Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

_____ _____

James J. Gardner Date

Privacy Preserving Medical Data Publishing

By

James Johnson Gardner

Doctor of Philosophy
Computer Science and Informatics

Li Xiong, Ph.D.
Advisor

Eugene Agichtein, Ph.D.
Committee Member

James Lu, Ph.D.
Committee Member

Andrew Post, M.D., Ph.D.
Committee Member

Accepted:

Lisa A. Tedesco, Ph.D.
Dean of the James T. Laney School of Graduate Studies

Date

Privacy Preserving Medical Data Publishing

By

James Johnson Gardner
M.S. Computer Science, Emory University, Atlanta, 2007

Advisor: Li Xiong, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2012

Abstract

Privacy Preserving Medical Data Publishing

By James Johnson Gardner

There is an increasing need for sharing of medical information for public health research. Data custodians and honest brokers have an ethical and legal requirement to protect the privacy of individuals when publishing medical datasets. This dissertation presents an end-to-end Health Information DE-identification (HIDE) system and framework that promotes and enables privacy preserving medical data publishing of textual, structured, and aggregated statistics gleaned from electronic health records (EHRs). This work reviews existing de-identification systems, personal health information (PHI) detection, record anonymization, and differential privacy of multi-dimensional data. HIDE integrates several state-of-the-art algorithms into a unified system for privacy preserving medical data publishing. The system has been applied to a variety of real-world and academic medical datasets. The main contributions of HIDE include: 1) a conceptual framework and software system for anonymizing heterogeneous health data, 2) an adaptation and evaluation of information extraction techniques and modification of sampling techniques for protected health information (PHI) and sensitive information extraction in health data, and 3) applications and extension of privacy techniques to provide privacy preserving publishing options to medical data custodians, including de-identified record release with weak privacy and multidimensional statistical data release with strong privacy.

Privacy Preserving Medical Data Publishing

By

James Johnson Gardner
M.S. Computer Science, Emory University, Atlanta, 2007

Advisor: Li Xiong, Ph.D.

A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2012

Acknowledgements

I am grateful to all individuals who have spoken with me and provided insight in my study of philosophy and science. I am most grateful to my advisor, Dr. Li Xiong. Your guidance, patience, and wisdom have improved my abilities in more ways than the contents of this dissertation. I also thank my committee members, Dr. Eugene Agichtein, Dr. James Lu, and Dr. Andrew Post. Your advice improved my education, teaching, and research that led to the ideas in this dissertation. I also thank my friends and family for the practical advice and discussions.

To my wife Kelly, brother Andy, Mom, and Dad

Contents

1 Introduction **1**

1.1 Privacy 2

1.2 Health Information DE-identification 3

1.2.1 Overview 4

1.2.2 Contributions 4

1.3 Organization 5

2 Background and Related Work **6**

2.1 Existing medical record de-identification systems 6

2.2 Privacy preserving data publishing 10

2.2.1 De-identification options specified by HIPAA 11

2.2.2 General anonymization principles 12

2.3 Formal principles 13

2.3.1 Weak privacy 14

2.3.2 Strong privacy 15

2.4 Discussion 20

3 HIDE Framework **21**

3.1 Overview 21

3.2 Health information extraction 23

3.3 Data linking 23

3.4 Privacy models 24

3.4.1 Weak privacy through structured anonymization 25

3.4.2 Strong privacy through differentially private data cubes 25

3.5 Heterogeneous Medical Data 26

3.5.1 Formats 26

3.5.2 Datasets used in this dissertation 27

3.6 Software 30

3.7 Discussion 31

4 Health Information Extraction **33**

4.1 Modeling PHI detection 34

4.2 Conditional Random Field background 37

4.2.1 Features and Sequence Labeling 37

4.2.2 From Generative to Discriminative 38

4.2.3 Definition 41

4.2.4 Parameter Learning 46

4.3 Metrics 50

4.4 Feature sets 51

4.4.1 Regular expression features 51

4.4.2 Affix features 52
4.4.3 Dictionary features 53
4.4.4 Context features 53
4.4.5 Experiments 53
4.5 Sampling 57
4.5.1 Cost-proportionate sampling 57
4.5.2 Random O-sampling 58
4.5.3 Window sampling 59
4.5.4 Experiments 59
4.6 Discussion 66

5 Privacy-Preserving Publishing 68

5.1 Weak privacy 69
5.1.1 Mondrian Algorithm 69
5.1.2 Count Queries on Extracted PHI 70
5.2 Strong privacy 72
5.2.1 Differentially private data cubes 73
5.2.2 DPCube algorithm 76
5.2.3 Temporal queries 79
5.3 Evaluations 82
5.3.1 Distribution accuracy 83
5.3.2 Information gain threshold 88
5.3.3 Trend accuracy 89

5.3.4 Temporal queries 90
5.3.5 Applying DPCube to temporal data 92
5.3.6 Applying tree-based approach to temporal data 93
5.4 Discussion 97

6 Conclusion and Future Work **98**
6.1 Integration 99
6.2 Extension of prefix tree approach 99
6.3 Combining unstructured data 101
6.4 Larger-scale statistical analysis 101
6.5 Clinical use cases 102
6.6 Conclusion 103

Chapter 1

Introduction

We are in the age where massive data collection, storage, and analysis is possible. Although this data has proven useful [31], data custodians have the ethical responsibility maintain the privacy of individuals in the data, especially in the health-care domain. Preserving the privacy of individuals in medical data repositories is not only an ethical requirement, but also mandated by law in the United States by the Health Insurance Portability and Accountability Act (HIPAA)[1].

This dissertation focuses on privacy preserving data publishing and solutions to limiting the risk of disclosing confidential information about individuals. Most research has focused on specific types of privacy breaches or attacks on specific data sets. This work focuses on privacy algorithms and methods that give the maximum amount of utility for a variety of analyses on hetero-

[1] `http://www.hhs.gov/ocr/privacy/`

geneous medical datasets. Multiple experiments show the ability of medical publishing practitioners to decide between the level of utility and privacy of data chosen for release.

1.1 Privacy

The goal of privacy preserving medical data publishing is to ensure that confidential patient data is not disclosed. Privacy models typically include consider three types of disclosure: identity, attribute, and inferential disclosure. Prevention of identity disclosure focuses on perturbing the records, so that any one record doesn't uniquely identify an individual with any outside data source. Attribute disclosure is prevented if no new information about a particular individual is disclosed after releasing the data. Inferential disclosure prevention involves removing the statistical properties of the released data, that allow for high confidence predictions of an individuals confidential information.

Methods for preventing unauthorized disclosure of information include: restricting access, restricting the data, and restricting the output. Restricting access by locking down the data is a relatively simple solution to the privacy problem, but it completely eliminates the utility of the data. It is critical that useful medical information be shared across research institutions. Restricting the data involves removing attributes or modifying the dataset with some form of generalization or perturbation of values. Restricting the output involves transforming the results of user queries while leaving the data

unchanged. The restricted data approach allows for much more widespread sharing and distribution of the data.

The tradeoff between privacy and utility has been the subject of much research and debate. A variety of models and techniques for preserving privacy have been explored by medical and privacy researchers. The privacy models can be classified into two types: weak and strong privacy. The terminologies of weak privacy and strong privacy are adopted in order to help elucidate these concepts to health care professionals and regulators.

A dataset is said to exhibit weak privacy if the privacy of individuals is ensured assuming the users with access to the data have some predetermined set of background knowledge, e.g. knowing that the user has access to voter registration or other public datasets. These privacy models are best suited when releasing individual records is required. A dataset with strong privacy ensures privacy without assuming the background knowledge of the attackers. These models are best suited when releasing aggregated statistics from the datasets. Chapter 2 presents formal privacy principles and techniques.

1.2 Health Information DE-identification

The main subject and contribution of this dissertation is the Health Information DE-identification (HIDE) software and framework developed to aid health data custodians and publishers with the publishing of sensitive medical information.

1.2.1 Overview

HIDE provides an end-to-end framework for publishing HIPAA-compliant, de-identified patient records, anonymized tables and differentially private data cubes (multi-dimensional histograms). The released data allows researchers to deduce important medical findings without compromising the privacy of individuals. This dissertation includes examples and solutions to problems faced by medical data publishers, researchers, and privacy advocates. The end result is a framework that encourages information sharing that allows also for the protection of individuals privacy.

1.2.2 Contributions

The main contributions of HIDE include: 1) a conceptual framework and software system for anonymizing heterogeneous health data [24, 26], 2) an adaptation and evaluation of information extraction techniques and modification of sampling techniques for protected health information (PHI) and sensitive information extraction in health data [25], and 3) applications and extension of privacy techniques to provide privacy preserving publishing options to medical data custodians, including de-identified record release with weak privacy [24, 26] and multidimensional statistical data release with strong privacy [76].

Each of these contributions was validated on real-world datasets and information gathering tasks. The framework provides medical data custodians and researchers with formal guarantees of privacy without having to rely on

the typical "common sense" approaches, which can help prevent oversight and unforeseen privacy leaks. The information extraction techniques and recall-enhancing sampling techniques studied on real-world medical data give practical expectations on the privacy that can be provided by automatic methods. The usage of formal privacy techniques give formal guarantees of privacy, which are typically lacking in honest brokers and data releasers data toolboxes. The extensions of multidimensional aggregated statistical privacy techniques provide guaranteed privacy for the difficult problem of determining the best partitioning of the data necessary to release useful privacy preserving statistics. Results in the final chapter show the utility of a variety of anonymization techniques and include extensions beyond those demonstrated in [76].

1.3 Organization

The remainder of this dissertation is organized as follows. Chapter 2 reviews the related work and gives initial background information. Chapter 3 discusses the HIDE framework in detail. Chapter 4 discusses information extraction techniques used for detection of PHI. Chapter 5 discusses privacy and anonymized release of heterogeneous data. Chapter 6 gives conclusion and future work.

Chapter 2

Background and Related Work

This chapter gives background information on techniques used for privacy-preserving publishing of medical records. Existing information extraction, structured anonymization, and differential privacy techniques are presented.

The remainder of this dissertation will use the terms medical reports, electronic health records (EHRs), and electronic health information (EHI) interchangeably.

2.1 Existing medical record de-identification systems

Previous approaches to de-identifying medical records follow a two step process. First they identify PHI in the text then replace the PHI with a placeholder such as "XXXXX" or "–XNAMEX–." The most common approaches

to de-identification are based on rules and dictionaries or statistical learning techniques. Efforts on de-identifying medical text documents in medical informatics community [63, 61, 67, 66, 30, 59, 4, 68] are mostly specialized for specific document types or a subset of HIPAA identifiers. Most importantly, they rely on simple identifier removal techniques without taking advantage of the research developments from data privacy community that guarantee a more formalized notion of privacy while maximizing data utility.

Extracting atomic identifying and sensitive attributes (such as name, address, and disease name) from unstructured data can be seen as an application of named entity recognition (NER) [49]. NER systems can be roughly classified into two categories and are both applied in medical domains for de-identification: rule-based and statistical learning-based. The rule-based (or grammar-based) techniques rely heavily on hand-coded rules and dictionaries. Depending on the type of identifying information, there are common approaches that can be used. For identifiers that are in a closed class with an exhaustive list of values such as geographical locations and names, common knowledge bases such as lists for area codes, common names, words that sound like first names (Soundex) can be used for lookups. Local knowledge such as first names of all patients in a specific hospital can be also used for specific dataset. For identifying information that follows certain syntactic pattern such as phone numbers and zip codes, regular expressions can be used to match the patterns. Common recording practices (templates) with respect to personal information can be utilized to build rules. For many cases, a mixture

of information including context such as prefix for a person name, syntactic features, dictionaries, and heuristics need to be considered. Such hand-crafted systems typically obtain good results, but at the cost of months of work by experienced domain experts. In addition, the rules that are used for extracting identifying information will likely need to change for different types of records (radiology, surgical pathology, operative notes) and across organizations (hospital A formats, hospital B formats). The software will become increasingly complex with growing rules and dictionaries.

The scrub system [63] is one of the earliest de-identification systems that locates and replaces HIPAA-compliant personally-identifying information for general medical records. The system uses rules and dictionaries to label and remove text that is identified as a name, an address, a phone number, etc. The medical document anonymization system with a semantic lexicon [55] is another system that uses rules to locate and removes personally-identifying information in patient records. The system builds rules based on the surrounding terms and information gleaned from a sematic lexicon to detect PHI. It removes explicit personally-identifying information such as name, address, phone number, and date of birth. An alternative approach that uses a dictionary of safe (guaranteed non-PHI) terms and removes all terms that are not in the list can be found in [7]. The Concept-Match algorithm steps through the record replacing all standard medical terms with the corresponding code, leaves all high frequency (stop words) and removes all other terms leaving a de-identified record. This technique has high recall, but suffers from lower

precision. DE-ID [30] is another system that uses rules and dictionaries developed at the University of Pittsburgh, where it is used as the de-identification standard for all clinical research approved by the Institutional Review Board (IRB). HMS Scrubber [6] is an open-source system implemented in Java that utilizes the header information associated with a record, rules for detecting common PHI (e.g. dates), and a dictionary of common names (and names associated with the institution). Any information that matches is then removed from the record. An alternative open-source system implemented in Perl using similar techniques as the HMS Scrubber can be found in [51].

The statistical (or machine) learning-based approaches have been applied to the NER problem with remarkable success. Much work has focused on modeling NER as a sequence labeling task, where each word in the text is classified as a particular type. Statistical sequence-labeling involves training classifiers to label the tokens in the text to indicate the presence (or absence) of an entity. The classifier uses a list of feature attributes for training and classification of the terms in new text as either identifier or non-identifier. The best performing systems use a variety of features.

An SVM-based system is proposed in [29] for de-identifying medical discharge summaries using a statistical SVM-based classification method. The system does not distinguish between different types of PHI but simply between PHI and non-PHI. Another approach using SVM is discussed in [60]. A variation of a decision tree is used to detect PHI in [65]. A CRF-based system is presented in [72]. The system uses regular expression and context features

and models the detection as a sequence labeling problem.

The limitations of the above systems are that they do not use formal privacy principles to guarantee privacy and it still remains an open question as to how much information must be removed (or modified) from text data so that we can ensure that the text is de-identified. Chapter 4 covers the health information extraction problem in more detail.

2.2 Privacy preserving data publishing

Currently, investigators or institutions wishing to use medical records for research purposes have three options: obtain permission from the patients, obtain a waiver of informed consent from their Institutional Review Boards (IRB), or use a data set that has had all or most of the identifiers removed. The last option can be generalized into the problem of de-identification or anonymization (both de-identification and anonymization are used interchangeably throughout this dissertation) where a *data custodian* distributes an anonymized view of the data that does not contain individually identifiable information to a *data recipient*.

Protected health information (PHI) is defined by HIPAA as individually identifiable health information. We use PHI to refer to protected health information and personal health information interchangeably, because it is possible to deduce the identity of a patient based only on the various attributes in the individuals records, not just specific identifiers. Identifiable information refers

to data that can be linked to a particular individual. Names and Social Security numbers are examples of direct identifiers. Age, gender, and zip codes are examples of indirect identifiers.

2.2.1 De-identification options specified by HIPAA

HIPAA defines three main methods for de-identifying records.

Full De-identification. Information is considered fully de-identified by HIPAA if all of the identifiers (direct and indirect) have been removed and there is no reasonable basis to believe that the remaining information could be used to identify a person. The full de-identification option allows a user to remove all explicitly stated identifiers.

Partial De-identification. As an alternative to full de-identification, HIPAA makes provisions for a limited data set[1] from which direct identifiers (such as name and address) are removed, but not indirect ones (such as age). The partial de-identification option allows a user to remove the direct identifiers.

Statistical De-identification. Statistical de-identification attempts to maintain as much "useful" data as possible while guaranteeing statistically acceptable data privacy. Many such statistical criteria and anonymization techniques have been proposed for structured data.

[1] limited data sets require data use agreements between the parties from which and to which information is provided.

2.2.2 General anonymization principles

The previous definitions provided by HIPAA are used by medical data custodians and honest brokers. At a higher level of abstraction, anonymization techniques can be classified into four main categories.

Data suppression. Full and partial de-identification as defined by HIPAA are forms of data suppression, where the value of the attributes are removed completely. The drawback is that this information is completely lost in the final release.

Data generalization. Generalization involves grouping (or binning) attributes into equivalence classes. Numeric attributes are discretized to a range similar to the construction of histogram bins, e.g. date of birth could be generalized to the year of birth. If a concept hierarchy exists, then categorical attributes can be replaced with values higher in the concept hierarchy, e.g. a city mentioned in the records could be generalized into the state where the city is located.

Data swapping. Data swapping modifies records by switching a subset of attributes between pairs of records.

Micro-aggregation. Micro-aggregation involves clustering records. For each cluster, the data values are replaced with a representative value that is typically the average value along each dimension in the cluster.

Macro-aggregation. In macro-aggregation, the individual records are never released, but aggregations of statistics over the population in the dataset are

released with some level of perturbation.

2.3 Formal principles

Privacy preserving data publishing and analysis has received much attention over the last decade [3, 17, 23]. At the first glance, the general problem of data anonymization has been extensively studied in recent years in the data privacy community. Most of the work has been focused on formalizing the notion of privacy through *identifiability* and developing computational approaches that guarantees sufficient privacy protection of a dataset. The seminal work by Sweeney, *et al.* shows that a dataset that simply has identifiers removed is subject to linking attacks [62].

Since then, a large body of work contributes to data anonymization that transforms a dataset to meet a privacy principle. These works have proven successful on structured data. These structured techniques do not provide the answer for anonymization or privacy on textual data, which is commonly found in EHI repositories. Chapters 4 through 6 describe the integration of some of these techniques for providing answers to common medical research queries used in heterogeneous medical data repositories.

We classify the privacy principles into weak privacy and strong privacy. Weak privacy refers to the release of a modified version of each record (input perturbation) because these techniques assume a certain level of background knowledge of the attackers, while strong privacy refers to the release

of perturbed statistics (output perturbation) and assumes nothing about the background knowledge of the attackers.

2.3.1 Weak privacy

The weak privacy models assume a reasonable limited background of the attackers. Techniques involving generalization, suppression (removal), permutation and swapping of certain data values so that it does not contain individually identifiable information including determining the presence of absence of an individuals record in a table can be found in [64, 34, 71, 5, 2, 22, 8, 80, 39, 40, 73, 79, 52, 42, 53].

In defining anonymization given a relational table T, the attributes are characterized into three types. Unique identifiers are attributes that identify individuals. Quasi-identifier set is a minimal set of attributes that can be joined with external information to re-identify individual records. We assume that a quasi-identifier is recognized based on the domain knowledge. Sensitive attributes are those attributes that an adversary should not be permitted to uniquely associate their values with a unique identifier.

