Deriving Private Data in Vertically Partitioned Data-based PPCF Schemes

Murat Okkalioglu, Mehmet Koc, and Huseyin Polat

***Abstract*— New companies might lack enough data for collaborative filtering. This problem can be overcome if different companies collaborate by sharing their data. However, these companies should protect their user data. Therefore, companies must take privacy measures to prevent data disclosure. There are some studies in the privacy-preserving collaborative filtering literature focusing vertically partitioned binary rated data for two-party cases. In this study, we focus on such two-party vertically distributed privacy-preserving collaborative filtering schemes and experimentally analyze privacy offered by a specific scheme against three different attack types in the literature. We show that data can be derived through such attacks.**

***Index Terms*— Privacy, collaborative filtering, binary, vertically partitioned data, attack scenarios**

I. Introduction

Amount of information that customers can process has some limits and this limit makes the decision making process difficult. E-companies might collect implicit (browsing, purchase history, time spent, etc.) and explicit data (ratings, reviews, etc.) about their customers for collaborative filtering (CF), which is a technique to offer predictions on user data [1]. A typical CF system is composed of an $n \times m$ matrix with n users and m items. The users can express their ratings as binary (*like* or *dislike*).

Since CF systems usually have a sparse matrix, data sparsity is a crucial problem [1]. User participation is important to obtain accurate referrals. However, users may be unwilling to participate in providing their true preference due to privacy risks like unsolicited marketing, price discrimination, unauthorized access, and government surveillance [1]. Thus, privacy-preserving collaborative filtering (PPCF) aims to protect privacy and provide accurate referrals [1]. Online vendors might collaborate by sharing their ratings to fill their missing ratings. Two companies could have ratings for the different set of items by the same customers. This type of partitioning is known as vertically partitioned data (VPD). There are different VPD-based PPCF schemes to offer predictions [2, 3].

Although PPCF schemes promise individual privacy, there are some studies arguing that they do not protect the privacy as much as believed [4, 5, 6, 7]. Data perturbed by randomization has predictable nature and data perturbed by this method can be extracted using spectral filtering (SF) [4]. Zhang et al. [5] propose two different techniques to disclose user ratings masked by randomization. The authors in [6] analyze how to derive rated items in PPCF, which are disguised by randomized response technique (RRT). In [7], the authors study three attack scenarios for a two-party horizontally partitioned data (HPD)-based PPCF scheme proposed in [3]. In this study, however, we examine how to derive private data in VPD-based binary PPCF scheme proposed in [3]. Our aim is to show *how much privacy can be achieved by a specific VPD-based PPCF binary scheme, where data is partitioned between two parties*.

II. Related work

Polat and Du [8] apply RRTs on binary data to disguise ratings. Kaleli and Polat [9] utilize RRTs to produce predictions on naïve Bayesian classifier (NBC). While the previous studies handle the central server-based scenarios, there are some other studies focusing on partitioned data [2, 3, 10, 11, 12]. The researchers in [10, 11, 12] also propose multi-party schemes. The authors in [10] utilize NBC for HPD- and VPD-based data. Self-organizing maps-based predictions are proposed for HPD- [11] and VPD-based [12] schemes.

Kargupta et al. [4] propose an SF technique to extract the original data perturbed by random perturbation. Their method is based on obtaining theoretical boundaries of maximum and minimum values of eigenvalues of the noise matrix. Some researchers study the bounds of the reconstruction error by SF methods [13]. Principal component analysis is also utilized to reconstruct the original data by exploiting data correlations [14]. When the correlation is high, more accurate reconstructions can be performed for random perturbation. Zhang et al. [5] examine how to derive true data in PPCF schemes. They utilize singular value decomposition and *k-means* clustering to reconstruct the original data. They apply *k-*

Manuscript received August 22, 2015. This work was supported by TUBITAK under Grant 113E262.

M. Okkalioglu is with the Computer Engineering Department, Yalova University, Yalova, 77200, TURKEY (e-mail: murat.okkalioglu@yalova.edu.tr).

M. Koc is with Electrical and Electronics Engineering Department, Seyh Edabali University, Bilecik, 11210, TURKEY (e-mail: mehmet.koc@bilecik.edu.tr).

