Mining Frequent Patterns with Differential Privacy

Luca Bonomi
(Supervised by Prof. Li Xiong)
Department of Mathematics & Computer Science
Emory University
Atlanta, USA
lbonomi@mathcs.emory.edu

ABSTRACT

The mining of frequent patterns is a fundamental component in many data mining tasks. A considerable amount of research on this problem has led to a wide series of efficient and scalable algorithms for mining frequent patterns. However, releasing these patterns is posing concerns on the privacy of the users participating in the data. Indeed the information from the patterns can be linked with a large amount of data available from other sources creating opportunities for adversaries to break the individual privacy of the users and disclose sensitive information. In this proposal, we study the mining of frequent patterns in a privacy preserving setting. We first investigate the difference between sequential and itemset patterns, and second we extend the definition of patterns by considering the absence and presence of noise in the data. This leads us in distinguishing the patterns between exact and noisy. For exact patterns, we describe two novel mining techniques that we previously developed. The first approach has been applied in a privacy preserving record linkage setting, where our solution is used to mine frequent patterns which are employed in a secure transformation procedure to link records that are similar. The second approach improves the mining utility results using a two-phase strategy which allows to effectively mine frequent substrings as well as prefixes patterns. For noisy patterns, first we formally define the patterns according to the type of noise and second we provide a set of potential applications that require the mining of these patterns. We conclude the paper by stating the challenges in this new setting and possible future research directions.

1. INTRODUCTION

In this paper, we are interested in the privacy preserving *frequent pattern mining* problem which is the process of finding (i.e. mining) repetitive patterns from the data without compromising the individual privacy. Frequent pattern mining has been first investigated by Agrawal et al. [1]. In that setting, the frequent patterns are represented as itemsets that are mined from transactional databases with the intent of analyzing the customer purchasing habits. The analysis of these patterns by finding the associations between the different items purchased by users in their shopping basket can help retailers in develop marketing strategies. In this proposal, we focus our study on mining sequential patterns in sequential data, such as trajectories and DNA sequences. In contrast to itemsets, sequential patterns are extensively used in those applications where it is important to capture the sequentiality of the real events in the data. An example is trajectory data, where the mining of sequential pattern could be use to predict traffic congestions.

Frequent patterns have drawn the attention of the data mining community over several decades leading to the development of many algorithmic techniques, see [11] for a survey of results. Although the mining of frequent patterns has been well studied and extended in various ways, the privacy concerns rising in the community are posing new challenges on this problem. The rapid growth of digital information from different sources (e.g. web, institutions, individuals etc) has been creating tremendous opportunities for disclosing personal information. While the concerns of revealing sensitive and private information have led researches to develop a variety of privacy models [8, 14, 21], the design of effective and efficient data mining algorithms with privacy is very challenging. In this scenario, we investigate the problem of releasing frequent patterns while preserving the individual privacy of the users that contribute to the data. Differential privacy [8] has become the de facto standard for research in data privacy since it provides strong and provable guarantees of privacy. Our goal is to study frequent pattern mining problem under the *differential privacy* model.

In this setting, only few works have been proposed to mine frequent patterns [3, 16, 25]. Although these techniques have been shown effective in certain scenarios, they have the following limitations. The mining process in these strategies is based on the concept of *support* for an itemset p which is defined as the number of transactions containing p. In this way, patterns that are repeated multiple times within the same record/transaction may not be reported as frequent. This may lead to a loss of informative patterns, such as periodic patterns in time-series data. Furthermore, patterns are represented as a subset of an universe of items and this may not well represent the sequentiality of important events in reality. As a final drawback these approaches do not considered the fact that real data are usually noisy, and therefore some patterns may be lost. The goal of this proposal consists in addressing these limitations and studying the new challenges in the privacy preserving frequent pattern mining problem as follows.

- We propose a distinction between two classes of patterns: *exact* and *noisy* patterns. In the former, we assume that the patterns appear in the data in an exact way (i.e. are preserved in the data), while in the latter the patterns are corrupted by noise.
- For exact patterns, we first present the state of the art techniques for privacy preserving frequent pattern mining and de-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Articles from this volume were invited to present their results at The 39th International Conference on Very Large Data Bases, August 26th - 30th 2013, Riva del Garda, Trento, Italy.
Proceedings of the VLDB Endowment, Vol. 6, No. 12
Copyright 2013 VLDB Endowment 2150-8097/13/10... \$ 10.00.

scribe their limitations. Second, we illustrate the challenges in mining frequent sequential patterns, and we describe our two proposed algorithms to mine these patterns [4, 5].

