Secure and Efficient Computation on Biomedical Data in a Distributed Environment

by

Md Nazmus Sadat

A thesis submitted to
The Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements
of the degree of

Master of Science

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada
May 2018

Thesis advisor

Noman Mohammed

Author

Md Nazmus Sadat

Secure and Efficient Computation on Biomedical Data in a Distributed Environment

Abstract

Recent advances in cost-effective and efficient genome sequencing technologies and scalable health data management systems have resulted in a biomedical data revolution. This massive data availability has created several exciting opportunities to accelerate clinical research, develop better and efficient diagnosis and prevention techniques for patients. In order to perform a comprehensive study by utilizing a large-scale dataset, multiple institutions need to collaborate with each other. However, policy and legal challenges in biomedical data sharing result in geographic inequities in access to data, which often hinders collaborative research. As constructing a centralized data repository is infeasible due to legal and policy constraints, a distributed network model is very effective in this scenario. In this model, to protect the data and summary statistics of participating sites, traditional cryptographic means are not readily applicable due to significant computation and communication overhead. In this thesis, I propose three different frameworks for secure and efficient biomedical data analysis in a distributed environment. These frameworks can securely perform genome-wide association studies, regression analysis, and clinical notes de-identification.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my gratitude to the Almighty for granting me the opportunity to write this thesis.

I am grateful to my supervisor, Dr. Noman Mohammed for giving me the opportunity to work under his supervision. I am very thankful for his guidance, suggestions, and feedback throughout the preparation of this thesis. My sincerest gratitude to the members of my committee, Dr. Olivier Tremblay-Savard and Dr. Ekram Hossain for their thoughtful comment and advice.

I am also grateful to all my collaborators, Dr. Xiaoqian Jiang, Dr. Shuang Wang, Dr. Feng Chen, and Md Momin Al Aziz (in no particular order) for their guidance. I am equally grateful to all the faculties and staff members of Department of Computer Science, University of Manitoba.

Finally, I would like to thank my parents for their immense sacrifices and inspirations throughout the years.

Dedicated to my parents for their unconditional support and encouragement

Chapter 1

Introduction

Rapid advances in human genome sequencing and scalable Electronic Medical Record (EMR) management systems have enabled effective use of biomedical data in a wide range of healthcare applications. The availability of such data offers numerous opportunities to accelerate clinical research, develop better and targeted therapies for patients. In this way, biomedical data can yield more effective healthcare by guiding medical decisions. Therefore, data-driven biomedical research is gaining popularity as it can identify potential correlations between a disease and a certain clinical profile, which improves the safety and efficacy of drug treatment and can also develop more effective prevention strategies. The outcomes of biomedical research are capable of providing new clues to disease biology by identifying genes and biological pathways correlated with the disease process. Some related tests attest this statement: women who have a higher risk for breast and ovarian cancer can be identified by the mutations in the BRCA1 and BRCA2 genes [1].

To minimize the sampling error and to increase the statistical accuracy of this type of research projects, large study populations are required. For instance, in observational drug effect studies, very large sample size is essential to distinguish between individual members of drug classes [2]. Therefore, data from different sources need to be brought together since a single organization does

not necessarily possess the required amount of data. However, biomedical data (for instance, human genome) is potentially re-identifiable, and can expose sensitive information, which raises privacy and security concerns about sharing such data. Several works have showed the vulnerability of human genomic data if data security is not ensured: re-identifying patients from an anonymized dataset [3], reconstructing allele frequencies for participating individuals [4], and predicting the likelihood of developing a disease [5]. As genomic information is shared among family members, the inappropriate exposure of individual genomic data may compromise their privacy as well [6]. In addition, new privacy threats are emerging [7].

Collaborative biomedical research projects among multiple institutions (cross-state, cross-provincial, or cross-border) must satisfy the strict policies that have been enforced to regulate privacy-sensitive data sharing. These policies vary in different regions of the world based on various criteria. For instance, there are several key differences between the US Health Insurance Portability and Accountability Act (HIPAA) and the Canadian Personal Information Protection and Electronic Documents Act (PIPEDA). Due to these differences in policies and potential misuse of the privacy-sensitive data, it is hard to establish centralized data repositories between different geographical entities. As a result, raw data never leaves the boundary of an institution or a government. For example, the Institute for Clinical Evaluative Sciences in Ontario (ICES) and Manitoba Centre for Health Policy (MCHP) do not allow the dissemination of raw data beyond the institution [8, Page 22]. Therefore, it is imperative to design a framework that is capable of performing computation using summary statistics from participating sites instead of depending on centralized storage of raw data. This distributed model, where data is analyzed in each local site and outputs are combined by a collaborating centre, can overcome substantial policy and legal obstacles to pooling the raw data [2].

