propagated for more than two steps, i.e., "the enemy of your enemy is not necessarily your friend". On the other hand, it is well known that one-step distrust propagation could be useful to improve trust inference accuracy [15]. Therefore, in our binary case, we only propagate positive trust ratings for the propagation operators, and add one-step distrust propagation as the last propagation step. In other words, if we use $\mathbf{T}_+$ to denote the trust matrix with only positive trust ratings, the four propagation operators can then be re-defined as $(\mathbf{T}_+)^t\mathbf{T}$, $(\mathbf{T}'_+)^t\mathbf{T}$, $(\mathbf{T}'_+\mathbf{T}_+)^t\mathbf{T}$, and $(\mathbf{T}_+\mathbf{T}'_+)^t\mathbf{T}$, respectively.

With the above two adaptations, we could now predict trust/distrust based on the sign of Eq. (6).

3.3 Discussions and Generalizations

We further present some discussions and generalizations of our optimization formulation.

First, it is worth pointing out that our formulation in Eq. (1) differs from the standard matrix factorization (e.g., SVD) as in the objective function, we try to minimize the square loss *only* on those observed trust pairs. This is because the majority of trust pairs are missing from the input trust matrix $\mathbf{T}$. As mentioned before, our basic problem setting in Eq. (1) is conceptually similar to the standard collaborative filtering, as in both cases, we aim to fill in missing values in a partially observed matrix (trustor-trustee matrix vs. user-item matrix). Indeed, if we fix the coefficients $\alpha_1 = \alpha_2 = \alpha_3 = 1$ and $\beta_1 = \beta_2 = \cdots = \beta_{4t-1} = 0$ in our formulation (i.e., Eq. (5) and Eq.(7)), it is reduced to the collaborative filtering algorithm in [16]. Our formulation goes beyond the standard collaborative filtering by (1) incorporating two other important properties in trust inference (i.e., trust bias and trust transitivity); and (2) learning their relative weights ($\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$). Our experimental evaluations show that such subtle treatments are crucial and they lead to further performance improvement over these existing techniques.

Second, although our model is a subjective trust inference metric where different trustors may form different opinions on the same trustee [17], [18], as a side product, the proposed model can also be used to infer an objective, unique trustworthiness score for each trustee. For example, this objective trustworthiness score can be computed based on the trustee matrix $\mathbf{G}$. We will compare this feature of the proposed model with a well studied objective trust inference metric EigenTrust [3] in our experimental evaluation.

Finally, we would like to point out that our formulation is flexible and can be generalized to other settings. For instance, we have already shown that our formulation could deal with both continuous and binary trust inference. In the continuous setting, we adopt the square loss function in the objective function. In other words, we implicitly assume that the residuals of the pair-wise trustworthiness scores follow a Gaussian distribution. Such assumption is commonly used when no prior knowledge about the residuals is known. Nonetheless, our upcoming proposed algorithm for continuous trust inference can be generalized to *any* Bregman divergence in the objective function. Also, we can naturally incorporate some additional constraints (e.g., non-negativity, sparseness, etc.) in the trustor and trustee matrices. Finally, after we infer all the parameters, we use a linear combination to compute the trustworthiness score $\hat{\mathbf{T}}(u, v)$. We can also generalize this linear form to other non-linear combinations, such as the logistic function. For the sake of clarity, we skip the details of such generalizations in the paper.

4 The Proposed MATRI Algorithm

In this section, we present a family of algorithms (MATRI) to solve the trust inference problems in Eq. (5) and Eq. (7), followed by some effectiveness and efficiency analysis.

4.1 The MATRI Algorithm for Continuous Case

We first show our algorithm for the continuous case (i.e., Eq. (5)). Unfortunately, the optimization problem in Eq. (5) is not jointly convex wrt the coefficients ($\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$) and the trustor/trustee matrices ($\mathbf{F}_0$ and $\mathbf{G}_0$) due to the coupling between them as well as the fact that most entries of the T matrix are unknown [19]. Therefore, instead of seeking for a global optimal solution, we try to find a local minima by alternatively updating the coefficients and the trustor/trustee matrices while fixing the other.

4.1.1 Sub-routine 1: Updating the Trustor/trustee Matrices

First, let us consider how to update the trustor/trustee matrices ($\mathbf{F}_0$ and $\mathbf{G}_0$) when we fix the coefficients ($\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$). For clarity, we define an $n \times n$ matrix $\mathbf{P}$ as follows:

$$\mathbf{P}(i, j) = \begin{cases} \mathbf{T}(i, j) - (\boldsymbol{\alpha}'[\mu, \mathbf{x}(i), \mathbf{y}(j)]' + \boldsymbol{\beta}' \mathbf{z}_{ij}) & \text{if } (i, j) \in \mathcal{K} \\ \text{'?'} & \text{otherwise,} \end{cases} \quad (8)$$

Algorithm 1 alternateUpdate($\mathbf{P}, \mathbf{F}_0, \mathbf{G}_0$).

Input: The $n \times n$ matrix $\mathbf{P}$, the $n \times r$ matrix $\mathbf{F}_0$, and the fixed $n \times r$ matrix $\mathbf{G}_0$
Output: The updated matrix $\mathbf{F}_1$ of $\mathbf{F}_0$

$\mathbf{F}_1 = \mathbf{F}_0$;
for i = 1 : n **do**
  $\mathbf{a}$ = the vector of column indices of existing elements in $\mathbf{P}(i, j)$ $(j = 1, 2, ..., n)$;
  column vector $\mathbf{d} = \mathbf{0}_{|\mathbf{a}|\times 1}$;
  matrix $\mathbf{G}_1 = \mathbf{0}_{|\mathbf{a}|\times r}$;
  for j = 1: $|\mathbf{a}|$ **do**
    $\mathbf{d}(j) = \mathbf{P}(i, \mathbf{a}(j))$;
    $\mathbf{G}_1(j, :) = \mathbf{G}_0(\mathbf{a}(j), :)$;
  end for
  $\mathbf{F}_1(i, :) = (\mathbf{G}_1'\mathbf{G}_1 + \lambda \cdot \mathbf{I}_{r\times r})^{-1}\mathbf{G}_1'\mathbf{d}$;
end for
return $\mathbf{F}_1$;

Algorithm 2 updateMatrix($\mathbf{P}$, r).

Input: The $n \times n$ matrix $\mathbf{P}$, and the latent factor size r
Output: The $n \times r$ trustor matrix $\mathbf{F}_0$, and the $n \times r$ trustee matrix $\mathbf{G}_0$

generate the $n \times r$ matrices $\mathbf{F}_0$ and $\mathbf{G}_0$ randomly;
while not convergent **do**
  $\mathbf{F}_0$ = `alternateUpdate`($\mathbf{P}$, $\mathbf{F}_0$, $\mathbf{G}_0$);
  $\mathbf{G}_0$ = `alternateUpdate`($\mathbf{P}'$, $\mathbf{G}_0$, $\mathbf{F}_0$);
end while
return $[\mathbf{F}_0, \mathbf{G}_0]$;

Algorithm 3 computePropagation($\mathbf{T}$, l, t).

Input: The $n \times n$ matrix trust $\mathbf{T}$, the latent factor size l, and the maximum propagation step t
Output: The propagation vector $\mathbf{z}_{ij}$ for all $(i, j) \in \mathcal{K}$

$[\mathbf{L}, \mathbf{R}]$ = `updateMatrix`($\mathbf{T}$, l);
for each $(i, j) \in \mathcal{K}$ **do**
  compute $\mathbf{z}_{ij}$ by Eq. (12);
end for
return $[\mathbf{z}_{ij}]$ $(i, j) \in \mathcal{K}$;

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are some fixed constants, and '?' means the rating is unknown.