- We propose a new definition of noisy patterns, and describe possible scenarios where the mining of these patterns is crucial. We explore the challenges in mining these patterns and state possible future directions to address this new mining problem. To the best of our knowledge, we are the first to study the mining of noisy patterns in a privacy preserving setting.

The rest of the paper is organized as follows. In Section 2, we describe the privacy model used. In Section 3, we study the problem of mining exact patterns, and we illustrate our solutions for mining sequential exact patterns in a privacy preserving way. In Section 4, we introduce the concept of noisy patterns and we present possible scenario as well as new challenges in this setting. At the end of that section, we present possible future research directions in designing algorithmic solutions for mining noisy patterns. Finally, we conclude our paper in Section 5.

2. PRIVACY MODEL

Differential privacy [8] aims to protect the disclosure of sensitive individual information when statistical data are released. The differential privacy mechanism guarantees that the computation returned by a randomized algorithm is insensitive to the change in any particular individual record in the input data.

DEFINITION 1 (DIFFERENTIAL PRIVACY [8]). *A non interactive privacy mechanism M achieves ϵ-differential privacy if for any two input sets (databases) D_A and D_B with symmetric difference of one (neighboring databases), and for any set of outcomes $S \subseteq Range(M)$, the following holds*

$$Pr[M(D_A) \in S] \leq exp(\epsilon) \times Pr[M(D_B) \in S] \quad (1)$$

ϵ is the *privacy parameter* (also known as privacy budget) which defines the privacy level of the mechanism. Higher values of ϵ lead to lower level of privacy, while smaller values pose a stronger privacy guarantee. In literature, there are two well established techniques to achieve differential privacy: *Laplace Mechanism* [9] and *Exponential Mechanism* [18]. Both these strategies are based on the concept of *global sensitivity* [9] of the function to compute.

DEFINITION 2 (GLOBAL SENSITIVITY [9]). *For any two neighboring databases D_A and D_B, the global sensitivity for any function $f : D \to \mathbb{R}^n$ is defined as:*

$$GS(f) = \max_{D_A, D_B} \|f(D_A) - f(D_B)\|_1 \quad (2)$$

The Laplace mechanism is used in our paper to construct differentially private algorithms, so we briefly discuss it below. Let f be a function, and ϵ be the privacy parameter, then by adding noise to the result $f(D)$ we obtain a differential privacy mechanism. The noise is generated from a Laplace distribution with probability density function $pdf(x|\lambda) = \frac{1}{2\lambda} e^{-|x|/\lambda}$, where the parameter λ is determined by ϵ and $GS(f)$.

3. EXACT PATTERN MINING

In this section, we state the definition for frequent pattern mining problem and challenges. We provide an overview of the existing work for mining frequent patterns with differential privacy and their limitations. Finally, we describe the problem of mining sequential patterns and our proposed solutions to address the challenges in this setting.

3.1 Problem Definition and Challenges

In this work, we start by considering patterns that appear in the data in an exact way. We call these patterns *exact patterns* since they preserve their structure in the data and they can be directly mined without requiring external information (e.g. noise level, similarity measure etc.). Mining frequent patterns is a central problem in the data mining field. This problem often occurs in the form of mining frequent *itemsets*, where a pattern p is a subset of a given set of items $\mathcal{I}$. The mining process for these patterns is based on the concept of *support*, that for an itemset p is defined as the number of transactions containing p. Therefore the frequent itemset mining problem requires to report all the itemsets whose support is greater than a given threshold.