The k-anonymity model provides an intuitive requirement for privacy in that no individual record should be uniquely identifiable from a group of k with respect to the quasi-identifier set. The set of all tuples in T containing identical values for the quasi-identifier set is referred to as equivalence class. T is k-anonymous if every tuple is in an equivalence class of size at least k. A k-anonymization of T is a transformation or generalization of the data

Table 2.1: Illustration of Anonymization

Name	Age	Gender	Zipcode	Diagnosis
Henry	25	Male	53710	Influenza
Irene	28	Female	53712	Lymphoma
Dan	28	Male	53711	Bronchitis
Erica	26	Female	53712	Influenza

Original Data

Name	Age	Gender	Zipcode	Disease
$*$	$[25-28]$	Male	[53710-53711]	Influenza
$*$	$[25-28]$	Female	53712	Lymphoma
$*$	$[25-28]$	Male	[53710-53711]	Bronchitis
$*$	$[25-28]$	Female	53712	Influenza

Anonymized Data

T such that the transformed dataset is k-anonymous. The l-diversity model provides an extension to k-anonymity and requires that each equivalence class also contains at least l well-represented distinct values for a sensitive attribute to avoid the homogeneous sensitive information revealed for the group. Table 2.3.1 illustrates one possible anonymization of the original table with respect to the quasi-identifier set $(Age, Gender, Zipcode)$ that satisfies 2-anonymity and 2-diversity.

2.3.2 Strong privacy

The weak privacy models assume limited background of the attackers. This may be acceptable in many scenarios (e.g. internal research by universities and hospitals), but for more widespread release of the information it is necessary to only release aggregate views of the data due to privacy concerns. Differential Privacy [19, 16, 17] is the most widely accepted strong privacy notion that

makes no assumptions on the attacker's background knowledge. Differential privacy requires that a randomized computation yields nearly identical output when performed on nearly identical input. The addition or modification of one record in a dataset is considered to be nearly identical input.

Most work on differential privacy has been studied under an interactive model, where the users can continually query the data until the desired level of privacy can no longer be guaranteed [19, 16]. Non-interactive differential privacy has been previously studied in [10, 21, 75].

Large repositories of medical data can be represented as data cubes for faster OLAP queries and learning tasks. Many aggregate datasets are released to the public without considering the privacy implications on those individuals involved. There is always a tradeoff between utility and privacy. Simply removing or replacing identifiers with statistically anonymized values (Chapter 5) does increase the privacy of the individuals in the dataset, but cannot guarantee the privacy of every individual in the dataset, because it is impossible to know the full background knowledge of any attacker. Differential privacy [18, 14] is widely accepted as one of the strongest known unconditional privacy guarantees and is a promising technique for standardizing the privacy practices of health institutions that desire to release data for statistical analysis [50].

This section outlines the various approaches to achieving differential privacy. There are two models for privacy protection [18]: the interactive model and the non-interactive model. In the interactive model, a trusted *curator* (e.g.

hospital) collects data from *record owners* (e.g. patients) and provides an access mechanism for *data users* (e.g. public health researchers) for querying or analysis purposes. The result returned from the access mechanism is perturbed by the mechanism to protect privacy. McSherry implemented the interactive data access mechanism into PINQ[47], a platform providing a programming interface through a SQL-like language, which was used as inspiration for the differentially private query interface in HIDE.

In the non-interactive model, the curator publishes a "sanitized" version of the data (typically in the form of a data cube), simultaneously providing utility for data users and privacy protection for the individuals represented in the data. There are a few works that studied general non-interactive data release with differential privacy. Blum, et al. [9] proved the possibility of non-interactive data release satisfying differential privacy for queries with polynomial VC-dimension, such as predicate queries and also proposed an inefficient algorithm based on the exponential mechanism. A data releasing algorithm for predicate queries using wavelet transforms with differential privacy as developed in [74]. Achieving optimal utility for a given sequence of queries as explored in [41, 33]. A mechanism that reduces error by ensuring consistency of the released differentially cuboids was developed in [13]. Formal definitions of privacy follow.

Definition 1. *A function A gives ϵ-differential privacy if for all neighboring*

data sets D_i and D_j, and all $S \subseteq Range(A)$,

$$\Pr[A(D_i) \in S] \leq exp(\epsilon) \cdot \Pr[A(D_j) \in S]. \tag{2.1}$$

Differential privacy is achieved by perturbing (adding noise to) the original data before release. This noise is a function of the $L1$-sensitivity of a given query.

Definition 2 ([15])**.** *For $f : \mathcal{D} \to \mathbb{R}^d$, the $L1$-sensitivity of f is*

$$S(f) = \max_{D_i, D_j} ||f(D_i) - f(D_j)||_1 \tag{2.2}$$

for all neighboring data sets D_i and D_j.

The symmetric exponential (Laplace) distribution has density function $p(x) \propto \exp(-|x|)$. The Laplace distribution is the most common distribution used as a noise function to achieve differential privacy. (Comment on optimality of Laplace noise)

Theorem 1. *Let X be the true answer for a given query Q. The randomized function $M(X) = |X| + Laplace(\Delta(Q)/\epsilon)$ ensures ϵ-differential privacy for query Q.*

Definition 3 (Error)**.** *A database mechanism A has (ϵ,δ)-error* [2] *for queries*

[2]This is called (ϵ, δ)-usefulness in the literature, but we find it odd that a lower value for ϵ implies higher usefulness.

in class C if with probability δ, for every $Q \in C$, and every database D, $A(D) = \hat{D}$, $|Q(\hat{D}) - Q(D)| \leq \epsilon$.

Theorem 2 ([18])**.** *Let F be a query sequence of length n. The randomized algorithm that takes as input database T then output $F'(T) = F(T) + \text{Lap}(S(F)/\epsilon)^n$ is ϵ-differentially private.*

The $L1$-sensitivity differs according to the type of query being performed on the original data. The focus of this chapter is on data cubes generated from count queries. Therefore, the sensitivity is always 1.

Theorem 3. *Parallel Composition [47] Let M_i be a differentially private query mechanism. Let D_i be arbitrary disjoint subsets of the input domain D. The sequence of $M_i(X \cap D_i)$ provides ϵ-differential privacy.*

Results for strong privacy typically include theoretical guarantees on the utility (or usefulness) of the data release. Definition 4 gives a formal definition of usefulness.

Definition 4. *[10] A database mechanism A is (ϵ, δ)-useful for queries in class C if with probability $1 - \delta$, for every $Q \in C$ and every database D, for $\hat{D} = A(D)$, $|Q(\hat{D}) - Q(D)| \leq \epsilon$.*

Set-valued data is a common format for inclusion in data cubes, e.g. How many patients of both disease A and disease B. Differentially private set-valued data publishing was presented in [11]. A similar method was applied to trajectory data publishing in [12]. Chapter 5 presents an application of the technique for publishing differentially private temporal medical data.

2.4 Discussion

The proposed definitions are accepted as standards in the privacy research community and have yet to be applied or accepted at a national scale for privacy practice in real-world scenarios. Technically, the definitions and techniques discussed in this dissertation have certain levels of privacy guarantees, but there are non-technical hurdles that need to be discussed in order for inclusion in practice. The safe-harbor method of removing identifiers remains the predominant technique for ensuring privacy, even though privacy researchers have shown the danger of assuming such informal techniques ensure privacy.

In any real world system it is necessary to keep a pointer back to the original data without exposing it to the end-users so that in cases of emergency or individuals with appropriate access levels can access the original data. This matter is an engineering and practice concern that is not discussed in detail in this dissertation nor in most privacy literature.

The remaining chapters present the first prototype system that aims to show real world applicability of releasing data with formal privacy guarantees, while easing the burden of honest brokers.

Chapter 3

HIDE Framework

Health Information DE-identification (HIDE) is a software and framework that allows data custodians to release "scrubbed" patient records, weakly-private tables through structured anonymization and strongly-private data cubes through differentially private aggregated statistics of the patients in the datastore. This chapter describes the components in the framework and the relationship between the components.

3.1 Overview

HIDE consists of a number of key integrated components that give an end-to-end privacy solution for heterogeneous data spaces. A data custodian for a medical institution will have access to both structured (SQL), semi-structured (HL7) and unstructured (text) electronic health records (EHRs). The utility

of these records is greatly enhanced by creating a patient-centric view of the data, where we have as complete a medical history of every patient generated from the records in the database as possible. This is useful for patient centric studies, but it is also necessary for guaranteed structured anonymization (Chapter 5). Extracting all personal health information (PHI) for each patient is referred to as health information extraction (HIE). HIE allows the data custodian to build a structured entry for each EHR. This process of gathering all records for an individual is referred to as data linking. After creating this structured patient-centric view of the data, it is then possible to release: the original text with statistically anonymized substitutions in place of the original words, statistically anonymized data tables containing individual records, and differentially private aggregated statistics through data cubes. Figure 3.1 presents an illustration of the framework.

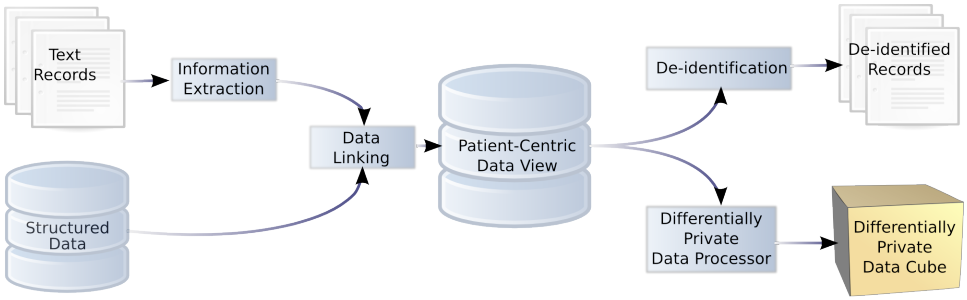


Figure 3.1: Integrated Framework Overview

Given a structured view of the integrated heterogeneous data, the *anonymization* component anonymizes the data using generalization and suppression (removal) techniques with different privacy models. Finally, using the gener-

alized values in the anonymized identifier view, we can remove or replace the identifiers in the original records, or release anonymized tables. The structured identifier view also provides the ability to generate aggregated statistics in the form of data cubes that are useful for determining trends for the population of patients in the datastore.

3.2 Health information extraction

HIDE uses a statistical learning approach, in particular, the Conditional Random Field framework as the basis for extracting identifying and sensitive attributes. HIDE allows data custodians and honest brokers with the ability to train CRF models that can then be used to automatically detect and extract PHI from textual EHRs. Chapter 4 contains more information and experiments using the HIDE PHI extractor.

3.3 Data linking

In relational data it is useful to assume each tuple corresponds to an individual entity. This mapping is not usually present in a heterogeneous data repository. For example, one patient may have multiple pathology and lab reports prepared at different times. In order to preserve privacy for individuals and apply data anonymization in this complex data space, the *data linking* component links relevant attributes (structured attributes or extracted attributes from

unstructured data) to each individual entity and produces a patient-centric representation of the data. The problem of data linkage is very hard, even for humans. FRIL is a probabilistic record linkage tool developed [35] to resolve the potential attribute conflicts and semantic variations to aid in linking records.

A novel aspect of the HIDE framework is that the data linking component and information extraction component form a feedback loop and are carried out in an iterative manner. Once attributes are extracted from unstructured information, they are linked or added to existing or new entities. Once the data are linked, the linked or structured information will in turn be utilized in the extraction component in the next iteration. The final output will be a patient-centric *identifier view* consisting of identifiers, quasi-identifiers, and sensitive attributes. This structured identifier view is also used to generate aggregated statistics in the form of data cubes.

3.4 Privacy models

HIDE allows for multiple data-release options of with varying privacy and utility. A data custodian can simply release all data associate with each patient including both the structured and textual data for each patient. The custodian also has the option of releasing the structured patient-centric identifier table or differentially private aggregated data cubes constructed from the structured view.

3.4.1 Weak privacy through structured anonymization

Once the person-centric identifier view is generated after attribute extraction and data linking it is now possible to use a variety of techniques for de-identifying the data. The text and structured tables can be released by substituting values in place of the original identifiers according to the full, partial techniques specified by HIPAA. This modified text can then be released providing higher levels of privacy for individuals in the dataset. Chapter 5 discusses the query utility of the k-anonymity [64] and its extension l-diversity [45] methods on real world data extracted from Emory pathology reports.

3.4.2 Strong privacy through differentially private data cubes

Differential privacy [18, 14] is widely accepted as one of the strongest known unconditional privacy guarantees and is a promising technique for standardizing the privacy practices of health institutions that desire to release data for statistical analysis [50]. Simply removing identifiers is not enough to protect (by theoretical guarantee) the identity of individuals. The aim is to provide methods that allow for the dissemination of aggregated statistics from datasets of patient health records while preserving the privacy of those individuals in the dataset. Analysis of large health datasets is made possible through creating data cubes (multidimentisonal histograms). HIDE provides a method for generating differentially private data cubes. The resulting data cubes can serve

as a sanitized synopsis of the raw database and, together with an optional synthesized dataset based on the data cubes, are useful to support count queries and other types of Online Analytical Processing (OLAP) queries and learning tasks. Chapter 6 describes the utility and methods of the HIDE DPCube algorithm.

3.5 Heterogeneous Medical Data

A major contribution of HIDE is support for heterogeneous data formats. The main goal was to create a framework and techniques for supporting a wide-variety of data input formats and optimizing algorithms so that a wide variety of medical research could be performed in a privacy-preserving manner.

3.5.1 Formats

Data formats can be categorized generally into three classes: structured, semi-structured, and unstructured.

There is a large amount of structured information in medical data repositories. These sources are commonly used for epidemiological studies. They are also useful because they are typically stored in data warehouses accessible by SQL[1] or other structured query mechanisms. Many data warehouses also provide researches with the ability to perform rapid execution of online analytical processing (OLAP) through data cubes. A data cube contains ag-

[1] `http://www.iso.org/iso/catalogue_detail.htm?csnumber=45498`

gregated statistics, e.g. counts, averages, along the various dimensions in the data cube. The dimensions in the cube are selected from the set of columns in the structured relational data tables.

The expansion of data and the new for sharing information has brought about standards for semi-structured data including XML[2]. In the medical field a standards organization called Health Level Seven International (HL7) has sought to standardize the exchange, integration, sharing, and retrieval of health information to support clinical practice[3]. These data formats allow researchers to more easily query for certain attributes within the text, but the sections of unstructured text still provide valuable information to researchers.

Unstructured data is the most common data format for EHRs. The majority of research interest for privacy in medical records has focused on textual forms such as clinical notes, SOAP (subjective, objective, assessment, patient care plan) notes, radiology and pathology reports.

3.5.2 Datasets used in this dissertation

A variety of medical datasets were used to validate the hypotheses and concepts explored in this dissertation. This section briefly describes those datasets.

[2] `http://www.w3.org/XML/`
[3] `http://www.hl7.org/`

Surveillance, Epidemiology and End Results (SEER) Data

The Surveillance, Epidemiology and End Results (SEER) dataset [1] contains cancer statistics representing approximately 28 percent of the US population. The SEER research data include SEER incidence and population data associated by age, sex, race, year of diagnosis, and geographic areas. Chapter 6 uses the breast cancer section of this dataset to show that privacy-preserving views of this data can still produce useful information.

Emory Winship cancer data

The Emory Winship Cancer dataset contains 100 textual pathology reports we collected in collaboration with Winship Cancer Institute at Emory. In consultation with HIPAA compliance office at Emory, the reports were tagged manually with identifiers including name, date of birth, age, medical record numbers, and account numbers or *other* if the token was not one of the identifying attributes. The tagging process involved initial tagging of a small set of reports, automatic tagging for the rest of the reports with our attribute extraction component using the small training set, and manual retagging or correction for all the reports. Chapters 4 and 5 give evaluations and details of PHI detection and query accuracy on statistically anonymized tables for this dataset, respectively.

i2b2 de-identification challenge data

The i2b2 de-identification challenge data [69] is a gold standard for evaluating medical record de-identification solutions. The i2b2 dataset consists of example pathology reports that have been re-synthesized with fake PHI. The reports are somewhat structured and have sentence structure. The training set consists of 669 reports and the testing set consists of 220 reports. Chapter 4 gives evaluations of PHI detection for this dataset.

PhysioNet nursing notes data

The PhysioNet nursing notes dataset [28] consists of re-synthesized nursing notes that are very sporadic and contain almost no sentence structure. Chapter 4 gives evaluations of PHI detection for this dataset.

Emory electronic medical record (EeMR) prescription data

Hey, what about doctor privacy? Typically privacy research on medical data has focused on patient privacy. In order to show the privacy preserving temporal data publishing protecting doctor privacy, the Emory electronic Medical Record (EeMR) prescription dataset was selected.This dataset contains all the e-prescription information written by doctors at Emory University and Affiliated Hospitals. It also contains demographic information on each doctors including age, sex, and locations of residence over the doctor's entire residency in the hospital system. Chapter 5 explores publishing differentially private data that is useful for temporal queries and includes combining these

temporal sequences with other structured demographic information for more complex queries.

3.6 Software

The HIDE software has been demonstrated in [27, 76]. HIDE is a web-based application that utilizes the latest web-technologies. HIDE is written in Python on top of the Django[4] web application framework. It uses Apache CouchDB[5] as the document storage engine. HIDE provides users (primarily honest brokers and de-identification researchers) with the ability to either manually or automatically label (annotate), de-identify, anonymize, and analyze the data. HIDE provides a web-based annotation interface (javascript) that allows iterative annotation of documents and training of the classifier for detecting PHI. This allows the user to quickly create training sets for the CRF classifier. HIDE uses the CRFSuite [54] package for the underlying CRF implementation. Although the framework allows for the integration of an iterative attribute extraction and data linking components, the data linking component of HIDE is supplied externally by the FRIL[35] tool. The extraction and linking can be made iterative by using the HIDE and FRIL tools iteratively for generating features and building higher accuracy extraction models and linking of patient records. HIDE was integrated into the caTIES[6] de-

[4] `http://www.djangoproject.com/`
[5] `http://couchdb.apache.org/`
[6] `http://caties.cabig.upmc.edu/`

identification pipeline. The software package can be configured to use HIDE as a de-identification option for pathology reports in the caTIES database. HIDE can import data from a variety of sources. The system is currently being implemented and tested in real-world settings by multiple institutions. More details can be found at the HIDE project[7] and code[8] web pages.

3.7 Discussion

The HIDE software provides functionality for giving strong and weak privacy guarantees through the safe-harbor method. The underlying algorithms and classifier training are suitable for including in a larger software package for a larger scale analytics information warehouse. There some remaining issues that should be addressed in the software including access security to the servers, providing linkages to the original data, and potential scaling issues including database access and integration. The underlying CouchDB database in HIDE can scale to provide a large amount of data, but doesn't fit into the standard paradigm of structured schema (SQL) databases. These implementation issues would need to be addressed or handled by another aspect of an analytics software solution while HIDE could be used as a library for dealing with the de-identification and privacy issues in the data.

The next two chapters describe some scenarios and results obtained using the HIDE software for detecting PHI and the effects of applying different

[7] `http://mathcs.emory.edu/hide/`
[8] `http://code.google.com/p/hide-emory`

formal privacy techniques on the utility of the released data. These studies show promise for some fundamental tasks required of honest brokers.

Chapter 4

Health Information Extraction

The de-identification of medical records is of critical importance in any health informatics system in order to facilitate research and sharing of medical records. Information extraction (IE) is defined as the process of automatically extracting structured information from unstructured or semi-structured documents. When applied to patient records it is called health information extraction (HIE). HIE is an active field of research [48].

> *CLINICAL HISTORY: 56 year old female with a history of B-cell lymphoma (Marginal zone, SH-02-22222, 6/22/01). Flow cytometry and molecular diagnostics drawn.*

Figure 4.1: A Sample Pathology Report Section

Figure 4.1 shows a sample pathology report section with personally identifying information such as age and medical record number highlighted. This chapter describes the *Information Extraction* component of HIDE and summarizes some of the work in [24, 26, 25], including a comprehensive study of

the features necessary to extract PHI, accuracy on three representative textual EHR datasets and sampling techniques used to enhance the recall of extraction.

