Publishing Differentially Private Medical Events Data

Sigal Shaked, Lior Rokach

Publishing Differentially Private Medical Events Data

Sigal Shaked and Lior Rokach

The Department of Information Systems Engineering, Ben-Gurion University, Israel *
shaksi/liorrk@post.bgu.ac.il

Keywords: data synthetization, privacy preserving data publishing, Markov model, clustering, sequential patterns, differential privacy, medical events

Abstract. Sequential data has been widely collected in the past few years; in the public health domain it appears as collections of medical events such as lab results, electronic chart records, or hospitalization transactions. Publicly available sequential datasets for research purposes promises new insights, such as understanding patient types, and recognizing emerging diseases. Unfortunately, the publication of sequential data presents a significant threat to users' privacy. Since data owners prefer to avoid such risks, much of the collected data is currently unavailable to researchers. Existing anonymization techniques that aim at preserving sequential patterns lack two important features: handling long sequences and preserving occurrence times. In this paper, we address this challenge by employing an ensemble of Markovian models trained based on the source data. The ensemble takes several optional periodicity levels into consideration. Each model captures transitions between times and states according to shorter parts of the sequence, which is eventually reconstructed. Anonymity is provided by utilizing only elements of the model that guarantee differential privacy. Furthermore, we develop a solution for generating differentially private sequential data, which will bring us one step closer to publicly available medical datasets via sequential data. We applied this method to two real medical events datasets and received some encouraging results, demonstrating that the proposed method can be used to publish high quality anonymized data.

1 Introduction

A large amount of sequential medical event data has been gathered in the recent years. Studies based on this data can help address challenges in the medical field and may lead to new discoveries. Unfortunately, the publication of sequential data is accompanied by a real threat to users' privacy. Even when such data is not widely published as it might be in an academic study, shared datasets such as AOL's web querying history and Netflix's movie ratings (containing information which is publicly available to users of the services), have been vulnerable. Users

* This work was supported in part by Deustche Telekom Labs.

have been identified based on linking the published data with externally available data. Since data owners prefer to avoid this type of risk, the collected data remains inaccessible to scientists.

The work of De Montjoye et al. [10] emphasizes the magnitude of the privacy risk, as based on human mobility data spanning 15 months among 1.5 million users, the authors show that human mobility traces are somewhat "unique in the crowd", by demonstrating that four randomly chosen locations in an hour rounded resolution are enough to re-identify 95% of the users. Since four events can be obtained using very little outside information, their study represents a major threat to individuals' privacy.

There are two main traditional privacy models regarding record linkage, where the attacker reveals the owner of a record. The first aims at preventing an attacker from linking to a record owner, based on some quasi identifying attributes (QID) that were gained from external sources. K-anonymity [12] is usually adopted for this privacy model, demanding that each record is indistinguishable from at least $k-1$ other records with respect to the QID. [8, 7] place a human agent for defining the background knowledge of the hypothetical enemy.

The second privacy model, differential privacy [4] aims to ensure that by using the published data the attacker gains as less additional knowledge as possible. It checks that the removal or addition of a single record does not significantly affect results of the querying function. Typically, differential privacy is achieved by adding noise to the outcome of a query. A randomized algorithm satisfies $\varepsilon-$differential privacy if the ratio between the probability that the algorithm outputs any output on a dataset and the probability that it outputs the same output on a dataset that differ by exactly one record, is bounded by a constant.

Some methods have been suggested for anonymizing sequential data such as that used in the previously mentioned cases. Ghasemzadeh et al. [6] anonymize sequential data using a probabilistic flow graph, which is a tree representing transition probabilities between pairs of time and location. Violating sequences are suppressed in order to achieve LK-privacy. The inclusion of time in each state might worsen the model's sparsity. While this might be necessary for effective passenger flow analysis, it is unnecessary for simpler analysis tasks. Pensa et al. [11] use a prefix tree of transitions between states and a pruning technique to ensure k-anonymity in sequential data. Each pruned trajectory propagates an increase in the support of the most similar trajectory in the prefix tree. The last two methods use tree-based techniques, which do not scale well to large domains (complexity increases quadratically with the number of transitions). Furthermore, these approaches are built based on partition-based privacy models and therefore provide limited privacy protection. We prefer to apply differential privacy that makes no assumptions about the attacker's existing knowledge.

Chen et al. [3] propose a sanitization algorithm to generate differentially private sequential data by making use of a noisy prefix tree based on the underlying variable-length n-gram model behind the data. Each node holds the count of sequences described by the nodes in the current branch, and Laplace noise is added to these counts. Their method lacks some features that are required

in order for it to be applied in certain domains, such as medical events. First, input sequences are truncated to a predefined length, since the method does not perform well for long sequences. Second, the input data does not include a time dimension; this method can therefore neither support the generation of occurrence times, nor take into account possible dependencies of the sequential patterns with some other attributes that appear in the data.