In this proposal, we are interested in mining *sequential* patterns that are in the form of a sequence rather than a subset of items. Therefore, a pattern of length n in our case is a sequence of symbols $p = a_0 a_1 \dots a_{n-1}$ where each symbol a_i belongs to a finite alphabet Σ. Later in the paper, we will refer to these type of patterns and transactions also as strings. We denote with $|p|$ the length of the pattern p. Furthermore, we say that a pattern p *occurs* at position i in a string $x = x_0 x_1 \dots x_{m-1}$ if there exists an integer $0 \leq i \leq m - n$ such that $x_{i+j} = a_j$ for $j = 0, \dots, n-1$. For any pattern p we denote by $f_x(p)$ the number of occurrences of the pattern p in x. When a set of N strings $D = \{x^0, x^1, \dots, x^{N-1}\}$ is given as a input, we define by $F_D(p) := \sum_{i=0}^{N-1} f_{x^i}(p)$ the frequency of the pattern p in D. Using this notation, the frequent pattern mining problem that we consider is defined as follows.

PROBLEM 1 (FREQUENT PATTERN MINING). *Given a positive integer k, report the list $\mathcal{F}_k$ of the top-k most frequent patterns in the input set D.*

To protect the individual privacy, in this section we describe two mining algorithms which return a list $\mathcal{F}_k$ that satisfies the rigorous privacy model of differential privacy.

Challenges. Two major challenges arise in mining frequent exact patterns in our setting. First, classical itemsets mining algorithms consider mining patterns that are unordered. Therefore, in that setting the number of possible distinct patterns of length n is $\binom{|\mathcal{I}|}{n}$, where $|\mathcal{I}|$ the size of the universe of items. On the other hand, in our approach the patterns considered are substrings which are ordered. Hence in our model, the number of possible distinct patterns of length n is $|\Sigma|^n$ which could result in a larger output size space. Second, the global sensitivity for counting the occurrences of a patterns is very large. For instance, the deletion or insertion of a string y in the dataset D could change the count of a pattern up to $O(|y|)$ since the pattern may occur multiple times within the same string. Therefore from the differential privacy prospective, the presence of long strings in the dataset could lead to a large amount of noise injected resulting in overall poor utility performance. To overcome these challenges we describe two strategies. The first approach has been proposed in [5] and it uses a *prefix tree* structure to partition the data and estimate the frequency of the patterns using the frequency of the prefixes. The second strategy is a *two-phase* approach [4] that first generates a candidate set of frequent patterns and successively it refines their count on a transformed version of the dataset where the sensitivity of the count query can be properly reduced.

3.1.1 Existing works for frequent itemset mining

The frequent itemset problem is well studied in literature but only recently the privacy concerns have drawn the attention of the scientific community to this problem with new challenges. Starting from the work of Bhaskar et al. [3], which first proposed a differential privacy preserving solution for this problem several techniques have been developed. The PrivBasis approach proposed in [16] introduces a new mining technique based on the concept of basis set which allows to efficiently and effectively construct the set of frequent itemsets from a small set of short patterns. Recently, Zeng et al. [25] pointed out the hardness of the differentially private frequent itemsets mining problem by investigating the trade-off between privacy and utility. As a negative result the authors showed that in order to achieve certain level of utility and privacy guarantee we could incur an extremely high privacy cost due to the presence of long transactions that increase the sensitivity of the count query for the itemsets. Motivated by this observation, the authors proposed a novel technique based on the idea of carefully truncating the transactions to mitigate the effect of the noise while providing high utility results.

Limitation of existing approaches. The classical notion of frequent pattern is based on the concept of support. In this way the task of counting frequency of a pattern is relaxed into counting the number of strings in the input dataset where the pattern appears. However, in some settings, for example time-series and genomic data, the notion of support is unable to capture patterns that have repetitive occurrences within the same string. Therefore, in order to report those repetitive patterns as frequent we use the notion of occurrence rather than support. Another important aspect related to the classical notion of patterns is the fact that the itemsets do not preserve the order of the symbols. This could be a considerable drawback in scenarios where the sequence of symbols may represent interest events in reality. Furthermore from the privacy prospective, these approaches require to know the value k as a input parameter since the privacy mechanism depends on this value. This means that any change in this parameter requires additional privacy cost.