4.1 Modeling PHI detection

Extracting identifiers from textual EHRs can be seen as an application of named entity recognition (NER). NER is the aspect of information extraction that seeks to locate and classify atomic elements in text into predefined categories such as the names of persons, organizations, locations, expressions of time, quantities, monetary values, percentages, etc. The main approaches for NER can be classified into rule-based or statistical (machine learning)-based methods. Rule-based systems can be quite powerful, but they lack the portability necessary for multiple institutions to quickly adopt a software package based on such techniques.

The statistical learning techniques use a list of features (or attributes) to train a classification model that at runtime can classify the terms in new text as either a term of an identifying or non-identifying type. These models typically learn the categories of tokens based on context not simply based on lexicons or rules, but also have the ability to incorporate this information. The most frequently applied techniques use either maximum entropy models (MEMM), hidden Markov models (HMM), support vector machines (SVM), or conditional random fields (CRF). Statistical techniques have the advantage

that they can be ported to other languages, domains or genres of text much more rapidly and require less work overall.

Sequence labeling is the process of labeling each token in a sequence with a label corresponding to features of the token in the sequence. One of the most common examples of sequence labeling is part-of-speech (POS) tagging, where each token in the sequence is labeled with its corresponding part-of-speech. Detecting PHI in medical text is very similar, except that the labels correspond to whether or not the term is (or is part of) a name, date, medical record number (MRN), *etc.* If the term is not PHI, it is labeled with an "O."

> *CLINICAL HISTORY: <age>56</age> year old female with a history of B-cell lymphoma (Marginal zone, <id>SH-02-22222</id>, <date>6/22/01</date>). Flow cytometry and molecular diagnostics drawn.*

Figure 4.2: A Sample Marked Pathology Report Section

Figure 4.2 shows an example pathology report with the PHI surrounded by SGML tags. Our task is to train the computer to label the sequence of tokens in the pathology report with the correct PHI labels corresponding to the tags. In order to predict the correct label for a token it is necessary to build features for each token that can be used to calculate the probability of a label given the set of features. This set of features (corresponding to and including the token) are referred to as a feature vector. This sequence of feature vectors is then used in the machine learning framework for predicting PHI and for training the underlying classifier.

PHI extraction in HIDE consists of *training* and *labeling* phases. In order

Label	Token	ALPHA?	NUMBER?	PREV_WORD	NEXT_WORD	PRE1	SUF1
O	HISTORY	1	0	CLINICAL	56	H	Y
age	56	0	1	HISTORY	year	7	7
O	year	1	0	56	old	y	r
O	old	1	0	year	female	o	d

Table 4.1: Example subset of features in feature vectors generated from marked report section.

for HIDE to automatically label the PHI in the document it must first be trained on how to predict the correct labels. The training phase consists of (1) tokenizing the records in the gold-standard training set, (2) building the feature vector for each token, and (3) constructing a statistical model of the feature vectors corresponding to the known labels. The labeling phase consists of (1) tokenizing the record, (2) building the feature vector for each token, and (3) predicting the correct label sequence given the feature vector sequence.

The Conditional Random Field (CRF) framework [37] was developed for the sequence labeling task. A CRF takes as input a sequence of feature vectors, calculates the probabilities of the various possible labelings (whether it is a particular type of identifying or sensitive attribute) and chooses the one with maximum probability. The probability of a labeling is a function of the feature vectors associated with the tokens. More specifically, a CRF is an undirected graphical model that defines a single log-linear distribution function over label sequences given the observation sequence (feature vector sequence). The CRF is trained by maximizing the log-likelihood of the training data. HIDE uses the CRF framework for learning and automatically detecting PHI in EHRs. The next section describes CRFs in more detail.

4.2 Conditional Random Field background

This section includes background information on the Conditional Random Field framework. This section explains the intuition behind the formulation of CRFs and helps elucidate these concepts through detailed explanations.

4.2.1 Features and Sequence Labeling

Given an observation sequence $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and a set of labels $\mathcal{L}$, the goal in a sequence labeling problem is to assign the correct label sequence $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ where y_i is the label assigned to x_i and each $y_i \in \mathcal{L}$. Each $x_i \in \mathbf{x}$ is usually represented as a vector of features where each feature is either 0 or 1 depending on whether or not that feature is true of the observation sequence at x_i. E.g. each word in the input sequence is associated with a set of feature values. Each row in Table 4.2 shows the features that are calculated for the sequence for each word in the example sentence. The n prev word features are actually represented as more than three features but it is written in this way for compactness. The third row states that the feature corresponding to the 1st previous word being 'think' is true and the feature corresponding to the 1st previous word being 'I' is false. The third column actually represents as many features as there are unique words in the sequence.

word	CAPS	1 prev word	2 prev word	label
I	true	NA	NA	PRP
think	false	I	NA	VBP
it	false	think	I	PRP
's	false	it	think	BES
a	false	's	it	DT
pretty	false	a	's	RB
good	false	pretty	a	JJ
idea	false	good	pretty	NN

Table 4.2: Data representation of part-of-speech tagging as a sequence labeling problem.

4.2.2 From Generative to Discriminative

Hidden Markov Models (HMMs) [57] are often used to perform sequence labeling tasks. An HMM is a finite state automaton with stochastic state transitions and observations. More formally, an HMM in sequence labeling defines a state transition probability for the hidden label sequence $\mathbf{y}$, and an observation probability for the observation sequence $\mathbf{x}$. In our example the POS tags are the label sequence and the words (and features) are the observation sequence. The POS tags are called hidden because we only observe the words sequence and not the POS. The probability of a label sequence $\mathbf{y}$ and an observation sequence $\mathbf{x}$ for an HMM is based on the assumption that the probability of transitioning from one state to another is only based on a history window of previous states and the current observation probability depends only on the hidden state that produced the observation. If the history window is one, i.e. the transition to the current state depends only on the previous state then we have a first-order HMM. If the window is two we have a second-order HMM.

It is possible to have arbitrarily high order for an HMM but the time for training the HMM increases exponentially. Using this notation and assumption a first-order HMM would compute the probability of a label sequence given the observation sequence as

$$p(\mathbf{y}, \mathbf{x}) = p(\mathbf{x}|\mathbf{y})p(\mathbf{y}) = \prod_{i=1}^{n} p(x_i|y_i)p(y_i|y_{i-1}). \tag{4.1}$$

HMMs are a generative (directed graph) model, which means that it defines a joint probability distribution $p(\mathbf{x}, \mathbf{y})$. In order to define a joint distribution the model must enumerate all possible observation sequences. Thus, each observation x_i can only depend on y_i for the inference problem to remain tractable. As a result determining the relationship between multiple interacting features from the observation sequence is not tractable, i.e. HMMs cannot model non-independent or overlapping features since the features for the prior probability $p(x_i|y_i)$ only depend on the current state. It is possible to extend the HMM to a higher order but doing this increases computation time and still doesn't allow for modeling non-independent or overlapping features.

The limitations of generative models invites the question — How can we design a model that doesn't have to make so many independence assumptions? The answer lies in conditional probability. Instead of constructing a model that computes $p(\mathbf{x}, \mathbf{y})$, we can model the conditional probability $p(\mathbf{y}|\mathbf{x})$. We can label the observation sequence $\mathbf{x}$ with the label sequence $\mathbf{y}$ that maximizes the conditional probability $p(\mathbf{y}|\mathbf{x})$. Models that perform this task are called

discriminative models rather than generative models.

Maximum Entropy Markov Models (MEMMs) [46] are well-known discriminative models used in part-of-speech tagging, text segmentation and information extraction. MEMMs are based on the maximum entropy framework where the underlying principle is that the best model for given data is the model that is consistent with the data while making the least amount of assumptions. The best model is the model that has the highest entropy, or equivalently the model that is closest to the uniform distribution. An MEMM is defined similarly to an HMM except that the state transition and observation probabilities are replaced with one function $p(y_i|y_{i-1}, x_i)$ that gives the probability of the current state given the previous state and current observation. In a MEMM the posterior $p(\mathbf{y}|\mathbf{x})$ is computed directly as opposed to the HMM where Bayes' Rule is used and we indirectly compute the posterior as $p(\mathbf{x}|\mathbf{y})p(\mathbf{y})/p(\mathbf{x})$, but in computation we drop the denominator because the denominator is the same for each possible label, i.e. the best sequence labeling is computed as $\text{argmax}_{\mathbf{y}'}\, p(\mathbf{y}'|\mathbf{x}) = \text{argmax}_{\mathbf{y}'}\, p(\mathbf{x}|\mathbf{y}')p(\mathbf{y}')$. By using state observation transition functions we can model transitions in terms of non-independent features of observations of the form $f_j(x, y)$ where each feature is dependent upon the current observation and the current state. These features correspond to the features in Table 4.2. The exponential form for the probability distribution (or transformation function) that has maximum

entropy given an MEMM is

$$p(y_i|y_{i-1}, x_i) = \frac{1}{Z(x_i, y_{i-1})} \exp\left(\sum_j \lambda_j f_j(x_i, y_i)\right). \tag{4.2}$$

where the λ_i are the parameters to be learned and $Z(x_i, y_{i-1})$ is a normalizing factor that ensures that the distribution sum to one across all possible values for y_i, i.e. the previous state y_{i-1} is used in the normalization constant and not represented in the feature vector of x_i for the model. MEMMs define the transition functions locally. We will see in the next section that CRFs use a similar definition except that the CRF defines a single exponential model for the entire sequence of labels given the observation sequence.

4.2.3 Definition

Conditional random fields (CRFs) are a probabilistic framework for labeling and segmenting sequential data. CRFs are discriminative models, i.e. they model the conditional probability $p(\mathbf{y}|\mathbf{x})$ where $\mathbf{x}$ is a sequence of observations and $\mathbf{y}$ is a sequence of labels.

Definition

Assume that $\mathbf{x}$ is a random variable over observations sequences, and $\mathbf{y}$ is a random variable over corresponding label sequences. Let $G = (V, E)$ be a graph such that each $v \in V$ corresponds to each $y_v \in \mathbf{y}$. If each $y_v \in \mathbf{y}$ obeys the Markov property with respect to G, then $(\mathbf{x}, \mathbf{y})$ is a **conditional**

random field. The Markov property is an assumption that the probability of the state associated with vertex $v \in G$ is conditionally independent of all of the vertices that are not neighbors of v given all the neighbors of v, i.e. $p(y_v|\mathbf{x}, y_w, w \neq v) = p(y_v|\mathbf{x}, y_w, w \sim v)$ where $w \sim v$ means w and v are neighbors in G.

In sequence labeling it is natural and useful to assume that the graph G is a chain, i.e. each label is dependent on the previous and next labels. Given that the graph of the label sequence is a tree (a chain is the simplest example of a tree) then the distribution over the label sequence $\mathbf{y}$ given $\mathbf{x}$ has the form

$$p(\mathbf{y}|\mathbf{x}) \propto \exp\left(\sum_{e \in E,k} \lambda_k f_k(e, \mathbf{y}|_e, \mathbf{x}) + \sum_{v \in V,k} \mu_k g_k(v, \mathbf{y}|_v, \mathbf{x})\right) \tag{4.3}$$

where $\mathbf{x}$ is an observation sequence, $\mathbf{y}$ is a label sequence, $\mathbf{y}|_S$ represents the set of components of $\mathbf{y}$ associated with the subgraph $S \subset G$, f_k, g_k are the feature functions, and the λ_k, μ_k are the weights of features f_k, g_k. The features denoted with f_k are related to transitions between states and those with g_k are related to the current observation. E.g. if the word at position x_i is "Computer" in the sequence we may say that the feature "CAPITALIZED" is true. In our notation $g_k(x_i, \mathbf{y}|_{x_i}, \mathbf{x}) = 1$ where g_k is the feature corresponding to capitalized words in the observation sequence. Note that f_k and g_k can be any real valued fixed functions. Figure 4.3 gives a graphical representation of a chain structured CRF where each feature function is dependent upon pairs of adjacent label vertices and the entire observation sequence.

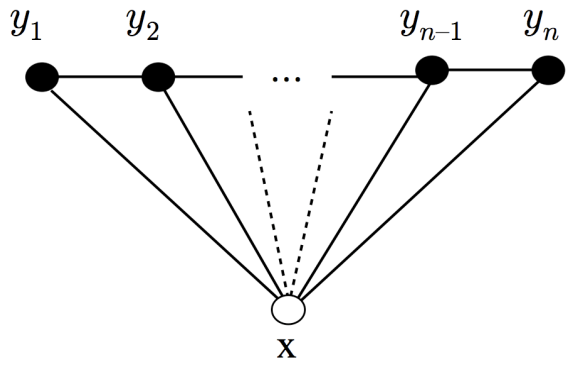


Figure 4.3: A linear-chain CRF where the variables y_i are labels and x_i are observations. Each label state transition function is dependent on the entire observation sequence.

If we ignore the distinction between the f_k and g_k features and let $F_j(\mathbf{y}, \mathbf{x})$ represent the sum of the feature function values for f_j over the entire observation sequence, i.e.

$$F_j(\mathbf{y}, \mathbf{x}) = \sum_{i=1}^{n} f_j(y_i, y_{i-1}, \mathbf{x}, i),$$

we can rewrite (4.3). The probability of given a label sequence $\mathbf{y}$ and an observation sequence $\mathbf{x}$ is

$$p_\lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_j \lambda_j F_j(\mathbf{y}, \mathbf{x})\right) \tag{4.4}$$

where $Z(\mathbf{x})$ is a normalization factor and the λ_j are to be learned by the model. Equations (4.2) and (4.4) are similar. In fact, MEMM and CRFs use very similar training algorithms (see Section 4.2.4).

HMMs, MEMMs and linear-chain CRFs graphical models are similar in structure. Figure 4.4 shows the dependencies of states in HMMs, MEMMs, and

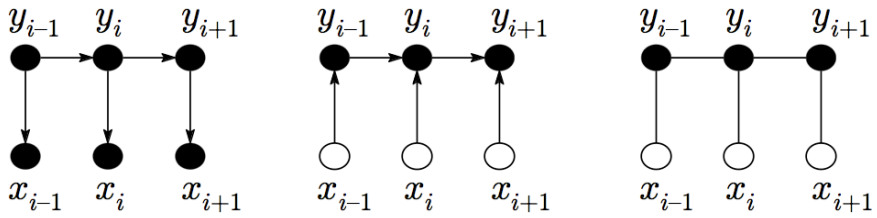


Figure 4.4: Dependency diagrams of states in HMMs (left), MEMMs (center), and a linear-chain CRF. An open circle indicates that the variable is not generated by the model.

CRFs. The edges between states represent the dependencies of the transition functions in the models. A directed edge from node x to y in the graph indicates a one way dependency of node y on x, i.e. the probability of y depends on x. A non-directed edge between x and y indicates that x and y are conditonally independent of all other nodes in the model given the values of x and y and are dependent on one another. Note also that each label node of the CRF in Figure 4.4 is dependent upon the current observation rather than the entire observation sequence. This differs from Figure 4.3. The diagrams are a model of how the feature functions are calculated. If any of the features used in the model are calculated based on the entire training instance then the CRF would have a model similar to that of Figure 4.3. If every feature is calculated based on only the current observation then the CRF would be of the form in Figure 4.4.

CRF Matrix Form

A chain-structured CRF can be expressed in matrix form. We can then use these matrices to efficiently compute the unnormalized probability of a label sequence given an observation sequence. For ease of notation we augment our chain-structured CRF with extra start and stop states with labels y_0 and y_{n+1} respectively. Let $M_i(\mathbf{x})$ be a $|\mathcal{L}| \times |\mathcal{L}|$ matrix with elements

$$M_i(y', y|\mathbf{x}) = \exp\left(\sum_j \lambda_j f_j(y', y, \mathbf{x}, i)\right). \tag{4.5}$$

Each matrix has an entry that represents an unnormalized probability of transferring from label y' to label y given the observation sequence $\mathbf{x}$, i.e. each matrix is the representational equivalent of the exponential transition function in MEMMs. The conditional probability of the label sequence given the parameters is

$$p_\lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{i=1}^{n+1} M_i(y_{i-1}, y_i|\mathbf{x}) \tag{4.6}$$

The normalization constant can be computed from the $M_i(\mathbf{x})$ matrices using closed semi-rings [70] as

$$Z(\mathbf{x}) = \left[\prod_{i=1}^{n+1} M_i(\mathbf{x})\right]_{\text{start,stop}}. \tag{4.7}$$

4.2.4 Parameter Learning

In order to use the CRF model we have constructed, it is necessary to determine the λ parameters from the training data. Assuming there are N i.i.d. training instances of the form $\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}$ which are the observation feature values and associated label for training instance i. We want to find the values of each $\lambda_j \in \lambda$ that maximize the likelihood $p(\{\mathbf{y}^{(i)}\}|\{\mathbf{x}^{(i)}\}, \lambda)$. This can be accomplished by maximizing the log-likelihood

$$\begin{aligned} L(\lambda) &= \sum_{i=1}^{N} \log p_\lambda(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) \\ &= \sum_{i=1}^{N} \left(\log \frac{1}{Z(\mathbf{x}^{(i)})} + \sum_j \lambda_j F_j(\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) \right). \end{aligned} \tag{4.8}$$

This function is concave and guarantees convergence to the global maximum. Setting the gradient of this function to zero and solving does not always yield a closed form solution. Thus, it is necessary to use iterative scaling or gradient-based methods to estimate the values of λ.