Our suggested method addresses these gaps. We suggest a new privacy preservation method for sequential data, which generates differentially private synthetic data while preserving the sequential patterns of the source data. Data quality is maintained by using an ensemble of Markovian models, each of which captures another level of periodicity that considers dependency with an influencing attribute. Privacy is provided by using only the models' elements that fulfill differential privacy. In order to support the generation of occurrence times, each model also maintains transition times. Long sequences are handled by being partitioned into shorter parts, which are then clustered, condensed, and anonymized. The original long sequences are eventually reconstructed using a secondary ensemble of Markovian models.

To the best of our knowledge, this is the first use of Markovian generators to help ensure privacy preservation. Markovian generators have been applied to the task of sequence generation in other domains; for text generation, music composition and wind speed prediction.

The main contribution of this work is twofold; 1) demonstrating the generation of a differentially private medical events dataset, and 2) overcoming gaps in existing methods for sequential data anonymization, by providing a solution that incorporates generation of the time dimension, handles anonymization of long sequences, and considers dependencies of the sequential patterns with some predefined influencing attributes.

We evaluated the suggested method based on real world medical events data and received some encouraging results regarding its use.

The rest of this paper is organized as follows: in Sect. 2 background information and basic definitions are provided. Sect. 3 describes the model, and Sect. 4 presents the algorithm. In Sect. 5 we analyze performance and discuss experimental results. Finally, Sect. 6 concludes this work.

2 Background and Basic Definitions

2.1 Sequential data

Let $E = e_1, e_2, .., e_{|E|}$ be the universe of all possible states within a sequence, where the meaning of a state varies from one domain to another. A state in a sequence can be, for example, a charted event like a ventilator setting, or a laboratory value as appears in medical electronic chart data. Other examples of states in the medical events domain include procedures that appear in medical billing data, or a hospital unit that takes care of a patient within data that describes transfers of patients within different units in the hospital.

A transactional sequential dataset D contains records of the form $\langle O, t, e \rangle$. Each record indicates a state e that occurred at time t and is attributed to an object O.

In this work, we apply sequence reconstruction for gaining the sequences' anonymity while preserving their other features; we use Markovian generators for this task. As can be seen in (1), each state e_i adds a multiplication with the probability $Pr(e_i|e_{i-1})$ to transfer from the previous state to e_i. The probability of accurately reconstructing a sequence, therefore, decreases with the sequence's size:

$$p(S) = Pr(e_1)\Pi_{i=2}^{|S|} Pr(e_i|e_{i-1}) \tag{1}$$

It is also harder to apply differential privacy to long sequences, since the longer the sequence is, the less support it has. We therefore divide the original sequence of states S into several shorter sequences s_i.

Considering all states that are attached to a certain object as a single sequence S may yield long sequences; for example, when the data describes hourly resolution transactions for a specific patient over three years, the object is comprised of a sequence with around $3 * 365 * 24 = 26,280$ states. In order to avoid the shortcomings that come with long sequences, we initially split long sequences into shorter ones, as discussed in Subsection 3.3. This results in several possible sequences per objects which necessitates the following revision to our dataset definition:

Definition 1. *(a sequential dataset): A sequential dataset D contains records of the form $\langle O, t, e \rangle$. Each record indicates a state e that occurred at time t and is attributed to sequence s of object O.*

Definition 2. *(a sequence): A sequence s is an ordered list of states $s = e_1 \rightarrow e_2 \rightarrow .. \rightarrow e_{|s|}$, where $e_i \in E$. Records in a sequential dataset that belong to a single sequence s_1 share the same sequence Id (s) and object Id (O), and their times t_i dictate the order of states e_i in the sequence. A transition exists between two records that are attached to the same sequence, and have sequential times t_i and t_{i+1}. The transition time, therefore, is equal to $t_{i+1} - t_i$.*

Consecutive states in a sequence can be identical (for example: $s = e_3 \rightarrow e_2 \rightarrow e_2$). For convenience, we denote the original long sequence with S and the divided shorter sequence with s.

2.2 Cluster of sequences

Our suggested method is based on the assumption that input objects share some common sequential patterns. In order to recognize common patterns, we initially cluster parts of sequences into groups. We measure the similarity between sequences by applying MinHash [2], a locality sensitive hashing (LSH) method that is often used for reducing dimensionality. With LSH the similarity is measured as the ratio of common hashed tokens for the two compared sequences using

a family of hash functions H; the w-shingling (n-grams) method σ converts a sequence into a set of tokens.

$$LSH(s_1, s_2) = \frac{count_{h \in H}[h(\sigma(s_1)) = h(\sigma(s_2))]}{|H|} \tag{2}$$

Using the MinHash technique, the hash signature of a sequence s, $h(s)$, is constructed of minimal hash values of the tokens in the sequence. Using θ to hash each shingle z, we can hash a sequence s as follows:

$$h(s) = \min_{\forall_{z \in \sigma(s)}} (\theta(z)) \tag{3}$$

The MinHash technique can be used to efficiently measure the distance between sequences of unfixed size. Its low complexity stems from the fact that only $|H|$ comparisons are eventually made in order to estimate the distance between two sequences.