Based on the above definition, Eq. (5) can be simplified (by ignoring some constant terms) as:

$$\min_{\mathbf{F}_0, \mathbf{G}_0} \sum_{(i,j)\in\mathcal{K}} \left(\mathbf{P}(i, j) - \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)'\right)^2 + \lambda||\mathbf{F}_0||_{fro}^2 + \lambda||\mathbf{G}_0||_{fro}^2. \quad (9)$$

Therefore, updating the trustor/trustee matrices when we fix the coefficients unchanged becomes a standard matrix factorization problem for missing values. Many existing algorithms (e.g., [16], [20], [21]) can be plugged in to solve Eq. (9). In our experiment, we found the so-called alternating strategy, where we recursively update one of the two trustee/trustor matrices while keeping the other matrix fixed, works best and thus recommend it in practice. For simplicity, let us consider how to update $\mathbf{F}_0$ when $\mathbf{G}_0$ is fixed. Updating $\mathbf{G}_0$ when $\mathbf{F}_0$ is fixed can be done in a similar way. By fixing $\mathbf{G}_0$, Eq. (9) can be further simplified as follows:

$$\min_{\mathbf{F}_0} \sum_{(i,j)\in\mathcal{K}} \left(\mathbf{P}(i, j) - \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)'\right)^2 + \lambda||\mathbf{F}_0||_{fro}^2. \quad (10)$$

In fact, the above optimization problem in Eq. (10) now becomes convex wrt $\mathbf{F}_0$. It can be further decoupled into many independent sub-problems, each of which only involves a single row in $\mathbf{F}_0$:

$$\min_{\mathbf{F}_0(i,:)} \sum_{j,(i,j)\in\mathcal{K}} \left(\mathbf{P}(i, j) - \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)'\right)^2 + \lambda||\mathbf{F}_0(i, :)||^2. \quad (11)$$

The optimization problem in Eq. (11) can now be solved by the standard ridge regression wrt the corresponding row $\mathbf{F}_0(i, :)$.

Alg. 1 presents the overall solution for updating the trustor matrix $\mathbf{F}_0$. Based on Alg. 1, we present Alg. 2 to alternatively update the trustor and trustee matrices $\mathbf{F}_0$ and $\mathbf{G}_0$. The algorithm first generates two random $n \times r$ matrices for $\mathbf{F}_0$ and $\mathbf{G}_0$. At each iteration, the algorithm then alternatively calls Alg. 1 to update the two matrices. The iteration ends when the stopping criteria are met, i.e., either the L_2 norm between successive estimates of both $\mathbf{F}_0$ and $\mathbf{G}_0$ is below our threshold ξ_1 or the maximum iteration step m_1 is reached.

4.1.2 Sub-routine 2: Computing Trust Propagation

Directly computing the propagation vector $\mathbf{z}_{ij}$ $(i, j) \in \mathcal{K}$ is computationally inefficient as it involves the multiplications of matrices of $n \times n$. To address this issue, we propose the following procedure (Alg. 3) to compute the trust propagation vectors. In Alg. 3, we first factorize the input trust matrix into two $n \times l$ low rank matrices $\mathbf{L}$, $\mathbf{R}$ (step 1); and use them as the base to compute the trust propagation vectors. By doing so, we only need to compute the matrix power or multiplications of $l \times l$, where $l \ll n$.

Notice that in step 1, instead of the standard SVD, we call Alg. 2 to get the two low rank matrices. In this way, we implicitly fill in the missing values in the partially observed matrix $\mathbf{T}$ before performing the propagation. This has the additional advantage to mitigate the sparsity or coverage problem in trust propagation where some trustor and trustee might not be connected with each other [22].

$$\begin{cases} \mathbf{T}^t(i, j) = \mathbf{L}(i, :)\left(\mathbf{R}'\mathbf{L}\right)^{t-1}\mathbf{R}(j, :)' \\ \left(\mathbf{T}'\right)^t(i, j) = \mathbf{R}(i, :)\left(\mathbf{L}'\mathbf{R}\right)^{t-1}\mathbf{L}(j, :)' \\ \left(\mathbf{T}'\mathbf{T}\right)^t(i, j) = \mathbf{R}(i, :)\left(\left(\mathbf{L}'\mathbf{L}\right)\left(\mathbf{R}'\mathbf{R}\right)\right)^{t-1}\left(\mathbf{L}'\mathbf{L}\right)\mathbf{R}(j, :)' \\ \left(\mathbf{T}\mathbf{T}'\right)^t(i, j) = \mathbf{L}(i, :)\left(\left(\mathbf{R}'\mathbf{R}\right)\left(\mathbf{L}'\mathbf{L}\right)\right)^{t-1}\left(\mathbf{R}'\mathbf{R}\right)\mathbf{L}(j, :)'. \end{cases} \quad (12)$$

4.1.3 Sub-routine 3: Updating the Coefficients

Here, we consider how to update the coefficients ($\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$) when we fix the trustor/trustee matrices.

If we fix the trustor and trustee matrices ($\mathbf{F}_0$ and $\mathbf{G}_0$) and let:

$$\mathbf{P}(i, j) = \begin{cases} \mathbf{T}(i, j) - \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)' & \text{if } (i, j) \in \mathcal{K} \\ \text{'?'} & \text{otherwise.} \end{cases} \quad (13)$$

Algorithm 4 MATRI($\mathbf{T}$, $\mathcal{K}$, r, l, t, u, v).

Input: The $n \times n$ partially observed trust matrix $\mathbf{T}$, the set of observed trustor-trustee pairs $\mathcal{K}$, the latent factor size r, the low rank l for trust propagation, the maximum propagation step t, trustor u, and trustee v

Output: The estimated trustworthiness score $\hat{\mathbf{T}}(u, v)$

Pre-computation stage:

1: compute bias: $[\mu, \mathbf{x}, \mathbf{y}]$ = `computeBias`($\mathbf{T}$) by Eq. (4);
2: compute propagation: $\mathbf{z}_{ij}$ = `computePropagation`($\mathbf{T}, l, t$), $(i, j) \in \mathcal{K}$;
3: initialize $\alpha_1 = \alpha_2 = \alpha_3 = 1$, $\beta_1 = \beta_2 = ... = \beta_{4t-1} = 0$;
4: **while** not convergent **do**
5: **for** each $(i, j) \in \mathcal{K}$ **do**
6: $\mathbf{P}(i, j) = \mathbf{T}(i, j) - (\boldsymbol{\alpha}'[\mu, \mathbf{x}(i), \mathbf{y}(j)]' + \boldsymbol{\beta}'\mathbf{z}_{ij})$;
7: **end for**
8: $[\mathbf{F}_0, \mathbf{G}_0]$ = `updateMatrix`($\mathbf{P}$, r);
9: **for** each $(i, j) \in \mathcal{K}$ **do**
10: $\mathbf{P}(i, j) = \mathbf{T}(i, j) - \mathbf{F}_0(i, :)\mathbf{G}_0(j :,)'$;
11: **end for**
12: $[\boldsymbol{\alpha}, \boldsymbol{\beta}]$ = `updateCoefficient`($\mathbf{P}$, μ, $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}_{ij}$) by Eq. (15);
13: **end while**

On-line query response stage:

14: **return** $\hat{\mathbf{T}}(u, v) = \mathbf{F}_0(u, :)\mathbf{G}_0(v, :)' + \boldsymbol{\alpha}'[\mu, \mathbf{x}(u), \mathbf{y}(v)]' + \boldsymbol{\beta}'\mathbf{z}_{uv}$;

Eq. (5) can then be simplified (by dropping constant terms) as:

$$\min_{\boldsymbol{\alpha}, \boldsymbol{\beta}} \sum_{(i,j) \in \mathcal{K}} \left(\mathbf{P}(i, j) - \left(\boldsymbol{\alpha}' \left[\mu, \mathbf{x}(i), \mathbf{y}(j) \right]' + \boldsymbol{\beta}' \mathbf{z}_{ij} \right) \right)^2 + \lambda \|\boldsymbol{\alpha}\|^2 + \lambda \|\boldsymbol{\beta}\|^2. \quad (14)$$

To simplify the description, let us introduce another scalar k to index each pair (i, j) in the observed trustor-trustee pairs $\mathcal{K}$, that is, $(i, j) \in \mathcal{K} \rightarrow k = \{1, 2, \ldots, |\mathcal{K}|\}$. Let $\mathbf{b}$ denote a vector of length $|\mathcal{K}|$ with $\mathbf{b}(k) = \mathbf{P}(i, j)$. We also define a $|\mathcal{K}| \times (4t+2)$ matrix $\mathbf{A}$ as: $\mathbf{A}(k, 1) = \mu$, $\mathbf{A}(k, 2) = \mathbf{x}(i)$, $\mathbf{A}(k, 3) = \mathbf{y}(j)$, $\mathbf{A}(k, 4{:}4t + 2) = \mathbf{z}'_{ij}$, $(k = 1, 2, \ldots, |\mathcal{K}|)$.