Iterative Scaling

Recall from section 4.2.1 that we are considering two types of features functions f_k and g_k. In this section λ_k and μ_k update equations correspond to f_k and g_k features respectively. Iterative scaling algorithms update the weights of the parameter λ_k by $\lambda_k = \lambda_k + \delta\lambda_k$ and μ_k by $\mu_k = \mu_k + \delta\mu_k$. We now discuss a method for learning the parameters based on the improved iterative scaling

(IIS) algorithm in [56]. The IIS update $\delta\lambda_k$ for feature f_k is the solution of the expected value of f_k. That is,

$$\begin{aligned}\tilde{E}[f_k] &= \sum_{\mathbf{x},\mathbf{y}} \tilde{p}(\mathbf{x},\mathbf{y}) \sum_{i=1}^{n+1} f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) \\ &= \sum_{\mathbf{x},\mathbf{y}} \tilde{p}(\mathbf{x}) p(\mathbf{y}|\mathbf{x}) \sum_{i=1}^{n+1} f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) e^{\delta\lambda_k T(\mathbf{x},\mathbf{y})} \end{aligned} \tag{4.9}$$

where $\tilde{p}(\cdot)$ is the empirical distribution of variable $\cdot$ and

$$T(\mathbf{x},\mathbf{y}) = \sum_{i,k} f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) + \sum_{i,k} g_k(v_i, \mathbf{y}|_{v_i}, \mathbf{x})$$

is the total feature count. $\tilde{E}[g_k]$ has a similar form. The solution involves an exponential sum which is intractable for large sequences. Lafferty, *et al.* [36] present an algorithm based on the concept of a *slack feature* as a normalization constant for computing the $\delta\lambda_k$ and $\delta\mu_k$. Let

$$s(\mathbf{x},\mathbf{y}) = S - \sum_i \sum_k f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) - \sum_i \sum_k g_k(v_i, \mathbf{y}|_{v_i}, \mathbf{x}).$$

S is a constant large enough that $s(\mathbf{x}^{(i)}, \mathbf{y}) \geq 0$ for all $\mathbf{y}$ and observation vectors $x^{(i)}$ in the training set. If we set $T(\mathbf{x},\mathbf{y}) = S$ in (4.9), then we can use a dynamic programming method analogous to the forward-backward algorithm

used in HMM inference. The forward vectors are defined as

$$\alpha_0(y|\mathbf{x}) = \begin{cases} 1 & \text{if } y = \text{start} \\ 0 & \text{otherwise} \end{cases}$$

and

$$\alpha_i(\mathbf{x}) = \alpha_{i-1}(\mathbf{x}) M_i(\mathbf{x}).$$

The backward vectors are defined as

$$\beta_{n+1}(y|\mathbf{x}) = \begin{cases} 1 & \text{if } y = \text{stop} \\ 0 & \text{otherwise} \end{cases}$$

and

$$\beta_i(\mathbf{x})^\top = M_{i+1}(\mathbf{x}) \beta_{i+1}(\mathbf{x}).$$

Given the α and β vectors the update equations are

$$\begin{aligned} \delta\lambda_k &= \frac{1}{S} \log \frac{\tilde{E}[f_k]}{E[f_k]} \\ \delta\mu_k &= \frac{1}{S} \log \frac{\tilde{E}[g_k]}{E[g_k]}, \end{aligned}$$

where

$$
\begin{aligned}
E[f_k] &= \sum_{\mathbf{x}} \tilde{p}(\mathbf{x}) \sum_{i=1}^{n+1} \sum_{e_i=(y',y)} f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) \frac{\alpha_{i-1}(y|\mathbf{x}) M_i(y', y|\mathbf{x}) \beta_i(y|\mathbf{x})}{Z(\mathbf{x})} \\
E[g_k] &= \sum_{\mathbf{x}} \tilde{p}(\mathbf{x}) \sum_{i=1}^{n} \sum_{v_i=y} g_k(v_i, \mathbf{y}|_{v_i}, \mathbf{x}) \frac{\alpha_i(y|\mathbf{x}) \beta_i(y|\mathbf{x})}{Z(\mathbf{x})}.
\end{aligned}
$$

In a very similar form to HMMs the marginal probability of label $\mathbf{y}_i = y$ modeled by a linear-chain CRF is given by

$$
p(\mathbf{y}_i = y|\mathbf{x}) = \frac{\alpha_i(y|\mathbf{x}) \beta_i(y|\mathbf{x})}{Z(\mathbf{x})}. \tag{4.10}
$$

An alternative algorithm with slightly faster convergence that is based on a similar idea is discussed in [36]. These iterative scaling algorithms converge quite slowly. It is therefore necessary to utilize numerical optimization techniques for efficient training of CRFs.

L-BFGS

In order to optimize equation (4.8) it is necessary to find the zero of the gradient function

$$
\nabla L(\lambda) = \sum_k \left[F(\mathbf{y}^{(k)}, \mathbf{x}^{(k)}) - E_{p_\lambda(\mathbf{y}|\mathbf{x}^{(k)})}[F(\mathbf{y}, \mathbf{x}^{(k)})] \right]. \tag{4.11}
$$

Limited memory BFGS (L-BFGS) [43] is the de facto way to train a CRF model by optimizing (4.8). L-BFGS is a limited memory quasi-Newton method for

large scale optimization. L-BFGS is a second-order method that estimates the curvature using previous gradients and updates rather than having to compute the inverse of the Hessian. Typically it is necessary to store 3 to 10 pairs of previous gradients and updates to approximate the curvature [58].

4.3 Metrics

Typical metrics for information extraction and sequence labeling experiments include precision (positive predictive value), recall, and the F_1 metrics. True positives (TP) are those PHI which are correctly labeled as PHI, false positives (FP) are those tokens that are labeled as PHI when they should be labeled as "O," true negatives (TN) are those tokens correctly labled as "O" and false negatives (FN) are those tokens that should be labeled as PHI but are marked as "O." Precision (P) or the positive predictive value is defined as the number of correctly labeled identifying attributes over the total number of labeled identifying attributes, or equivalently $P = TP/(TP + FP)$. Recall (R) is defined as the number of correctly labeled identifying attributes over the total number of identifying attributes in the text, equivalently $R = TP/(TP + FN)$. F_1 is defined as the harmonic mean of precision and recall $F_1 = 2(P \cdot R)/(P + R)$. It is worth noting that sensitivity is defined the same as recall and specificity is defined as the number of correctly labeled non-identifying attributes over the total number of non-identifying attributes in the text. It is not useful to report specificity because the non-identifying attributes are

dominating compared to the identifying attributes so specificity will be always close to 100% which is not very informative.

4.4 Feature sets

A key to the CRF classifier is the selection of the feature set. Examples of features of a token include previous word, next word, and things such as capitalization, whether special characters exists, or if the token is a number, etc. The features used in HIDE were largely influenced by suggestions in the executable survey of biomedical NER systems [38]. Table 4.1 shows example feature vectors based on the sample marked report. The features can be categorized into regular expression, affix, dictionary, and context features.

4.4.1 Regular expression features

Regular expression features are those features that are generated by matching regular expressions to the tokens in the text. The value for a given regular expression is active (specifically the value for the feature is set to 1 in the CRF framework) if the token matches the regular expression. These features are useful for detecting medical record numbers and phone numbers. The regular expression features are fairly standard and similar to those in [72]. Table 4.3 contains the list of all regular expression features used in HIDE.

Regular Expression	Name
^[A-Za-z]$	ALPHA
^[A-Z].*$	INITCAPS
^[A-Z][a-z].*$	UPPER-LOWER
^[A-Z]+$	ALLCAPS
^[A-Z][a-z]+[A-Z][A-Za-z]*$	MIXEDCAPS
^[A-Za-z]$	SINGLECHAR
^[0-9]$	SINGLEDIGIT
^[0-9][0-9]$	DOUBLEDIGIT
^[0-9][0-9][0-9]$	TRIPLEDIGIT
^[0-9][0-9][0-9][0-9]$	QUADDIGIT
^[0-9,]+$	NUMBER
[0-9]	HASDIGIT
^.*[0-9].*[A-Za-z].*$	ALPHANUMERIC
^.*[A-Za-z].*[0-9].*$	ALPHANUMERIC
^[0-9]+[A-Za-z]$	NUMBERS_LETTERS
^[A-Za-z]+[0-9]+$	LETTERS_NUMBERS
-	HASDASH
’	HASQUOTE
/	HASSLASH
‘~!@#$%\^&*()\-=_+\[\]{}\|;’:\",./<>?]+$	ISPUNCT
(-|\+)?[0-9,]+(\.[0-9]*)?%?$	REALNUMBER
^-.*	STARTMINUS
^\+.*$	STARTPLUS
^.*%$	ENDPERCENT
^[IVXDLCM]+$	ROMAN
^\s+$	ISSPACE

Table 4.3: List of regular expression features used in HIDE

4.4.2 Affix features

The prefix and suffix of a token are affix features. HIDE uses the prefixes and suffixes of length one, two and three for each token. E.g., if the token is “diagnosis” the affix features of `PRE1_d`, `PRE2_di`, `PRE3_dia`, `SUF1_s`, `SUF2_is`, and `SUF3_sis` would be active. These features can be useful for detecting certain classes of terms that have common prefixes or suffixes, e.g. disease names.

4.4.3 Dictionary features

HIDE can use any number of dictionaries. If a phrase (or token) is encountered that matches any of the entries in the dictionary a feature indicating that each token is contained in the dictionary is added to the feature vector. Suppose that "John" is in a dictionary file called `male_names_unambig`. If "John" occurs in the text, then the feature `IN_male_names_unambig` would be active in the feature vector associated with the token "John." HIDE currently uses all of the dictionaries from the PhysioNet de-identification webpage[1].

4.4.4 Context features

Previous words, next words, and occurrence counts are examples of context features. Sibanda and Uzuner [60] demonstrate that context features are important features for de-identification. HIDE includes the previous and next four tokens, and the number of occurrences of the term scaled by the length of the sequence in each feature vector

4.4.5 Experiments

This section describes the results of PHI extraction experiments conducted on the Emory Winship cancer and i2b2 challenge datasets.

[1] `http://www.physionet.org/physiotools/deid/`

Emory Winship cancer data

The Emory dataset experiments were conducted using 10-fold cross-validation in which the dataset of 100 records was divided into 10 subsets and 9 subsets were used for training and the other was used for testing and it was repeated 10 times (once for each subset). Table 4.4 summarizes the effectiveness of PHI extraction from HIDE on the Emory dataset.

Table 4.4: Effectiveness of PHI Extraction

Overall Accuracy: 0.982			
Label	Prec	Recall	F1
Medical Record Number	1.000	0.988	0.994
Account Number	0.990	1.000	0.995
Age	1.000	0.963	0.981
Date	1.000	1.000	1.000
Name (Begin)	0.970	0.970	0.970
Name (Intermediate)	1.000	0.980	0.990

i2b2 challenge data

Table 4.5 presents results on the i2b2 challenge where 669 documents were used for training and tested against a 220 document holdout test set.

When using the full feature set HIDE PHI extraction was able to achieve precision of 0.967, recall of 0.986 and F-Score of 0.977. This result is slightly better than the Carafe system [72] which reported a f-score of 0.975 when counting only true positives. If the Carafe system uses the feature sets described here, then theoretically it should achieve very similar or equivalent

Overall Accuracy: 0.967			
Label	Prec	Rec	F1
Age	1.0	0.667	0.8
Date (Begin)	0.996	0.999	0.998
Date (Intermediate)	0.998	0.998	0.998
Doctor (Begin)	0.985	0.992	0.988
Doctor (Intermediate)	0.986	0.985	0.985
Hospital (Begin)	0.982	0.981	0.981
Hospital (Intermediate)	0.984	0.949	0.966
ID (Begin)	0.990	0.997	0.994
ID (Intermediate)	0.720	0.981	0.830
Location (Begin)	0.906	0.807	0.853
Location (Intermediate)	0.980	0.787	0.873
Patient (Begin)	1.0	0.959	0.979
Patient (Intermediate)	1.0	0.972	0.986
Phone (Begin)	1.0	0.948	0.973
Phone (Intermediate)	1.0	0.902	0.948

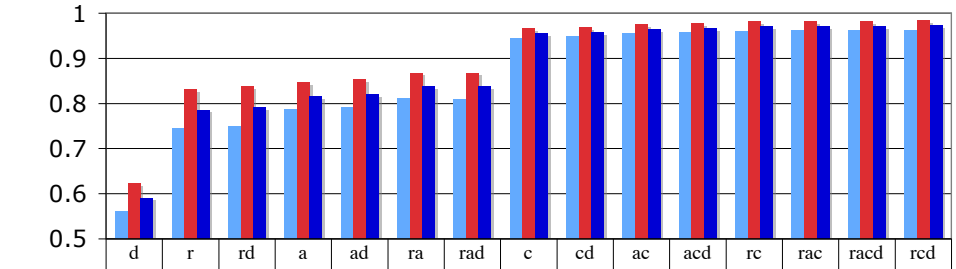
Table 4.5: Results on the i2b2 training and testing challenge data.

results. The most commonly missed PHI are the ID (Intermediate), which correspond to missing the continuation of a medical record number, e.g. detecting <id>1234</id>-123 instead of <id>1234-123</id>. HIDE achieved precision of 0.998, recall of 0.999, and f-score of 0.999 when counting true positives and negatives (without including spaces as tokens) as reported in the standard i2b2 challenge metrics.

Effect of features on PHI extraction

The feature experiments show all subsets of regular expression, affix, dictionary, and context features. Figure 4.5 shows the overall term-level results for all subsets of the features. We calculated the $p - values$ for a paired t-

test against the using all features (racd). The experiments indicate that the most important features for this task in increasing order are: dictionary (p-value $< 10^{-7}$), affix (p-value $< 10^{-6}$), regular expression (p-value $< 10^{-5}$), and context features (p-value $< 10^{-4}$). Using only the context features the classifier achieves f-score of 0.955. The experiments indicated that rcd performed slightly better than racd, but was not statistically significant (p-value > 0.326). The regular expression features are the second most effective. The affix features are third. The least important features were the dictionary features. This is likely due to the fact that many of the terms in the text that are in the dictionaries are not PHI. In practice, it is necessary to have clean dictionaries to ensure meaningful statistics for the features generated by the dictionaries.



	d	r	rd	a	ad	ra	rad	c	cd	ac	acd	rc	rac	racd	rcd
Precision	0.562	0.745	0.749	0.788	0.792	0.811	0.81	0.944	0.948	0.956	0.958	0.961	0.962	0.962	0.963
Recall	0.623	0.832	0.839	0.847	0.853	0.868	0.868	0.967	0.969	0.975	0.977	0.982	0.982	0.982	0.984
F-Score	0.591	0.786	0.792	0.816	0.821	0.838	0.838	0.955	0.958	0.965	0.967	0.971	0.972	0.972	0.973

Figure 4.5: Figure showing dictionary, affix, regular expression, and context features in order of increasing importance.

4.5 Sampling

A comprehensive evaluation was necessary to thoroughly understand the effects of different feature sets and potential impacts of sampling and their tradeoffs between the often conflicting goals of precision (or positive predictive value), recall (or sensitivity), and efficiency. Any medical de-identification system requires high recall of PHI, but the precision must be acceptable. It is possible to detect PHI with high precision in many types of highly unstructured data, but the recall is sometimes low.

The overwhelming number of "O" tags biases the classifier into predicting "O" as the label. A simple technique for removing some of this bias is to remove the number of "O" in the training set. This will increase the recall of most labels at the cost of decreasing precision (positive predictive value). This section describes sampling techniques and variations of cost-proportionate rejection sampling that can be used to increase the accuracy of de-identification using statistical classifiers.

4.5.1 Cost-proportionate sampling

One of the drawback of the CRF classifier is its long training time. Naive Bayes classifiers, on the other hand, are very fast to train and evaluate. As expected the Bayesian classifier performs poorly, especially for the relatively rare types of identifying and sensitive attributes. This is mainly due to the fact that the non-identifying terms (or the terms with *other* class label in our

classification system) comprise more than 99% of the total terms and hence the prior probability for most of the identifying attributes are extremely small. In addition, the classifier missed quite a large portion of directly identifying attributes such as names. This is considered detrimental compared to a classifier that misses a few indirect identifiers such as age or address attributes. In general, different cost (risk of identifying a person) can be associated with failing to de-identify certain individual types attributes.

The above observations motivated the development of a *prioritized* classification approach called cost-proportionate rejection sampling [78]. The basic idea is that random examples from the original dataset (the feature set of all tokens in our case) are chosen and added to the training set based on specified probability for each instance. The probability of being added to the training set is based on the class label of the instance. By assigning different probabilities to different class labels we force our training set to contain more or less instances of particular classes. For example, since missing a name attribute incurs a higher cost or a higher risk of identification of individuals, we assign a higher probability for the name class. As a result, tokens that are tagged as a name, have a higher probability making it into the training set, and hence boost the extraction accuracy for the name attribute.

4.5.2 Random O-sampling

Random O-samping keeps every non-"O" label and selects every "O" label with probability p. The intuition behind this method is a version of cost-

proportionate rejection sampling, except that the order of the training data is always preserved and the non-"O" labels are always selected. This method decreases the number of "O" labels the classifier sees and thus, the classifier will choose the "O" label less often with the overall effect of increasing recall.

4.5.3 Window sampling

In window sampling we keep every non-"O" label and a window of size k around that label. The intuition behind this method is similar to the random O-sampling except that it treats all "O" labeled terms not "near" PHI as noise to the classifier as we are more interested in detecting PHI than non-PHI. The window sampling technique can be quite useful for tweaking the precision and recall sequence labeling classifiers such as CRFs.

4.5.4 Experiments

This section describes shows the results of sampling experiments performed on the Emory Winship cancer, i2b2 challenge, and PhysioNet nursing notes datasets. The results show that HIDE has excellent performance and is tunable to adjust to an honest brokers precision and recall requirements.

Cost-proportionate rejection sampling

The effect of cost-proportionate rejection sampling on a Naive Bayes classifier for PHI extraction is now presented. For cost proportionate sampling,

Table 4.6 shows the probabilities used for each type of attribute. A file with 200,000 examples using the sampling from the original feature file with 106,255 examples was generated for training.

Table 4.6: Probability Values Used in Cost-Proportionate Sampling

Label	Probability
Medical Record Number	.2
Account Number	.2
Age	.3
Date	..5
Name (Begin)	1
Name (Intermediate)	1
Other	.1

Table 4.7 and 4.8 present the extraction results in precision, recall and $F1$ metric for each identifying attribute (class) as well as the overall accuracy without and with rejection sampling, respectively.

Table 4.7: PHI Extraction Accuracy using Naive Bayes

Overall Accuracy: 0.75			
Label	Prec	Recall	F1
Medical Record Number	0.915	0.9627	0.938
Account Number	0	0	0
Age	1	0.5223	0.6802
Date	1	1	1
Name (Begin)	1	0.9746	0.987
Name (Intermediate)	1	0.4053	0.5754

The results from the Naive Bayes with biased rejection sampling are much better than those without the biased rejection sampling. The results for Naive

Table 4.8: PHI Extraction Accuracy using Naive Bayes with Prioritized Sampling

Overall Accuracy: 0.98			
Label	Prec	Recall	F1
Medical Record Number	0.9176	0.9962	0.9552
Account Number	0	0	0
Age	1	0.9924	0.9963
Date	1	1	1
Name (Begin)	1	1	1
Name (Intermediate)	1	1	1

Bayes with biased rejection sampling are comparable or even better than the CRF-based classifier for certain attributes. This is somewhat surprising to considering the simplicity of Bayesian and complexity of the CRF classifier. The good results achieved by the Bayesian method is largely due to the sampling technique and the fairly homogeneous pathology report structure in the dataset.

Random-O sampling

We performed experiments on the i2b2 and PhysioNet datasets with varying probability for random-O sampling. A history size of four surrounding tokens was kept constant in order to retain some context information.

Figures 4.6 and 4.7 show the effects of the random O-sampling with various selection probabilities. When the selection probabillity is small the system is biased toward recall and when p is large the precision and recall begin to converge.

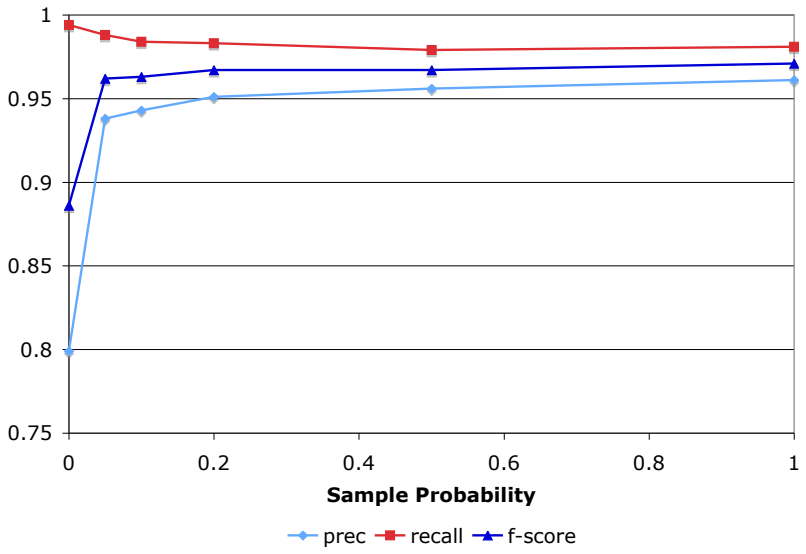


Figure 4.6: Effect of random O-sampling selection probability and a fixed history size of 4 on the i2b2 cross-validation data.