To protect the confidentiality of shared biomedical data, different secure computation techniques have been proposed. Most of these techniques solely rely on cryptographic means, which

Table 1.1: Summary of the contributions of this thesis

Frameworks	Data		Secure Computation Technique	
	Genomic	Clinical	Homomorphic Encryption	Secure Hardware
Chapter 3	✓		✓	✓
Chapter 4	✓	✓	✓	✓
Chapter 5		✓	✓	

are very inefficient to handle a computation task of a practical scale. A promising alternative is made possible by recent advances in trusted execution environments (TEEs) such as Intel Software Guard Extension (Intel SGX) [9]. SGX enables critical code to process sensitive data in an isolated hardware environment that is protected from the traditional operating system and other system softwares. Such protection is available in Intel's mainstream CPUs (Skylake and Kaby lake). However, the system design of Intel SGX requires an enclave program to use resources (memory, I/O, etc.) partially or fully controlled by the untrusted system softwares, and thus potentially subjects it to side-channel attacks. By observing operations on the shared resources, an adversary could infer sensitive information inside the enclave.

1.1 Contributions

This thesis explores different approaches to develop a practical secure computation framework that is both secure and efficient. The proposed frameworks adopt two secure computation techniques: *homomorphic encryption* and *secure hardware*. Table 1.1 summarizes the secure computation techniques of the proposed frameworks and the nature of data they operate on.

Following we discuss the technical contributions of this thesis in detail.

SAFETY: Secure gwAs in Federated Environment Through a hYbrid solution

We present SAFETY, a hybrid framework, which can securely perform GWAS on federated genomic datasets using homomorphic encryption and secure hardware component of Intel SGX to ensure high efficiency and privacy at the same time. Different experimental settings show the efficacy and applicability of such hybrid framework in secure conduction of GWAS. To the best of our knowledge, this hybrid use of homomorphic encryption along with Intel SGX is not proposed or experimented to this date. Our proposed framework, SAFETY is up to 4.82 times faster than the best existing secure computation technique. The results of this chapter appear in IEEE/ACM Transactions on Computational Biology and Bioinformatics [10].

Secure and Efficient Regression Analysis Using a Hybrid Cryptographic Framework

We designed, developed, and evaluated a hybrid cryptographic framework, which can securely perform regression analysis, a fundamental machine learning algorithm using somewhat homomorphic encryption and Intel SGX to ensure both privacy and efficiency at the same time. Experimental results demonstrate that our proposed method provides a better trade-off in terms of security and efficiency than solely secure hardware-based methods. In addition, there is no approximation error. The computed model parameters are exactly similar to the plaintext results. The results appear in Journal of Medical Internet Research - Medical Informatics [11].

A Secure and Efficient Framework for Distributed Clinical Notes De-identification

We propose a novel protocol based on private set intersection and secure thresholding for a distributed data de-identification task. We extended a previous filtering-based method [12] to de-identify data from distributed sources and demonstrated the feasibility of using homomorphic encryption to devise an efficient multi-party protocol. We optimized naive homomorphic encryption by incorporating a Single Instruction, Multiple Data (SIMD) technique and number theoretic trans-

forms. Experimental results show that our proposed method can simultaneously ensure data privacy and preserve data utility. The outcome of this research project has been submitted to American Medical Informatics Association Annual Symposium 2018.

1.2 Organization of this Thesis

This thesis is organized as follows:

- Chapter 2 discusses necessary background materials, which are utilized by the different methods proposed in this thesis.
- Chapter 3 shows how to securely conduct genome-wide association studies on a federated dataset using a trusted hardware assisted framework.
- Chapter 4 discusses hybrid cryptographic framework for performing regression analysis over distributed data in a secure and efficient way.
- Chapter 5 describes a novel secure protocol based on private set intersection and secure thresholding to identify uncommon and low-frequency terms in clinical notes from distributed sources.
- Chapter 6 concludes the thesis.

Chapter 2

Background

In this chapter, we will introduce some of the concepts, which are required to understand the proposed methods in this thesis. Also, some general related works will be discussed. More specific related works will be discussed in corresponding chapters.

2.1 Preliminaries

2.1.1 Genomics

Genes may have different variants which are located at a particular position or locus, on a chromosome. The different variants of a gene at a locus are called alleles. This kind of genetic variations in DNA sequences has a significant influence on disease and phenotype. The most common form of genetic variants is Single Nucleotide Polymorphism (SNP). A SNP is a genetic variation, which refers to an alteration of a single nucleotide (A, T, C, or G). A SNP is generally bi-allelic: there are two alleles at a SNP locus.