Definition 3. *(a cluster of sequences): A cluster C is a group of similar sequences s_i, according to the LSH similarity measure; $C = s_1, s_2, ..s_n$.*

The cluster's centroid represents sequential patterns that are commonly made by members of this cluster. When the cluster's members are similar, the cluster's centroid can be used to maintain sequential patterns of a higher quality than those maintained based on unclustered data. A cluster's centroid is represented as two Markovian chains, one for the times and the other for the states, as described later in Subsection 3.1. An object can be attached to several sequences and can therefore be attached to several different clusters.

Definition 4. *(the similarity between two sequential datasets): The similarity between an origin dataset D, and an anonymized dataset D' is measured as the mean LSH similarity between each sequence in D' and it's nearest neighbor in D. The distance is the complementary to one of the similarity.*

$$sim\left(D, D^{'}\right) = \sum_{s^{'} \in D^{'}} \frac{max_{\forall s \in D}\left(LSH\left(s, s^{'}\right)\right)}{|D^{'}|} \tag{4}$$

2.3 Differential privacy

The differential privacy model [5] guards against privacy breaches by ensuring that any computation made on the data by a randomized algorithm is insensitive to the presence of a single record. Applying this notion for sequential data, therefore, requires bounding the influence of each individual record.

Definition 5. *(differential privacy): A data generation method M provides a $\varepsilon-$differential privacy if, for any two datasets D and D' that differ on a single record (state transition), and for any possible output $R \in range(M)$, the probability to achieve the same output may only differ by a constant:*

$$pr\left(M(D) = R\right) \ = Pr\left(M(D') = R\right) \times e^{\mathcal{E}} \tag{5}$$

According to the composition property of differential privacy, a sequence of differentially private computations also provides differential privacy:

Theorem 1. *[5]: Let M_i be an $\varepsilon - differential\ privacy$ computation, then a sequence of n computations over a dataset D provides $(\sum_{i=1}^{n} \mathcal{E}_i)$-differential privacy.*

We can therefore compose a data generation method from several computational phases. Each phase, though, adds another level of noise ε_i to the anonymized data, which might damage the quality of the data.

2.4 Markov model

Predicting a sequence of states requires some heavy computations, especially when it comes to long sequences. A common approach is to adopt the Markov independence assumption. In an m-order Markov model, the probability for the appearance of a state in a sequence depends only the previous m states of the sequence. A 2-order Markov model has the lowest computational cost, since it only examines the previous state when predicting the current state.

$$Pr\left(e_{i+1}{=}a \mid e_1, e_2, .., e_i\right) {=} Pr\left(e_{i+1}{=}a \mid e_i\right) \tag{6}$$

In order to generate a sequence of states, frequencies of starting states and state transitions are collected; let F_D denote the frequency of a given term according to dataset D, so that:

$$StartProb\left(a\right) {=} \mathcal{F}_D\left(e_1{=}a\right) \tag{7}$$

$$TransitionProb(a,b) {=} \mathcal{F}_D\left(e_{i+1}{=}b \mid e_i{=}a\right) \tag{8}$$

3 The Model

3.1 State & time Markovian model (STMM)

Two Markovian chains are required in order to supply a solution that includes generation of occurrence times. While the first chain handles transitions between states, the second chain models navigation along the time dimension. The two chains are combined, since each time transition matches a specific state transition.

Definition 6. *(a state and time Markovian model (STMM)): a STMM model maintains statistics regarding a Markovian chain of transitions between states and its matching transition times according to dataset D.*

$$STMM(D) = \left\{ \begin{array}{l} \forall a \in range(E) \mid StartStateFreq(a), \\ \forall t \in \{1..24\} \mid StartTimeFreq(t), \\ \exists T \leftarrow \left\{ \forall_{i=1}^{|D-1|} \left(e_i, e_{i+1}\right) \right\} \Rightarrow \forall a, b \in range\left(T\right) \mid TransitionProb(a,b) \end{array} \right\} \tag{9}$$

where T denotes transitions within dataset D, and the maintained statistics are as follows:

Two types of statistics regarding the start of the chain:

$$StartStateFreq(a) = \mathcal{F}_D\left(e_1 = a\right) \tag{10}$$

$$StartTimeFreq(t) = \langle \mathcal{F}_D\left(t_1 = t\right), \mathcal{L}_D(t_1 = t) \rangle \tag{11}$$

e_1 is the starting state, and t_1 is its occurrence time. $StartStateFreq(a)$ is therefore the frequency of records in D that contain the state a, and $StartTimeFreq(t)$ is the frequency of records in D that start at time t, as well as the estimated (mean and standard deviation of) duration $\mathcal{L}_D$ for transitions in D that start at time t.