Window sampling

We performed experiments on the i2b2 and PhysioNet datasets with varying history sizes for window sampling. Figures 4.8 and 4.9 show the effects of the window sampling with various selection probabilities. When the history size is small the system is biased toward recall and when history size is large the precision and recall begin to converge.

These results show that by decreasing the window size the classifier can detect all PHI. Neamatullah, *et. al* [51] report precision of 0.967 and recall of 0.749 on the full PhysioNet dataset of 1836 notes. We were only able to import a fraction of these from the site, but we believe our system would have similar results to those we have reported here on the full corpus. At a

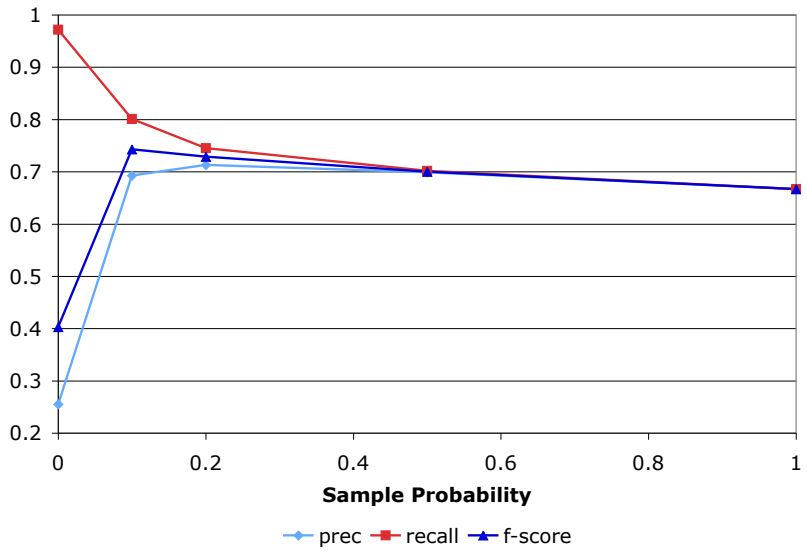


Figure 4.7: Effect of random O-sampling selection probability and a fixed history size of 4 on the PhysioNet cross-validation data

similar level of recall .972 we obtain precision of .255 with a history size of 4. I believe that similar or better performance can be achieved with extremely accurate lexicons and more training data. The contribution is that the window sampling allows users to tweak the system to perform as well as hand tailored rule-based systems for recall.

Figure 4.10 shows relates the window sampling and random sampling techniques on the i2b2 dataset. It was observed that the window sampling has higher recall initially and requires a longer history window in order for precision to increase to a more acceptable level relative to the results from the random O-sampling. The figure was scaled between 0 and .5 indicating the amount of training examples relative to the full dataset and shifted to start at

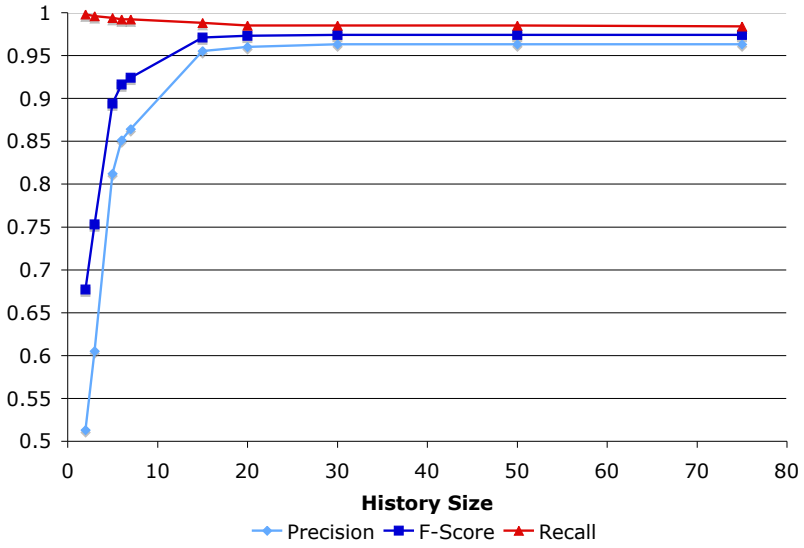


Figure 4.8: Effect of window history size for window filtering on i2b2 cross-validation data. History size of 10 gives a window of 20 tokens.

0.

In general, the CRF approach achieves the best overall result. In particular, it is much better at detecting Account Number, which neither Naive Bayes approach ever detects. The effectiveness of HIDE is largely contributed to the CRF modeling technique and the extensive set of features shown useful for personal health information extraction.

Performance

The HIDE system has integrated the CRFSuite [54], which is one of the fastest CRF implementations. The CRF is trained using the CRFSuite application with the L-BFGS [44] algorithm. The L-BFGS algorithm stops when the log-likelihood on the training data improves by no more than 10^{-5} from the

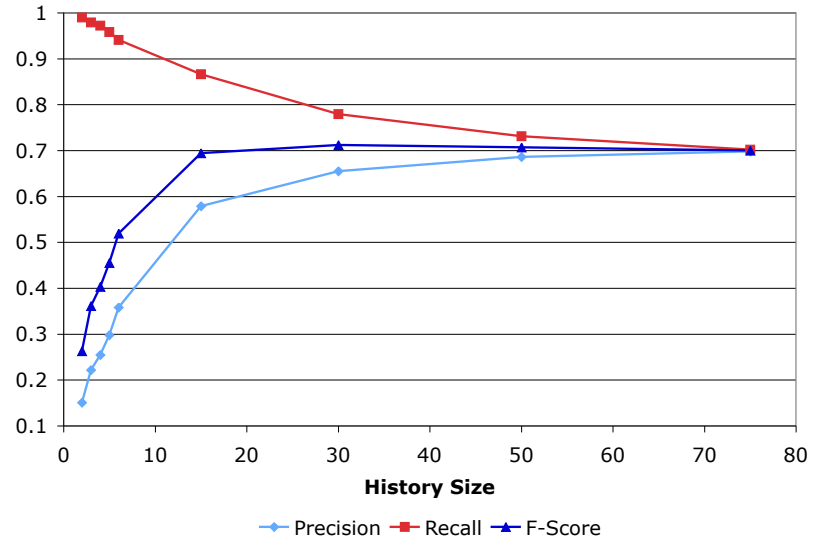


Figure 4.9: Effect of window history size for window filtering on PhysioNet cross-validation data.

previous iteration. The training time for the full i2b2 669 report training set with all features was 51 minutes, 39 seconds.

In order to determine performance, all ten CRF cross-validation training sets for each history size were simultaneously trained. The numbers reported are the average runtimes. The training time to build all ten models for the PhysioNet data was 24 seconds. The training time to build all ten models for the i2b2 cross-validation dataset with no sampling was 12 minutes, 24 seconds (744 seconds).

Figure 4.11 shows the training time vs. window history size training time on the i2b2 dataset. The training time increases with the history size. Setting the correct sampling rate can allow users to optimize HIDE for their different speed, precision, and recall requirements.

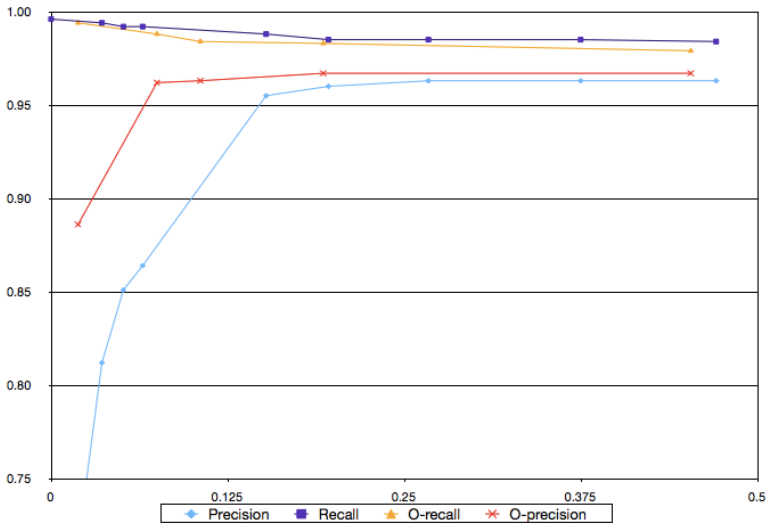


Figure 4.10: Precision and recall curves relating O-sampling to Window sampling on the i2b2 cross-validation data with x-axis scaled to indicate relative amount of training examples.

4.6 Discussion

This chapter described the information extraction component of the HIDE framework and demonstrated that context features are the most important for de-identification as well as shown the effect of a variety of features. We described the window sampling technique for tweaking the time, precision, and recall performance of the system. HIDE has proven to be one of the best systems at PHI detection. Note also that HIDE can extract a much broader set of information than most existing de-identification systems that typically focus on a subset of HIPAA identifiers. By redefining PHI as personal health information and focusing on it's detection, the information extraction

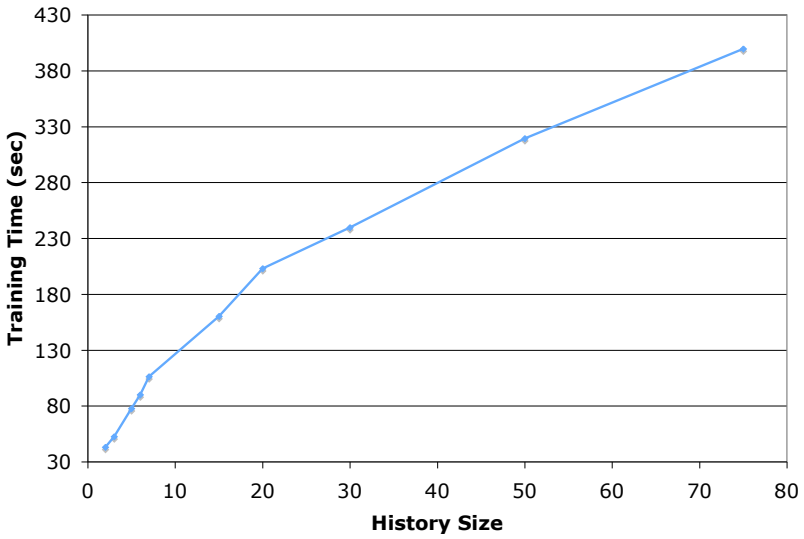


Figure 4.11: Training Time (seconds) vs. History Size on i2b2 dataset

component provides a foundation for privacy preserving data analysis. Using this PHI extraction capability an honest broker can release private views of datasets that include both structured and unstructured records.

The information extraction techniques used by HIDE can be used to provide highly accurate PHI detection. It's still a matter of policy of whether an institution will allow for a completely automated solution. Even if an entirely automated solution is unacceptable by an IRB, these tools can be used by honest brokers to more quickly remove identifying information from the data. The next chapter details the formal privacy preserving techniques used in HIDE for releasing structured views of the underlying text data.

Chapter 5

Privacy-Preserving Publishing

While the research on data anonymization has made great progress, its practical utilization in medical fields lags behind. An overarching complexity of medical data, but often overlooked in data privacy research, is data heterogeneity. HIDE addresses this issue by providing data custodians with the ability to create patient-centric structured data tables. Structured anonymization techniques can then be applied. This chapter discusses the structured anonymization methods providing private record release with weak privacy and multidimensional aggregated statistical data with strong privacy techniques employed in HIDE and presents evaluation results on queries similar to those used by medical researchers.

5.1 Weak privacy

HIDE provides the ability for data custodians to release data where individual records have been modified according to and satisfying the k-anonymization and l-diversity principles discussed in Chapter 2. This option is given to honest brokers who are comfortable with the level of privacy afforded by these techniques. This option is recommended for sharing within medical institutions, but these options are likely not to satisfy the staunchest privacy advocates for general public release.

5.1.1 Mondrian Algorithm

HIDE includes an implementation of the Mondrian algorithm [40] that guarantees k-anonymity and an extended Incognito algorithm that guarantees l-diversity [45]. The Mondrian algorithm uses greedy recursive top-down partitioning of the (multidimensional) quasi-identifier domain space. It recursively chooses the split attribute with the largest normalized range of values, and (for continuous or ordinal attributes) partitions the data around the median value of the split attribute. This process is repeated until no allowable split remains, meaning that a particular region cannot be further divided without violating the anonymity constraint, or constraints imposed by value generalization hierarchies. Algorithm 1 outlines the greedy Mondrian partitioning algorithm.

The Incognito algorithm generates the set of all possible k-anonymous full-

Algorithm 1 Mondrian Algorithm [40]

Require: *partition*: the partition to be split
Ensure: set of partitions satisfying privacy principle

```
Let partitions be a list of partitions
if partition can be split and ensure privacy principle then
    dim ← choose_dimension()
    splitVal ← find_best_split(partition, dim)
    leftPartition ← {t ∈ partition : t.dim ≤ splitVal}
    rightPartition ← {t ∈ partition : t.dim > splitVal}
    add Mondrian(leftPartition) to partitions
    add Mondrian(rightPartition) to partitions
else
    add partition to partitions
end if
return partitions
```

domain generalizations, with an optional tuple suppression threshold. Based on the subset property, the algorithm begins by checking single-attribute subsets of the quasi-identifier, and then iterates, checking k-anonymity and l-diversity with respect to increasingly large subsets. The next section outlines the effect of applying these techniques to real-world data extracted from pathology reports.

5.1.2 Count Queries on Extracted PHI

In many public health and outcome research studies, a key step involves subpopulation identification where researchers may wish to study a certain demographic population, such as males over 50, and learn classification models based on demographic information and clinical symptoms to predict diagnosis.

This section presents query accuracy experiments on 100 ages extracted from the textual pathology reports using the information extraction component in HIDE discussed in Chapter 4. We applied different de-identification options on the original dataset. For full de-identification, all the identifying attributes were removed. For partial de-identification, only direct identifiers, including name and record numbers, were removed, but did not remove indirect ones such as age. For statistical de-identification, we removed the direct identifiers and generalized age attribute using the k-anonymization algorithm. The utility of the anonymized data is evaluated through a set of queries.

To evaluate the effectiveness of different de-identification options, we ran a set of queries for a sub-population selection on the de-identified dataset and measured the query precision defined as % of correct reports being returned. Concretely, we randomly generated 10,000 queries with a selection predicate of the form $age > n$ and $age < n$ to select the corresponding reports (patients). Given a selection predicate $age > 45$, a report with age attribute anonymized to the range [40-50] would also be returned. Thus the query result gives perfect recall but varying precision and we report the query precision below.

Figure 5.1 presents the query precision on the de-identified dataset using different de-identification options with varying k in k-anonymization based statistical de-identification. It can be observed that partial de-identification offers 100% precision as it did not de-identify age attribute. However, such de-identification provides limited privacy protection. On the other hand, full de-identification provides the maximum privacy protection, but suffers a low

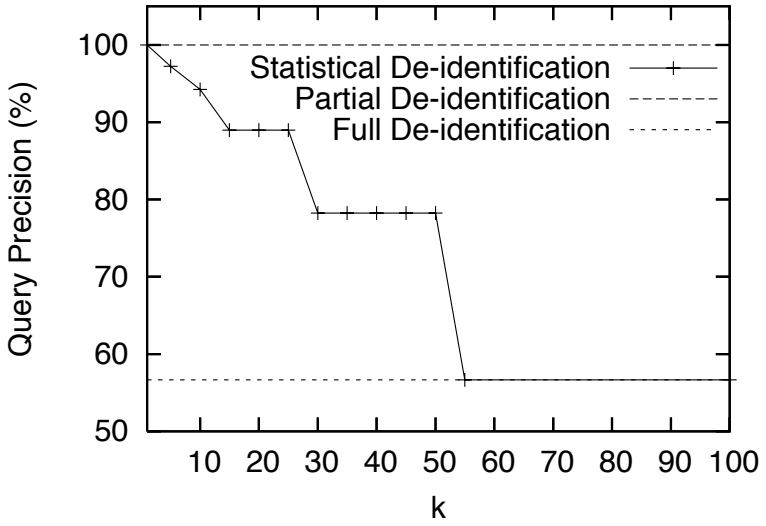


Figure 5.1: Effectiveness of De-Identification

query precision. Statistical de-identification offers a tradeoff that provides a guaranteed privacy level while maximizing the data utility. As expected, the larger the k, the better the privacy level and the lower the query precision as the original data are generalized to a larger extent. Intuitively, the error increases based on an increased ratio between k and the number of records in the dataset.

5.2 Strong privacy

Chapter 2 gave formal definitions of differential privacy. In this chapter we discuss the novel methods used in HIDE for generating differentially private data cubes, and for generating differentially private prefix trees for temporal

data publishing. HIDE implements and utilizes the DPCube algorithm for publishing differentially private data cubes. This section describes data cubes and methods for achieving useful non-interactive differential privacy without a priori knowledge of the set of queries that will be posed by users based on information gain.

5.2.1 Differentially private data cubes

Data cubes are a generalization of the typical two dimensional histogram that can be used to view data from a number of perspectives. A data cube is an n-dimensional abstraction that consists of 2^n cuboids that represent aggregations of counts along chosen dimensions. The base cuboid consists of *cells* that contain the counts of records with the values according to the values along all dimensions. For illustration purposes, let's assume that we have a 3-dimensional data cube with dimensions: age, sex, and ICD-10 diagnosis code. An example cell would represent the number of 32 year old females with diagnosis code of C50[1]. If the number of dimensions of a cuboid is less than n we aggregate the counts along the unchosen dimensions. These aggregations are typically called *slices* of the data cube. If we slice along all values for all dimensions we get the count of the number of records in the dataset. If we slice along age and sex then we aggregate the counts of all the cells with given age and sex, ignoring the value for the diagnosis code. If we slice along no dimensions we end up with the number of records represented by the data

[1] `http://apps.who.int/classifications/apps/icd/icd10online/?gc50.htm+c50`

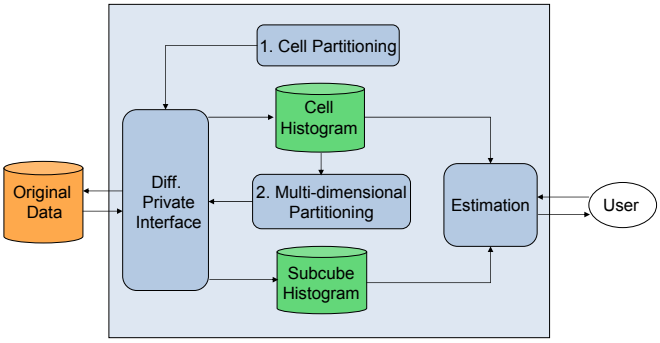


Figure 5.2: Differentially Private Data Cube Release

cube.

