On Addressing Efficiency Concerns in Privacy-Preserving Mining

Shipra Agrawal[1], Vijay Krishnan[2], and Jayant R. Haritsa[1]

[1] Dept. of Computer Science & Automation
Indian Institute of Science, Bangalore 560012, INDIA
{shipra,haritsa}@csa.iisc.ernet.in
[2] Dept. of Computer Science & Engineering
Indian Institute of Technology, Mumbai 400076,INDIA
vijay@cse.iitb.ac.in

Abstract. Data mining services require accurate input data for their results to be meaningful, but privacy concerns may influence users to provide spurious information. To encourage users to provide correct inputs, we recently proposed a data distortion scheme for association rule mining that simultaneously provides both privacy to the user and accuracy in the mining results. However, mining the distorted database can be orders of magnitude more time-consuming as compared to mining the original database. In this paper, we address this issue and demonstrate that by (a) generalizing the distortion process to perform symbol-specific distortion, (b) appropriately chooosing the distortion parameters, and (c) applying a variety of optimizations in the reconstruction process, runtime efficiencies that are well within an order of magnitude of undistorted mining can be achieved.

Keywords: Privacy, Data Mining, Efficiency

1 Introduction

The knowledge models produced through data mining techniques are only as good as the accuracy of their input data. One source of data inaccuracy is when users deliberately provide wrong information. This is especially common with regard to customers who are asked to provide personal information on Web forms to e-commerce service providers. The compulsion for doing so may be the (perhaps well-founded) worry that the requested information may be misused by the service provider to harass the customer. As a case in point, consider a pharmaceutical company that asks clients to disclose the diseases they have suffered from in order to investigate the correlations in their occurrences – for example, "Adult females with malarial infections are also prone to contract tuberculosis". While the company may be acquiring the data solely for genuine data mining purposes that would eventually reflect itself in better service to the client, at the same time the client might worry that if her medical records are either inadvertently or deliberately disclosed, it may adversely affect her employment opportunities.

Recently, in [11], we investigated whether customers can be encouraged to provide correct information by ensuring that the mining process cannot, with any reasonable

degree of certainty, violate their privacy, but at the same time produce sufficiently accurate mining results. The difficulty in achieving these goals is that privacy and accuracy are typically contradictory in nature, with the consequence that improving one usually incurs a cost in the other [2].

Our study was carried out in the context of extracting *association rules* from large historical databases, a popular mining process [3] that identifies interesting correlations between database attributes, such as the one described in the pharmaceutical example. For this framework, we presented a scheme called **MASK** (Mining Associations with Secrecy Konstraints), based on a simple probabilistic distortion of user data, employing random numbers generated from a pre-defined distribution function. It is this distorted information that is eventually supplied to the data miner, along with a description of the distortion procedure. A special feature of MASK is that the distortion process can be implemented at the data source itself, that is, at the *user machine*. This increases the confidence of the user in providing accurate information since she does not have to trust a third-party to distort the data before it is acquired by the service provider.

Experimental evaluation of MASK on a variety of synthetic and real datasets showed that, by appropriate setting of the distortion parameters, it was possible to simultaneously achieve a high degree of privacy and retain a high degree of accuracy in the mining results. While these results were very encouraging, a problem left unaddressed was characterizing the *runtime efficiency* of mining the distorted data as compared to directly mining the original data. That is, the question "Is privacy-preserving mining of association rules more expensive than direct mining?" was not considered in detail. Our subsequent analysis has shown that this issue is indeed a *major concern*: Specifically, we have found that on typical market-basket databases, privacy-preserving mining can take as much as *two to three orders of magnitude* more time as compared to direct mining. Such enormous overheads raise serious questions about the viability of supporting privacy concerns in data mining environments.

In this paper, we address the runtime efficiency issue in privacy-preserving association rule mining, which to the best of our knowledge has never been previously considered in the literature. We demonstrate that it is possible to bring the efficiency to *well within an order of magnitude* with respect to direct mining, while retaining satisfactory privacy and accuracy levels. This improvement is achieved through changes in both the distortion process and the mining process of MASK, resulting in a new algorithm that we refer to as **EMASK** (Efficient MASK). Our new design is validated against a variety of synthetic and real datasets.

2 Background Information

In this section, we present background information about the privacy metrics used in association rule mining, and the MASK privacy-preserving mining algorithm [11].