3.2 Prefix-Tree Approach

The first algorithm we describe is called PT miner, and it has recently proposed in [5]. The idea behind our PT miner is based on the fact that the frequency of a pattern p can be computed as the sum of the frequencies of those prefixes having p as a suffix. In this way the overall sensitivity is reduced to the sensitivity of counting query for prefixes which is 1. To efficiently query the dataset in input we partition the set of strings in a top-down fashion according to their prefixes. Our mining technique is reported in Algorithm 1. Starting from the root node with empty prefix which represents the entire dataset, the length of the prefixes are progressively increased at line 8 in the algorithm forming a new partition in the tree. The frequency of each prefix is obtained by issuing a counting query on the original dataset where we employ the Laplace Mechanism to guarantee differential privacy. When the prefix is not frequent (i.e. comparison against a threshold value, at line 11 of the algorithm) the partitioning process on the node stops, this reduces the computational cost of our algorithm meanwhile it minimizes the loss on the estimated count for the prefixes. To maximize the utility results of our approach we develop several techniques to properly allocate the privacy budget $\bar{\epsilon}$ which determines the noise injected for each node in the tree. Finally, the tree is traversed and the list of the top-k patterns is returned. We have shown in [5] that the entire tree satisfies differential privacy. This represents a considerable advantage with respect to the existing approaches since this structure can be also used to perform other mining tasks (e.g. mining prefixes). Furthermore, the construction process does not depends on the length of the pattern required to be mined or on the value of k. Therefore,

Algorithm 1 Private Prefix-Tree Miner

```
 1: procedure PT MINER($D, \epsilon, k$)
    Input: dataset $D$; privacy parameter $\epsilon$; $k$
    Output: $\mathcal{T}$ private Prefix-Tree

 2:     $\mathcal{T}$ formed by the $root$
 3:     Use a queue $Q$
 4:     $Q \leftarrow root$
 5:     while ($Q$ is not empty) do
 6:         $node \leftarrow Q.remove$
 7:         for (every symbol $a$ in the alphabet $\Sigma$) do
 8:             create a node $cur$ with prefix $node.prefix + a$
 9:             allocate privacy budget $\bar{\epsilon}$
10:             perturb the $count$ with $Lap(1/\bar{\epsilon})$
11:             if ($cur$ is not frequent) then
12:                 Attach $cur$ as a child of $node$ in $\mathcal{T}$
13:                 $Q \leftarrow cur$
14:             end if
15:         end for
16:     end while
17:     traverse $\mathcal{T}$ to return the list of top-$k$ patterns
18: end procedure
```

any change in these parameters does not produce additional privacy cost while the previous approaches suffer this limitation.

In the rest of the section, we describe how this mining algorithm has been applied to effectively solve the problem of privacy preserving record linkage for string records.

3.2.1 Privacy Preserving Record Linkage

Record linkage [22, 10] is the process of identifying records that refer to the same real world entity across different sources. It is extensively used in many applications, for example, in linking medical data of the same patient across different hospitals in the country or in collecting the credit history of users from several sources. However, many of these data may contain sensitive personal information that could disclose individual privacy. The objective is to allow two parties to identify records that are close to each other according to some distance functions, such that no additional information about the data records other than the result is disclosed to either party. There are several techniques to solve this problem, and they are grouped into tree major families: Secure Multiparty Computation (SMC) [17, 24], secure transformation [2, 7, 19, 20], and hybrid methods [12, 23, 13, 15]. While these techniques have been shown to be effective in several scenarios the tradeoff between privacy and utility (accuracy and computational cost) leaves open many challenges.

Frequent Grams Embedding. In the privacy preserving setting, we proposed in [5] a novel secure transformation technique that embeds the original string records into vectors, where the matching is performed by a third party (not necessary trusted). The key idea consists in using a transformation that is based on the patterns (grams) mined from the original datasets so that the structure of the data is preserved. We compared our solution with the state of the art secure transformation technique [19], and our proposed approach provides better scalability and stronger privacy while achieves comparable utility results. In the rest of this section, we briefly illustrate the key components in our approach.

In our approach, we adopt a three party model, where two parties hold their data and a third party is in charge of the matching. Our protocol proceeds as follows: each of data holder transforms the original records into vectors that are sent to the third party which performs the matching.