Another type of statistic is maintained regarding transitions along the chain:

$$Transition(a, b) = \langle \mathcal{F}_D(e_{i+1} = b \mid e_i = a), T_D(e_{i+1} = b \mid e_i = a) \rangle \tag{12}$$

where $Transition(a, b)$ holds both the frequency $\mathcal{F}_D$ of a transition $a \rightarrow b$ according to dataset D, and the (mean and standard deviation of) transition time T_D for $a \rightarrow b$ according to D.

3.2 Modelling state transitions

In order to generate an anonymized dataset of high quality, we attempt to capture as many characteristics of the data as possible, and to consider several optional periodicity levels that appear in the data. In order to accomplish this, we gather a number of models and design an ensemble based method; each model focuses on a certain trend at a specific level of accuracy. Let's clarify this by providing the two following definitions:

Definition 7. *(influencing factor): an influencing factor f is a direct or derived attribute within dataset D, which the sequence is assumed to depend upon. For each possible value of this attribute $v \in range(f)$, a model is trained to represent a specific sequential trend.*

We can assume, for example, that diverse sequential patterns exist on different weekdays, so that $f = weekday$ and $v \in \{Sun, Mon, Tue, Wed, Thu, Fri, Sat\}$.

Definition 8. *(support level): A support level l is a set of four possible categories $l \in \{cluster\&factor, factor, cluster, all\}$.*

The various categories will derive the homogeneity level of the trained model; categories with higher support level train more models in the ensemble, each addresses a smaller population with higher homogeneousness. The most accurate support level is $cluster\&factor$; models at this level are trained based on members of a given cluster C, while considering dependencies with factor f. The consideration of dependencies with a factor means that a separated STMM is created for each value v of that factor. The following support level is $factor$; here the model is trained based on the entire population of D, while considering

dependencies with factor f. The next support level is *cluster*, where the model is trained based on members of a given cluster C, while considering no influencing factors. The least accurate level is *all*, in which a single model is trained based on the entire population of D, while considering no influencing factors.

Definition 9. *(a state and time Markovian model ensemble (STE)): An ensemble of state and time Markovian models is the set of STMM models that were trained according to each support level ℓ, given a clustered sequential dataset D'', a set of clusters C, and a set of factors F.*

$$STE(D,\ell,C,F) = \bigcup_{\forall l \in \ell} \begin{cases} \forall c \in C, v \in f, f \in F | STMM(D''_{cluster=c \cap f.value=v}) & l = c \times f \\ \forall v \in f, f \in F | STMM(D''_{f.value=v}) & l = factor \\ \forall c \in C | STMM(D''_{cluster=c}) & l = cluster \\ STMM(D'') & l = all \end{cases} \quad (13)$$

3.3 Dividing sequences into shorter parts

Existing methods for differentially private sequence generation only handle short sequences [3, 1]. In these techniques the source sequences are initially truncated into a predefined length, and the truncated set is then used as an input for the sanitization process. In this work we address this gap and provide a solution that also suits long sequences. For this task we apply the concept of dividing long sequences into smaller parts and then reconnect the parts; in this approach most of the training work is performed on short sequences, including clustering, state transitions modeling, and model anonymization. By processing shorter sequences, our method obtains more homogenous clusters that better reflect common patterns within the data.

The division of a single long sequence into smaller sections can be performed artificially according to a predefined sequence length, but a more natural separation could take place by analyzing the input data. Dealing with data that includes a time dimension, for example, comes with the benefit of being able to estimate the natural separation into sections based on the distribution of transition times within the data. Transition time is the time it takes to reach from one state to another, which is actually derived from the difference between timestamps of two successive object's records. We assume that some regularity in transition times or distances exists while a certain part of a sequence is active. Whenever the regularity seems to change drastically, we infer the occurrence of a stop, and therefore the sequence is split at this point.

Let's demonstrate this analytic process as it was conducted on the TRANSFERS dataset that was used in our experiment (described in more detail later).

The histogram is based on the first 25,000 records in the data. As can be seen in Fig. 1, there seems to be some regularity of transition times that is distributed around the zero bar, diminishing at around 16,800 hours. The small peaks after this regularity indicate the duration of various stops.