2.1 Privacy Metric

Since the mechanism for achieving privacy is to *distort* the user data before it is subject to the mining process, we measure privacy with regard to the probability with which

distorted entries can be *reconstructed*. In short, our privacy metric is: "With what probability can a random 1 or 0 in the true matrix be reconstructed"? For the sake of presentation simplicity, we will assume in the sequel that the user is only interested in privacy for 1's (this appears reasonable to expect in a market-basket environment, since the 1's denote specific actions, whereas the 0's are default options). The generalization to providing privacy for both 1's and 0's is straightforward – the details are given in [1].

Privacy can be computed at two levels: *Basic Privacy (BP)* and *Re-interrogated Privacy (RP)*. In basic privacy, we assume that the miner does not have access to the distorted data matrix after the completion of the mining process. Whereas, in re-interrogated privacy, the miner can use the mining output (i.e. the association rules) to subsequently *re-interrogate* the distorted database, possibly resulting in reduced privacy.

2.2 The MASK Algorithm

Given a customer tuple with 1's and 0's, the MASK algorithm has a simple distortion process: Each item value (i.e. 1 or 0) is either kept the same with probability p or is flipped with probability $1 - p$. All the customer tuples are distorted in this fashion and make up the database supplied to the miner – in effect, the miner receives a *probabilistic function* of the true customer database. For this distortion process, the Basic Privacy for 1's was computed to be

$$\mathcal{P}_1(p) = 1 - \frac{s_0 \times p^2}{s_0 \times p + (1-s_0) \times (1-p)} - \frac{s_0 \times (1-p)^2}{s_0 \times (1-p) + (1-s_0) \times p} \quad (1)$$

where s_0 is the average support of individual items in the database and p is the distortion parameter mentioned above.

Since the privacy graph as a function of p has a large range where it is almost constant, it means that we have considerable flexibility in choosing the p value – in particular, we can choose it in a manner that will *minimize the error* in the subsequent mining process. Specifically, the experiments in [11] showed that choosing $p = 0.9$ (or, equivalently, $p = 0.1$, since the graph is symmetric about $p = 0.5$) was most conducive to accuracy.

The concept of Re-interrogated Privacy was not considered in [11]; we include it in this paper and compute both Basic Privacy and Re-interrogated Privacy for EMASK (see Section 3.2 for details).

Run time efficiency of MASK While MASK is successful in achieving the dual objectives of privacy and accuracy, its runtime efficiency proves to be rather poor. For example, Figure 1 shows the running time (on log scale) of MASK, implemented as an extension to Apriori[4], as compared to Apriori[4] itself, for various settings of the minimum support parameter. The database used in this experiment (T10.I4.D1M.N1K as per naming convention of [4]) was created using the IBM generator [4]. The graph in Figure 1 show that there are huge differences in the running times of the two algorithms – specifically, mining the distorted database can take as much as *two to three orders of magnitude* more time than mining the original database. Obviously, such enormous overheads make it difficult for MASK to be viable in practice.

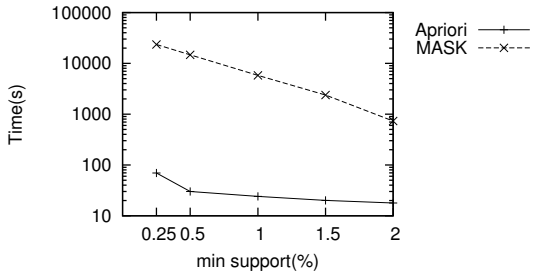

Fig. 1. Comparison of run time of Apriori and MASK (log scale)

The reasons for the huge overheads are the following:

Increased database density: This overhead is inherent to the methods employing random distortion method to achieve privacy. The random perturbation methods flip 0's to 1's to hide the original 1's. Due to the generation of false 1's, the *density* of the database matrix is increased considerably. For example, given a supermarket database with an average transaction length of 10 over a 1000 item inventory, a flipping probability as low as 0.1 increases the average transaction length by an order of magnitude, i.e. from 10 to 108. The reason that flipping of true 1's to false 0's cannot compensate for this increase is that the datasets we are considering here are sparse databases, with the number of 0's orders of magnitude larger than the number of 1's. Hence the effect of flipping 0's highly dominates the effect of flipping 1's.