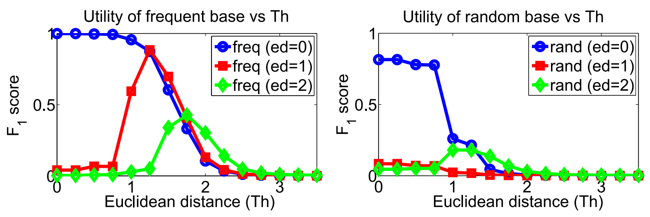

Figure 1: Utility results with different bases. (Left) Base formed with frequent variable length patterns, (Right) Base formed with random patterns.

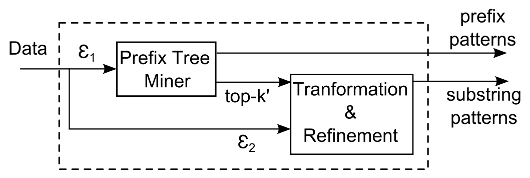

Figure 2: Algorithm overview

The transformation of the data is crucial both for the privacy, security and as well as for the utility. To perform the embedding the data holders have to agree on the information necessary for applying the map. Our intuition is that this information should be data-dependent so that the structure of the data could be well preserved in the transformation. In our approach, we use a common set of frequent patterns (grams) of variable length, called base, that are determined by the data holders before applying the transformation. This choice is motivated by the fact that a data-dependent base better represents the data with respect to a random base that at the contrary does not reflect the data changes. Figure 1 illustrates the benefits of using a base formed with frequent patterns. For different values of threshold in the embedded space, we report the F_1 score for matching records with several values of edit operations ed allowed in the matching. From these results, we can see that the use of frequent patterns as components in the base provides a considerable improvement in representing the data in the embedded space over a set of random patterns. Although the use of a base of frequent grams provides good utility results, sharing directly this information between the data holders may incur privacy leakages for individual records. In order to use the base formed by the frequent patterns without compromising the individual privacy we employ the PT miner algorithm illustrated in the previous section. Each party sets the privacy parameter ϵ, and it constructs the prefix tree. Then the tree is traversed and the list of frequent variable length patterns is reported for each party. This set of frequent patterns is shared among the two parties and a common base of k frequent patterns is computed. When the base is decided, each party embeds its own records into vectors, where each vector is a projection of the original string on the base. These vectors are then sent to the third party which computes the matching pairs by using the Euclidean distance.

3.3 A Two-Phase Approach

The second approach we describe is a two-phase algorithm that we proposed in [4]. This approach significantly improves our previous solution for mining frequent patterns while it allows to preserve the information about the frequent prefixes. In fact, the prefix tree approach may turn out to be not very effective when the frequent patterns are not well captured by the prefixes. A possible solution to this problem consists in mining the patterns in apriori fashion directly from the original dataset. Although this strategy looks appealing, it incurs a large amount of noise due to the high sensitivity of the count query for the patterns. To address this problem we use a two-phase algorithm as reported in Figure 2. In the first phase, we use an enhanced **prefix tree miner** to release an ϵ_1-differentially private prefix tree. We will use this tree to first directly mine prefix patterns, and second to generate a set of candidate patterns to use in the second phase. In this second step, we use a **transformation and refinement** procedure to first construct a sketch of the dataset and second refine the count of the candidate patterns. We use the remaining ϵ_2 privacy budget to query the transformed dataset and estimate the final counts.

Our strategy presents several advantages. First, the prefix tree in the first phase allows us to achieve high utility in mining frequent prefixes. Second, using a set of candidate patterns helps us to limit the number of possible frequent patterns to examine in the second phase. In fact, rather than considering the entire universe of patterns, we focus only on k' candidates. Third, we refine the count of the mined patterns in the top-k' by issuing counting queries on a transformed version of the dataset. In this way, we show that the sensitivity of the counting query can be reduced leading to a smaller amount of perturbation noise to be injected, thus the final top-k patterns are more accurate. In the rest of the section, we briefly describe the two phases separately.

3.3.1 Prefix Tree Miner

We use an enhanced prefix tree mining algorithm to mine the prefixes and the candidate set of frequent substrings. Our mining procedure makes use of statistical properties of the data to maximize the quality of the counts in the nodes of the tree. In fact we use the information from a statistical model to calibrate the noise injected with the Laplace Mechanism in the tree, so that we minimize the effect of the perturbation noise on the overall utility. For the tree construction procedure we proposed two algorithms. The first one uses a static sample of the data that is given in input as background knowledge. The second approach instead handles the case where this a priori information is not available, and it takes advantages of the partially constructed noisy tree to define a dynamic statistical model which is used to calibrate the noise.