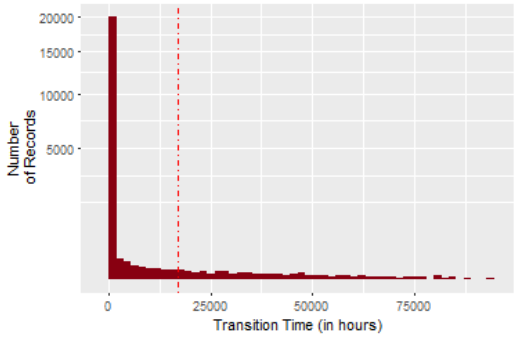

Fig. 1: A histogram of transition times for the $TRANSFERS$ data, based on 25,000 sampled transitions; the red line indicates the recognized splitting point (16,800 hours between transitions).

3.4 Reconstructing divided sequences

Processing shorter sequences facilitates the extraction of patterns with higher quality and enables their anonymization, but it requires an additional phase that connects the short sections into a whole sequence again. In order to reconnect sections of a divided long sequence, we use another Markovian chain that traverses from one section to another until connecting the entire original sequence back together. As mention before, each cluster represents some common sequential pattern. Assuming that members of each cluster are similar, by sampling a cluster we are able to ascertain (roughly) the pattern of a specific section of the sequence. The generation of a long sequence starts by sampling the initial cluster, according to which the first section of the sequence is generated. Each of the following sections of the sequence is then added to the sequence by sampling the transition to the following cluster and generating a short section of the sequence accordingly.

In order to strengthen our methods reconstruction ability, instead of modeling cluster transitions using a single Markovian chain, we suggest using an ensemble of such models that also takes into account some predefined factors. For this task we use another ensemble of STMMs which we call a cluster transitions ensemble (CTE).

Definition 10. *(CTE): An ensemble of state and time Markovian models, which is a set of STMM models that were trained according to each accuracy level ℓ, given a sequence of clusters per object $D_{clusters}$, and a set of factors F.*

$$CTE(D_{clusters}, \ell, F) = \bigcup_{\forall l \in \ell} \begin{cases} \forall v \in f, f \in F | STMM(D_{clusters_{f.value=v}}) & l = factor \\ STMM(D_{clusters}) & l = all \end{cases} \quad (14)$$

Since CTE represent chains of transitions between clusters, the state in this chain is the cluster. The starting state frequency $StartStateFreq(a)$ therefore

represents the frequency of a specific starting cluster a and the estimated size of the starting sequence according to $D_{cluster}$; a minor difference from the use of STMM in STE is that in CTE we preserve statistics for the mean number of sections per sequence, while in STE we preserve statistics for mean duration per sequence. In a similar manner, $StartTimeFreq(t)$ is the frequency of sequences starting at time t.

The transition frequency $Transition(a, b)$ represents the frequency of transitioning from cluster a to cluster b, as well as the estimated time between the starting times of subsequent sequences in these two clusters, according to $D_{cluster}$.

3.5 Anonymizing the model

As stated in Definition 5, a method M realizes differential privacy if it provides the same output as it would have provided in the absence of each possible input record.

Denoting the source data as D and the same data with one less record as D', differential privacy applies the following constraint to each possible calculation:

$$\frac{Pr\left(M(D) = R\right)}{Pr\left(M(D') = R\right)} \leq e^{\varepsilon} \tag{15}$$

Applying differential privacy in our method, therefore, requires that all calculations in the model maintain this constraint. The pertinent statistics in our basic STMM model are frequencies of starting states, starting times, and transitions. By maintaining a count alongside the frequencies, we can easily check whether a given statistic meets the constraint; this is done by dividing the current frequency with the frequency achieved by removing a single record (from both numerator and denominator). For example, if the frequency of transition $a \rightarrow b$ is 0.3, based on three occurrences of this transition out of 10 transitions from a elsewhere, we must check the following constraint:

$\frac{Pr(transition(D)=a\rightarrow b)}{Pr(transition(D')=a\rightarrow b)} = \frac{3/10}{(3-1)/(10-1)} \leq e^{\varepsilon}$

The ε parameter calibrates the amount of the permitted difference between calculations with and without a single record; using higher ε values permits a greater difference and therefore reduces the anonymity level. In the current example, the constraint is supplied for $\varepsilon = 0.4$, but it is not supplied for $\varepsilon = 0.3$. If the ε parameter is predefined as 0.3, the transition $a \rightarrow b$ is suppressed from the model, and statistics are normalized and checked for the differential privacy constraint again; this continues until no further suppressions are made.

3.6 Additional statistics

Some additional statistics are maintained in order to facilitate the data generation process. First, a list is compiled that, for each date maintains the number of starting objects, as well as the number of sections (mean and standard deviation) that form a starting object on this date. Next, some generic mean and standard

deviation statistics are maintained for state transition time, cluster transition time, and sequence duration, as well as starting hour frequencies which are also kept. These statistics are held without the enforcement of differential privacy; it is possible to anonymize these statistics, but our assumption is that they are not considered sensitive.

