Privacy preserving trusted social feedback

Anirban Basu[§], Juan Camilo Corena[§], Shinsaku Kiyomoto[§], Stephen Marsh[†]
Jaideep Vaidya[‡], Guibing Guo[¶], Jie Zhang[¶], Yutaka Miyake[§]

[§]KDDI R&D Laboratories, Inc., 2-1-15 Ohara, Fujimino-shi, Saitama 356-8502, Japan
{basu, corena, kiyomoto, miyake}@kddilabs.jp
[†]Faculty of Business and IT, UOIT, 2000 Simcoe St N, Oshawa, ON L1H 7K4, Canada
stephen.marsh@uoit.ca
[‡]MSIS Department, Rutgers, 1, Washington Park, Newark, NJ 07102, USA
jsvaidya@rutgers.edu
[¶]School of Computer Engineering, NTU, 50 Nanyang Avenue, Singapore 639798
{gguo1, zhangj}@ntu.edu.sg

ABSTRACT

With the growth of social networks, recommender systems have taken advantage of the social network graph structures to provide better recommendation. In this paper, we propose a privacy preserving trusted social feedback (TSF) system, in which users obtain feedback on questions or items from their friends. It is different from and independent of a typical recommender system because the responses from friends are not automated but tailored to specific questions. TSF can be used to complement the results from a recommender system. Our experimental prototype runs on the Google App Engine and utilises the Facebook social network graph. In our experimental evaluation, we have looked at users' perceptions of privacy and their trust in the prototype as well as the performances on the client side and the cloud side.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering

General Terms

Human Factors, Security, Algorithms

Keywords

Privacy, Trust, Recommendation, Social Network

1. INTRODUCTION

Rating based collaborative filtering (CF) has become the de-facto standard for generating personalised recommendations in order to help users cope with the problem of ever increasing information overload. In general, CF schemes work with the similarities or differences between users, or items, or from some kind of compact notations such as matrix factorisations. With the growth of online social networks, newer CF schemes have been proposed to take advantage of the social link graph structures and some sort of weight on each relevant edge in the graph. Sometimes paths through the graph (or friend-of-a-friend relations) are discovered to find opinions that are used by the recommender systems. Privacy of rating data from participating users is an important concern especially when such recommender systems are deployed on public cloud platforms, and privacy preserving recommendation is a well researched area.

We take a look at social aware recommendation from a different angle. We postulate that the strength of a social relation is often one's asymmetric personal perception of another in a particular context that changes over time. We refer to this as *trust* in this paper[1]. The asymmetric nature of personal perception means that a's trust on b is likely to be different from b's trust on a. This should affect the way one believes a recommendation from a friend. Recommendations based on a community of opinions do not generally consider this interpersonal and contextual trust. In tune with this understanding of trust and modelling after the real society, we observe that a recommendation could have two stages in its lifecycle. The first stage helps the user obtain an automated targeted recommendation based on the opinion of a community. This stage could also use the user's history amongst other information. In the second stage, the user asks for the aggregate feedback from her friends, in her social network, regarding the aforementioned recommendation. She attaches a certain level of contextual trust to each friend that she asks the question. This non-automated second stage is what we call the *trusted social feedback* (TSF). Assuming that such a recommender system will be deployed on a cloud, the TSF proposal must be privacy preserving. Although limited to the scope of feedback for item recommendation, we are looking into extending TSF to a generalised question-answer service in the future.

We note that automated personal trust transitivity is debatable and subjective. It exists but modelling it is difficult. Jøsang et al. in [16] go as far as saying "[...] all mathemati-

[1]Apart from this notion of trust, we refer to the concept of foreground trust [6] in section 4.

cal operators for trust transitivity proposed in the literature must be considered *ad hoc*; they represent attempts to model a very complex human phenomenon as if it were lendable to analysis by the laws of physics". The authors propose a radically different interpretation of trust transitivity based on subjective logic. The authors observe that in order for transitivity to function, the *advisor* must, in some way, communicate his/her trust on the *trust target* to *the originator relying party*. Thus, we rule out automatic estimation of propagated trust.

The rest of the paper is organised as follows. We describe our proposal of trusted social feedback in section 2 and its security analysis in section 3. This is followed by the experimental evaluation of our system in section 4. The state-of-the-art is described in section 5 before we conclude in section 6.