It is challenging (in fact, the discovery of the set of optimal partitions is NP-hard [20]) to release a differentially private data cube, due to the exponential number of cuboids and access mechanisms imposed to ensure differential privacy. It remains an open problem to find efficient algorithms for many domains. An overview of the differentially private release mechanism of HIDE is shown in Figure 5.2. These mechanisms show progress toward efficient algorithms for a variety of medical related queries.

The goal is to optimize the utility of the differentially private data presented to the user. It has been shown that given an exact query workload presented by the user, it is possible to estimate the queries necessary for querying the original data to minimize the amount of noise necessary to add to the original data [33, 77].

Algorithm 2 was proposed in [77] as an efficient approximation algorithm for finding useful ϵ-differentially private data cubes. The key component of Algorithm 2 is the multidimensional partitioning Step 3. We want to find the partitioning that maximizes the utility of the released D_p data cube.

For a query issued by users, the estimation component uses the histograms and generates an answer using inference or estimation techniques. An interactive differential privacy interface provided by HIDE is used to provide differentially private access to the raw database. An algorithm submits a sequence of queries to the interface and generates differentially private data cubes of the raw database. The resulting data cubes can serve as a sanitized synopsis of the raw database and, together with an optional synthesized dataset based on the data cubes, are useful to support count queries and other types of Online Analytical Processing (OLAP) queries and learning tasks.

The remaining sections describe DPCube and demonstrate the feasibility and applicability of the approach through an empirical study on real data. Applications and modifications of existing techniques and evaluations on real-world heterogeneous query problems are also presented. The integration of the DPCube algorithm and implementation of the prefix tree approach for temporal data privacy and utility into HIDE gives health professionals the ability to generate summary statistics of guaranteed privacy from heterogeneous data respositories.

5.2.2 DPCube algorithm

In this chapter, we assume to have no real knowledge of the exact queries a user will give and thus our estimation component will heuristically approximate the most useful data cube through greedily determining the partitions based on the information gain using an algorithm similar to decision tree construction. The DPCube algorithm used in HIDE implements and extends the multidimensional partitioning technique in [77]. Specifically, DPCube follows a two-phase approach: Phase 1 creates a differentially private multidimensional partitioning of the data cube based on the information gain. Phase 2 uses this partitioning to compute a differentially private data cube with higher utility. DPCube extends the standard kd-tree based partitioning algorithm by using the information gain on the noisy data to determine the heuristically optimal partitioning. DPCube utilizes the information gain for a particular split point as a heuristic that determines the utility of splitting a data cube into two sub-cubes. The information gain based algorithm described in the next section is generally useful for sparse data cubes that contain "clusters" of points with relatively uniform distributions.

Definition 5. *Let $p(x_i)$ be the probability of a datapoint having value x_i for the dimension of interest and let $I(x_i)$ be the indicator value indicating whether the datapoint has value x_i. The information entropy is defined as* $H(X) = \sum_{i=1}^{n} p(x_i)\, I(x_i) = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i)$

Definition 6. *Let D be a distribution on one dimension of a data cube. Let*

D_1 and D_2 be the distributions after splitting on some value a and dimension. The information gain from splitting on a is defined as $IG(D, a) = H(D_1) + H(D_2) - H(D)$

For a detailed description of entropy, information gain, and decision tree construction, see [32].

The algorithm software has been demonstrated in [76]. DPCube's integration into the HIDE software gives practitioners an entire toolkit of privacy preserving data publishing techniques on both structured and unstructured data.

Algorithm 2 DPCube Algorithm

Require: D : original database; ϵ: the overall privacy budget
Ensure: D_p : differentially private data cube
1: **Partition** the original database into cells using cell partitioning.
2: get **NoisyCount** of each cell using privacy parameter ϵ_1, where $\epsilon_1 < \epsilon$, generating a cell data cube D_c.
3: **Partition** D_c using multidimensional partitioning.
4: **Partition** the original database based on the partition keys returned from step 3.
5: get **NoisyCount** of each partition using privacy parameter $\epsilon - \epsilon_1$ and generate the partition data cube, D_p
6: **return** D_p

DPCube uses an innovative partitioning strategy that seeks to produce close to uniform partitions similar to decision tree construction. It starts from the root node which covers the entire space. At each step, a splitting dimension and a split value from the range of the current partition on that dimension are chosen to divide the space into subspaces. The algorithm repeats until

no splitting choice increases the amount of information gain. In contrast to kd-tree construction which desires a balanced tree, our main goal is to generate partitions that are most useful from an information theoretic perspective. Specifically, the algorithm determines the dimension and value of the dimension to split that gives the highest information gain. If no dimensions in the cube can be split leading to information gain higher than a specified threshold, the cube is no longer split. Algorithm 3 describes the partitioning strategy in pseudo-code. The splitting function is defined in Algorithm 4.

Algorithm 3 DPCube Partitioning Algorithm

Require: *cube*: input data cube; *ig*: information gain threshold;
Ensure: *partitions* : heuristically optimal partitioning based on information gain

```
1: Intialize partitions = ()
2: (lCube, rCube) = SplitCubeOnMaxGain(cube, ig
3: if lCube empty and rCube empty then
4:    return (cube)
5: end if
6: if lCube not empty then
7:    for partition in SplitCubeOnMaxGain(lCube, ig) do
8:       append partition to partitions
9:    end for
10: end if
11: if rCube not empty then
12:    for partition in SplitCubeOnMaxGain(rCube, ig) do
13:       append partition to partitions
14:    end for
15: end if
16: return partitions
```

Theorem 4. *DPCube produces ϵ-differentially private data cubes.*

Algorithm 4 SplitCubeOnMaxGain

Require: *cube*: input data cube; *ig*: information gain threshold;
Ensure: 2-tuple representing left and right subcubes of *cube*

```
1: Determine value, dimension, with highest infogain
2: if infogain > ig then
3:     return (lCube,rCube) where lCube and rCube are partitioned along
       dimension with lCube having values less or equal than value and rCube
       having larger than value
4: else
5:     return (lCube,rCube) where lCube and rCube are both empty
6: end if
```

Proof. The original data is accessed by an ϵ_1-differentially private mechanism in the first phase ($\epsilon_1 < \epsilon$) and by an $(\epsilon - \epsilon_1)$-differentially mechanism. Invoking Theorem 3 shows DPCube is ϵ-differentially private. □

Health care researchers ask a variety of query types in order to answer important health questions. These queries are usually of the form of counts over given predicates, queries ranging from counts and histogram generation to complex queries over large aggregated data cubes and temporal trend queries.

5.2.3 Temporal queries

The DPCube approach allows researchers to understand general trends over the population of the data, but doesn't easily support count queries with predicates that require temporal ranges or trends for specific individuals. For example, suppose a researcher poses the question "How many doctors eRx writing increased over there tenure?" This could be approximated by asking "What is the trend of eRx over time?" It is impossible to know whether the

same doctors at time point 0 are contributing to time point t. In order to support aggregations over temporal range queries scoped on a specific set of individuals, the techniques described in [11, 12] were applied to generate more utility from differentially private release of temporal data. The algorithm builds a differentially private prefix tree which gives three possible options for release: 1) the tree can be released and data users can perform queries on the tree, 2) when a system receives a query for temporal trends it can answer the query using the prefix tree, or 3) the tree can be used to regenerate a sanitized dataset generated from the prefix tree. Algorithm 5 shows the differentially private prefix tree algorithm.[2]

Algorithm 5 BuildDifferentiallyPrivatePrefixTree

Require: D : original temporal dataset; L : set of possible values for entries in D; ϵ: privacy budget; $height$: maximum depth of generated prefix tree;
Ensure: $DPTree$: differentially private prefix tree
1: Create an empty prefix tree PT
2: Add all data from D into PT
3: Let $DPTree = noisyTree(root(PT), L, \epsilon, height, 0)$
4: **return** DPTree

Algorithm 6 shows the recursive algorithm for adding noise to a prefix tree resulting in a differentially private data structure.

Sharing the privacy budget between the temporal release and data cube release allows for the publishers to maintain differential privacy.

[2]I modified the initial definition to be a recursive method for ease of implementation and to extend the method to use heuristic techniques discussed in Chapter 6.

Algorithm 6 noisyTree: Build Differentially Private Prefix Tree

Require: N : currentNode; L : set of possible values for entries in D; ϵ: total privacy budget; *height* : maximum depth of generated prefix tree; *depth* : current depth in tree

Ensure: tree modified with noise

1: **if** $depth <= height$ **then**
2: $budgetToUse = \epsilon / height$
3: $threshold = 2 * \sqrt{2} / budgetToUse$
4: $realCount = N.count$
5: $noisyCount = \lfloor realCount + laplace(1/budgetToUse) \rfloor$
6: $N.count = noisyCount$
7: **if** $N.count >= threshold$ **then**
8: **for** $c \in L$ **do**
9: **if** c is not child of N **then**
10: add c as child of N
11: **end if**
12: $noisyTree(N.child(c), L, epsilon, height, depth + 1)$
13: **end for**
14: **else**
15: **for** $c \in L$ **do**
16: remove c if child of N
17: **end for**
18: **end if**
19: **else**
20: remove all children of N
21: **end if**

5.3 Evaluations

We performed a variety of experiments addressing a variety of heterogeneous queries on a variety of datasets. It is common to use aggregated population statistics to determine mortality rates over ranges of time and also for determining if there is a large enough proportion of individuals fitting specific criteria for clinical trials. This section demonstrates the feasibility and utility of differentially private data cubes for such studies. Empirical results and figures are presented that show the benefit of the DPCube approach.

All of the privacy preserving algorithms are tested and compared based on the Surveillance, Epidemiology and End Results (SEER) [1] breast cancer dataset. The following dimensions (and cardinality) of the dataset were chosen to generate the data cubes: sex (2), age (130), diagnosis year (36), behavior code (2), lab confirmation (9), death code (2), other death code (2). After filtering out patients with unknown data, the dataset contains 22,174 breast cancer patient records between 1973 and 2008.

The year of diagnosis and age at diagnosis including the death status were sliced from the original data cube and serve as a basis for analysis. Figures 5.3 and 5.4 show the original histograms sliced from the higher dimensional data cube.[3] The goal is to release a data cube that when sliced will generate histograms closest to the original while giving a guaranteed level of privacy.

[3]All figures in this chapter use blue to indicate other cause of death, and green to indicate death as a result of cancer.

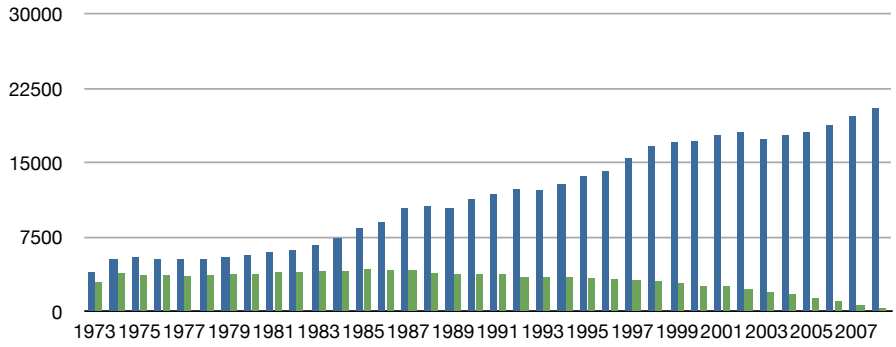


Figure 5.3: Original histogram from (year of diagnosis, death) sliced from original datacube.

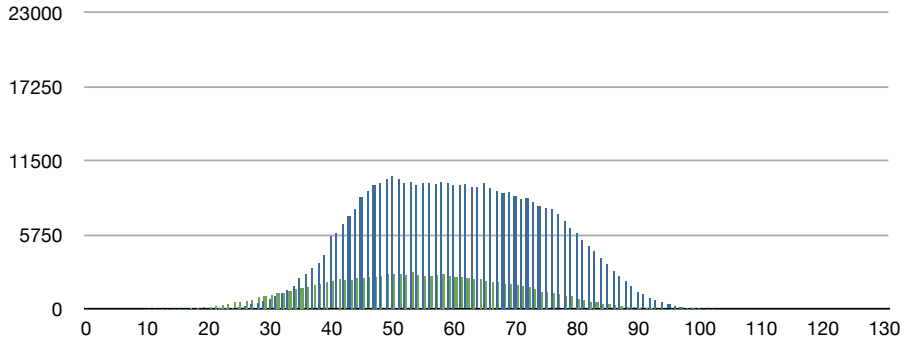


Figure 5.4: Original histogram from (age at diagnosis, death) sliced from original datacube.

5.3.1 Distribution accuracy

This section show the distribution of counts from after applying differentially private algorithms compared with the originals. It is demonstrated that DPCube outperforms the cell-based baseline on this high-dimensional dataset.

Baseline

Our baseline algorithm is simply adding noise to every cell of the data cube according to the Laplacian distribution with the appropriate privacy parameter. Decreasing the privacy budget has the effect of adding more noise, hence more privacy for individuals involved. In general, the less privacy budget (hence, more noise) cause the distributions to become closer to uniform. It is a task of regulators and honest brokers to determine what is an acceptable level of utility for the tradeoff between privacy and utility.

Figures 5.5 and 5.6 show the effect of decreasing the privacy budget. Decreasing the privacy budget shows that the distribution is moving closer to uniform and hence less utility and more privacy.

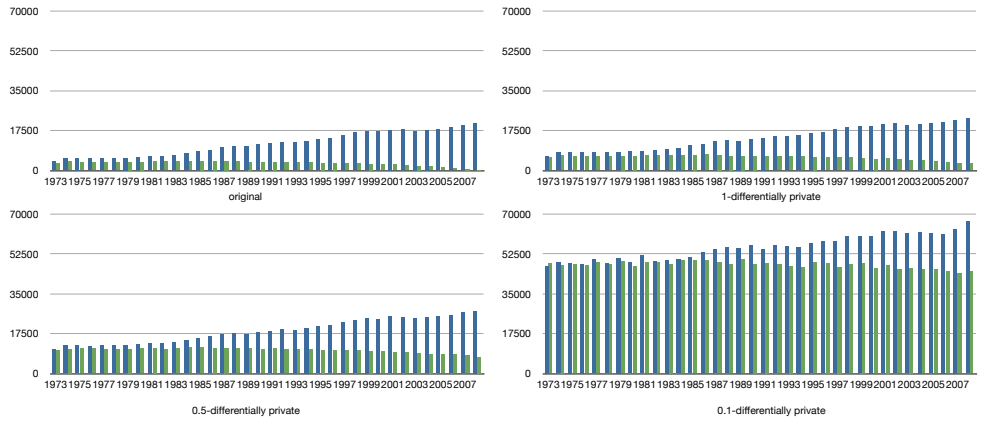


Figure 5.5: Differentially private histograms from (year of diagnosis, death) sliced from privacy preserving datacube produced by adding noise to every cell.

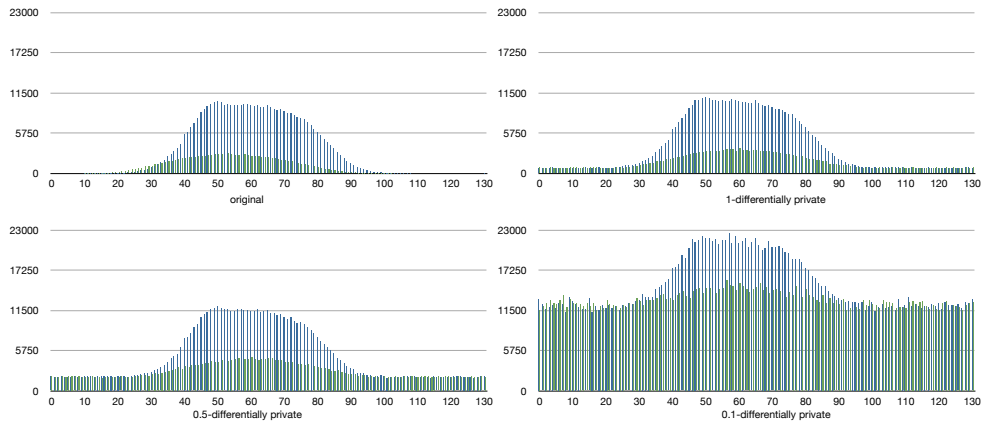


Figure 5.6: Differentially private histograms from (age at diagnosis, death) sliced from privacy preserving datacube produced by adding noise to every cell.

DPCube

DPCube produces distributions closer to the original than the baseline cell based approach. Slices over (age, deathcode) and (diagnosis year, deathcode) where taken from the data cubes to verify this hypothesis. In all of our DPCube experiments we set $\epsilon_1 = \epsilon/2$, i.e. the original cell histogram is $\epsilon/2$-private. We then partition the data according to Algorithm 3. We then generate the final data cube by querying the partitions with $\epsilon/2$ as the privacy parameter. The resulting data cubes are ϵ-differentially private.

The DPCube approach results in errors between 2,227 and 3056 while the baseline gives errors between 6,787 and 7,447 for individual queries on a histogram created by querying for counts of those individuals diagnosed between years 1973 to 2008. Figures 5.7 and 5.8 shows the distribution of counts on the ϵ-differentially private released data cubes. Is is shown that the histograms

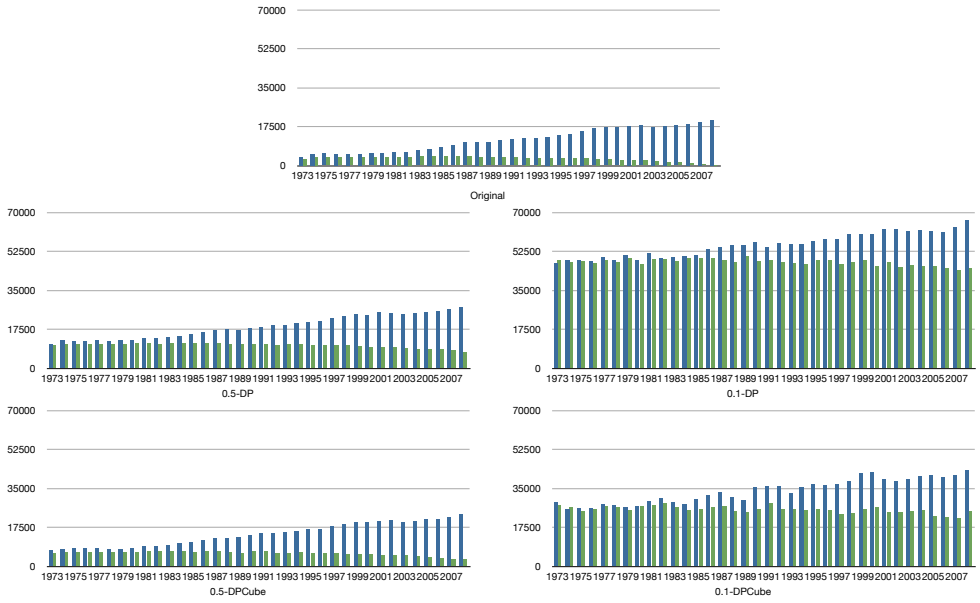


Figure 5.7: Histograms of death cause after cancer diagnosis relative to year of diagnosis showing effect of privacy parameter and algorithm.

provided by DPCube are closer than the baseline algorithm. The histograms show both those individuals who died as a result of cancer and those who either are still living or died for other reasons.

Figure 5.9 shows that the DPCube algorithm results in less error than the baseline approach. These results show that DPCube produces distributions closer to the original with the same level of privacy as the standard cell-based approach.