3.3.2 Transformation & Refinement

In this phase, we refine the count of the candidate patterns generated in the first phase by issuing a counting query. However due to its high sensitivity we could incur a large perturbation noise if it is applied on the original dataset directly. Therefore, our idea consists in introducing a new representation of the original dataset where we can control the sensitivity of the count query and at the same time preserve the frequent patterns. This transformation process consists in representing the original strings using vectors that summarize the contribution of each string to the counts of the candidate patterns. The final counts of the patterns can be easily determinate with this representation. However to reduce the sensitivity for the count query, the strings that are too long are carefully transformed by keeping only the part that has maximum impact on the final counts. Under this representation, the counts of the candidate patterns are refined by querying the transformed dataset and the final frequent patterns are reported. For the details of this technique, we point the interested readers to [4].

Compared with the state of the art of sequential pattern mining [6], our approach provides comparable results for frequent substring patterns, while achieving significantly superior results for mining frequent prefixes at the same time.

4. NOISY PATTERN MINING

In real scenarios data is typically subject to noise and measurement errors that may affect the frequent patterns. It is possible that even a small amount of noise may compromise the set of the frequent patterns. To illustrate this phenomena we consider the following example.

EXAMPLE 1. *Let $D = \{ccabbb, bbab, cabccab\}$ be a dataset in input containing three strings. The list of the top-3 frequent exact patterns of length 2 is $F_3 = \{(ab, 4), (bb, 3), (ca, 3)\}$. In the list of the frequent patterns, in addition to report each pattern we also display its frequency. However, if the first string is subject to one error and it becomes "ccabab", the new top-3 patterns will be $F_3 = \{(ab, 5), (ca, 3), (cc, 2)\}$, where the pattern "cc" replaces the pattern "bb".*

In the example above, we saw how the presence of one error in one of the data records made the pattern "*bb*" that originally was reported in the top-k to become infrequent. In this scenario, it is critical to design mining algorithms that take into account the presence of the noise, so that all the frequent patterns are captured. For this reason, in this section we first extend the definition of exact patterns with the presence of errors and second we investigate suitable applications where mining these patterns is crucial. We denote these patterns as *noisy* patterns since their structure depends on the noise presents in the original data.

4.1 New Problem Definition

To better understand the mining problem for these noisy patterns and design algorithmic solutions we first consider a further distinction on the type of patterns that the noise can induce.

Pattern with mismatches. When the input dataset is corrupted by noise or error multiple occurrences of a pattern could be lost. To handle this problem, we make use of similarity measure between strings, and we compute the frequency of a pattern p in the input set as the number of occurrences where p appears in an approximate way (i.e. the pattern occurs up to a maximum number of errors allowed). We formalize this concept in the following definition.

DEFINITION 3 (θ-APPROXIMATE OCCURRENCE). *Given a distance function between strings $d(\cdot, \cdot)$ and a threshold value θ, we say that a pattern p has a θ-approximate occurrence at position i in x if there exists a substring y of x, defined as $y = x_i x_{i+1} \dots x_m$ such that $d(p, y) \leq \theta$.*

Depending on the specific domain, several distance functions could be chosen to better represent the type of noise. Some possible functions that can be considered are the Hamming distance and the Edit distance.

EXAMPLE 2. *Let $x = abcbaabcacb$ be a record in the input dataset D and let d be the Hamming distance. The pattern "baa" has two 1-approximate occurrences in x: one at position 6 (where it appears with one mismatch), and one at position 3 (where it appears in an exact way).*

Therefore, the mining problem in this case consists in mining the top-k patterns where their θ-approximate occurrences are computed with respect to a given θ value.

Pattern with gaps. In Example 1, we saw that the noise perturbs the pattern by corrupting some of its symbols. In other cases, the noise can introduce a gap so that the pattern is broken into parts that co-occur in the dataset in input. In this scenario, we define the gapped pattern p as a triplet in the form $p = (m_1, m_2, \Delta)$, where m_1 and m_2 are the two exact components of the pattern and Δ is the maximum distance allowed within them.