As a result of increased transaction lengths, counting the itemsets in the distorted database requires much more processing as compared to the original database, substantially increasing the mining time. In [9], a technique for compressing large transactions is proposed – however, its utility is primarily in reducing storage and communication cost, and not in reducing the mining runtime since the compressed transactions have to be decompressed during the mining process.

Counting Overhead: In the MASK distortion scheme, a k-itemset may be distorted to produce any of 2^k combinations. For example, a '11' may distort to '00', '01', '10' or '11'. In order to accurately reconstruct the support of the k-itemset via the reconstruction procedure as explained in [11], we need the counts of all these 2^k combinations in the distorted database. Thus in principle we need to keep track of 2^k counts for each k-itemset. To reduce these counting overheads, MASK took the approach of maintaining *equal* flipping probabilities for both 1's and 0's – with this assumption, the number of counters required is only k [11]. Further, only $k-1$ of the counts need to be explicitly maintained since the sum of the counts is equal to the database cardinality, $dbsize$ – the counter chosen to be implicitly counted was the one with the expected largest count. While these counting optimizations do appreciably reduce runtime costs, yet the overhead in absolute terms remains very significant.

3 The EMASK Algorithm

In this section, we describe how EMASK, the algorithm that we propose in this paper, is engineered to address the above-mentioned efficiency concerns with the basic MASK technique.

3.1 The Distortion Process

The new feature of EMASK's distortion process is that it applies *symbol-specific* distortion – that is, 1's are flipped with probability $(1 - p)$, while 0's are flipped with probability $(1 - q)$. Note that in this framework the original MASK can be viewed as a special case of EMASK wherein $p = q$.

The idea here is that MASK generates false items to hide the true items that are retained after distortion, resulting in an increase in database density. But, if a fewer number of true items are retained, a fewer number of false items need to be generated, and we can minimize this density increase. Thus the goal of reduced density could be achieved by reducing the value of p and increasing the value of q (specifically increasing it to beyond 0.9). However, note that a decrease in p value increases the distortion significantly which can reduce accuracy of reconstruction. Also, q or the non-flipping probability of 0's cannot be increased to very high values as it can decrease the privacy significantly. Thus, the choices of p and q have to be made *carefully* to obtain a combination of p and q values, such that q is high enough to result in decreased transaction lengths, but privacy and accuracy are still achievable.

We have developed estimators of privacy and accuracy using which we can *analytically* select the appropriate p and q values for the distortion and reconstruction process. Due to space limitations, we do not include these details here, but refer the reader to [1].

3.2 Privacy Estimation for EMASK

As mentioned earlier, privacy can be computed at two levels: Basic Privacy (BP) and Reinterrogated Privacy (RP). We now quantify the privacy provided by EMASK with regard to both these metrics.

Basic Privacy The basic privacy measures the probability that given a random customer C who bought an item i, her original entry for i, i.e. '1', can be accurately reconstructed from the distorted entry, *prior* to the mining process. We calculate this basic privacy on the lines of calculations of privacy in [11] to get the following expression for privacy of a '1' (the derivation details are available in [1]:

$$P_1(p, q) = 1 - \frac{p^2 s_0}{s_0 p + (1 - s_0)(1 - q)} - \frac{(1-p)^2 s_0}{s_0 (1-p) + (1 - s_0) q} \tag{2}$$

Reinterrogation privacy Reinterrogation privacy takes into account the reduction in privacy due to the knowledge of the *output* of the mining process – namely, the association rules, or equivalently, the support of frequent itemsets [4]. Privacy breach due to reinterrogation stems from the fact that an individual entry in the distorted database may not reveal enough information to reconstruct it, but on seeing a long frequent itemset in the distorted database and knowing the distorted and original(reconstructed) supports of the itemset, the miner may be able to predict the presence of an item of the itemset with more certainty. As an example situation, suppose the reconstructed support of a 3-itemset present in a transaction distorted with $p = 0.9, q = 0.9$, is 0.01. Then the probabiity of this 3-itemset coming from a '000' in the original transaction is as low as $0.1 * 0.1 * 0.1 * 0.99 = 0.00099$. Thus, with the knowledge of support of higher length itemsets the miner can predict the presence of an item of the itemset in the original transaction with higher probability.

Due to space limitations we skip the details of the procedure to calculate the reinterrogation privacy, and refer the reader to [1].