Then, the coefficients ($\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$) can be updated by solving the following ridge regression problem, which is equivalent to Eq. (14):

$$\boldsymbol{\gamma} = [\boldsymbol{\alpha};\ \boldsymbol{\beta}] = \texttt{argmin}_{\boldsymbol{\gamma}} ||\mathbf{b} - \mathbf{A}\boldsymbol{\gamma}||^2 + \lambda \|\boldsymbol{\gamma}\|^2. \quad (15)$$

4.1.4 Putting Everything Together: MATRI

Putting everything together, we propose Alg. 4 for the continuous trust inference problem in Eq. (5). The algorithm first computes trust bias (step 1) and trust propagation (step 2). Next, after an initialization step (step 3), the algorithm begins the alternating procedure (step 4-13). At each iteration, it first fixes the coefficients ($\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$), and updates the trustor matrix $\mathbf{F}_0$ and trustee matrix $\mathbf{G}_0$ (step 5-8). Next, the algorithm fixes $\mathbf{F}_0$ and $\mathbf{G}_0$, and uses ridge regression in Eq. (15) to update the coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ (step 9-12). We use the following criteria to terminate the alternating procedure: either the L_2 norm between successive estimates of both $\mathbf{F}_0$ and $\mathbf{G}_0$ is below our threshold ξ_2 or the maximum iteration step m_2 is reached. Finally, the algorithm outputs the estimated trustworthiness score from the given trustor u to the trustee v using Eq. (6).

It is worth pointing out that step 1-13 in the algorithm can be pre-computed and their results (including $\mathbf{F}_0$, $\mathbf{G}_0$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, μ, $\mathbf{x}$, $\mathbf{y}$, $\mathbf{L}$,$\mathbf{R}$,$\mathbf{L}'\mathbf{R}$,$\mathbf{R}'\mathbf{L}$,$\mathbf{L}'\mathbf{L}$ and $\mathbf{R}'\mathbf{R}$) can be stored in the pre-computational or off-line stage. When an on-line trust inference request arrives, the proposed MATRI only needs to apply step 14 to return the inference result, which only requires a constant time.

4.2 The MATRI Algorithm for Binary Case

As described in Section 3.2, there are two major differences (i.e., loss function and trust propagation) between binary trust inference and continuous trust inference in our formulation. We now present the corresponding algorithms to handle these two differences.

First, we cannot apply the alternating strategy used in the continuous case to solve Eq. (7) due to the sigmoid loss function. Therefore, we adopt the stochastic gradient descent method to search for a local minima. That is, we randomly pick an observed trust rating $\mathbf{T}(i, j)$ in $\mathcal{K}$, and then update the i^{th} row of $\mathbf{F}_0$, the j^{th} row of $\mathbf{G}_0$ and the coefficient vector. The sub-procedure for the picked rating $\mathbf{T}(i, j)$ is shown in Eq. (16), where k is the index of pair (i, j), η_1 and η_2 are the learning rates, and $\boldsymbol{\gamma}$ and $\mathbf{A}$ are defined in Eq. (14). In our algorithm, we sweep through the observed trustor-trustee pairs in each iteration, and stop the iteration when either the L_2 norm between successive estimates of both $\mathbf{F}_0$ and $\mathbf{G}_0$ is below our threshold ξ_3 or the maximum iteration step m_3 is reached. Finally, the algorithm will return the trust/distrust sign based on Eq. (6).

Next, we show how to compute trust propagation for the binary case. Similar to the continuous case, we still resort to low rank matrices. Let $\mathbf{L}_+$ and $\mathbf{R}_+$ denote the low rank matrices derived from $\mathbf{T}_+$. The propagation vectors can now be computed by the sub-procedure shown in Eq. (17).

To obtain $\mathbf{L}$ and $\mathbf{R}$ (or equivalently, $\mathbf{L}_+$ and $\mathbf{R}_+$), we also use sigmoid loss and adopt stochastic gradient descent method. The sub-procedure for computing $\mathbf{L}$ and $\mathbf{R}$ is shown in Eq. (18).

$$\begin{aligned}
\mathbf{F}_0(i, :) &= \mathbf{F}_0(i, :) - \eta_1 \left(\frac{\partial g(\mathbf{T}(i, j), \mathbf{A}(k, :)\boldsymbol{\gamma} + \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)')}{\partial \mathbf{F}_0(i, :)} + \lambda \mathbf{F}_0(i, :) \right) \\
\mathbf{G}_0(j, :) &= \mathbf{G}_0(j, :) - \eta_1 \left(\frac{\partial g(\mathbf{T}(i, j), \mathbf{A}(k, :)\boldsymbol{\gamma} + \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)')}{\partial \mathbf{G}_0(j, :)} + \lambda \mathbf{G}_0(j, :) \right) \\
\boldsymbol{\gamma} &= \boldsymbol{\gamma} - \eta_2 \left(\frac{\partial g(\mathbf{T}(i, j), \mathbf{A}(k, :)\boldsymbol{\gamma} + \mathbf{F}_0(i, :)\mathbf{G}_0(j, :)')}{\partial \boldsymbol{\gamma}} + \lambda \boldsymbol{\gamma} \right) \quad (16)
\end{aligned}$$

$$\begin{cases} (\mathbf{T}_+^t\mathbf{T})(i,j) &= \mathbf{L}_+(i,:)(\mathbf{R}'_+\mathbf{L}_+)^{t-1}(\mathbf{R}'_+\mathbf{L})\mathbf{R}(j,:)' \\ ((\mathbf{T}'_+)^t\mathbf{T})(i,j) &= \mathbf{R}_+(i,:)(\mathbf{L}'_+\mathbf{R}_+)^{t-1}(\mathbf{L}'_+\mathbf{L})\mathbf{R}(j,:)' \\ ((\mathbf{T}'_+\mathbf{T}_+)^t\mathbf{T})(i,j) &= \mathbf{R}_+(i,:)((\mathbf{L}'_+\mathbf{L}_+)(\mathbf{R}'_+\mathbf{R}_+))^{t-1} \\ & \qquad (\mathbf{L}'_+\mathbf{L}_+)(\mathbf{R}'_+\mathbf{L})\mathbf{R}(j,:)' \\ ((\mathbf{T}_+\mathbf{T}'_+)^t\mathbf{T})(i,j) &= \mathbf{L}_+(i,:)((\mathbf{R}'_+\mathbf{R}_+)(\mathbf{L}'_+\mathbf{L}_+))^{t-1} \\ & \qquad (\mathbf{R}'_+\mathbf{R}_+)(\mathbf{L}'_+\mathbf{L})\mathbf{R}(j,:)' \end{cases} \tag{17}$$

$$\begin{aligned} \mathbf{L}(i,:) &= \mathbf{L}(i,:) - \eta_1\left(\frac{\partial g(\mathbf{T}(i,j),\mathbf{L}(i,:)\mathbf{R}(j,:)')}{\partial \mathbf{L}(i,:)} + \lambda\mathbf{L}(i,:)\right) \\ \mathbf{R}(j,:) &= \mathbf{R}(j,:) - \eta_1\left(\frac{\partial g(\mathbf{T}(i,j),\mathbf{L}(i,:)\mathbf{R}(j,:)')}{\partial \mathbf{R}(j,:)} + \lambda\mathbf{R}(j,:)\right). \end{aligned} \tag{18}$$

Actually, the stochastic gradient descent method as described above could also be used for the square loss in continuous trust inference problem. For example, we can use this method for computing the propagation vector, or to directly solving Eq. (5). We will experimentally evaluate all the possible combinations in the next section.

4.3 Algorithm Analysis

Here, we analyze the effectiveness and efficiency of our algorithms.

The effectiveness of the proposed MATRI can be summarized in Lemma 1, which says that overall, it finds a local minima solution for both continuous and binary trust inference. Given that the original optimization problems (Eq. (5) and Eq. (7)) are not jointly convex wrt the coefficients ($\boldsymbol{\alpha}$, $\boldsymbol{\beta}$) and the trustor/trustee matrices ($\mathbf{F}_0$ and $\mathbf{G}_0$), such a local minima is acceptable in practice.

Lemma 1 (Effectiveness of MATRI). *Fixing the propagation vector $\mathbf{z}_{ij}$,* MATRI *finds local minima for the optimization problems in Eq.* (5) *and Eq.* (7).

Proof Sketch. For the continuous case: First, Eq. (11) is convex and therefore step 10 in Alg. 1 finds the global optima for updating a single row in the matrix $\mathbf{F}_0$. Notice that the optimization problem in Eq. (11) is equivalent to that in Eq. (10), and thus Alg. 1 finds the global optimal solution for the optimization problem in Eq. (10). Next, based on the alternating procedure in Alg. 2, we have that Alg. 2 finds a local minima for the optimization problem in Eq. (9). Finally, based on the alternating procedure in Alg. 4, we have that Alg. 4 finds a local minima for the optimization problem in Eq. (5).