DEFINITION 4 (GAPPED PATTERN OCCURRENCE). *A gapped pattern $p = (m_1, m_2, \Delta)$ has an occurrence at position i in the string x if there exists two substrings of x denoted as y_1 and y_2 where $y_1 = x_i x_{i+1} \cdots x_{i+|m_1|-1}$ and $y_2 = x_j x_{j+1} \cdots x_{j+|m_2|-1}$, such that y_1 and y_2 are occurrences of m_1 and m_2 respectively and $j - i - |m_1| + 1 \leq \Delta$.*

In our case, we measure the distance Δ between the components as the number of symbols between the tail of the first component to the head of the second one; however other measures can be adopted.

EXAMPLE 3. *Let $x = abcbaabcacb$ be a record in the input dataset D. The gapped pattern $(ab, cb, 2)$ occurs twice in x. It appears at position 0 of the string x where the gap between component is 0, and at position 5 where the gap is 2.*

Furthermore, the previous definition can be extended allowing the two components m_1 and m_2 to occur with some errors. Notice that, in general cases the pattern p could have more than two components; however it can always be decomposed as a set of gapped patterns with two components.

4.2 Potential Application Studies

A first possible scenario for mining these noisy patterns falls in the computational biology field. Indeed, the information conserved in biological structures (e.g. DNA, RNA, proteins) can be easily represented in terms of sequences of symbols, and this fact makes the search and discovery of patterns extensively used in Biology. One example of these informative patterns are the transcription factor binding sites (TFBSs) that are generally represented as gapped patterns. The discovery/mining of these sites is very challenging both biologically and computationally. In fact, TFBSs play an important role in the regulation of the gene expression and their prediction and location make it possible to understand the presence of inheritance diseases in the organism. The discovery of these sites is very hard due to the length of the biological sequence involved and the presence of noise intervened in the course of the evolution.

A second possible scenario where mining noisy patterns plays an important role is in monitoring time-series. Time-series data generally appear in the form of noisy numeric values that often can be discretized into symbols. For example, we can take the time series representing the blood pressure of a patient and convert it into a sequence of symbols on the alphabet $\Sigma = \{H, N, L\}$, where each symbol represents a high, normal and low level of blood pressure respectively. When the discretized sequence is constructed we perform the mining of noisy patterns to study the relationships between the presence of repetitive noisy patterns and the condition of the patient. In this way, we can use these patterns as features to diagnose the patient so that doctors can rapidly intervene with adequate medical treatments.

4.3 Challenges and Future Directions

The presence of noise in the data rises new challenges in mining patterns. First of all, when a pattern is allowed to occur with some errors the process for counting its occurrences is not trivial. For example, for a string x of n symbols the number of exact patterns of length l that can occur in x is $O(n)$. On the contrary, for noisy patterns given a maximum number of errors allowed θ there can be $O(l^\theta |\Sigma|^\theta n)$ patterns occurring in x. This explosion in the number of patterns is due to the presence of noise that allows to a large amount of patterns that are not present in the original data to be mined from the data. Another challenge is related to the privacy model. Although differential privacy is the state of the art for data privacy techniques, it is not clear how to translate the privacy needs that rise in biological scenarios into this privacy model. For example, in the case of DNA sequences we could think to construct privacy mechanisms that allow to perform analysis and data mining tasks while at the same time it protects the user (DNA owner) to be identified.

In mining noisy patterns we plan to consider two potential solutions. First, for mining gapped patterns in the form (m_1, m_2, Δ) a future direction consists in equipping our prefix tree with backward links of length Δ, so that when the component m_2 is mined from the suffix of a node in the tree we can use the link to quickly identify the component m_1 of the gapped pattern. Second, in mining pattern with mismatches we plan to investigate how to keep track of the threshold value θ while the prefix tree is traversed. In this way, we are able to estimate the θ-approximate occurrences of the noisy patterns in the data.