2. TRUSTED SOCIAL FEEDBACK

Irrespective of the means by which the generalised recommendation is obtained, the user can ask people in her social network for trusted social feedback (TSF) on a query. For our purposes, *feedback* is a numeric rating in response to a *query*. A query is defined as a question for soliciting an opinion on an item or topic of interest. For instance, a query could be "What is your opinion on the Canon 5D Mark III DSLR camera?".

The feedback acts as a trust empowering information aid to the user in making a choice. In the simplest case, the feedback is an average of the feedback inputs from all friends within one degree of separation, each weighted by the directional trust the user has on that friend. This is similar to the model presented in the FilmTrust work by Jennifer Golbeck [9]. The feedback is obtained per query. Because of the dynamic nature of queries as well as the trust levels specified during queries, no feedback can be pre-defined or stored on the cloud platform that hosts the social network.

In order to preserve privacy, TSF must ensure the non-disclosure of: (a) the directional trust values in a query to the friends and to the social network; and (b) the feedback from a particular friend of the user to the social network and the user.

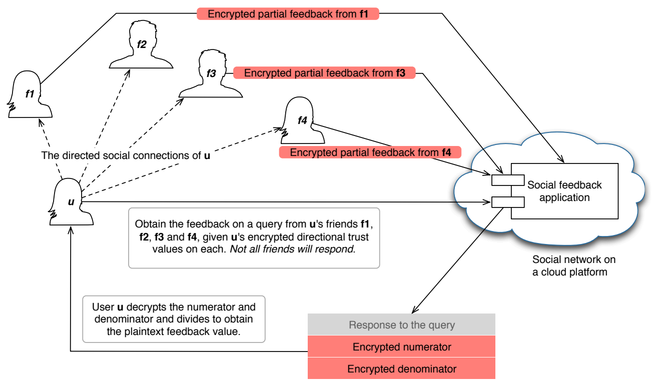
The Paillier public-key cryptosystem [22] exhibits additively homomorphic properties, which we utilise in our proposal. Denoting encryption and decryption functions by $\mathcal{E}()$ and $\mathcal{D}()$ respectively, the encryption of the sum of two plaintext messages m_1 and m_2 is the modular product of their individual ciphertexts:

$$\mathcal{E}(m_1 + m_2) = \mathcal{E}(m_1) \cdot \mathcal{E}(m_2) \tag{2.1}$$

while, the encryption of the product of one plaintext messages m_1 and a plaintext integer multiplicand π is the modular exponentiation of the ciphertext of m_1 with π as the exponent:

$$\mathcal{E}(m_1 \cdot \pi) = \mathcal{E}(m_1)^{\pi}. \tag{2.2}$$

Let us denote the directional trust from user a to friend b as $\mathcal{T}_{a \to b}$, the feedback from a friend i on a query k as $\omega_{i,k}$ and the total number of friends responding to the query as n. The trust value and the individual feedback value are discrete integers. The trusted feedback on query k for user



Encrypted partial feedback from **f1**
f1
f2
f3
Encrypted partial feedback from **f3**
f4
Encrypted partial feedback from **f4**
The directed social connections of **u**
u
Social feedback application
Obtain the feedback on a query from **u**'s friends **f1**, **f2**, **f3** and **f4**, given **u**'s encrypted directional trust values on each. *Not all friends will respond.*
Social network on a cloud platform
User **u** decrypts the numerator and denominator and divides to obtain the plaintext feedback value.
Response to the query
Encrypted numerator
Encrypted denominator

Figure 2.1: Overview of the trusted social feedback mechanism.

u is given as:

$$\mathcal{F}_{u,k} = \frac{\sum_{i|i \neq u}^{n} \omega_{i,k} \mathcal{T}_{u \to i}}{\sum_{i|i \neq u}^{n} \mathcal{T}_{u \to i}} \tag{2.3}$$

This computation can be performed over the (additively homomorphic) encrypted domain for user u as:

$$\mathcal{F}_{u,k} = \frac{\mathcal{D}(\prod_{i|i \neq u}^{n} \mathcal{E}(0, r_i) \mathcal{E}(\mathcal{T}_{u \to i})^{\omega_{i,k}})}{\mathcal{D}(\prod_{i|i \neq u}^{n} \mathcal{E}(\mathcal{T}_{u \to i}))} \tag{2.4}$$