For the binary case: In each iteration, Eq. (16) essentially defines a gradient descent procedure, which would never improve the cost of Eq. (7). Such a procedure will converge when a local minima is found. Lemma 1 holds. □

The time complexity of the proposed MATRI is summarized in Lemma 2, which says that MATRI (1) requires *constant* time for on-line query response (e.g., step 14 in Alg. 4) and (2) scales *linearly* wrt the number of users and the number of the observed trustor-trustee pairs in the pre-computational stage (e.g., step 1-13 in Alg. 4).

Lemma 2 (Time Complexity of MATRI). *Fixing r, l and t as constants, (P1)* MATRI *requires $O(nm_1m_2 + |\mathcal{K}|m_1m_2)$ time for pre-computation in the continuous case, where m_1 and m_2 are the maximum iteration number in Alg. 2 and Alg. 4;* MATRI *requires $O(|\mathcal{K}|m_3+n)$ time for pre-computation in the binary case, where m_3 is the maximum iteration number of the gradient descent method; and (P2)* MATRI *requires $O(1)$ for on-line query response in both continuous and binary cases.*

Proof. We first prove (P1) for the continuous case: In Alg. 1, the time cost for step 1 is $O(nr)$. Let a_i denote the number of elements in $\mathbf{a}$ of the i^{th} iteration. The time cost for step 3-5 is then $O(a_ir)$ since we store $\mathbf{P}$ in sparse format. We need another $O(a_ir)$ time in the inner iteration (step 6-9). The time cost of step 10 is $O(r^3 + a_ir^2)$. Therefore, the total time cost for the algorithm is $O(nr) + O(\sum_i(r^3 + a_ir^2)) = O(nr^3 + |\mathcal{K}|r^2)$ where $\sum_i a_i = |\mathcal{K}|$.

Next, in Alg. 2, the time cost for step 1 is $O(nr)$. As shown before, we need $O(nr^3+|\mathcal{K}|r^2)$ time for both step 3 and step 4. The total time cost is $O(nr^3m_1 + |\mathcal{K}|r^2m_1)$.

Similarly, in Alg. 3, we need $O(nl^3m_1 + |\mathcal{K}|l^2m_1)$ time for step 1. For the iteration in step 2-4, by computing and storing the power of $\mathbf{L}'\mathbf{R}$,$\mathbf{R}'\mathbf{L}$,$\mathbf{L}'\mathbf{L}$ and $\mathbf{R}'\mathbf{R}$, we need $O(nl^2 + tl^3 + |\mathcal{K}|tl^2)$ time. The total time cost for Alg. 3 is $O(nl^3m_1 + |\mathcal{K}|l^2m_1 + tl^3 + |\mathcal{K}|tl^2)$.

Finally, in Alg. 4, the time cost for step 1 is $O(|\mathcal{K}|)$ as we store $\mathbf{T}$ in sparse format. As shown before, step 2 needs $O(nl^3m_1 + |\mathcal{K}|l^2m_1 + tl^3 + |\mathcal{K}|tl^2)$ time. In the iteration, step 5-7 needs $O(|\mathcal{K}|t)$ time. Step 8 costs $O(nr^3m_1 + |\mathcal{K}|r^2m_1)$ time. We need $O(|\mathcal{K}|r)$ time for step 9-11, and $O(|\mathcal{K}|t^2+t^3)$ time for step 12. The time cost of the iteration of step 4-10 is $O(nr^3m_1m_2 + |\mathcal{K}|r^2m_1m_2 + |\mathcal{K}|t^2m_2 + t^3m_2)$. In practice, we fix r, l and t as small constants. Therefore, the total time cost of Alg. 4 is $O(nm_1m_2 + |\mathcal{K}|m_1m_2)$.

We next prove (P1) for the binary case: Here, we only need to analyze the time cost of bias computation, propagation computation, and the computation of parameters (i.e., $\mathbf{F}_0$, $\mathbf{G}_0$ and $\boldsymbol{\gamma}$). As already shown in the continuous case, we need $O(|\mathcal{K}|)$ time for bias computation. For propagation computation, we need $O(|\mathcal{K}|l)$ time for each iteration over Eq. (18) and thus we need $O(|\mathcal{K}|lm_3)$ time for obtaining $\mathbf{L}$, $\mathbf{R}$, $\mathbf{L}_+$ and $\mathbf{R}_+$. As for the computation of $\mathbf{z}_{ij}$, the time cost is $O(nl^2 + tl^3 + |\mathcal{K}|tl^2)$ which is the same as the continuous case. The total time cost for propagation computation is $O(|\mathcal{K}|lm_3 + nl^2 + tl^3 + |\mathcal{K}|tl^2)$.

Next, to compute $\mathbf{F}_0$, $\mathbf{G}_0$ and $\boldsymbol{\gamma}$, the time cost for each iteration over Eq. (16) is $O(|\mathcal{K}|r + |\mathcal{K}|t)$, and the overall time cost is thus $O(|\mathcal{K}|rm_3 + |\mathcal{K}|tm_3)$. In total, by fixing r, l and t as small constants, the time cost of MATRI for the binary case is $O(|\mathcal{K}|m_3 + n)$.

(P2) Step 14 in Alg. 4 only require $O(r + tl^2) = O(1)$ time, and we only need $O(1)$ time to decide the sign of the result for the binary case, which completes the proof. □

The space complexity of MATRI is summarized in Lemma 3, which says that MATRI requires *linear* space wrt the number of users and the number of the observed trustor-trustee pairs.

Lemma 3 (Space Complexity of MATRI). *Fixing r, l and t as constants,* MATRI *require $O(|\mathcal{K}| + n)$ space.*

Proof. For the continuous case: First, in Alg. 1, we need $O(|\mathcal{K}|+nr)$ space for the input since we store $\mathbf{P}$ in sparse format. We need $O(nr)$ space for step 1 and $O(1)$ space

TABLE 2
High Level Statistics of *advogato* and *PGP* Data Sets

Data set	Nodes	Edges (postive/negative)	Avg. degree	Avg. clustering [23]	Avg. diameter [24]	Date
advogato-1	279	2,109	15.1	0.45	4.62	2000-02-05
advogato-2	1,261	12,176	19.3	0.36	4.71	2000-07-18
advogato-3	2,443	22,486	18.4	0.31	4.67	2001-03-06
advogato-4	3,279	32,743	20.0	0.33	4.74	2002-01-14
advogato-5	4,158	41,308	19.9	0.33	4.83	2003-03-04
advogato-6	5,428	51,493	19.0	0.31	4.82	2011-06-23
PGP	38,546	317,979	16.5	0.45	7.70	2008-06-05
Epinions	131,828	841,372 (717,667/123,705)	12.8	0.24	6.37	2003-08-12
Slashdot	82,140	549,202 (425,072/124,130)	13.4	0.09	5.77	2009-02-21

for step 2. We need another $O(nr)$ space for step 3-5. For step 6-9 we only need $O(1)$ space. We need $O(nr + r^2)$ space for step 10. Among the different iterations of the algorithm, we can re-use the space from the previous iteration. Finally, the overall space cost is $O(|\mathcal{K}|+nr+r^2)$.

Next, in Alg. 2, we need $O(|\mathcal{K}|)$ space for input. We need $O(nr)$ space for step 1. Step 3 and step 4 need $O(|\mathcal{K}| + nr + r^2)$ space. The space for each iteration can be re-used. The total space cost is $O(|\mathcal{K}| + nr + r^2)$.

Next, in Alg. 3, we need $O(|\mathcal{K}|)$ space for input. Step 1 requires $O(|\mathcal{K}| + nl + l^2)$ space. Step 2-4 requires $O(tl^2)$ space for the storage of the power of $\mathbf{L}'\mathbf{R}$,$\mathbf{R}'\mathbf{L}$,$\mathbf{L}'\mathbf{L}$ and $\mathbf{R}'\mathbf{R}$, and $O(|\mathcal{K}|t)$ space for the storage of propagation vectors. The total space cost is $O(|\mathcal{K}|t + nl + tl^2)$.