3.3 The EMASK Mining Process

Having established the privacy obtained by EMASK's distortion process, we now move on to presenting EMASK's technique for estimating the true supports of itemsets from the distorted database. In the following discussion, we first show how to estimate the supports of 1-itemsets (i.e. singletons) and then present the general n-itemset support estimation procedure. In this derivation, it is important to keep in mind that the miner is provided with both the distorted matrix as well as the distortion procedure, that is, he *knows* the values of p and q that were used in distorting the true matrix.

Estimating Singleton supports We denote the original true matrix by T and the distorted matrix, obtained with distortion parameters p and q, as D. Now consider a random individual item $\mathbf{i}$. Let c_1^T and c_0^T represent the number of 1's and 0's, respectively, in the $\mathbf{i}$ column of T, while c_1^D and c_0^D represent the number of 1's and 0's, respectively, in the $\mathbf{i}$ column of D. With this notation, we estimate the support of $\mathbf{i}$ in T using the following equation:

$$\mathbf{C^T = M^{-1} C^D} \tag{3}$$

where

$$M = \begin{bmatrix} p & 1-q \\ 1-p & q \end{bmatrix} \; C^D = \begin{bmatrix} c_1^D \\ c_0^D \end{bmatrix} \; C^T = \begin{bmatrix} c_1^T \\ c_0^T \end{bmatrix}$$

Estimating n-itemset Supports It is easy to extend Equation 3, which is applicable to individual items, to compute the support for an arbitrary n-itemset. For this general case, we define the matrices as:

$$C^D = \begin{bmatrix} c^D_{2^n-1} \\ . \\ . \\ . \\ c^D_1 \\ c^D_0 \end{bmatrix} \quad C^T = \begin{bmatrix} c^T_{2^n-1} \\ . \\ . \\ . \\ c^T_1 \\ c^T_0 \end{bmatrix}$$

Here c^T_k should be interpreted as the count of the tuples in T that have the binary form of k (in n digits) for the given itemset (that is, for a 2-itemset, c^T_2 refers to the count of 10's in the columns of T corresponding to that itemset, c^T_3 to the count of 11's, and so on). Similarly, c^D_k is defined for the distorted matrix D.

The column matrices can be simplified by observing that the distortion of an entry in the above distortion procedure depends only on whether the entry is 0 or 1, and *not* on the column to which the entry belongs, rendering distortion of all combinations with same number of 1's and 0's equivalent. Hence the above matrices can be represented as:

$$C^D = \begin{bmatrix} c^D_n \\ . \\ . \\ . \\ c^D_1 \\ c^D_0 \end{bmatrix} \quad C^T = \begin{bmatrix} c^T_n \\ . \\ . \\ . \\ c^T_1 \\ c^T_0 \end{bmatrix}$$

where c^T_k should be interpreted as the count of the tuples in T that have a binary form with k 1's (out of n entries) for the given itemset. For example , for a 2-itemset, c^T_2 refers to the count of 11's in the columns of T corresponding to that itemset, c^T_1 to the count of 10's and 01's, and c^T_0 to the count of 00's. Similarly, c^D_k is defined for the distorted matrix D.

Each entry $m_{i,j}$ in the matrix $\mathbf{M}$ is the probability that a tuple of the form corresponding to c^T_j in T goes to a tuple of the form corresponding to c^D_i in D. For example, $m_{2,1}$ for a 2-itemset is the probability that a 10 or 01 tuple distorts to a 11 tuple. Accordingly, $m_{2,1} = p(1-q)$. The basis for this formulation lies in the fact that in our distortion procedure, the component columns of an n-itemset are distorted *independently*. Therefore, we can use the product of the probability terms. In general,

$$m_{i,j} = \sum_{k=max(0,i+j-n)}^{min(i,j)} {}^jC_k \; p^k(1-p)^{(j-k)} \quad {}^{n-j}C_{i-k} \; q^{(n+k-i-j)}(1-q)^{(i-k)} \quad (4)$$