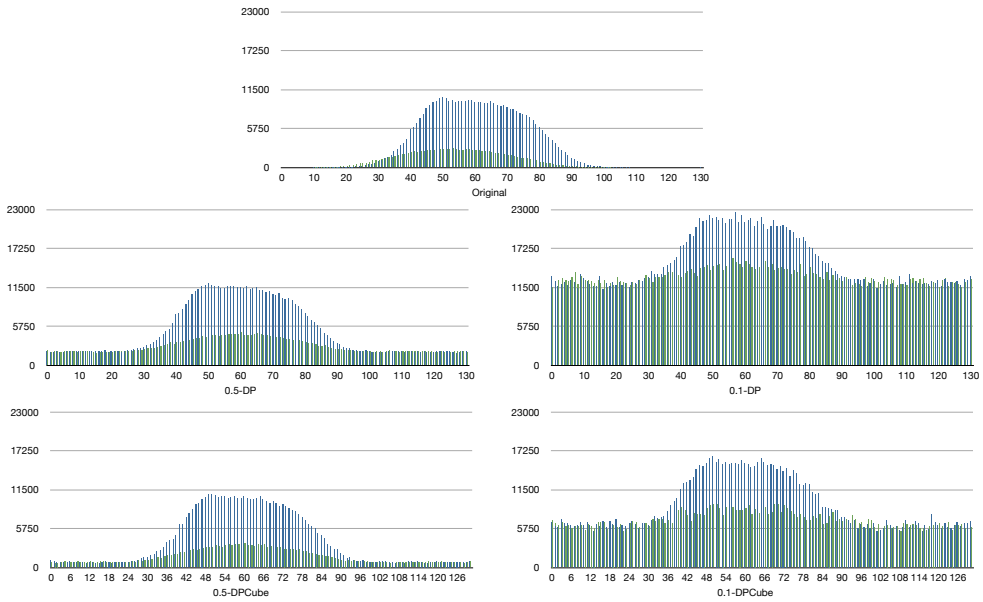


Figure 5.8: Histograms of death cause after cancer diagnosis relative to year of diagnosis showing effect of privacy parameter and algorithm.

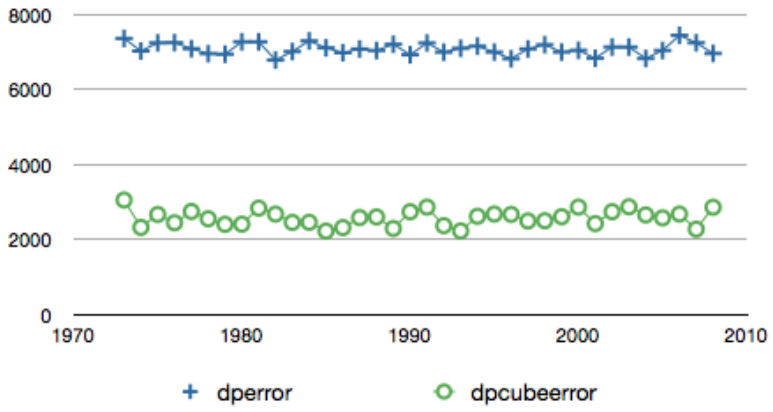
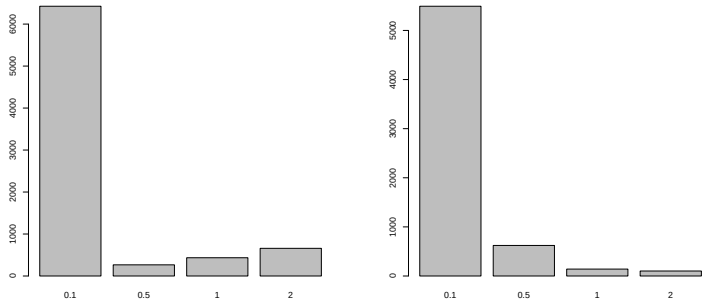


Figure 5.9: Count error for queries over 0.5-differentially private data cube with each year of diagnosis for individuals that died as a result of breast cancer.

5.3.2 Information gain threshold

For those datasets with lots of noise, it is apparent that drilling further gives better results. For those datasets with little noise, drilling deeper gives poorer results due to the fact that more subcubes are generated, and in the final phase more noise is added to the final histogram. For datasets with little noise, it makes sense to add more noise in phase one and less in phase 2. Figure 5.10 indicates this phenomena. Histograms on this data are included to make this more clear. This phenomena deserves further investigation. The figures indicate the nature of such heuristic algorithms and show the data dependence of such techniques.



(a) (b)

Figure 5.10: Average error vs epsilon on (death,age) subcube with information gain threshold of 1 (a) and 0.1 (b).

5.3.3 Trend accuracy

This next experiment shows that the DPCube generated data cubes preserve the trends of the original data better than the cell based approach. A metric similar to AAPC[4] was calculated to see the effect on trend analysis after adding noise according to the DPCube algorithm. The average slope (AS) where, slope is $b_i = (y_i - y_{i-1})/(x_i - x_{i-1})$ and $AS(data) = \sum b_i/|data|$ is compared on the original and differentially private histograms. The AS on the data in Figure 5.11 are -76.14 (original), -87.6 (baseline), and -81.63 (DPCube). Not only are the DPCube counts closer to the original, the algorithm gives closer results to the original average slope statistics.

The cell-based errors for this slice are always over-predictions, because negative values are not included in cells. Any negative values get set to 0 because having negative counts in a histogram doesn't make sense and we don't want the user to have to handle negative values returned from the histograms (most software dealing with counts probably do not elegantly handle negatives). Because this is a sparse dataset the DPCube algorithm partitions out the large blocks of cells with nearly zero count and averages this error across all cells. These experiments shows that the information gain based approach gives both better count and trend accuracy for this specific type of query on a sparse data cube.

[4] `http://surveillance.cancer.gov/joinpoint/aapc.html`

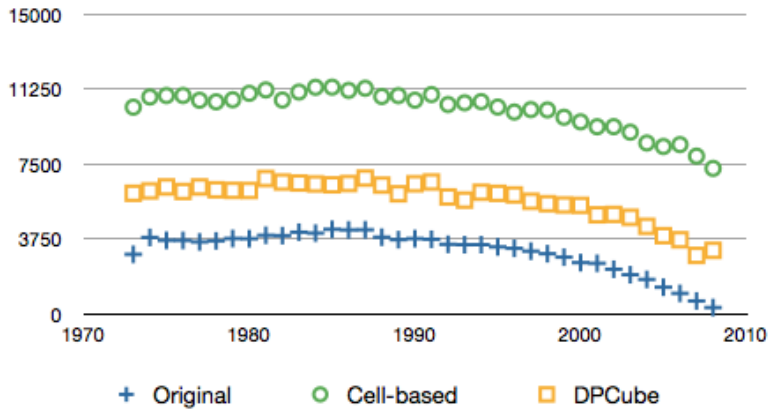


Figure 5.11: Plots showing numbers of deaths caused by cancer relative to year of diagnosis on original data, 0.5-differentially private cell-based (baseline) algorithm, and 0.5-differentially private histogram based on DPCube.

5.3.4 Temporal queries

Temporal query support in HIDE was evaluated using the EeMR e-prescription dataset. The results show that a non-interactive dataset can be released that supports complex queries involving aggregations of demographic and temporal information from the data. The temporal data contained the the National Provider ID (NPI) and average number of medications per patient seen for each each month for each doctor. Many doctors started at different times, therefore the data was normalized so that each doctor stared at month one. This preprocessing allows for the detection of trends for the counts of eRx writing for doctors in residence. Doctors with less than 9 months of residency were also removed from the dataset. After filtering, the dataset consisted of 517 doctor temporal sequences. The data was smoothed into "quarters" where

we took the average over three months spans for each doctor. We randomly augmented the data by sampling with replacement 10,000 entries in order to get a large enough dataset to apply differential privacy principles. The data was normalized to indicate doctors who averaged zero, low (0 to 3), medium (3 to 6), or high (6 and higher) medication counts.

Figure 5.12 shows the trends of a random selection of four doctors in the dataset. Most doctors tend to write on average more eRx per patient over time, but some trend downward or exhibit "zig-zag" patterns. We performed experiments seeing if demographic information could be used to cluster doctors by trend, but without success. This led us to believe that demographic information alone is not a good indicator of trend.

Even though we were unable to classify or cluster trends with doctor demographics we were able to see clear trends in the temporal data. The goal is to provide differentially private release of the data that still preservers the ability to perform trend queries in aggregate. The utility of these trends can be evaluated by measuring the error for temporal queries of varying length. Prefix trees are useful for determining counts of datasets satisfying a series of values. E.g. An example temporal query of length three would be "How many doctors averaged 4 prescriptions the first month, 6 the second, and 8 the third?"

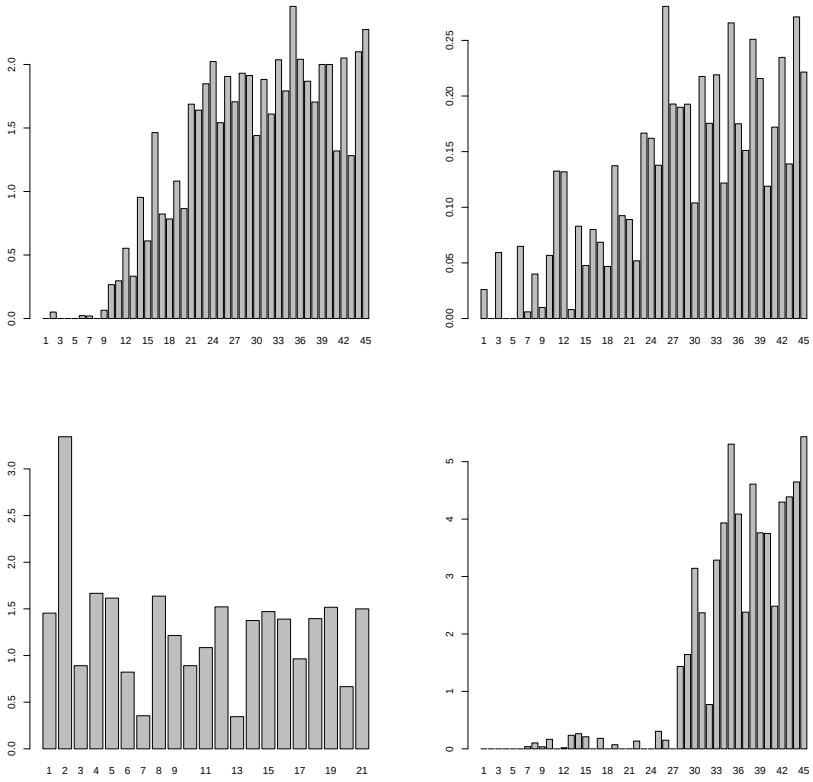


Figure 5.12: Random selection of doctors where x value is month of residence and y value is the average number of eRx prescribed.

5.3.5 Applying DPCube to temporal data

This experiment was conducted using the augmented EeMR dataset, where each row in the original data consisted of seven attributes corresponding to first seven months of residence in the Emory Hospital system. DPCube was applied to the aggregations over this data. The error was significantly worse than the standard cell-based approach. The standard cell based approach shows L_1 error of 12,616 while DPCube shows L_1 error of 106,235. If a single

large cell value randomly in the data is not partitioned by itself, that count get's distributed throughout all the cells within the partition. This can lead to large errors from a single partitioning error, i.e. the major errors come when a partition is selected where one cell has an extremely large count over cells around it. These results show that the DPCube approach is ill suited for datasets with extremely skewed local distributions. Taking the average count and distributing this count in surrounding cells causes the majority of error.

5.3.6 Applying tree-based approach to temporal data

The prefix tree based approach was implemented in HIDE to support queries over temporal trends. Let L be the length of a trend query and let D be the cardinality of domain of values. The next experiments show the average errors for queries that ask about the relative increase in doctor eRx writing. Due to the high error from DPCube and the exponential number of cells $L \cdot D^L$ that must be in the cube to represent temporal trends, HIDE has integrated and adapted the prefix tree based approach from [12] to support temporal range queries.

One measure for determining the accuracy of a differentially private temporal data release is to look at the average error for temporal queries of given lengths. The number of queries of length L on a domain of size D is D^L. Figure 5.13 shows the amount of error per query over lengths 1 through 7. The error follows an exponential pattern. This is due to the large amount of branches that are pruned due to too low of support thresholds. The recur-

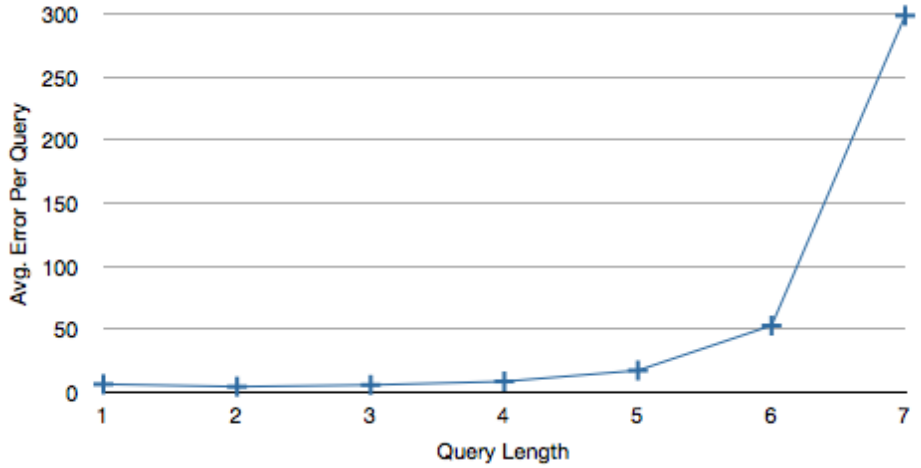


Figure 5.13: Average query error per query length with threshold of 1 standard deviation from 0

sive approach limits the total amount of noise that must be added over all queries as opposed to taking a count over every possible query and adding noise, which would require adding more noise and also compounding use of the privacy budget.

Figures 5.14 and 5.15 show the average number of counts returned and relative error for queries of given length, respectively. As expected, the average count decreases over query length.

A more interesting experiment is to determine the amount of doctors who tend to write more or less eRx over their residence. More specifically, "How many doctors have negative [-1.5, -0.5), zero [-0.5, 0.5), or positive [0.5, 1.5) slopes over their residence?" Figure 5.16 shows the average error of queries of doctors with relatively constant eRx writing through doctors with substan-

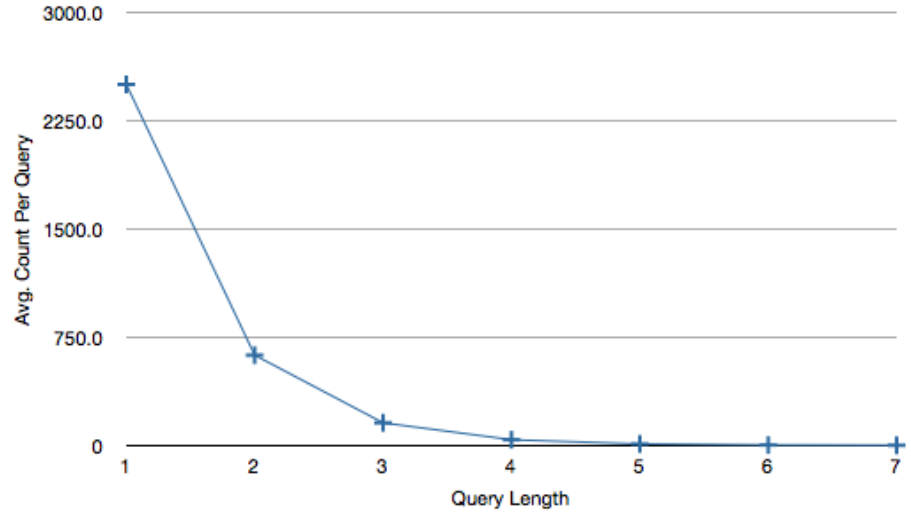


Figure 5.14: Average count of doctors vs query length

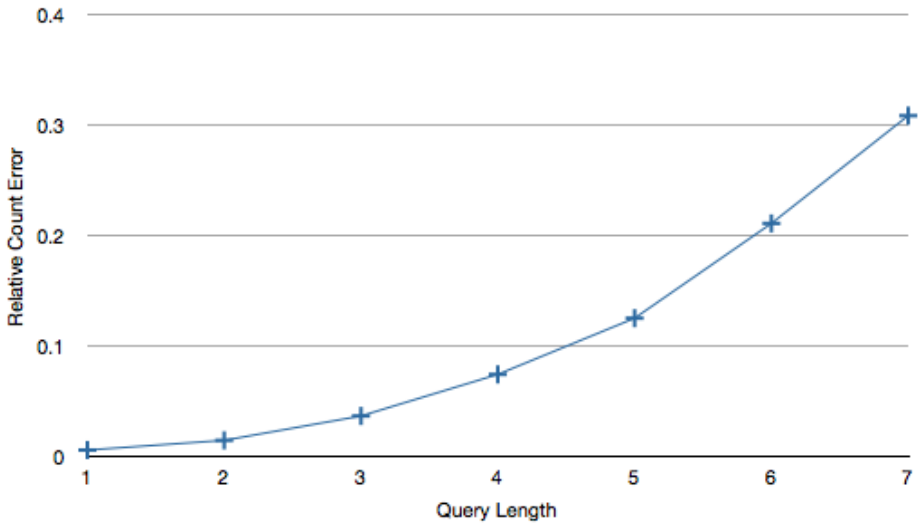


Figure 5.15: Relative error of temporal queries vs query length

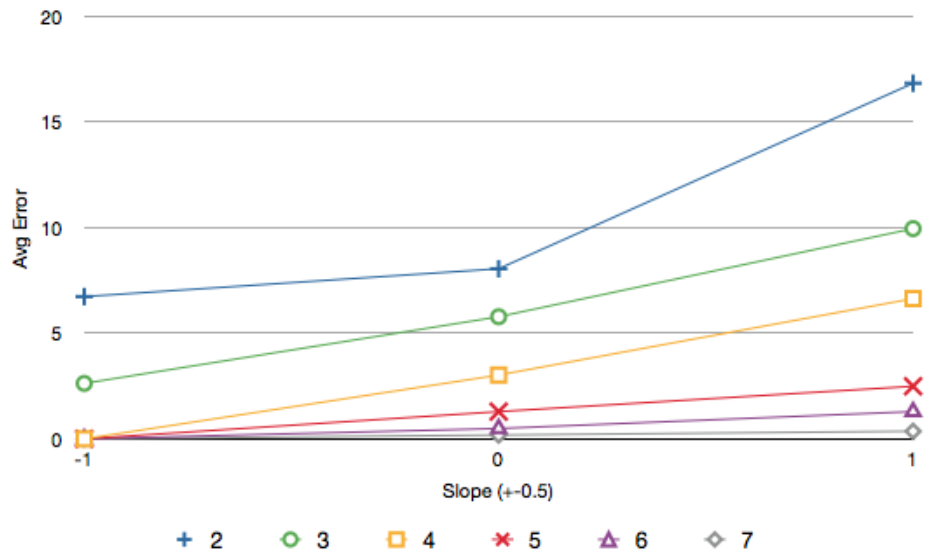


Figure 5.16: Average error of temporal queries vs slope for each query length that can satisfy the slope constraint.

tially increased eRx writing (high slope).