The encryption of zero performed by the friend i, (denoted as $\mathcal{E}(0, r_i)$) ensures[2] that the encrypted partial feedback from friend i, i.e., $\mathcal{E}(0, r_i)\mathcal{E}(\mathcal{T}_{u \to i})^{\omega_{i,k}}$ does not reveal $\omega_{i,k}$ despite the cloud's knowledge of $\mathcal{E}(\mathcal{T}_{u \to i})$, unless the user u and the cloud collaborate. The formal proof is in section 3.2. The trusted social feedback mechanism is illustrated in figure 2.1 and is described in algorithm 2.1[3]. While sending the question, the user attaches an encrypted trust value for each friend to the question such that when a friend responds, the response is homomorphically multiplied by the trust value. The cloud aggregates those individual responses from the friends and sends back the aggregate response to the user after a threshold number of friends have responded.

As trust is personal and idiosyncratic [7], our proposed feedback mechanism are only there for trust empowerment, not to enforce a trust decision on the user. What the user does with the feedback is solely her choice. Therefore, a mathematical model for trust transitivity over multiple degrees of separation in the social network graph is often inadequate and meaningless because the model would tend to suggest a particular trust level. Trust is also sensitive to changes over time and context. In our proposal, the trust values can be as short-lived as a single query, which caters for temporal changes. The user can solicit the response to her query from a selected group of friends, thereby enabling context sensitive trust values for friends. Thus, the queries in TSF are short-lived and context sensitive.

Untrust [18], which can be expressed in our proposed feedback mechanism, is also context sensitive. This means that

[2]The notations $\mathcal{E}(x, r_u)$ and $\mathcal{E}(x)$ are synonymous, i.e., encryption performed by the user u. The random number notation is used only when the operation is performed by some other user i with u's public key, i.e., $\mathcal{E}(x, r_i)$.

[3]The $\cdot$ is used to denote multiplication for the sake of readability.

Algorithm 2.1 Computing the trusted social feedback for user u on item k.

```
Require: Additively homomorphic encrypted domain for user $u$, i.e., $\mathcal{E}$ and corresponding public key.
Require: Encrypted directional trust $\mathcal{E}(\mathcal{T}_{u\to i})$ from user $u$ to each friend $i$.
 1: for each encrypted directional trust $\mathcal{E}(\mathcal{T}_{u\to i})$ do
 2:   if $i$ wishes to respond then
 3:     $i$ computes encrypted partial feedback,
          $\psi_i \leftarrow \mathcal{E}(0,r_i)\cdot\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}$
 4:     social network updates encrypted trusted feedback,
          $\Psi \leftarrow \Psi \cdot \psi_i$
 5:     social network updates encrypted response cardinality,
          $\eta \leftarrow \eta \cdot \mathcal{E}(\mathcal{T}_{u\to i})$
 6:   end if
 7: end for
 8: return encrypted trusted feedback, $\Psi$.
 9: return encrypted response cardinality, $\eta$.
10: user $u$ obtains the trusted social feedback, $\mathcal{F}_{u,k} = \frac{\mathcal{D}(\Psi)}{\mathcal{D}(\eta)}$.
```

Alice could trust her friend Bob for an opinion on cloud security but at the same time untrust him regarding any opinion on quantum entanglement. Untrust can prove useful to accept negative feedback or reject positive feedback from untrusted friends for specific queries. In our current prototype, we do not model untrust.

3. SECURITY ANALYSIS

3.1 Adversary model

In this discussion, the word *cloud* will be considered synonymous with *social network* in terms of threats because a social network is usually deployed on a cloud environment. Thus, the internal privacy threats to the social network can arise from the cloud infrastructure. We assume that the parties involved in this process are honest but curious. Therefore, attacks involving collaborations between the cloud and the attacker are not considered as realistic threats although we have described some such possible attacks. For a malicious user, a specialised attack for partial response disclosure is also described in section 3.3.

3.1.1 Curious user, multi-query and sybil attacks

The user can run multiple queries requesting the feedback on the same question from the same set of friends. In doing so, and by varying the user's directional trust on each friend, the user can acquire the information necessary to reveal the feedback provided by each friend. However, the feedback response is slow and some friends may choose not to respond. Furthermore, the feedback from the same person may vary over time. Therefore, using a multi-query attack is not guaranteed to succeed. To further enhance the privacy of the feedback, a friend can perturb his/her feedback input in bounded integral ranges – an avenue we have left open for future work.