Finally, in Alg. 4, we need $O(n)$ space for step 1 and $O(|\mathcal{K}|t + nl + tl^2)$ space for the step 2. We need $O(t)$ space for step 3. In the iteration, we need $O(|\mathcal{K}|)$ space for step 5-7, and $O(|\mathcal{K}| + nr + r^2)$ space for step 8. For step 9-11, we can re-use the space of step 5-7. Step 12 needs $O(|\mathcal{K}|t + t^2)$ space. For each iteration, the space can be re-used. The total space cost of Alg. 4 is $O(|\mathcal{K}|t + nr + nl + r^2 + tl^2 + t^2) = O(|\mathcal{K}| + n)$.

For the binary case: We need $O(nr + t)$ space for the initialization of $\mathbf{F}_0$, $\mathbf{G}_0$ and $\boldsymbol{\gamma}$. The space cost of bias computation and propagation computation is the same as the continuous case, which is $O(n)$ and $O(|\mathcal{K}|t + nl + tl^2)$, respectively. Over each iteration of Eq. (16), we need $O(|\mathcal{K}|t + nr)$ space, and the space can be re-used in the following iterations. Therefore, the total space cost of MATRI for the binary case is $O(|\mathcal{K}|t + nr + nl + tl^2) = O(|\mathcal{K}| + n)$, which completes the proof. □

5 Experimental Evaluation

In this section, we present experimental evaluations, after we introduce the data sets. All the experiments are designed to answer the following questions:

- *Effectiveness*: How accurate is the proposed MATRI for trust inference?
- *Efficiency*: How fast is the proposed MATRI? How does it scale?

5.1 Data Sets Description

We evaluate our MATRI on four data sets: two for continuous case and two for binary case.

For the continuous case, the first data set is *advogato*[1]. It is a trust-based social network for open source developers. To allow users to certify each other, the network provides 4 levels of trust assertions, i.e., '*Observer*', '*Apprentice*', '*Journeyer*', and '*Master*'. These assertions can be mapped into real numbers which represent the degree of trust. To be specific, we map '*Observer*', '*Apprentice*', '*Journeyer*', and '*Master*' to 0.1, 0.4, 0.7, and 0.9, respectively (a higher value means more trustworthiness).

The second data set for the continuous case is *PGP* (short for Pretty Good Privacy) [25]. *PGP* adopts the concept of 'web of trust' to establish a decentralized model for data encryption and decryption. Similar to *advogato*, the web of trust in *PGP* data set contains 4 levels of trust as well. In our experiments, we also map them to 0.1, 0.4, 0.7, and 0.9, respectively.

For the binary case, we also use two widely used benchmark data sets, i.e., the *Epinions* data set and the *Slashdot* data set [26]. *Epinions* is a product review website. In the website, a user can tag others as trust/distrust according to their product review ratings, and such trust/distrust relationships can in turn help to select suitable product reviews for the given user. We represent these trust/distrust relationships in the data set as +1/-1 where +1 means trust and -1 means distrust.

The forth data set is *Slashdot*, which is a technology-related news website. Similar to *Epinions*, *Slashdot* also provides the mechanism to allow users to tag each other as "friends" or "foes", and these tags are represented as +1/-1 in the data set.

Table 2 summarizes the basic statistics of the four resulting partially observed trust matrices $\mathbf{T}$. Notice that for the *Epinions* and *Slashdot* data sets, we also show the number of positive and negative edges. In both data sets, around 80% edges are positive. For the *advogato* data set, it contains six different snapshots, i.e., *advogato-1*, *advogato-2*,..., *advogato-6*, etc. We use the largest snapshot (i.e., *advogato-6*) in the following unless otherwise stated.

5.2 Effectiveness Results

We now show the effectiveness results of MATRI. For continuous trust inference, we use both *advogato* (i.e., *advogato-6*) and *PGP*. We hide a randomly selected sample of 500 observed trustor-trustee pairs as the test set, and apply the proposed MATRI as well as other existing methods on the remaining data set to infer the trustworthiness scores for those hidden pairs. To evaluate and compare the accuracy, we report both the root mean squared error (RMSE) and the mean absolute error (MAE) between the estimated and the true trustworthiness scores. For the binary case, we use both *Epinions* and *Slashdot*. To avoid the unbalance problem

1. http://www.trustlet.org/wiki/Advogato_dataset.

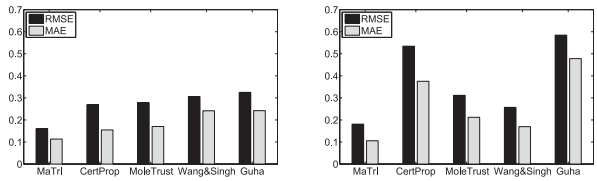


Fig. 3. Comparisons with subjective trust inference models. Lower is better. The proposed MATRI significantly outperforms all the other existing models wrt both RMSE and MAE on both data sets. (a) *advogato* data set. (b) *PGP* data set.

in prediction, we randomly select 250 trust samples and 250 distrust samples, and hide these 500 observed trustor-trustee pairs as the test set. As to evaluation metrics, we report the prediction error which is the percentage of incorrectly predicted signs. In our experiments, we set $r = l = 10$, $m_1 = 100$, $m_2 = m_3 = 10$ and $\xi_1 = \xi_2 = \xi_3 = 10^{-6}$ unless otherwise stated. For the maximum propagation step t, we fix it to 6 due to the "six-degree separation".

5.2.1 Effectiveness Results for Continuous Case

(A) Comparisons with Existing Subjective Trust Inference Methods. For the continuous case, we first compare the effectiveness of MATRI with several benchmark trust propagation models, including *CertProp* [25], *MoleTrust* [17], *Wang&Singh* [27], [28], and *Guha* [15]. For all these subjective methods, the goal is to infer a pair-wise trustworthiness score (i.e., to what extent the user u trusts another user v).

The result is shown in Fig. 3. We can see that the proposed MATRI significantly outperforms all the other trust inference models wrt both RMSE and MAE on both data sets. For example, on *advogato* data set, our MATRI improves the best existing method (CertProp) by 40.7% in RMSE and by 26.7% in MAE. As for *PGP* data set, the proposed MATRI improves the best existing method (Wang&Singh) by 29.6% in RMSE and by 37.8% in MAE. To further show the statistical significance, we also conducted t-test on the prediction errors of our method and the compared methods, and found that the improvement of our method is significant with p-values less than e^{-6} on both data sets. Overall, the proposed MATRI leads to 26.7% - 40.7% improvement over these best known competitors in prediction accuracy. The results suggest that multi-aspect of trust indeed plays a very important role in the inference process.

(B) Performance Gain Analysis of MATRI. Let us take a close look at where the performance gain of the proposed MATRI comes from. Recall that in the proposed MATRI, we aim to integrate the three important properties of trust, that is, *multi-aspect*, *trust bias* and *trust propagation*. We next analyze how each of these properties improves the trust inference accuracy. The result is shown in Table 3. In Table 3, 'Basic form' only considers multi-aspect of trust by setting the coefficients for trust bias as well as those for trust propagation as 0*s*; 'Basic form + propagation' ignores the trust bias; 'Basic form + bias' ignores the trust propagation; and MATRI is the proposed method that integrates all three properties. We also show the result of the best known competitors, i.e., CertProp for *advogato* and Wang&Singh for *PGP*, in the table for comparison.

TABLE 3
Performance Gain Analysis of MATRI

RMSE/MAE	advogato	PGP
Best known competitor	0.269 / 0.155	0.257 / 0.169
Basic form	0.256 / 0.194	0.265 / 0.155
Basic form + propagation	0.174 / 0.124	0.214 / 0.124
Basic form + bias	0.168 / 0.119	0.189 / 0.116
MATRI	0.159 / 0.113	0.181 / 0.105

Smaller is better. Both trust propagation and trust bias further improve trust inference accuracy.

As we can see from Table 3, the performance of 'Basic form' which only considers the multi-aspect property is already close to the best known competitors. When trust propagation and trust bias are incorporated, both of them significantly improve trust inference accuracy. For example, on *advogato* data set, trust propagation helps to obtain 32.0% and 36.1% improvements in RMSE and MAE, respectively. Further, trust bias improves RMSE and MAE by 8.6% and 8.9%, respectively. To further show the usefulness of propagation when trust bias is incorporated, we also test the statistical significance of the propagation features by conducting t-test on the prediction errors between the 'Basic form + bias' method and the MATRImethod. The result shows that the improvement of incorporating propagation is indeed significant, i.e., the p-values on both data sets are less than 0.01. Overall, the performance gain analysis result confirms our hypothesis that in addition to multi-aspect, both trust propagation and trust bias also play important roles in trust inference.