H. Polat is with the Computer Engineering Department, Anadolu University, Eskisehir, 26470, TURKEY (phone: +90 222 321 3550; fax: +90 222 323 9501; e-mail: polath@anadolu.edu.tr).

means clustering to get the data in groups for discrete and continuous valued data. They discretize the continuous data into *k* segment and an item is assigned to the median value of the segment it belongs to after clustering. Binary PPCF scheme in [8] is investigated in terms of disclosing which items are rated [6]. The authors utilize publicly collected information about the target data set and manage to retrieve this private information.

Our approach in this paper is to deal when data is partitioned vertically between two parties with binary data. Okkalioglu et al. [7] study how much privacy is offered when data is partitioned horizontally between two-parties. They utilize possible attack techniques (*acting as an active user* and *knn-based*) and propose an attack technique, *perfect match* attack. We extend our prior study [7] for a VPD-based binary PPCF scheme.

III. Preliminaries

Polat and Du [3] offer a top-*N* (*TN)* prediction scheme for the active user, *a*, among the item list (N_a) she wants a prediction. Privacy measures are taken by the parties to overcome possible privacy breaches. From now on, *A* and *B* will denote each party. This scheme selects the users who have high positive and negative correlations with *a* claiming that accuracy might be increased if the best similar and dissimilar users are selected as follows:

$$W_{au} = \frac{t(R_s) - t(R_d)}{t(R)} \quad (1)$$

In Eq. (1), W_{au} is the similarity weight between the user *u* and *a*. $t(R_s)$, $t(R_d)$, and $t(R)$ are the number of similarly, dissimilarly, and commonly rated items by both *u* and *a*, respectively.

After determining W_{au}, neighbors are picked on *best-N* or *threshold* methods. In the *best-N* neighbors' selection, *N* number of users with the highest correlations is picked as neighbors. *Threshold* neighbors' selection method picks its neighbors among the users whose correlations surpassing a threshold (τ_n) value. Next, number of *likes* (l_j) and *dislikes* (d_j) among the selected neighbors are estimated, where *j* is item number. Then, $ld_j = l_j - d_j$ is calculated. If $ld_j > 0$, the item will be liked by *a*. Otherwise, it will be disliked.

Private similarity computation protocol (PSCP) is utilized to privately compute the similarities. PSCP considers the cases where *a*'s incoming query is sparse or dense. Either party who receives the incoming query can apply PSCP. Assume that *A* applies the protocol as a master party. First, *A* finds the number of rated items, *M*, by *a* and the protocol follows two different path on its density. In the sparse case, if *M* is less than [*m/2*], then *A* finds the unrated items of *a*. A random number Y_{Aa} is created by *A* from the range (1, *m-M*). Y_{Aa} number of cells is selected to fill using private default vote computation protocol calculating a default vote for the items on ld_j values. The dense case handles the situation, where *M* is greater than *m/2*. A uniform random number Y_{Ar} is selected by *A* from the range (1, *M*). Then, *A* randomly selects Y_{Ar} items among the *M* rated items and removes them from *a*'s query. In VPD, target items (N_a) that *a* is looking for a prediction might belong to different parties. This scheme offers two different cases. The first one deals with the case, where all N_a items belong to the one of the parties. The second case is designed when N_a items are shared between parties. The first case is called *Case-All* and the second one refers to *Case-Split*.

Items belong to both parties in VPD. N_a items might belong to the one of parties. *Case-All* scheme deals with this special case assuming that *B* has all items [3]: (1) *a* sends her corresponding ratings to both parties and N_a to only *B*. *A* computes the required values to calculate similarities using PSCP. (2) *A* sends the partial similarity values to *B* through *a*. B finds its own partial similarity values between users it holds and *a*. Then, *B* calculates the final similarities (W_{au}) adding the partial similarities received from *A* to its own calculated ones. (3) *B* selects the best neighbors. (4) ld_j values are calculated and sorted by *B*. *TN* is returned to *a*. In *Case-Split*, N_a items are split between parties: (1) *a* sends a query and her ratings to both parties. *B* finds the partial similarities between its users and *a* using PSCP; and sends them to *A* through *a*. (2) *A* computes its own partial similarities. Then, *A* finds the similarities by adding values from *B*. (3) *A* selects the best N_n neighbors. It lets *B* know which neighbors are selected and the similarity signs. (4) *A* forms a neighborhood by employing random N_n and τ_n. *A* computes ld_{Aj} with this new neighborhood and lets *B* know them. *B* then calculates ld_j values by adding ld_{Aj} values from *A* to ld_{Bj} values. *B* finally sends *TN* list to *a*.