However, in a *sybil attack* the user asks a question to one real friend and a number of sybil identities. Upon receiving the responses, the asker can find out the exact response from the real friend given the knowledge of those from the sybil identities. Our model is not resistant against sybil attacks, which we aim to investigate in the future.

3.1.2 Curious cloud, man-in-the-middle attack

Despite the query itself being sent in clear text, the directional trust values from the user to the friends and the partial feedback from each friend are both in the encrypted domain of the user. Even though the cloud knows the encrypted directional trust value, it cannot decipher the actual feedback from any friend since encrypted zero, i.e., $\mathcal{E}(0, r_i)$, is homomorphically added by each friend thus making the encrypted trusted feedback component probabilistic. The cloud, however, can tell which friends responded to the query.

3.1.3 Curious friend

The friend cannot determine the directional trust value because it is encrypted by the user's public key.

3.1.4 Collaborative attacks

If the user and the cloud collaborate then all the partial feedbacks can be deciphered since the cloud will be able to decrypt partial feedback values with the help of the user. If a friend and the cloud collaborate, the friend can learn how many other friends responded to the query but it cannot decipher the actual individual feedback values. If the user and a friend collaborates, they can learn about each others' secrets – the directional trust value and the feedback.

3.1.5 Out-of-the-range attacks

Both the friend and the cloud can encrypt arbitrary numbers and send them to the user in the response. Homomorphic range check protocols [23] may be applicable to protect those scenarios but this falls within the remits of future work.

3.2 Proof of obfuscation by encryption of zero

Since the numeric feedback on item k from a friend, i, is in a fixed discrete integral range, the cloud can attempt to learn it by pre-computing all possible values[4] of $\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}$ using a trial-and-error method of dividing what the friend sends by the pre-computed value to eliminate the obfuscating encryption of zero. Let us assume that the correct value of $\omega_{i,k}$ in question is ω_1 and a wrong value is ω_2. This is what happens.

3.2.1 Case A: correct pre-computed value

If the cloud used the correct pre-computed value: $\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_1}$, we have:

$$\begin{aligned}\frac{\mathcal{E}(0,r_i)\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}}{\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_1}} &= \mathcal{E}(0,r_i)\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}-\omega_1}\\ &= \mathcal{E}(0,r_i)\end{aligned}$$

Now, the cloud computes:

$$\begin{aligned}\frac{\mathcal{E}(0,r_i)\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}}{\mathcal{E}(0,r_i)} &= \mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}\\ &= \mathcal{E}(\mathcal{T}_{u\to i})^{\omega_1}\end{aligned}$$

[4]Note that this homomorphic multiplication has deterministic values.

Thus, the cloud obtains the same value as the one it pre-computed.

3.2.2 *Case B: wrong pre-computed value*

If the cloud used a wrong pre-computed value: $\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_2}$, we have:

$$\frac{\mathcal{E}(0,r_i)\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}}{\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_2}} = \mathcal{E}(0,r_i)\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}-\omega_2}$$

Now, the cloud computes:

$$\begin{aligned}\frac{\mathcal{E}(0,r_i)\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}}{\mathcal{E}(0,r_i)\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}-\omega_2}} &= \mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}-\omega_{i,k}+\omega_2}\\ &= \mathcal{E}(\mathcal{T}_{u\to i})^{\omega_2}\end{aligned}$$

Here again, the cloud obtains the same value as the one it pre-computed.

Since the results from both the right and the wrong guesses are indistinguishable, the cloud cannot guess which one is the true value of $\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}$ and hence $\omega_{i,k}$.

3.3 A specialised partial response disclosure attack

Our construction is not inherently secure against a malicious user wishing to know the responses of her friends from the aggregate encrypted values. This attack consists of creating a vector with several coordinates inside a single encrypted value. These coordinates can be read independently by the malicious user. Consider an x-bit number and treat it as a vector of dimension y, where each coordinate is represented using $\frac{x}{y}$ bits. If operations are performed on this vector with no individual coordinate exceeding $2^{\frac{x}{y}}-1$; then there is no loss of information for that coordinate. The following example illustrates this idea.