(C) Comparisons with Existing Matrix Factorization Methods. We also compare MATRI with some existing matrix factorization methods: *SVD*, the low rank approximation Alg. [29] for link sign prediction (referred to as *HCD*), and the collaborative filtering Alg. [16] for recommender systems (referred to as *KBV*).

The result is shown in Table 4. As we can see from the table, MATRI again performs best on both data sets. *SVD* performs poorly as it treats all the unobserved trustor-trustee pairs as zero elements in the trust matrix $\mathbf{T}$. MATRI outperforms *HCD* as *HCD* was essentially tailored to predict the *binary* trust/distrust relationship and it ignored the other two important properties (i.e., trust bias and trust propagation). MATRI also outperforms *KBV*. For example, MATRI improves *KBV* by 11.5% in RMSE and by 16.5% in MAE on *PGP* data set. As mentioned before, *KBV* can be viewed as a special case of the proposed MATRI if we (1) fix all the bias coefficients as 1*s* and (2) ignore the trust propagation. This result indicates that by (1) incorporating the trust propagation and (2) simultaneously learning the relative weights of trust propagation and trust bias, MATRI leads to further performance improvement.

TABLE 4
Comparisons with *SVD*, *HCD* [29], and *KBV* [16]

RMSE/MAE	advogato	PGP
SVD	0.629 / 0.579	0.447 / 0.306
HCD	0.269 / 0.219	0.314 / 0.216
KBV	0.179 / 0.125	0.217 / 0.133
MATRI	0.159 / 0.113	0.181 / 0.105

Smaller is better. MATRI *performs best.*

TABLE 5
Comparisons with *EigenTrust*

RMSE/MAE	advogato	PGP
EigenTrust	0.700 / 0.653	0.519 / 0.371
MATRI	0.290 / 0.203	0.349 / 0.280

Smaller is better. MATRI *is better than EigenTrust wrt both RMSE and MAE on both data sets.*

(D) Comparisons with Existing Objective Trust Inference Methods. Although our MATRI is a subjective trust inference metric, as a side product, it can also be used to infer an objective trustworthiness score for each trustee. To this end, we set $r = 1$ in MATRI algorithm, ignore the trust propagation vectors $\mathbf{z}_{ij}$, and aggregate the resulting trustee matrix/vector $\mathbf{G}_0$ with the bias (the global bias μ and the trustee bias $\mathbf{y}$). We compare the result with a widely-cited objective trust inference model *EigenTrust* [3] in Table 5. As we can see, MATRI outperforms EigenTrust in terms of both RMSE and MAE on both data sets. For example, on *advogato* data set, MATRI is 58.6% and 68.9% better than EigenTrust wrt RMSE and MAE, respectively.

5.2.2 Effectiveness Results for Binary Case

(E) Comparisons with Existing Binary Trust Inference and Link Sign Prediction Methods. Under binary trust inference, we compare the effectiveness of MATRI with several benchmark trust prediction models and link sign prediction models, including the aforementioned *Guha* [15] and *HCD* [29], as well as *MOI-5* and *HOC-5* [30].

The result is shown in Fig. 4. We can see that the proposed MATRI significantly outperforms all the other models wrt prediction error on both data sets. For example, on *Slashdot* data set, our MATRI improves the best existing method (*Guha*) by 19.4% in prediction error. Compared to the proposed MATRI, *Guha* as well as *MOI-5* and *HOC-5* (which are all actually based on propagation or social balance theory) ignore multi-aspect and trust bias, while *HCD* ignores propagation and trust bias. This again indicates that all the three properties are helpful to improve trust inference accuracy in the binary case.

5.3 Efficiency Results

We now present the efficiency results of MATRI. For efficiency experiments, we report the average wall-clock time. All the experiments were run on a machine with two 2.4GHz Intel Cores and 4GB memory.

(A) Speed Comparison. We first present the on-line response of MATRI in the continuous case. We compare MATRI with the trust propagation models, i.e., *CertProp*, *MoleTrust*, *Wang&Singh*, and *Guha*. Again, we use the *advogato-6* snapshot and *PGP* in this experiment, and the result is shown in Fig. 5. Notice that the y-axis is in the logarithmic scale.

Fig. 4. Comparisons with benchmark binary trust inference and link sign prediction models. Lower is better. The proposed MATRI significantly outperforms all the other existing models wrt prediction error on both data sets.

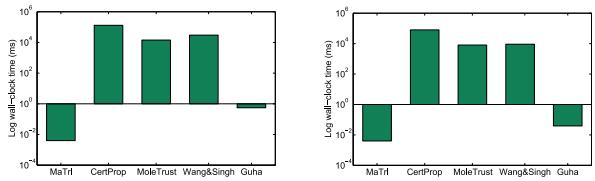


Fig. 5. Speed comparison. MATRI is much faster than all the other trust propagation methods. (a) Wall-clock time on *advogato* data set. (b) Wall-clock time on PGP data set.

We can see from the figure that the proposed MATRI is much faster than all the alternative methods on both data sets. For example, MATRI is up to 32,000,000x faster than CertProp. This is because once we have inferred the trustor/truestee matrices as well as the coefficients for the bias and propagation, it only takes *constant* time for MATRI to output the trustworthiness score. Among all the alternative methods, Guha is the most efficient. This is because its main workload can also be completed in advance. However, the pre-computation of Guha needs additional $O(n^2)$ space as the model fills nearly all the missing elements in the trust matrix, making it unsuitable for large data sets. In contrast, our MATRI only requires $O(|\mathcal{K}| + n)$ space.

(B) Scalability. Next, we present the scalability result of MATRI by reporting the wall-clock time of the pre-computational stage (i.e., step 1-13 in Alg. 4) in the continuous case. For *advogato* data set, we directly report the results on all the six snapshots (i.e., *advogato-1*, ..., *advogato-6*). For *PGP*, we use its subsets to study the scalability. The result is shown in Fig. 6, which is consistent with the complexity analysis in Section 4.3. As we can see from the figure, MATRI scales linearly wrt to both n and $|\mathcal{K}|$, indicating that it is suitable for large-scale applications. The scalability result for the binary case is similar, and we omit the figures for brevity.

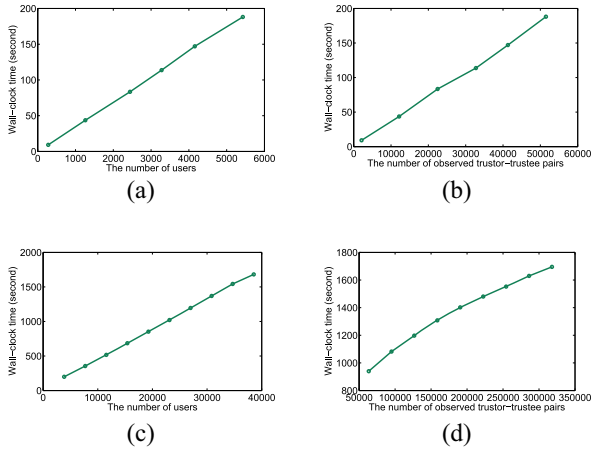


Fig. 6. Scalability of the proposed MATRI for continuous case. MATRI scales linearly wrt the data size (n and $|\mathcal{K}|$). (a) Wall-clock time vs. n on *advogato*. (b) Wall-clock time vs. $|\mathcal{K}|$ on *advogato*. (c) Wall-clock time vs. n on *PGP*. (d) Wall-clock time vs. $|\mathcal{K}|$ on *PGP*.

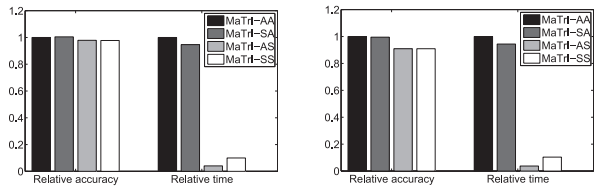


Fig. 7. Comparisons of alternative solutions of MATRI. Compared to MATRI-AA, MATRI-SS and MATRI-AS are more than 10*x* faster while preserving more than 90% accuracy on both data sets. (a) *advogato* data set. (b) *PGP* data set.