However, other statistics are maintained, in adherence with differential privacy. For the sake of "fixing" a transition whenever relevant parts in the model were suppressed during the anonymization process, we maintain frequencies of neighbors for states, as well as for clusters. By using these statistics we can reach an indirect neighbor whenever no direct neighbor was found in the model. The frequency for each pair of states is based on the number of sequences that contain these two states (although not necessarily successively). In a similar manner, for each pair of clusters, the frequency is based on the number of objects that are attached to these two clusters.

4 Synthetic Data Generator

Our anonymization method adopts the sanitization-based approach. It trains a model, based on the assumption that input objects share some common sequential patterns, and generates data accordingly. We use the source sequential dataset as an input, and deliver an equivalent anonymized data as the output. Statistics and patterns can then be extracted from the output using various queries. This approach imposes no limitations on the analyzing capabilities, as opposed to techniques that anonymize data mining results.

The generation process includes three main phases: 1) sampling the daily number of new starting objects; 2) sampling start and end times for each section in the object's sequence, as well as an attached cluster, which represents the general pattern for this section; 3) filling these time slots with a sequence of state and occurrence time pairs.

Since we deal with an ensemble of models, it is possible that relevant models lead to the sampling of different values. For example, sampling a starting state which has two factors in the model (weekday and hour), requires the combination of two models. If the current generation point is Monday at 6:00, then the starting state' frequencies of the 6:00 model should be combined with those of Monday's model. We use weighted means for combining matching values within the various relevant models.

In order to emphasize the effect of less common trends, we manipulate the weights so that models with lower support gain more weight. In our example, when the Wednesday model has the support of 20,000 transactions, and the 6:00 model has the support of 100 transactions, we give more weight to the latter, since in this example, 6:00 is more specific. The inversion of the weights is calculated as one divided by the support of each model, with an additional step of normalizing the inverted weights so that their total sum is one.

5 Performance Analysis

We consider both quality and anonymity of the synthesized data while evaluating our suggested method. We use two quality measures; the first is the distance between two sequential datasets as presented in Definition 4. This measures the mean similarity between each object in the anonymized data and its nearest object in the source data. The second measure is the intersection of the top 20 frequent patterns in the two compared datasets (only 2-gram patterns were considered, as a fast estimation).

Anonymity is measured as the mean support for statistics in the model. We also consider the percentage of suppressed records in the model in order to gain a deeper understanding of the trade-off between quality and anonymity.

We applied our proposed method to two tables of the MIMIC-III (Medical Information Mart for Intensive Care III) database [9], which is a large, freely-available database comprising de-identified health-related data associated with over 40,000 patients who stayed in critical care units at Beth Israel Deaconess Medical Center between 2001 and 2012. The $TRANSFERS$ table contains physical locations for patients throughout their hospital stay. The $CPTEVENTS$ table contains current procedural terminology (CPT) codes, which facilitate billing for procedures performed on patients. A sample of patient data from each of the tables was used to evaluate our method, as described in Table 1.

The described methods were implemented in R and ran using AWS (community AMI ID: ami-753e7c10; instance type: m4.xlarge). The number of clusters was set to 15% of the number of sequences. Sequence duration, weekday, and day hour were predefined as factors that are considered when generating a sequence of states, and weekday, month, and year were predefined as factors that are considered when re-constructing a long sequence from shorter sections. The separation into shorter sections was conducted according to transition times of at least 16,800 hours in the $TRANSFERS$ data, as described in Subsection 3.3. In the $CPTEVENTS$ data which contains no valid data in the time dimension, a fabricated time dimension was calculated according to the order of procedures ($ticketID\_seq$); splitting by transition times is irrelevant in this case. We examined the performance of our suggested method under different privacy budgets (as set by the ε parameter), as well as under different settings of the LSH similarity measure (number of hash functions and shingle size). The anonymization process was repeated ten times for each combination of four different values of

Table 1: Attributes of the experimental data

Dataset	Objects	Sequences	Max records per object	Records	States	Defined state
$TRANSFERS$	4,370	4,655	138	24,996	85	$curr\_warid + eventtype$ (unit+operation)
$CPTEVENTS$	3,171	3,171	116	25,000	117	cpt_cd (procedure)

the ε parameter (0.05, 0.35, 0.65, and infinity), three possible numbers of hash functions (100, 250, and 500), and three different shingle sizes (1, 2, and 3). The main characteristics for each of the experimental data samples are described in Table 1, including the number of objects, sequences, records, and states for each sample, and the maximal number of records per object. The defined state is also described.