1. Assume $x = 16$ and $y = 4$, then each coordinate can represent values in the range $[0 \quad 15]$.

2. The user asks four friends, i.e., $f_1 \ldots f_4$ a question using the following trust values (spaces introduced for readability), represented as bit sequences.

$$\begin{aligned}\mathcal{T}_{u\to f_1} &= 0000\,0000\,0000\,0001\ [decimal:1]\\ \mathcal{T}_{u\to f_2} &= 0000\,0000\,0001\,0000\ [decimal:32]\\ \mathcal{T}_{u\to f_3} &= 0000\,0001\,0000\,0000\ [decimal:512]\\ \mathcal{T}_{u\to f_4} &= 0001\,0000\,0000\,0000\ [decimal:8192]\end{aligned}$$

3. Each friend provides his/her response in the range $[1 \quad 15]$ weighted by the ingress trust value, i.e., $\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}$. The encryption of zero is left out for simplicity because it does not stop this attack, which happens in the plaintext domain.

4. The cloud aggregates the resultant numerator as:

$$\prod_{i|i\neq u}^{n}\mathcal{E}(\mathcal{T}_{u\to i})^{\omega_{i,k}}$$

Since none of the coordinates in the numerator has a value greater than 15, the malicious user can extract the answer from each friend by reading the decrypted numerator, 4-bits at a time. The technique also works for trust values where the particular non-zero nibble is greater than 0001, for example $0010\,0000\,0000\,0000\ [decimal:16384]$. In that case, the malicious user simply needs to adapt the coordinate length accordingly and once extracted, divide it by the original trust value assigned to that particular friend.

To prevent this attack, a proof stating that the trust values are in a given range is necessary. Alternatively, if the number of friends asked is large enough in comparison with the bit space of the plaintext trust values then the bit manipulations will overlap, thus making it impossible for the attacker to identify individual ratings.

4. EXPERIMENTAL EVALUATION

In this section we present the results from the user studies, the performance evaluation of the speed of cryptographic primitives on the web front-end and the speed of the essential functions of the prototype at the back-end. Our experimental prototype runs on the Google App Engine for Java. The application uses Facebook to perform user login and to determine social connections.

4.1 Measuring perception of privacy and foreground trust

The user study evaluated how users perceived the application's ability to preserve privacy and to measure users' trust in the application, which relates to the concept of foreground trust [6]. It is different from the trust between friends that we have discussed so far. Dwyer et al. in [7], suggested that a reduction of uncertainty is positively correlated with the increase of trust. Thus, a measure of uncertainty is used to infer trust. In our user study with 12 participating users, we have employed pre-use and post-use questionnaires to determine the changes in uncertainty. The users are highly technically competent and were aware of this research work before using the prototype. Table 4.1 shows that the uncertainty in the users' responses usually declined, thus suggesting a likely increase in foreground trust.

	Q1	**Q2**	**Q3**	**Q4**
U1	Reduced	No change	Reduced	Reduced
U2	Reduced	Reduced	Reduced	No change
U3	No change	Reduced	Reduced	Reduced
U4	Reduced	Reduced	Increased	No change
U5	Reduced	Reduced	Reduced	Reduced
U6	No change	Reduced	No change	Reduced
U7	Reduced	No change	Reduced	Reduced
U8	No change	Reduced	Reduced	No change
U9	Reduced	Reduced	Reduced	No change
U10	No change	Reduced	Reduced	Reduced
U11	Reduced	Reduced	Reduced	Reduced
U12	Reduced	Reduced	No change	Reduced

(a) Change in response uncertainties for each question per user.

	Increased	**No change**	**Reduced**
Q1	0 (0%)	4 (33%)	8 (67%)
Q2	0 (0%)	2 (17%)	10 (83%)
Q3	1 (8%)	2 (17%)	9 (75%)
Q4	0 (0%)	4 (33%)	8 (67%)

(b) Change in response uncertainties per question.

Table 4.1: Change in uncertainties associated with 4 questions and 12 users. See the actual questions in section 4.1.1.

4.1.1 *User questionnaire*

Q1 How is your understanding about what you can do with this application?

Q2 How well do (did) you feel that the application will preserve (preserved) the privacy of the personal trust levels that you have on your friends, and the privacy of the responses from your friends?

Q3 How useful do you think is this application?

Q4 How likely are you to use such an application, if available publicly?

Each question was followed by a question to measure uncertainty: *How certain are you about your previous response?*. Responses to each question was recorded in a 7-point Likert scale [17].