3.4 Eliminating Counting Overhead

We now present a simple but powerful optimization by which the extra overhead of counting all the combinations generated by the distortion can be *eliminated completely*. This optimization is based on the following basic formula from set theory: Given a universe U, and subsets A and B, $N(A' \cap B) = N(B) - N(A \cap B)$, where N is the

number of elements, i.e. cardinality, of the set denoted by the bracketed expression. This formula can be generalized[3] to

$$N(A'_1 A'_2 ... A'_m B_1 B_2 ... B_n)$$
$$= N(B_1 B_2 .. B_n) + \sum_{k=1}^{m} \sum_{\{x_1,...,x_k\} \subset \{1,..,m\}} (-1)^k N(A_{x_1} A_{x_2} ... A_{x_k} B_1 B_2 .. B_n)$$

Using this formula the counts of all the combinations generated from an n-itemset can be calculated from the counts of the itemset and the counts of its subsets which are available from previous passes over the distorted database. *Hence, we need to explicitly count only the '111...1's, just as in the direct mining case*.

For example, during the second pass we need to explicitly count only '11's which makes $N(A \cap B)$ available at the end of the second pass. The counts of the remaining combinations, '10', '01' and '00' can then be calculated using the following set of formulae:

$$10 : N(A \cap B') = N(A) - N(A \cap B)$$
$$01 : N(A' \cap B) = N(B) - N(A \cap B)$$
$$00 : N(A' \cap B') = N(U) - N(A) - N(B) + N(A \cap B)$$

The above implies that the extra calculations for reconstruction are performed only at *the end of each pass*, the rest of the pass being exactly the same as that of the original mining algorithm. Further, the only additional requirement of the above approach as compared to traditional data mining algorithms such as *Apriori* is that we need to have available, at the end of each pass, the counts of all frequent itemsets generated during the previous passes. However, this requirement is easily met by keeping a hash table in memory of these previously identified frequent itemsets.

4 Performance Framework

EMASK aims at simultaneously achieving satisfactory performance on three objectives: privacy, accuracy and efficiency. While privacy is determined as outlined in Section 3.2, the specific metrics used to evaluate EMASK's performance w.r.t. accuracy and efficiency are given below.

4.1 Accuracy Metrics

We evaluate two kinds of mining errors, Support Error and Identity Error, in our experiments:

Support Error (ρ) :

This metric reflects the (percentage) average relative error in the reconstructed support values for those itemsets that are correctly identified to be frequent. Denoting the number of frequent itemsets by $|f|$, the reconstructed support by rec_sup and

[3] $N(B_1 B_2 .. B_n)$ is replaced by $N(U)$ if $n = 0$

the actual support by act_sup, the support error is computed over all frequent itemsets as

$$\rho = \frac{1}{\mid f \mid} \Sigma_f \frac{\mid rec\_sup_f - act\_sup_f \mid}{act\_sup_f} * 100$$

Identity Error (σ) :

This metric reflects the percentage error in identifying frequent itemsets and has two components: σ^+, indicating the percentage of false positives, and σ^- indicating the percentage of false negatives. Denoting the reconstructed set of frequent itemsets with R and the correct set of frequent itemsets with F, these metrics are computed as:

$$\sigma^+ = \frac{|R-F|}{|F|} * 100 \qquad \sigma^- = \frac{|F-R|}{|F|} * 100$$

4.2 Efficiency Metric

This metric determines the runtime overheads resulting from mining the distorted database as compared to the time taken to mine the original database. This is simply measured as the inverse ratio of the running times between Apriori[4] on the original database and executing the same code augmented with EMASK (i.e. EMASK-Apriori) on the distorted database. Denoting this slowdown ratio as Δ, we have

$$\Delta = \frac{RunTime\ of\ EMASKApriori}{RunTime\ of\ Apriori}$$

For ease of presentation, we hereafter refer to this augmented algorithm simply as EMASK.

4.3 Data Sets

We carried out experiments on a variety of synthetic and real datasets. Due to space limitations, we report the results for only two representative databases here:

1. A synthetic database generated from the IBM Almaden generator [4]. The synthetic database was created with parameters T10.I4.D1M.N1K (as per the naming convention of [4]), resulting in a million customer tuples with each customer purchasing about ten items on average.
2. A real dataset, BMS-WebView-1 [15], placed in the public domain by Blue Martini Software. This database contains click-stream data from the web site of a (now defunct) legwear and legcare retailer. There are about 60,000 tuples with close to 500 items in the schema. In order to ensure that our results were applicable to large disk-resident databases, we scaled this database by a factor of ten, resulting in approximately 0.6 million tuples.