Figures 2 and 3 report the quality of the anonymized data according to two quality measures: mean distance and top frequent patterns intersection rate. These measures are described under various privacy budgets, as well as various settings of the LSH distance measure.

As illustrated in Fig. 2, the higher the privacy budget (the ε parameter) is, the lower the mean distance which is obtained, indicating that the anonymized data is more similar to the source data using higher privacy budgets. This mainly occurs since as anonymity demands become more restrictive, uncommon trends are suppressed, and the common trends remain quite similar to their source form. 90 to 100% similarity was obtained for the $CPTEVENTS$ data, where higher support for common patterns exists, and 76 to 81% similarity for the $TRANSFERS$ data, where there is more pattern variety. We also examined

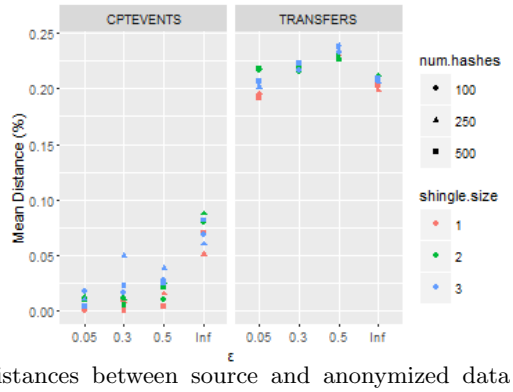

Fig. 2: Mean distances between source and anonymized data (y dimension) versus anonymity level as set by the ε parameter (x dimension) for the $CPTEVENTS$ and $TRANSFERS$ data (on the left and right, respectively). Three examined shingle sizes for the LSH distance measure are distinguished by color, and three examined number of hash functions used by the LSH are indicated by shape. One shingle was found to provide lower distance according to the Friedman test.

the influence of different settings of the LSH distance measure; this measure was used for measuring similarity between sequences in the clustering phase. One shingle was found to provide lower distance according to the Friedman test (at a 95% significance level). No additional significant difference was found within the various examined settings.

As presented in Fig. 3, there is no clear trend of the frequent patterns intersection rate measure with regard to the increase in privacy budget (the ε parameter). Moreover, no significant differences in this measure were found within the various examined settings of the LSH distance measure.

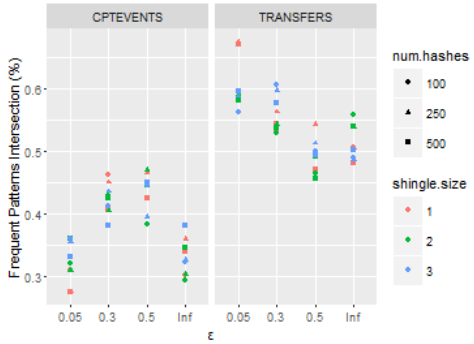

Fig. 3: Frequent patterns intersection rate between source and anonymized data (y dimension) versus anonymity level as set by the ε parameter (x dimension) for the $CPTEVENTS$ and $TRANSFERS$ data (on the left and right, respectively). Three examined shingle sizes for the LSH distance measure are distinguished by color, and three examined number of hash functions used by the LSH are indicated by shape.

Figures 4 and 5 report the anonymity of the generated data according to the mean support rate measure and examine the tradeoff between quality and anonymity according to the suppression rate measure. These measures are described under various privacy budgets (the ε parameter), as well as various settings of the LSH distance measure (number of hash functions and shingle size). As presented in Fig. 4, the higher the privacy budget (the ε parameter) is, the few records are suppressed. This demonstrates the tradeoff between anonymity and quality, since the suppression provides anonymity at the expense of damaging data quality. We also examined the influence of different settings of the LSH distance measure; smaller shingle sizes provides lower suppression rates; using 100 hash functions also decreases suppression rate. These findings are supported by Friedman tests (at a 95% significance level).

As illustrated in Fig. 5, the higher the privacy budget (the ε parameter) is, the lower mean support is provided. This can be explained by the fact that as the privacy budget increases, less common trends (which tend to have lower support) influence. We also examined the influence of different settings of the LSH distance measure; one shingle provides higher mean support (supported by Friedman tests at a 95% significance level).