IV. Attack Scenarios

We describe the possible three attack scenarios that can be applied to VPD-based PPCF schemes; and evaluate their performance in terms of privacy. *The first aspect* of privacy preserves actual ratings and *the second aspect* protects rated items.

A. Acting as an active user in multiple scenarios

If there is a malicious party, it can try sending multiple queries to learn the other party's matrix. Consider a case, where the malicious party sends multiple queries and alters only one cell each time. The malicious party can track the changes in the output (similarity weights) and decide the rating value of the items whose value has been manipulated. Assume that the malicious party invokes an initial query and stores the similarity weights. Then, it manipulates a single rating and sends the altered query to the other party to learn the new similarity weights. After receiving similarity weights for the manipulated query, it compares them with the ones from the earlier query. If there is an increase in the similarity weight between *a* and *u*, then the manipulated value is kept by the user. The malicious party can reach such a decision because the increase in the similarity weights means a higher correlation between *a* and *u*. If there is a decrease, then it concludes that *u* keeps the value in the first query. If there is no change in the similarity value, this means that the

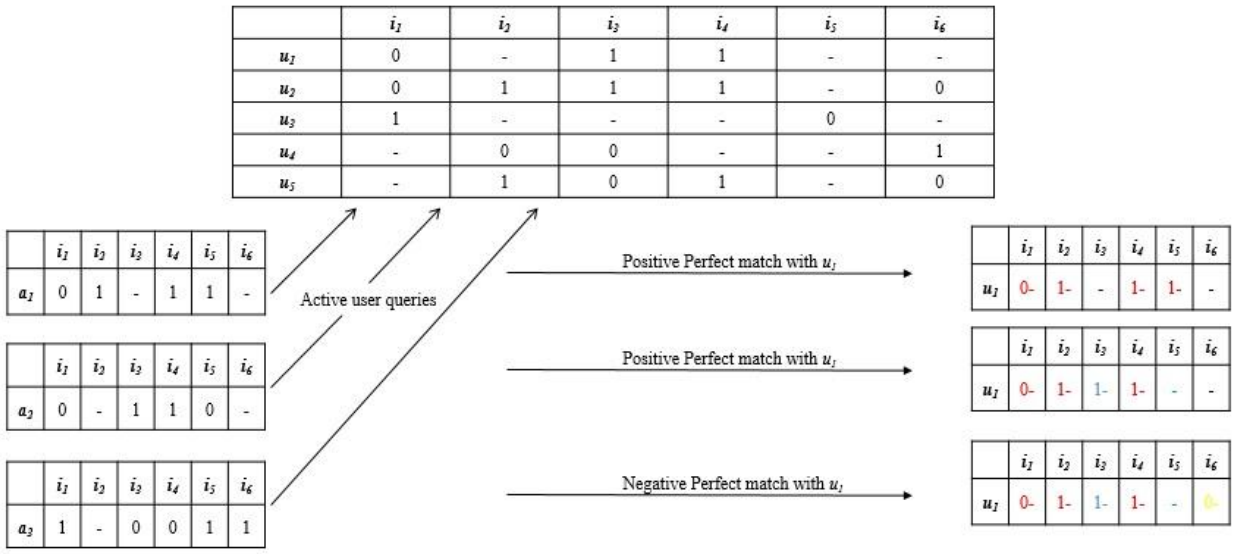

Fig. 1. Perfect match attack

manipulated item is not rated by *u*. This notion can be applied for each user so that the whole matrix can be disclosed in such a scenario. This attack is a threat for both privacy aspects. *Case-All* and *Case-Split* VPD-based PPCF schemes are subjected to this attack. Collaborating parties have to exchange the partial similarity values of individuals to calculate the final similarities for both schemes. This interaction makes it possible to perform this attack because the malicious party will have an access to the similarities for each user. Due to the PSCP, the success of this attack is affected. We examine how effective PSCP is against this attack using trials.

B. *knn-based scenarios*

This attack is proposed by Calandrino et al. [15]. It assumes that the attacker has a history of ratings of a target user and appends *k* fake users into the CF system with an exact copy of known history. When a prediction is requested for one of the fake users, it is highly probable that *k* neighbors will be selected among *k-1* fake users and the target user. Since *k-1* users are identical, the predictions are expected to come from the target user. This attack discloses the data of whether an item is rated or not, so the second privacy aspect is under threat. Since both VPD-based schemes utilize the best neighbors selected among all neighbors, *knn-based* attack can be performed. Additionally, the party intending for the attack does not need to have a history of the target user. Parties have already had a history of each user. Thus, the malicious party can use its own part of the ratings as the history of the target user and manipulate neighbors by inserting *k* fake users. We hypothesize that PSCP is not effective to prevent from this attack because randomly removing or appending ratings into *a*'s vector does not change the similarity weights between *a* and *k-1* users plus the target user.