4.2 Performance

The speed at which a feedback can be obtained depends almost entirely on the speed at which friends respond to the question; and to some extent on the speed of cryptographic operations and that too on the client-side because the speed of the limited cryptographic operations on the cloud-side is usually negligible compared to delays caused by network latencies, cloud instance initialisations, and datastore access. For every partial response submitted by a friend, the cloud is responsible for exactly two homomorphic additions, see line 4 in algorithm 2.1. We present a comparison of performances of cryptographic primitives on the client side. We have built a Google Web Toolkit wrapper for an optimised Javascript implementation of the Paillier cryptosystem using the Stanford Javascript BigInteger library. The result of each test, in table 4.2, is a rounded-off average from 50 runs. The tests were carried out on Windows 8, running on a 64-bit 3.4GHz Intel i7-3770 dual quad-core processor with 16GB RAM. The versions of the browsers are: Chrome 28.0.150072m, Firefox 22.0, IE 10.0.9200.16599 and Safari 5.1.7. IE and Safari failed to finish the tests when the cryptosystem was set to 1024 bits, so we used a 512 bits cryptosystem for our tests.

	Chrome	**Firefox**	**IE**	**Safari**
KG-512	69	57	441	1032
E-512	23	16	195	367
HA-512	2	2	9	27
HM-512	11	8	104	214
D-512	23	15	195	371

Table 4.2: Performances of Paillier in Javascript. Times are in milliseconds. KG: key generation, E: encryption, HA: homomorphic add; HM: homomorphic multiplication; D: decryption. The number suffixed to these abbreviations indicate cryptosystem bit size.

Using the F1 (600MHz) instance class and the high-replication datastore of the Google App Engine, the averages of the times taken for the different servlet calls are shown in table 4.3. The time taken for a particular function call also includes the time taken to execute any intermediate servlet filters, for instance the filter that verifies the logged-in users.

Servlet:Action	**Call count**	**Time (ms)**
`profile:getProfile`	121	3128
`profile:getTopUsers`	207	1816
`profile:savePublicKey`	36	2012
`qaserv:answerQuestion`	195	233
`qaserv:askQuestion`	81	1826
`qaserv:myNotifications`	552	783
`qaserv:myQuestions`	262	1779

Table 4.3: The average times taken for various servlet function calls. The `profile` servlet is responsible for user profile specific functions while `qaserv` deals with questions and their responses.

5. RELATED WORK

Herlocker et al.'s work [13] is one of the older works on automated collaborative filtering algorithms. Golbeck's work [9] on FilmTrust utilised trust in social networks for movie recommendations. Guo's work [12] is the closest to ours in the way they combined opinions of neighbours in a social network, weighted by trust values. Unlike our proposal, the paper used the concept of trust propagation and it does not preserve privacy in the aggregation process. Trust propagation is a hard-to-model subjective concept. Two recent proposals: [16] and [20] describe interesting ways of looking at trust propagation. Jamali and Ester [15] employed matrix factorisation to deduce trust propagation, which was then used in collaborative filtering. TidalTrust [10] and MoleTrust [19] are similar with the latter considering ratings from a maximum depth only, in a breadth first search over a trust network to compute a prediction. In [21], authors suggested that the traditional emphasis on user similarity in recommender systems was overstated, and proposed two trust based recommender system solutions. TrustWalker [14] used a random walk method to combine item-based collaborative filtering with trust-based recommendation.

Privacy preserving collaborative filtering (PPCF) has been studied by many [1–3, 5, 11, 25, 26]. Existing work can be classified into using either cryptographic or perturbation techniques to preserve privacy. Very few of these proposals have been tested on real world cloud platforms. Canny's work [4] utilised factor analysis and homomorphic encryption for PPCF; while in [11], the authors computed PPCF from a combination of random perturbation and secure multiparty computation. Polat's works [24, 26] have used randomisations to preserve privacy. Several question-answer services exist, including commercial ones, such as Yahoo! Answers, Aardvark. Fleming's thesis [8] proposed a privacy enhanced question-answer system based on stigmergic routing where privacy is provided by plausible deniability in a decentralised network.

The field of social aware recommendation is relatively new in comparison with traditional recommender systems. Our proposal is about trust empowerment because we see trust as an idiosyncratic, context sensitive, neither entirely rational nor subjective feeling that changes over time [7]. Our privacy preserving solution also considers privacy from a user-centric perspective. To the best of our knowledge, there is no single social aware recommender system that is privacy preserving and views trust from a trust empowerment instead of an enforcement approach.