(C) Comparisons of the Alternatives of MATRI. As mentioned before, the stochastic gradient descent method (SGD) could also be used for the continuous trust inference problem in computing propagation vector and solving Eq. (5). We now experimentally evaluate the efficiency of all the four alternatives of MATRI. We use MATRI-AA to denote the original MATRI, MATRI-SA to denote the case when we use SGD in the propagation step, MATRI-AS to denote the case when we use SGD to solve Eq. (5), and MATRI-SS to denote the case when we substitute SGD for both propagation and Eq. (5). The result is shown in Fig. 7, where we also plot the effectiveness results for comparison. Notice that in the figure, we show the relative accuracy and relative wall-clock time based on MATRI-AA. As we can see, MATRI-SS and MATRI-AS are more than 10*x* faster than MATRI-AA and MATRI-SA, while preserving more than 90% accuracy on both data sets. Such tradeoffs between efficiency and effectiveness could be explored by specific applications.

6 RELATED WORK

In this section, we briefly review related work, including trust propagation models, multi-aspect trust inference models, etc.

Trust Propagation Models. To date, a large body of trust inference models are based on trust propagation where trust is propagated along connected users in the trust network, i.e., the web of locally-generated trust ratings. Based on the interpretation of trust propagation, we further categorize these models into two classes: *path interpretation* and *component interpretation*.

In the first category of path interpretation, trust is propagated along a path from the trustor to the trustee, and the propagated trust from multiple paths can be combined to form a final trustworthiness score. For example, Wang *et al*. [27], [28] as well as Hang *et al*. [25] propose operators to concatenate trust along a path and aggregate trust from multiple paths. Liu *et al*. [31] argue that not only trust values but social relationships and recommendation role are important for trust inference. In contrast, there is no explicit concept of paths in the second category of component interpretation. Instead, trust is treated as random walks on a graph or on a Markov chain [4]. Examples of this category include [1], [15], [17], [32], [33].

The proposed MATRI integrates the trust propagation with two other important properties, i.e., the multi-aspect of trust and trust bias. In addition, our multi-aspect model offers a natural way to speed up on-line query response; as well as to mitigate the sparsity or coverage problem in trust inference where some trustor and trustee might not be connected with each other - both are known limitations with the current trust propagation models [10], [22].

Multi-Aspect Trust Inference Models. Social scientists have explored the multi-aspect property of trust for several years [8]. In computer science, there also exist a few trust inference models that *explicitly* explore the multi-aspect property of trust. For example, Xiong and Liu [11] model the value of the transaction in trust inference; Wang and Wu [34] take competence and honesty into consideration; Tang *et al*. [12] model aspect as a set of products that are similar to each other under product review sites; Sabater and Sierra [35] divide trust in e-commerce environment into three aspects: price, delivering time, and quality.

However, all these existing multi-aspect trust inference methods require some additional side information other than the locally-generated trust ratings, such as the value of transaction, user's preference, product categories, etc. These methods become infeasible when such side information is not available. In contrast, MATRI directly characterizes the multi-aspect of trust solely based on the locally-generated trust ratings; and therefore it has a broader applicability.

Trust Bias in Trust Inference. In sociology, it was discovered a long time ago that *trust bias* is an integral part in the final trust decision [9]. Nonetheless, this important aspect has been largely ignored in most of the existing trust inference models. One exception is from Nguyen *et al*. [13], which learns the importance of several trust bias related features derived from a social trust framework. Recently, Mishra *et al*. [36] propose an iterative algorithm to compute trust bias. Different from these existing works, our focus is to incorporate various types of trust bias as specified factors/aspects to increase the accuracy of trust inference.

Collaborative Filtering vs. Trust Inference. Multi-aspect or low rank approximation methods have been extensively studied in collaborative filtering [16], [19], [20]. These work provides rich methodologies to capture the multi-aspect of trust by viewing the trust inference as a collaborative filtering problem. The proposed MATRI takes one step further by (1) incorporating trust bias and trust propagation; and (2) learning their relative weights.

On the application side, the goal of collaborative filtering is to predict users' flavors of items. It is interesting to point out that (1) on one hand, trust between users could help to predict the flavors as we may give a higher weight to the recommendations provided by trusted users; (2) on the other hand, trust itself might be affected by the similarity of flavors since users usually trust others with a similar taste [37]. Although out of the scope of this paper, using recommendation to further improve trust inference accuracy might be an interesting topic for future work.

Link Sign Prediction Models. There are also some recent work on link sign prediction. Leskovec *et al*. [26] formulate the link sign prediction problem and apply social balance theory to solve the problem. Later on, Chiang *et al*. [30] generalize the two-step triad structure used in [26] to longer steps, and show that such generalization leads to further effectiveness improvement. Hsieh *et al*. [29] employ the low rank matrices and formulate the link sign prediction

problem as a matrix completion problem. These methods can all be adapted in our binary trust inference problem. However, their methods either only consider trust transitivity, or only consider multi-aspect, while the proposed MATRI incorporates multi-aspect, transitivity and trust bias into one single model.

Other Related Work. The concept of stereotype for trust inference is studied by Liu *et al*. [22] and Burnett *et al*. [38]. These methods learn the stereotypes from the user profiles of the trustees that the trustor has interacted with, and then use these stereotypes to reflect the trustor's first impression about unknown trustees. Several other work focuses on trust dynamics [39] and the relationship between trust and similarity [37], [40], [41].

7 CONCLUSION

In this paper, we have proposed a trust inference model, as well as a family of algorithms to apply the model to both continuous and binary inference scenarios. The basic idea of the proposed MATRI is to leverage the multi-aspect property of trust by characterizing several aspects/factors for each trustor and trustee based on the existing trust relationships. In addition, MATRI incorporates the trust propagation and trust bias; and further learns their relative weights. By integrating all these important properties, our experimental evaluations on real benchmark data sets show that MATRI leads to significant improvement over several benchmark approaches in prediction accuracy, for both quantifying numerical trustworthiness scores and predicting binary trust/distrust signs. The proposed MATRI is also nimble - it is up to 7 orders of magnitude faster than the existing trust propagation methods in the on-line query response, and in the meanwhile it enjoys the linear scalability for the pre-computational stage in both time and space. Future work includes investigating the capability of MATRI to address the trust dynamics.

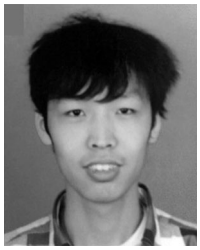
ACKNOWLEDGMENTS

This work was supported in part by the National 973 Program of China (2009CB320702), and in part by the National 863 Program of China (2012AA011205), and the National Natural Science Foundation of China (91318301, 61021062, 61073030, 61100037). It is partly supported by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053, the U.S. Defense Advanced Research Projects Agency (DARPA) under Agreement Number W911NF-12-C-0028, and the US National Science Foundation under Grant IIS-1017415.