C. *Perfect match attack*

This attack is proposed in [7] for an HPD-based binary PPCF scheme. We apply this attack for VPD-based scheme. The attack exploits the similarity value exchanged between parties. Each party calculates the partial similarity values. Although PSCP is applied by each party as a privacy measure, we believe that this scheme is subjected to this attack. Assume that *B* acts as the master party and no privacy measure is taken. *A* calculates the similarities between its user and *a*, sends them to *B*. If the similarity between any user of *A* and *a* is either 1 or -1, such similarities are called as *perfect matches* [7]. This means that the commonly rated items between these two users are either identical or opposite, respectively. Hence, *B* can conclude that the corresponding user who has a perfect match with *a* either identically voted or not voted for any of its items if the similarity is 1. If the similarity is -1, they either vote opposite or not voted for any of the items. The attack is depicted in Fig. 1.

Both VPD-based schemes calculate the similarities by two-party collaboration. One party calculates its own partial similarities and sends them to the master party for similarity calculation. Then, the similarities are calculated by adding up the incoming partial similarity values with the ones calculated off-line. The master party can use these interim similarity values received from the other part to identify perfect matches. Thus, this attack is applicable for both VPD-based schemes.

V. EXPERIMENTS

Experiments are performed using MovieLens Million (MLM) and Jester data sets. MLM contains one million ratings from 6,040 users for 3,952 items. It was collected by GroupLens (www.cs.umn.edu/research/GroupLens). Ratings are on a 5-star scale. Jester is a dense data set and ratings are continuous scale between -10 and 10

TABLE I
EMPIRICAL OUTCOMES FOR THE THREE ATTACKS

		NP				***WP***			
		Case-All		***Case-Split***		***Case-All***		***Case-Split***	
		MLM	***Jester***	***MLM***	***Jester***	***MLM***	***Jester***	***MLM***	***Jester***
Acting as an active user attack	***Precision***	1.000	1.000	1.000	1.000	0.028	0.592	0.030	0.760
	Recall	0.293	0.288	0.603	0.760	0.205	0.191	0.380	0.690
knn-based attack	***Precision***	0.178	0.828	0.136	0.808	0.204	0.807	0.085	0.801
	Recall	0.374	0.716	0.368	0.706	0.245	0.537	0.259	0.703
Perfect match attack	$Precision_1$	0.037	0.913	0.034	0.894	0.010	0.761	0.010	0.648
	$Recall_1$	0.173	0.443	0.189	0.441	0.017	0.042	0.017	0.032
	$Precision_2$	0.976	0.157	0.978	0.193	0.992	0.251	0.992	0.373
	$Recall_2$	0.331	0.514	0.344	0.537	0.098	0.077	0.086	0.111

(http://eigentaste.berkeley.edu/dataset/). *Precision* and *recall* are used as evaluation metrics. *Precision* is the ratio of how many retrieved items are related. *Recall* is the ratio of how many relevant items are selected. MLM ratings greater than or equal to 3 are converted to 1 (like) and the rest converted to 0 (dislike). Jester ratings greater than 2 are rated 1 and the rest is rated 0. Experiments are repeated 10 times. 500 and 2,000 users are picked randomly from MLM and Jester, respectively. Jester users are picked among the users with at least 60% rated cells for a denser data set. We display the outcomes in Table 1, where *WP* means privacy measures are taken while *NP* means no privacy measures are taken.

A. Experiment I

In this experiment, multiple active queries are sent by the malicious party to derive information. An active query is created with the random density between 10% and 50% and it is sent to the master party. PSCP can mitigate this attack. Thus, we devise the attack against VPD-based schemes with and without PSCP to measure the success of PSCP. Each query sent to the server is altered by PSCP. This means that the partial similarities returned from the other party does indeed belong to the altered active queries. As a result, it is expected that PSCP would be an effective privacy measure because each subsequent queries will be almost independent from each although they are distinct with only one item.