5. CONCLUSIONS

In this paper, we described the problem of mining frequent patterns in a privacy preserving scenario. We compared the concept of sequential pattern with the classical itemset pattern and made a distinction of patterns into exact and noisy. For exact patterns, we presented the state of the art for privacy preserving mining algorithms and described our mining techniques for sequential patterns. For noisy patterns, we proposed two definitions to capture the effects of the noise in the data. We pointed out possible scenarios where the mining of these patterns is central as well as the challenges in developing efficient mining algorithms. Future works include the extension of our privacy preserving prefix tree to mine noisy patterns, and developing privacy preserving techniques to handle genomic data.

6. REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, SIGMOD '93, pages 207–216, New York, NY, USA, 1993. ACM.

[2] A. Al-Lawati, D. Lee, and P. McDaniel. Blocking-aware private record linkage. In *Proceedings of the 2nd international workshop on Information quality in information systems*, IQIS '05, pages 59–68, 2005.

[3] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta. Discovering frequent patterns in sensitive data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 503–512, 2010.

[4] L. Bonomi and L. Xiong. A two-phase algorithm for mining sequential patterns with differential privacy. *To appear in the Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM)*, 2013.

[5] L. Bonomi, L. Xiong, R. Chen, and B. C. M. Fung. Frequent grams based embedding for privacy preserving record linkage. In *Proc. of the 21st ACM Conference on Information and Knowledge Management (CIKM)*, page 5 pages, Maui, HI, October 2012. ACM Press.

[6] R. Chen, G. Acs, and C. Castelluccia. Differentially private sequential data publication via variable-length n-grams. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12, pages 638–649, New York, NY, USA, 2012. ACM.

[7] T. Churches and P. Christen. Some methods for blindfolded record linkage. *BMC Medical Informatics and Decision Making*, 4(1):9, 2004.

[8] C. Dwork. Differential privacy. In *in ICALP*, pages 1–12. Springer, 2006.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. 2006.

[10] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Trans. Knowl. Data Eng.*, 19(1):1–16, 2007.

[11] J. Han, H. Cheng, D. Xin, and X. Yan. Frequent pattern mining: current status and future directions. *Data Min. Knowl. Discov.*, 15(1):55–86, Aug. 2007.

[12] A. Inan, M. Kantarcioglu, E. Bertino, and M. Scannapieco. A hybrid approach to private record linkage. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, pages 496–505, 2008.

[13] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino. Private record matching using differential privacy. In *Proceedings of the 13th International Conference on Extending Database Technology*, EDBT '10, pages 123–134, 2010.

[14] D. Kifer and J. Gehrke. l-diversity: Privacy beyond k-anonymity. In *In ICDE*, page 24, 2006.

[15] M. Kuzu, M. Kantarcioglu, A. Inan, E. Bertino, E. Durham, and B. Malin. Efficient privacy-aware record integration. In *Proceedings of the 16th International Conference on Extending Database Technology*, EDBT '13, pages 167–178, New York, NY, USA, 2013. ACM.

[16] N. Li, W. H. Qardaji, D. Su, and J. Cao. Privbasis: Frequent itemset mining with differential privacy. *PVLDB*, 5(11):1340–1351, 2012.

[17] Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. *Journal of Privacy and Confidentiality*, 1(5), 2009.

[18] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. FOCS '07. 48th Annual IEEE Symposium on*, pages 94 –103, oct. 2007.

[19] M. Scannapieco, I. Figotin, E. Bertino, and A. K. Elmagarmid. Privacy preserving schema and data matching. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, SIGMOD '07, pages 653–664, 2007.

[20] R. Schnell, T. Bachteler, and J. Reiher. Privacy-preserving record linkage using bloom filters. *BMC Medical Informatics and Decision Making*, 9(1):41, 2009.

[21] L. Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, Oct. 2002.

[22] W. Winkler. Overview of record linkage and current research directions. Technical Report Statistics #2006-2, 2006.

[23] M. Yakout, M. J. Atallah, and A. K. Elmagarmid. Efficient private record linkage. In *ICDE*, pages 1283–1286, 2009.

[24] A. C.-C. Yao. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162 –167, oct. 1986.

[25] C. Zeng, J. F. Naughton, and J.-Y. Cai. On differentially private frequent itemset mining. *Proc. VLDB Endow.*, 6(1):25–36, Nov. 2012.