REFERENCES

[1] C. Ziegler and G. Lausen, "Propagation models for trust and distrust in social networks," *Inform. Syst. Front.*, vol. 7, no. 4, pp. 337–358, 2005.
[2] A. Jøsang and R. Ismail, "The Beta reputation system," in *Proc. 15th Bled Electron. Comm. Conf.*, vol. 160. Bled, Slovenia, Jun. 2002.
[3] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The Eigentrust algorithm for reputation management in P2P networks," in *Proc. 12th Int. Conf. WWW*, Budapest, Hungary, 2003, pp. 640–651.
[4] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *Proc. 2nd ISWC*, Sanibel Island, FL, USA, 2003, pp. 351–368.
[5] D. Cartwright and F. Harary, "Structural balance: A generalization of Heider's theory," *Psychol. Rev.*, vol. 63, no. 5, pp. 277–293, 1956.
[6] G. Liu, Y. Wang, and M. Orgun, "Trust transitivity in complex social networks," in *Proc. AAAI*, 2011, pp. 1222–1229.
[7] D. Gefen, "Reflections on the dimensions of trust and trustworthiness among online consumers," *ACM SIGMIS Database*, vol. 33, no. 3, pp. 38–53, 2002.
[8] D. Sirdeshmukh, J. Singh, and B. Sabol, "Consumer trust, value, and loyalty in relational exchanges," *J. Marketing*, vol. 66, no. 1, pp. 15–37, 2002.
[9] A. Tversky and D. Kahneman, "Judgment under uncertainty: Heuristics and biases," *Sci.*, vol. 185, no. 4157, pp. 1124–1131, 1974.
[10] Y. Yao, H. Tong, F. Xu, and J. Lu, "Subgraph extraction for trust inference in social networks," in *Proc. IEEE/ACM Int. Conf. ASONAM*, Istanbul, Turkey, 2012, pp. 163–170.
[11] L. Xiong and L. Liu, "Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 7, pp. 843–857, Jul. 2004.
[12] J. Tang, H. Gao, and H. Liu, "mTrust: Discerning multi-faceted trust in a connected world," in *Proc. 5th ACM Int. Conf. WSDM*, Washingtion, DC, USA, 2012, pp. 93–102.
[13] V. Nguyen, E. Lim, J. Jiang, and A. Sun, "To trust or not to trust? Predicting online trusts using trust antecedent framework," in *Proc. 9th IEEE ICDM*, Miami, FL, USA, 2009, pp. 896–901.
[14] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. KDD*, New York, NY, USA, 2008, pp. 426–434.
[15] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in *Proc. 13th Int. Conf. WWW*, New York, NY, USA, 2004, pp. 403–412.
[16] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Comput.*, vol. 42, no. 8, pp. 30–37, 2009.
[17] 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.
[18] B. Lang, "A computational trust model for access control in P2P," *Sci. China Inform. Sci.*, vol. 53, no. 5, pp. 896–910, 2010.
[19] R. Bell, Y. Koren, and C. Volinsky, "Modeling relationships at multiple scales to improve accuracy of large recommender systems," in *Proc. 13th ACM SIGKDD Int. Conf. KDD*, New York, NY, USA, 2007, pp. 95–104.
[20] H. Ma, M. Lyu, and I. King, "Learning to recommend with trust and distrust relationships," in *Proc. 3rd ACM Conf. RecSys*, New York, NY, USA, 2009, pp. 189–196.
[21] A. Buchanan and A. Fitzgibbon, "Damped Newton algorithms for matrix factorization with missing data," in *Proc. IEEE CVPR*, vol. 2. Washington, DC, USA, 2005, pp. 316–322.
[22] X. Liu, A. Datta, K. Rzadca, and E. Lim, "Stereotrust: A group based personalized trust model," in *Proc. 18th ACM CIKM*, Hong Kong, China, 2009, pp. 7–16.
[23] D. Watts and S. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[24] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations," in *Proc. 11th ACM SIGKDD Int. Conf. KDD*, Chicago, IL, USA, 2005, pp. 177–187.
[25] C.-W. Hang, Y. Wang, and M. P. Singh, "Operators for propagating trust and their evaluation in social networks," in *Proc. 8th Int. Conf. AAMAS*, Budapest, Hungary, 2009, pp. 1025–1032.
[26] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proc. 19th Int. Conf. WWW*, Raleigh, NC, USA, 2010, pp. 641–650.
[27] Y. Wang and M. P. Singh, "Trust representation and aggregation in a distributed agent system," in *Proc. 21st Nat. Conf. AAAI*, 2006, pp. 1425–1430.
[28] Y. Wang and M. P. Singh, "Formal trust model for multiagent systems," in *Proc. 20th IJCAI*, San Francisco, CA, USA, 2007, pp. 1551–1556.
[29] C. Hsieh, K. Chiang, and I. Dhillon, "Low rank modeling of signed networks," in *Proc. 18th ACM SIGKDD Int. Conf. KDD*, Beijing, China, 2012, pp. 507–515.
[30] K.-Y. Chiang, N. Natarajan, A. Tewari, and I. S. Dhillon, "Exploiting longer cycles for link prediction in signed networks," in *Proc. 20th ACM CIKM*, Glasgow, Scotland, U.K., 2011, pp. 1157–1162.

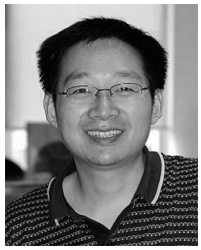
[31] G. Liu, Y. Wang, and M. Orgun, "Optimal social trust path selection in complex social networks," in *Proc. 24th AAAI*, 2010, pp. 1391–1398.
[32] U. Kuter and J. Golbeck, "Sunny: A new algorithm for trust inference in social networks using probabilistic confidence models," in *Proc. 22nd Nat. Conf. AAAI*, 2007, pp. 1377–1382.
[33] K. Nordheimer, T. Schulze, and D. Veit, "Trustworthiness in networks: A simulation approach for approximating local trust and distrust values," in *Proc. Int. Conf. IFIPTM*, vol. 321. Morioka, Japan, 2010, pp. 157–171.
[34] G. Wang and J. Wu, "Multi-dimensional evidence-based trust management with multi-trusted paths," *Future Gener. Comput. Syst.*, vol. 27, no. 5, pp. 529–538, 2011.
[35] J. Sabater and C. Sierra, "Reputation and social network analysis in multi-agent systems," in *Proc. 1st Int. Joint Conf. AAMAS*, Bologna, Italy, 2002, pp. 475–482.
[36] A. Mishra and A. Bhattacharya, "Finding the bias and prestige of nodes in networks based on trust scores," in *Proc. 20th Int. Conf. WWW*, New York, NY, USA, 2011, pp. 567–576.
[37] J. Golbeck, "Trust and nuanced profile similarity in online social networks," *ACM TWEB*, vol. 3, no. 4, Article 12, Sept. 2009.
[38] C. Burnett, T. Norman, and K. Sycara, "Bootstrapping trust evaluations through stereotypes," in *Proc. 9th Int. Conf. AAMAS*, Toronto, ON, Canada, 2010, pp. 241–248.
[39] J. Tang, H. Liu, H. Gao, and A. Das Sarmas, "eTrust: Understanding trust evolution in an online world," in *Proc. 18th ACM SIGKDD Int. Conf. KDD*, Beijing, China, 2012, pp. 253–261.
[40] C. Ziegler and J. Golbeck, "Investigating interactions of trust and interest similarity," *Decis. Support Syst.*, vol. 43, no. 2, pp. 460–475, 2007.
[41] R. Xiang, J. Neville, and M. Rogati, "Modeling relationship strength in online social networks," in *Proc. 19th Int. Conf. WWW*, New York, NY, USA, 2010, pp. 981–990.



Yuan Yao is a Ph.D. student in the Department of Computer Science and Technology, Nanjing University, China. He received the B.Sc. degree in computer science from Nanjing University in 2009. His current research interests include social network analysis, trust inference, and reputation systems.

Hanghang Tong is currently an Assistant Professor with the School of Computing, Informatics, Decision Systems Engineering, Arizona State University, Phoenix, AZ, USA. Before that, he was a Research Staff Member at IBM T.J. Watson Research Center and a Post-Doctoral Fellow in Carnegie Mellon University, Pittsburgh, PA, USA. He received the M.Sc. and Ph.D. degrees from Carnegie Mellon University in 2008 and 2009, both in machine learning. His current research interests include large scale data mining for graphs and multimedia. His research has been funded by NSF, DARPA, and ARL. He has received several awards, including the Best Paper Award in CIKM 2012, SDM 2008, and ICDM 2006. He has published over 60 referred articles and more than 20 patents. He has served as a program committee member in top data mining, databases, and artificial intelligence venues.

Xifeng Yan is an Associate Professor at the University of California at Santa Barbara, CA, USA, where he holds the Venkatesh Narayanamurti Chair of Computer Science. He received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign, IL, USA, in 2006. He was a Research Staff Member at the IBM T. J. Watson Research Center between 2006 and 2008. His current research interests include modeling, managing, and mining graphs in information networks, computer systems, social media, and bioinformatics. His works were extensively referenced, with over 7,000 citations per Google Scholar and thousands of software downloads. He received the NSF CAREER Award, the IBM Invention Achievement Award, the ACM-SIGMOD Dissertation Runner-Up Award, and the IEEE ICDM 10-year Highest Impact Paper Award. He is a member of the IEEE.



Feng Xu received the B.S. and the M.S. degrees from Hohai University, China, in 1997 and 2000, respectively. He received the Ph.D. degree from Nanjing University, China in 2003. He is a Professor with the Department of Computer Science and Technology at Nanjing University. His current research interests include trust management, trusted computing, and software reliability.

Jian Lu received the B.Sc., M.Sc., and Ph.D. degrees in computer science from Nanjing University, China. He is currently a Professor in the Department of Computer Science and Technology, and the Director of the State Key Laboratory for Novel Software Technology at Nanjing University, China. He serves on the Board of the International Institute for Software Technology of the United Nations University (UNU-IIST). He also serves as the Director of the Software Engineering Technical Committee of the China Computer Federation. His current research interests include software methodologies, software automation, software agents, and middleware systems.