As seen in Table 1, *precision* results are perfect in *NP* case. Every item that this attack can recover obviously belongs to the original data. *Recall* rates denote the percentage of these recovered items to the total relevant items including the ones that are not recovered. We do not attempt to recover every item. Multiple queries whose density is randomly picked between 10% and 50% are sent to the master party. In *WP* case, it is clear that *precision* and *recall* results deteriorate due to PSCP. *Precision* rates with the sparse data set, MLM, is very low; this is mainly because observing unrated items is more difficult with PSCP. We send an active query by manipulating only one item from the original active query to recover the manipulated item. Two subsequent active queries have to produce the same similarity weight to mark the manipulated item to be recovered unrated. However, marking the manipulated item unrated is a rare possibility since PSCP alters each incoming query based on the density. MLM has many unrated items so it is highly possible that many unrated items are marked rated. *Precision* calculation is dramatically affected for MLM because the number of actually rated items is very low. Thus, making mistakes by marking unrated items as rated creates great number of false positives for a sparse data set. This possibility is still valid for the dense data set, Jester; yet unrated items are not as many as the sparse case, MLM, and false positive rate remains low. This means that the penalty is greater for sparse data sets in terms of false positive due to sparsity. *Recall* rates basically indicate that there is a decline of the percentage of recovered items in *WP* compared to *NP* due to PSCP's role on altering active queries.

B. Experiment II

This experiment performs *knn-based* attack, which requires history of a user and the parties in VPD-based schemes have such information. We set k as 100 and 100 fake users are appended into the data set for each test. This attack asks for a *TN* prediction for one fake user to pick the best k neighbors among the k-1 fake users and the target user whose history is known. There might be more than k best neighbors, then we pick the first k's. Thus, this attack could be less efficient if there are some other highly correlated users other than appended *k-1* users. Calculation of the similarity metric is an important factor here; it is computed by only considering the commonly rated items. Assume that a user with only one item rated could have the perfect similarity with the target user if the target user rated that item. Therefore, such a user could join among the best neighbors by only one cell. *knn-based* attack has such a limitation for VPD-based schemes due to the similarity metric. However, PSCP might not introduce significant privacy for the same reason. Appending or removing random entries into/from the query will have no effect on the similarity calculations with k-1 fake users. Since there is no change in the fake users vector, altering the active query has no effect on the commonly rated entries. We do not expect a clear trend in *WP* case in both *precision* and *recall*

rates. Consequently, PSCP is not expected to introduce a level of privacy.

There is no clear trend in both *precision* and *recall* rates as seen in Table 1 for this attack because the similarity calculation is determined by only commonly rated cells. All *k*-1 fake users and the target users in *NP* case are still among the best neighbors after PSCP (*WP*). There might be some users joining into or staying out of the best neighbors if PSCP removes some items from the active query. A user in the best neighbors can be kept out of the best neighbors if there remain no commonly rated items after PSCP. A new user might join into the best neighbors if all commonly rated items match after some items are removed from the active query by PSCP. Note that *k*-1 fake users and the target user similarity are not affected because their commonly rated items remain same. However, the other users among the best users might have some opposite rating for the appended items in the active query by PSCP. Thus, those users are kept out. Since *k*-1 fake users and the target user are not kept out of the best neighbors due to PSCP, there is no clear trend in terms of *precision* and *recall*. The success of the attack can vary based on how the nominated users for the best *k* neighbors change. It can be concluded that PSCP does not offer a level of privacy as hypothesized.

C. Experiment III

Perfect match attack is fulfilled here. Its success relies on finding perfect matches between the users and the query. Thus, we empirically test how much density would be better to capture more perfect matches. We perform tests with active queries of 5%, 10%, 20%, and 50% density rates without employing privacy measures. 5% is chosen as the best density rate for this attack. This attack reveals two privacy breaches. We evaluate different *precision* and *recall* metrics for these two breaches: (1) either the actual value of the relevant item or it is unrated and (2) unrated entries. The first $precision_1$ and $recall_1$ measure the first breach. Since there are two possibilities in this case, we perform a coin toss to decide. $Precision_2$ and $recall_2$ measure the success of the second breach. It gives statistics about the items which are successfully marked unrated.

For the first breach, a clear effect of PSCP can be seen in Table 1. When PSCP is applied, we see certain decrease in all cases. Although PSCP helps privacy compared to the *NP* case, $precision_1$ for Jester in *WP* is not negligible. $Recall_1$ shows that the amount of data disclosed can be considered very low in *WP*. Jester reveals more privacy breaches than MLM in *NP* and *WP* cases. Sparse data possibly captures perfect matches on small number of commonly rated items and the other rated items in the active query are either marked 1's, 0's or unrated in the user vector. This process deteriorates results. In terms of the second breach, results are better for MLM due to sparsity. However, *WP* case reveals significant amount of data for MLM. $Recall_2$ are too low for both sets while $precision_2$ are significant for MLM.