5 Experimental Results

We evaluated the privacy, accuracy and efficiency of the EMASK privacy-preserving mining process on the two datasets for a variety of minimum support values. Due to

space limitations, we present here the results only for 0.3% sup_{min} value, which represents a support low enough for a large number of frequent itemsets to be produced, thereby stressing the performance of the EMASK algorithm. The results are shown for the p and q values recommended by the analytical selection method presented in [1].

For the above framework, the performance of EMASK on the synthetic and real datasets is summarized in Table 1 for the privacy, accuracy and efficiency metrics. In this table, *BP* and *RP* denote the basic and re-interrogated privacies, respectively; Δ denotes the slowdown of EMASK; and the remaining columns show the accuracy metrics.

DataSet	**Distortion Parameters**		**Privacy**		**Accuracy**			**Efficiency**
	p	q	BP	RP	σ^+	σ^-	ρ	Δ
Synthetic	0.5	0.97	92.6	74	5.64	6.27	4.86	3.8
Real	0.5	0.98	94.3	79.5	4.36	4.82	4.35	2.4

Table 1. Performance Results

The results in Table 1 are very encouraging since they clearly demonstrate that, with a proper selection of distortion parameters, EMASK is able to get high values for *all three competing objectives*. Specifically, for the synthetic data set, the basic and re-interrogated privacies are above 74 percent, the accuracy on all measures is better than 90 percent (less than 10 percent error), and, the slowdown is only 3.8 (that is, EMASK performs only around four times as slow as Apriori). For the real data set, the performance numbers are even better with privacies above 79 percent, accuracy levels above 95 percent, and slowdown of 2.4.

Most pertinently with respect to the concerns of this paper, note that the Δ slowdown value is indeed a *huge improvement* from the several orders of magnitude inefficiency that was exhibited by MASK. To emphasize the efficiency gain, we plot the run

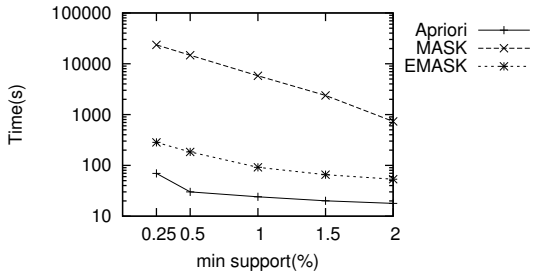

Fig. 2. Comparison of run time of Apriori, EMASK and MASK (log scale)

time of EMASK for the same environment as that used in Fig. 1 in Section 2.2, where we had compared the performance of MASK with Apriori. Figure 2 shows this resulting comparison of MASK, EMASK and Apriori, and we see here that while the MASK is two to three orders of magnitude worse than Apriori, EMASK is able to achieve run times well within an order of magnitude of Apriori.

Overall, our experiments indicate that by a careful choice of distortion parameter settings, it is possible to simultaneously achieve satisfactory privacy, accuracy, *and efficiency*. In particular, they show that there is a "window of opportunity" where these triple goals can be all met. The size and position of the window is primarily a function of the database density and could be quite accurately characterized with our estimation methods.

6 Related Work

The issue of maintaining privacy in association rule mining has attracted considerable attention in the recent past [6–10, 12–14]. However, to the best of our knowledge, none of these previous papers have tackled the issue of efficiency in privacy-preserving mining.

The work closest to our approach is that of [8, 9]. A set of randomization operators for maintaining data privacy were presented and analyzed in [8]. New formulations of privacy breaches and a methodology for limiting them were given in [9]. The problem of large transactions resulting from distortion was also mentioned in [9], but they addressed this problem from the perspective of reducing storage and communication costs, but not runtime mining efficiency. Specifically, they proposed a compression technique for reducing the effective size of the distorted database.

Another difference is that EMASK (and MASK) support a notion of "average privacy", that is, they compute the probability of being able to accurately reconstruct a *random* entry in the database. In contrast, [8, 9] evaluate the probability of accurately reconstructing a *specific* entry in the database. In [1], we present a detailed quantitative argument for why it appears fundamentally unlikely that efficient mining algorithms can be designed to support this stronger measure of privacy.

Finally, the problem addressed in [6, 7, 12, 13] is how to prevent *sensitive rules* from being inferred by the data miner – this work is complementary to ours since it addresses concerns about *output* privacy, whereas our focus is on the privacy of the *input* data. Maintaining input data privacy is considered in [10, 14] in the context of databases that are *distributed* across a number of sites with each site only willing to share data mining results, but not the source data.