6 Conclusions

In this paper, we proposed a novel privacy preservation approach for publishing differentially private sequential data based on an ensemble of Markovian models

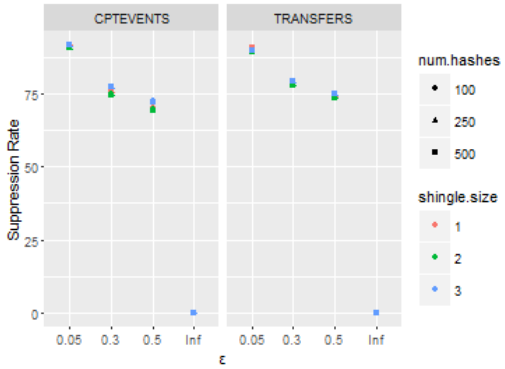

Fig. 4: Suppression rate between source and anonymized data (y dimension) versus anonymity level as set by the ε parameter (x dimension) for the $CPTEVENTS$ and $TRANSFERS$ data (on the left and right, respectively). Three examined shingle sizes for the LSH distance measure are distinguished by color, and three examined number of hash functions used by the LSH are indicated by shape. Lower shingle sizes, as well as smallest number of hash functions, were found to provide lower suppression rates.

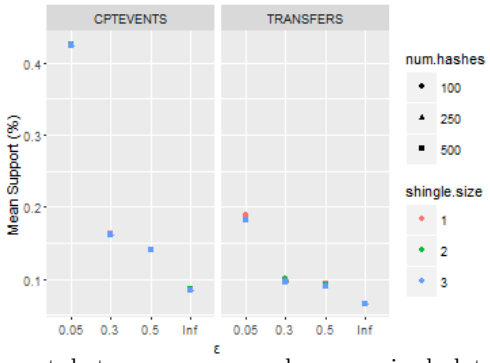

Fig. 5: Mean support between source and anonymized data (y dimension) versus anonymity level as set by the ε parameter (x dimension) for the $CPTEVENTS$ and $TRANSFERS$ data (on the left and right, respectively). Three examined shingle sizes for the LSH distance measure are distinguished by color, and three examined number of hash functions used by the LSH are indicated by shape. One shingle was found to provide higher mean support.

and demonstrated how it can be applied to anonymize medical events data. We designed a method for synthesizing sequential data, which allows published sequences to be used for a wider range of data analysis tasks, while preventing invasion or misuse of users' privacy.

Extensive experiments on actual medical events datasets demonstrated that our solution provides high quality anonymized data in terms of mean distance between the source and anonymized data. It also indicated that minimal performance demanding settings of the used similarity measure (one shingle and 100

hash functions) supply the best level of quality and anonymity, perhaps since using a more accurate distance measure results in less support for each pattern and increases the suppression rate.

Examining additional influencing factors such as patient attributes is essential in order to further improve the quality of the anonymized data and to refine our suggested method so it is applicable to additional datasets in the medical domain.

References

1. Bonomi, L., Xiong, L.: A two-phase algorithm for mining sequential patterns with differential privacy. In: Proceedings of the 22nd ACM international conference on Conference on information & knowledge management - CIKM '13. pp. 269–278. ACM Press, New York, New York, USA (2013)
2. Broder, A.Z., Charikar, M., Frieze, A.M., Mitzenmacher, M.: Min-wise independent permutations. In: Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98. pp. 327–336. ACM Press, New York, New York, USA (1998)
3. Chen, R., Acs, G., Castelluccia, C.: Differentially private sequential data publication via variable-length n-grams. In: Proceedings of the 2012 ACM conference on Computer and communications security - CCS '12. p. 638. ACM Press (2012)
4. Dwork, C.: Differential Privacy. Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP) pp. 1–12 (2006)
5. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC'06 Proceedings of the Third conference on Theory of Cryptography. Lecture Notes in Computer Science, vol. 3876, pp. 265–284. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
6. Ghasemzadeh, M., Fung, B.C., Chen, R., Awasthi, A.: Anonymizing trajectory data for passenger flow analysis. Transportation Research Part C: Emerging Technologies 39, 63–79 (2014)
7. Holzinger, A.: Interactive machine learning for health informatics: when do we need the human-in-the-loop? Brain Informatics 3(2), 119–131 (2016)
8. Kieseberg, P., Malle, B., Frühwirt, P., Weippl, E., Holzinger, A.: A tamper-proof audit and control system for the doctor in the loop. Brain Informatics pp. 1–11 (2016)
9. Lee, J., Scott, D.J., Villarroel, M., Clifford, G.D., Saeed, M., Mark, R.G.: Open-access MIMIC-II database for intensive care research. Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference 2011, 8315–8 (2011)
10. de Montjoye, Y.A., Hidalgo, C.A., Verleysen, M., Blondel, V.D.: Unique in the Crowd: The privacy bounds of human mobility. Scientific reports 3, 1376 (2013)
11. Pensa, R.G., Monreale, A., Pinelli, F., Pedreschi, D.: Pattern-preserving k-anonymization of sequences and its application to mobility data mining. CEUR Workshop Proceedings 397, 44–60 (2008)
12. Samarati, P.: Protecting respondents identities in microdata release. IEEE Transactions on Knowledge and Data Engineering 13(6), 1010–1027 (2001)