6. CONCLUSIONS

In this paper, we have presented a novel working prototype for obtaining feedback on queries from one's trusted friends in a privacy preserving manner. We have implemented and tested the prototype on the Google App Engine with Facebook, and have run a user study to evaluate foreground trust. In the future, we expect to run more exhaustive user tests and also extend this system further to a generic question answer service where people answering questions are domain experts instead of just friends.

7. REFERENCES

[1] A. Basu, J. Vaidya, and H. Kikuchi. Perturbation based privacy preserving Slope One predictors for collaborative filtering. In *Proceedings of the 6th IFIP WG 11.11 International Conference on Trust Management (IFIPTM), Surat, India*, 2012.

[2] A. Basu, J. Vaidya, H. Kikuchi, T. Dimitrakos, and S. K. Nair. Privacy preserving collaborative filtering for SaaS enabling PaaS clouds. *Journal of Cloud Computing: Advances, Systems and Applications*, 1(8), 2012.

[3] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci. Privacy-enhanced collaborative filtering. In *Proc. User Modeling Workshop on Privacy-Enhanced Personalization*, 2005.

[4] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, pages 238–245, New York, NY, USA, 2002. ACM.

[5] R. Cissée and S. Albayrak. An agent-based approach for privacy-preserving recommender systems. In *Proceedings of the 6th International Joint Conference on Autonomous agents and Multiagent Systems*, pages 1–8. ACM, 2007.

[6] N. Dwyer. *Traces of digital trust: an interactive design perspective*. PhD thesis, Victoria University, 2011.

[7] N. Dwyer, A. Basu, and S. Marsh. Reflections on measuring the trust empowerment potential of a digital environment. In *Proceedings of the IFIP WG11.11 International Conference on Trust Management (IFIPTM), Malaga, Spain*, 2013.

[8] S. Fleming. *An ant-inspired, deniable routing approach in ad hoc question & answer networks*. PhD thesis, University of Sussex, 2012.

[9] J. Golbeck. Generating predictive movie recommendations from trust in social networks. In *Proceedings of the 4th International Conference on Trust Management (iTrust), Pisa, Italy*. Springer, 2006.

[10] J. A. Golbeck. *Computing and applying trust in web-based social networks*. PhD thesis, University of Maryland, College Park, MD, USA, 2005. AAI3178583.

[11] S. Gong. Privacy-preserving collaborative filtering based on randomized perturbation techniques and secure multiparty computation. *IJACT: International Journal of Advancements in Computing Technology*, 3(4), 2011.

[12] G. Guo, J. Zhang, and D. Thalmann. A simple but effective method to incorporate trusted neighbors in recommender systems. In *Proceedings of the 20th International Conference on User Modeling, Adaptation, and Personalization (UMAP)*, pages 114–125. Springer, 2012.

[13] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999.

[14] M. Jamali and M. Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 397–406. ACM, 2009.

[15] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 135–142, New York, NY, USA, 2010. ACM.

[16] A. Jøsang, T. Aždersk, and S. Marsh. Trust transitivity and conditional belief reasoning. In *Trust Management VI*, pages 68–83. Springer, 2012.

[17] R. Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[18] S. Marsh and M. R. Dibben. Trust, untrust, distrust and mistrust – an exploration of the dark(er) side. In *Trust Management*, pages 17–33. Springer, 2005.

[19] P. Massa and P. Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 17–24, New York, NY, USA, 2007. ACM.

[20] T. Muller and P. Schweitzer. On beta models with trust chains. In *Trust Management VII*, pages 49–65. Springer, 2013.

[21] J. O'Donovan and B. Smyth. Trust in recommender systems. In *Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174. ACM, 2005.

[22] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology – EUROCRYPT'99*, volume 1592, pages 223–238. Springer, 1999.

[23] D. C. Parkes, M. O. Rabin, S. M. Shieber, and C. Thorpe. Practical secrecy-preserving, verifiably correct and trustworthy auctions. *Electronic Commerce Research and Applications*, 7(3):294–312, 2008.

[24] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 625–628. IEEE, 2003.

[25] H. Polat and W. Du. Privacy-preserving collaborative filtering on vertically partitioned data. *Knowledge Discovery in Databases: PKDD 2005*, pages 651–658, 2005.

[26] H. Polat and W. Du. Achieving private recommendations using randomized response techniques. *Advances in Knowledge Discovery and Data Mining*, pages 637–646, 2006.