7 Conclusions

In this paper, we have considered, for the first time, the issue of providing efficiency in privacy-preserving mining of association rules. Our goal was to investigate the possibility of simultaneously achieving high privacy, accuracy and efficiency in the mining process. We first showed how the distortion process required for ensuring privacy can have a marked negative side-effect of hugely increasing mining runtimes. Then, we presented our new EMASK algorithm that is specifically designed to minimize this side-effect through the application of symbol-specific distortion. We derived simple but effective formulas for estimating acceptable settings of the distortion parameters. We also presented a simple but powerful optimization by which all additional counting incurred by privacy preserving mining is moved to the end of each pass over the database.

Our experiments show that EMASK exploits a small window of opportunity which can simultaneously provide good privacy, accuracy and efficiency. Specifically, less than 5 times slowdown with respect to Apriori in conjunction with 70-plus privacies and 90-plus accuracies, were achieved with these settings. In summary, EMASK takes a significant step towards making privacy-preserving mining of association rules a viable enterprise.

Acknowledgements This work was supported in part by a Swarnajayanti Fellowship from the Dept. of Science & Technology, Govt. of India.

References

1. S. Agrawal, V. Krishnan and J. Haritsa, "Providing Efficiency in Privacy-Preserving Mining", *Tech. Rep. TR-2003-03, DSL/SERC, Indian Institute of Science*, 2003. `http://dsl.serc.iisc.ernet.in/pub/TR/TR-2003-03.pdf`
2. D. Agrawal and C. Aggarwal, "On the Design and Quantification of Privacy Preserving Data Mining Algorithms", *Proc. of 20th ACM Symp. on Principles of Database Systems*, May 2001.
3. R. Agrawal, T. Imielinski and A. Swami, "Mining association rules between sets of items in large databases", *Proc. of ACM SIGMOD Intl. Conf. on Management of Data*, May 1993.
4. R. Agrawal and R. Srikant, "Fast algorithms for mining association rules", *Proc. of 20th Intl. Conf. on Very Large Data Bases*, September 1994.
5. R. Agrawal and R. Srikant, "Privacy-Preserving Data Mining", *Proc. of ACM SIGMOD Intl. Conf. on Management of Data*, May 2000.
6. M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim and V. Verykios, "Disclosure Limitation of Sensitive Rules", *Proc. of IEEE Knowledge and Data Engineering Exchange Workshop*, November 1999.
7. E. Dasseni, V. Verykios, A. Elmagarmid and E. Bertino, "Hiding Association Rules by Using Confidence and Support", *Proc. of 4th Intl. Information Hiding Workshop*, April 2001.
8. A. Evfimievski, R. Srikant, R. Agrawal and J. Gehrke, "Privacy Preserving Mining of Association Rules", *Proc. of 8th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, July 2002.
9. A. Evfimievski, J. Gehrke and R. Srikant, "Limiting Privacy Breaches in Privacy Preserving Data Mining", *Proc. of ACM Symp. on Principles of Database Systems*, June 2003.
10. M. Kantarcioglu and C. Clifton, "Privacy-preserving Distributed Mining of Association Rules on Horizontally Partitioned Data", *Proc. of ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, June 2002.
11. S. Rizvi and J. Haritsa, "Maintaining Data Privacy in Association Rule Mining", *Proc. of 28th Intl. Conf. on Very Large Databases*, August 2002.
12. Y. Saygin, V. Verykios and C. Clifton, "Using Unknowns to Prevent Discovery of Association Rules", *ACM SIGMOD Record*, vol. 30, no. 4, 2001.
13. Y. Saygin, V. Verykios and A. Elmagarmid, "Privacy Preserving Association Rule Mining", *Proc. of 12th Intl. Workshop on Research Issues in Data Engineering*, February 2002.
14. J. Vaidya and C. Clifton, "Privacy Preserving Association Rule Mining in Vertically Partitioned Data", *Proc. of 8th ACM SIKGDD Intl. Conf. on Knowledge Discovery and Data Mining*, July 2002.
15. Z. Zheng, R. Kohavi and L. Mason, "Real World Performance of Association Rule Algorithms", *Proc. of 7th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, August 2001.