These results show that the slope statistics on this dataset are useful and can be released in a differentially private manner. These experiments have shown differentially private publishing of temporal data can be achieved with relatively low error. The next chapter discusses various extensions of the work presented in this chapter including using heuristics for reducing cascading error and saving the privacy budget for decisions that can't be easily inferred from the data.

5.4 Discussion

This chapter has described various experiments on real-world and augmented datasets. The algorithms discussed can be used to support a variety of privacy preserving data publishing options supporting a wide-range of desired queries for any medical data system.

In practice a data custodian would specify an initial privacy budget based on how sensitive the information for release is and what the desired level of privacy. Each of the strong private options would then share from this budget. For those parts of the system that are covered by the composition theorem each release would subtract from the privacy budget. If a data custodian wanted to release the EeMR Rx dataset demographics with support for temporal data queries, she would use half the budget to generate a data cube using DPCube and the other half releasing the temporal trends using the prefix tree approach or some similar allocation of the overall privacy budget.

Chapter 6

Conclusion and Future Work

This dissertation has presented the HIDE framework and system that provides an end-to-end solution for data custodians to generate a variety of privacy preserving medical data publishing options. We covered existing de-identification approaches, information extraction, named entity recognition, weak privacy methods, including k-anonymization and l-diversity and strong privacy through differential privacy. The implementation and research of HIDE advanced the applied knowledge of each of these fields through the study of the techniques on real-world and academic datasets. HIDE has also advanced the state-of-the-art on releasing differentially private data cubes. The work is a convincing proof-of-concept, yet there are several aspects that should be further explored. This final chapter describes the future work for HIDE including integration, proposal of an extension to the temporal sequence publishing algorithms discussed in Chapter 5, followed by a conclusion.

6.1 Integration

Incorporation of HIDE into a large analytics warehouse would allow for a comparison of prediction capabilities using both the private and non-private data. I believe that many studies can still be validated on the privacy preserving released data. A real-world study showing that combining both the structured and unstructured information contained in data warehouse and combined into a patient-centric view and then released in a privacy preserving manner would be of great utility to medical researchers and validate the work of privacy practitioners alike.

HIDE currently uses an external tool Fril [35] for doing the linking over the records in the dataset. Integrating the data linking and information extraction software pieces would greatly reduce the burden of data custodians and aid researchers in doing patient-centric studies.

I hope that our initial experiments and integration into into the Cancer Biomedical Informatics Grid (caBIG)[1] will lead to more engineering and research effort for building a production quality system.

6.2 Extension of prefix tree approach

This section describes proposed extensions to the prefix tree approach that I believe have promise for reducing error in differentially private trending analysis. Chapter 5 gave extensive studies of applying various approaches to the

[1]Cancer Biomedical Informatics Grid. https://cabig.nci.nih.gov/

differentially private data publishing of medical data. The DPCube approach has shown greater utility over various approaches but hasn't shown great improvement on sequential temporal data. This lead to using the prefix-tree based approach to the privacy-preserving temporal trend publishing task. The initial prefix tree approach has shown useful, but further investigation of the approach is warranted. We propose the following modification of the algorithm that hasn't yet shown great improvement on the EeMR data, but there are definitely datasets that the approach should show improvement.

I believe that for certain classes of data the following extension to the prefix tree approach will prove useful. Consider adding in a predictive component to the algorithm that attempts to correct the noisy counts and also be used to preserve the privacy budget for lower support paths in the tree. The model could only have knowledge of noisy counts to maintain the differential privacy requirement, but given that we can see relatively low error for certain trend queries on the noisy data means that it should be possible to predict the counts as we traverse the tree. In a high confidence prediction it would then be possible to use none or low amounts of the privacy budget and save this budget for use deeper in the recursion down the tree.

I performed some preliminary experiments with a Bayesian like classifier and a linear regression predictor, intuitively if a path has counts 0, 1, 2, 3 in sequence has very high support, then it's likely that the next most likely prediction will be 4. I have not seen an improvement (although not a decrease either) on the augmented EeMR eRx dataset in comparison to the standard

prefix tree approach at this point. The idea can likely be proven useful by determining on what types of paths it makes sense to make the predictions. Further research should help determine what is an appropriate way to compute confidence and what threshold of confidence warrants preserving the budget and trusting the prediction.

6.3 Combining unstructured data

One extremely important and interesting use-cases is determining the value of the textual data in combination with the structured information in an automatic way. It would be important for health analytics professionals and researchers if it were possible to do a differentially private release of data including both structured and unstructured information that could show an outcome previously not possible using even the non-anonymized structured data alone. This is important because it's arguable that at this point in time releasing textual data will always have a larger risk of privacy disclosure than releasing structured data, because the PHI detection algorithms aren't perfect and the structured data is generally better understood.

6.4 Larger-scale statistical analysis

Chapter 5 showed analysis of error for a variety of queries on real-world data, but more in-depth statistical analyses are necessary for determining the appli-

cability across multiple research institutions. This is especially important for any studies that must be replicated by different institutions. The addition of noise and removal of information will likely make replication difficult or impossible. A study showing the differences between research conclusions with and without perturbed data done independently by multiple research institutions would give more insight. This could prove to be an interesting research thread and extension of this work that is highly applicable in the medical domain.

6.5 Clinical use cases

An example real-world workflow of HIDE would likely consist of the following steps by an honest broker: 1) load original textual and structured data from private data repository of medical research institution into HIDE, 2) generate differentially private data cube for use by researchers. The data cube could then be freely queried for pre-research queries and prospective clinical trials to gauge whether or not there is likely enough data satisfying the needs of the clinician in the original dataset that could warrant seeking IRB approval or higher access rights to the data. Population-level or larger-scale observational research studies could potentially be done on the privacy preserving data release to determine trends or possible predictors for disease outcomes. Before publication or dissemination it would likely be necessary to perform the study on the original data, but this would allow for potentially more studies without the need for formal approval or large pools of patients that give consent for

some studies. Comparative effectiveness studies could also be possible following a similar workflow. Typically comparative effectiveness studies requires the use of a variety of data sources, which would likely be easier assuming a number of institutions release differentially private data cubes. If the study found promising trends across the diverse data sources, then it may be possible to reach out in a forum to promote other researchers to verify the findings on their own data where they may have higher access rights.

6.6 Conclusion

The ultimate goal of this work is to create a framework and system that can be used in practice at a large scale. I sincerely hope that this work will do society good by being adopted by a research hospital that proves the theory at a larger scale than presented in this dissertation. This work covered many life-like datasets, but still in a controlled environment. A system that can preserve the privacy of the individuals in the data, but be used as a powerful tool of inference and data analysis has the potential to change medical research as we know it. It is mostly a matter of legislation for determining what is an acceptable form of information sharing. This work has shown that structured and unstructured analytics, medical, and privacy research can be integrated into a science. HIDE is the first work to fuse these various fields into one coherent study. I believe our initial work will prove useful to the medical community and encourage responsible sharing for the advancement of health

and life.

Bibliography

[1] Surveillance, Epidemiology, and End Results (SEER) Program (www.seer.cancer.gov) Research Data (1973-2008), National Cancer Institute, DCCPS, Surveillance Research Program, Cancer Statistics Branch, released April 2011, based on the November 2010 submission.

[2] Charu C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, pages 901–909, 2005.

[3] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. *SIGMOD Rec.*, 29(2):439–450, 2000.

[4] R. Mahaadevan B. A. Beckwith, U. J. Balis, and F. Kuo. Development and evaluation of an open source software tool for deidentification of pathology reports. *BMC Medical Informatics and Decision Making*, 6(12), 2006.

[5] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE '05: Proceedings of the 21st International Con-*

ference on Data Engineering (ICDE'05), pages 217–228, Washington, DC, USA, 2005. IEEE Computer Society.

[6] B. A. Beckwith, R. Mahaadevan, , U. J. Balis, and F. Kuo. Development and evaluation of an open source software tool for deidentification of pathology reports. *BMC Medical Informatics and Decision Making*, 6(12), 2006.

[7] J.J. Berman. Concept-match medical data scrubbing. how pathology text can be used in research. *Arch Pathol Lab Med*, 127(6):680–6, 2003.

[8] E. Bertino, B.C. Ooi, Y. Yang, and R. H. Deng. Privacy and ownership preserving of outsourced medical data. In *ICDE*, 2005.

[9] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. *Stoc'08: Proceedings of the 2008 Acm International Symposium on Theory of Computing*, pages 609–617, 2008. 14th Annual ACM International Symposium on Theory of Computing MAY 17-20, 2008 Victoria, CANADA.

[10] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to non-interactive database privacy. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618, New York, NY, USA, 2008. ACM.

[11] R. Chen, N. Mohammed, B. C. M. Fung, B. C. Desai, and L. Xiong. Publishing set-valued data via differential privacy. *The Proceedings of the VLDB Endowment (PVLDB)*, 4(11):1087–1098, August 2011.

[12] Rui Chen, Benjamin C. M. Fung, and Bipin C. Desai. Differentially private trajectory data publication. *CoRR*, abs/1112.2020, 2011.

[13] Bolin Ding, Marianne Winslett, Jiawei Han, and Zhenhui Li. Differentially private data cubes: optimizing noise sources and consistency. In *Proceedings of the 2011 international conference on Management of data*, SIGMOD '11, pages 217–228, New York, NY, USA, 2011. ACM.

[14] C. Dwork. A firm foundation for private data analysis. *Commun. ACM.*, 2011.

[15] C Dwork, F McSherry, K Nissim, and A Smith. Calibrating noise to sensitivity in private data analysis. pages 265–284. Proceedings of the 3rd Theory of Cryptography Conference, 2006.

[16] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.

[17] Cynthia Dwork. Differential privacy: A survey of results. In Manindra Agrawal, Dingzhu Du, Zhenhua Duan, and Angsheng Li, editors, *Theory and Applications of Models of Computation*, volume 4978 of *Lecture Notes in Computer Science*, pages 1–19. Springer Berlin / Heidelberg, 2008.

[18] Cynthia Dwork. Differential privacy: A survey of results. In Manindra Agrawal, Ding-Zhu Du, Zhenhua Duan, and Angsheng Li, editors, *TAMC*, volume 4978 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2008.

[19] Cynthia Dwork, Frank Mcsherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *In Proceedings of the 3rd Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

[20] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 381–390, New York, NY, USA, 2009. ACM.

[21] Dan Feldman, Amos Fiat, Haim Kaplan, and Kobbi Nissim. Private coresets. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 361–370, New York, NY, USA, 2009. ACM.

[22] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *Proc. of the 21st IEEE International Conference on Data Engineering (ICDE 2005)*, pages 205–216, Tokyo, Japan, April 2005.

[23] Benjamin C. M. Fung, Ke Wang, Rui Chen, and Philip S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Comput. Surv.*, 42(4):1–53, 2010.

[24] James Gardner and Li Xiong. HIDE: An integrated system for health information de-identification. In *CBMS*, pages 254–259, 2008.

[25] James Gardner, Li Xiong, Fusheng Wang, Andrew Post, and Joel Saltz. An evaluation of feature sets and sampling techniques for statistical de-identification of medical records. *1st ACM International Health Informatics Symposium*, 2010.

[26] James J. Gardner and Li Xiong. An integrated framework for de-identifying unstructured medical data. *Data Knowl. Eng.*, 68(12):1441–1451, 2009.

[27] James J. Gardner, Li Xiong, Kanwei Li, and James J. Lu. Hide: heterogeneous information de-identification. In Martin L. Kersten, Boris Novikov, Jens Teubner, Vladimir Polutin, and Stefan Manegold, editors, *EDBT*, volume 360 of *ACM International Conference Proceeding Series*, pages 1116–1119. ACM, 2009.

[28] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circu-*

lation, 101(23):e215–e220, 2000 (June 13). Circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215.

[29] Y. Guo, R. Gaizauskas, I. Roberts, G. Demetriou, and M. Hepple. Identifying personal health information using support vector machines. In *Proceedings of the AMIA 2006 Workshop on Challenges in Natural Language Processing for Clinical Data*, Washington, November 2006.

[30] D. Gupta, M. Saul, and J. Gilbertson. Evaluation of a deidentification (de-id) software engine to share pathology reports and clinical documents for research. *American Journal of Clinical Pathology*, 2004.

[31] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24:8–12, 2009.

[32] J. Han and M. Kamber. *Data mining: concepts and techniques*. The Morgan Kaufmann series in data management systems. Elsevier, 2006.

[33] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially-private queries through consistency. In *VLDB*, 2010.

[34] Vijay S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD*, pages 279–288, 2002.

[35] P. Jurczyk, J. J. Lu, L. Xiong, J. D. Cragan, and A. Correa. Fril: A tool for comparative record linkage. In *AMIA 2008 Annual Symposium*, 2008.

[36] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[37] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[38] R. Leaman and Gonzalez G. Banner: An executable survey of advances in biomedical named entity recognition. In *Pacific Symposium on Biocomputing*, 2008.

[39] K. LeFevre, D. Dewitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *ACM SIGMOD International Conference on Management of Data*, 2005.

[40] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *IEEE ICDE*, 2006.

[41] Chao Li, Michael Hay, Vibhor Rastogi, Gerome Miklau, and Andrew McGregor. Optimizing linear counting queries under differential privacy. In *PODS '10: Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-*

SIGART symposium on Principles of database systems of data, pages 123–134, New York, NY, USA, 2010. ACM.

[42] Ninghui Li and Tiancheng Li. t-closeness: Privacy beyond k-anonymity and l-diversity. In *To appear in International Conference on Data Engineering (ICDE)*, 2007.

[43] Dong C. Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[44] Dong C. Liu, Jorge Nocedal, Dong C. Liu, and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[45] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*, page 24, 2006.

[46] Andrew Mccallum, Dayne Freitag, and Fernando Pereira. Maximum entropy markov models for information extraction and segmentation. pages 591–598. Morgan Kaufmann, 2000.

[47] McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data*, pages 19–30, New York, NY, USA, 2009. ACM.

[48] S M Meystre, G K Savova, K C Kipper-Schuler, and J F Hurdle. Extracting information from textual documents in the electronic health record: a review of recent research. *Yearbook of medical informatics*, 47(Suppl 1):128–144, 2008.

[49] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(7), 2007.

[50] Sharyl J. Nass, Laura A. Levit, and Lawrence O. Gostin. *Beyond the HIPAA Privacy Rule: Enhancing Privacy, Improving Health Through Research*. The National Academies Press, 2009.

[51] I Neamatullah, M Douglass, LH Lehman, A Reisner, M Villarroel, WJ Long, P Szolovits, GB Moody, RG Mark, and GD Clifford. Automated De-Identification of Free-Text medical records. *BMC Medical Informatics and Decision Making*, 8(32), 2008.

[52] Mehmet Ercan Nergiz, Maurizio Atzori, and Chris Clifton. Hiding the presence of individuals from shared databases. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 665–676, New York, NY, USA, 2007. ACM.

[53] Mehmet Ercan Nergiz, Maurizio Atzori, and Chris Clifton. Hiding the presence of individuals from shared databases. In *SIGMOD Conference*, pages 665–676, 2007.

[54] Naoaki Okazaki. Crfsuite: a fast implementation of conditional random fields (crfs), 2007.

[55] Ruch P, Baud RH, Rassinoux AM, Bouillon P, and Robert G. Medical document anonymization with a semantic lexicon. In *Proc AMIA Symp. 2000*, pages 729–733, 2000.

[56] Vincent Della Pietra, Vincent Della Pietra, and John Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:380–393, 1997.

[57] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

[58] Fei Sha and O Pereira. Shallow parsing with conditional random fields. pages 213–220, 2003.

[59] T. Sibanda and O. Uzuner. Role of local context in de-identification of ungrammatical fragmented text. In *North American Chapter of Association for Computational Linguistics/Human Language Technology*, 2006.

[60] Tawanda Sibanda and Ozlem Uzuner. Role of local context in automatic deidentification of ungrammatical, fragmented text. In *In Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL*, page 73, 2006.

[61] L. Sweeney. Guaranteeing anonymity when sharing medical data, the datafly system. In *Proceedings of AMIA Annual Fall Symposium*, 1997.

[62] L. Sweeney. k-anonymity: a model for protecting privacy. *International journal on uncertainty, fuzziness and knowledge-based systems*, 10(5), 2002.

[63] Latanya Sweeney. Replacing personally-identifying information in medical records, the scrub system. *Journal of the American Medical Informatics Association*, 1996.

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

[65] György Szarvas, Richárd Farkas, and Róbert Busa-Fekete. State-of-the-art anonymization of medical records using an iterative machine learning framework. *JAMIA*, 14(5):574–580, 2007.

[66] R. K. Taira, A. A. Bui, and H. Kangarloo. Identification of patient name references within medical documents using semantic selectional restrictions. pages 757–761, 2002.

[67] S. M. Thomas, B. Mamlin, and G. Schado adn C. McDonald. A successful technique for removing names in pathology reports. pages 777–781, 2002.

[68] O. Uzuner, Y. Luo, and P. Szolovits. Evaluating the state-of-the-art in automatic de-identification. *J Am Med Inform Assoc.*, 14(5), 2007.

[69] Ozlem Uzuner, Yuan Luo, and Peter Szolovits. Evaluating the State-of-the-Art in automatic de-identification. *JAMIA*, 14(5):550–563, 2007.

[70] Hanna M. Wallach. Conditional random fields: An introduction, 2004.

[71] K. Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: a data mining solution to privacy protection. In *Proc. of the 4th IEEE International Conference on Data Mining (ICDM 2004)*, November 2004.

[72] Ben Wellner, Matt Huyck, Scott Mardis, John Aberdeen, Alex Morgan, Leonid Peshkin, Alex Yeh, Janet Hitzeman, and Lynette Hirschman. Rapidly retargetable approaches to de-identification in medical records. *JAMIA*, 14(5):564—573, 2007.

[73] Xiaokui Xiao and Yufei Tao. Anatomy: Simple and effective privacy preservation. In *VLDB*, pages 139–150, 2006.

[74] Xiaokui Xiao, Guozhang Wang, and Johannes Gehrke. Differential privacy via wavelet transforms. *CoRR*, abs/0909.5530, 2009.

[75] Y. Xiao, L. Xiong, and C. Yuan. Differentially private data release through multidimensional partitioning. In *Secure Data Management (SDM), 7th VLDB Workshop, Lecture Notes in Computer Science, 2010*, 2010.

[76] Yonghui Xiao, James Gardner, and Li Xiong. Dpcube: Releasing differentially private data cubes for health information. *28th IEEE International Conference on Data Engineering (ICDE)*, 2012.

[77] Yonghui Xiao, Li Xiong, and Chun Yuan. Differentially private data release through multidimensional partitioning. *5th VLDB workshop on Secure Data Management*, 2010.

[78] Bianca Zadrozny, John Langford, and Naoki Abe. Cost-Sensitive learning by Cost-Proportionate example weighting. In *IEEE International Conference on Data Mining*, 2003.

[79] Qing Zhang, Nick Koudas, Divesh Srivastava, and Ting Yu. Aggregate query answering on anonymized tables. In *ICDE*, pages 116–125, 2007.

[80] Sheng Zhong, Zhiqiang Yang, and Rebecca N. Wright. Privacy-enhancing k-anonymization of customer data. In *PODS '05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 139–147, New York, NY, USA, 2005. ACM Press.