VI. Conclusion

We study three attack techniques that can be applied to VPD-based binary PPCF schemes proposed by Polat and Du [3]. *Acting as an active user attack* in multiple scenarios is very effective without privacy measures; however, PSCP provides privacy in terms of *precision* for sparse data set. *knn-based* attack performs similar results when privacy measures are taken and not taken. Results for *perfect match* attack displays that it is effective to exploit the first and second privacy breaches for denser and sparse data sets, respectively. PSCP has been originally designed for *acting as an active user in multiple scenarios attack* and it is successful to mitigate its effect. However, PSCP is not very useful for *knn-based* attack due to similarity weighting calculation.

As a future goal, we would like to consider if we can improve privacy provided by these schemes against three attacks. Multi-party PPCF schemes will be examined in terms of privacy breaches.

References

[1] A. Bilge, C. Kaleli, I. Yakut, I. Gunes, and H. Polat, "A Survey of Privacy-Preserving Collaborative Filtering Schemes," *Int. J. Softw. Eng. Knowl. Eng.*, Vol. 23, No. 08, pp. 1085-1108, 2013.

[2] Polat and W. Du, "Privacy-Preserving Collaborative Filtering on Vertically Partitioned Data" *Lect. Notes Comput. Sci.*, Vol. 3721, pp. 651-658, 2005.

[3] H. Polat and W. Du, "Privacy-Preserving Top-*N* Recommendation on Distributed Data," *J. Am. Soc. Inf. Sci. Technol.*, Vol. 59, No. 7, pp. 1093-1108, 2008.

[4] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the Privacy Preserving Properties of Random Data Perturbation Techniques," in *Proceedings of the 3rd IEEE International Conference on Data Mining*, Melbourne, FL, USA, 99-106, 2003.

[5] S. Zhang, J. Ford, and F. Makedon, "Deriving Private Information from Randomly Perturbed Ratings," in *Proceedings of the 6th SIAM International Conference on Data Mining*, Bethesda, MD, USA, 59-69, 2006.

[6] M. Okkalioglu, M. Koc, and H. Polat, "On the Discovery of Fake Binary Ratings," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, Salamanca, Spain, 901-907, 2015.

[7] M. Okkalioglu, M. Koc, and H. Polat, "On the Privacy of Horizontally Partitioned Binary Data-based Privacy-Preserving Collaborative Filtering" in *Proceedings of the 10th DPM International Workshop on Data Privacy Management*, Vienna, Austria, 2015.

[8] H. Polat and W. Du, "Achieving Private Recommendations Using Randomized Response Techniques," *Lect. Notes Comput. Sci.*, Vol. 3918, pp. 637-646, 2006.

[9] C. Kaleli and H. Polat, "Providing Private Recommendations Using Naïve Bayesian Classifier," *Advances in Soft Computing*, Vol. 43, pp. 168-173, 2007.

[10] C. Kaleli and H. Polat, "Privacy-Preserving Naïve Bayesian Classifier-based Recommendations on Distributed Data," *Comput. Intell.*, Vol. 31, No. 1, pp. 47-68, 2015.

[11] C. Kaleli and H. Polat, "Privacy-Preserving SOM-based Recommendations on Horizontally Distributed Data," *Knowledge-Based Syst.*, Vol. 33, pp. 124–135, 2012.

[12] C. Kaleli and H. Polat, "SOM-based Recommendations with Privacy on Multi-party Vertically Distributed Data," *Journal of the Operational Research Society*, Vol. 63, No. 6, pp. 826-838, 2012.

[13] S. Guo, X. Wu, and Y. Li, "Determining Error Bounds for Spectral Filtering based Reconstruction Methods in Privacy Preserving Data Mining," *Knowl. Inf. Syst.*, Vol. 17, No. 2, pp. 217-240, 2008.

[14] Z. Huang, W. Du, and B. Chen, "Deriving Private Information from Randomized Data," in *Proceedings of the 24th ACM SIGMOD International Conference on Management of Data*, 37-48, 2005.

[15] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov, "``You Might Also Like:" Privacy Risks of Collaborative Filtering," in *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, USA, 231-246, 2011.