In humans, chromosomes are inherited in pairs, one from each parent. Hence, at a particular locus, there is an allele in each of the two homologous chromosomes. These two alleles together are called genotype. On the other hand, the sequence of alleles along a chromosome is called a haplotype.

2.1.2 Intel SGX

Intel SGX is a set of extensions to the Intel architecture which mainly focuses on the problem of running applications on a remote machine administered by an untrusted party. SGX allows parts of an application to be executed inside secure segments of the CPU called *enclaves*. Untrusted entities including privileged software (kernel, hypervisor, etc.) cannot access enclaves. SGX ensures that the code and data within an enclave cannot be read or modified from outside the enclave.

There are two SGX features that play a vital role in provisioning of sensitive data to an enclave. These are called attestation and sealing.

- *Attestation:* SGX enclaves are created without privacy-sensitive data. Privacy-sensitive data are delivered after the enclave has been properly instantiated on the platform. The process of demonstrating that a piece of software has been properly instantiated within an enclave on an enabled platform is called *attestation* [13].

  Attestation demonstrates to a user that he is communicating with an application running inside an enclave. This demonstration is accomplished via a cryptographic signature that certifies the hash of the enclave's contents. The remote computer's administrator is able to load any program in an enclave. However, the user (who uses the remote computation service) will deny to load his data into an enclave if the hash of the contents does not match the desired value [14].

- *Sealing:* When an enclave is instantiated, SGX provides protections to its data while it is maintained inside the enclave. However, when the enclave process exits, the enclave will be destroyed and all associated data will be lost. If the data is required later, it needs to be stored outside the enclave. Sealing is the process of encrypting and storing data in a way such that only the same enclave would be able un-seal them back to their original form. In our framework, data sealing is not required since the data owners do not necessarily outsource their

data to the central server. Instead, they send certain local counts in response to researcher's query.

Memory Partition in Intel SGX

SGX partitions the main memory in two regions: protected region and unprotected region (as shown in Figure 2.1). Unprotected memory region is generally accessible. However, access to protected memory region is restricted. Data used by enclaves are swapped in and out of a segment of protected memory region, which is called *Enclave Page Cache (EPC)*. Enclave pages are stored in the EPC. Size of an enclave page is 4KB.

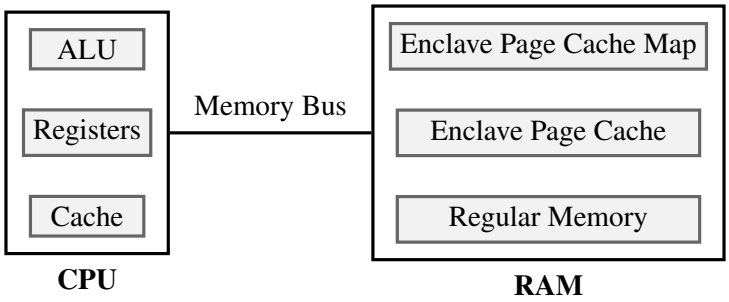


Figure 2.1: Memory partition in SGX.

Access to EPC is restricted on the hardware level. The *Enclave Page Cache Map (EPCM)* keeps track of which enclave is authorized to access which pages in the EPC. In EPCM, each page is mapped to a set of enclaves that are authorized to access it. SGX checks whether a specific enclave is authorized to access a page based on the EPCM.

2.1.3 Homomorphic Encryption

Homomorphic encryption allows performing computation on encrypted data without decrypting the data. Homomorphic encryption in a nutshell is: if $c_1 = \xi(m_1)$ and $c_2 = \xi(m_2)$ (where m_1 and m_2 are the plaintexts, c_1 and c_2 are the ciphertexts, and ξ is any randomized encryption function), we

can perform computation on c_1, c_2 and get the same output as if we were performing computation on m_1 and m_2.

The idea of an encryption scheme that is capable of performing arbitrary computation on encrypted data was first proposed by Rivest et al [15] in 1978. Since then, a number of homomorphic cryptosystems were invented. In Chapter 3, we have used a additively homomorphic cryptosystem called *Paillier cryptosystem* [16].

Paillier Cryptosystem

Paillier cryptosystem has two important properties that we utilized in chapter 3.

- *Probabilistic encryption:* If we encrypt the same message several times using Paillier cryptosystem, it generates different ciphertexts for the same plaintext.
- *Addition homomorphism:* For any public key n and arbitrary messages m_1, m_2,

$$\xi(m_1 + m_2) = (\xi(m_1) * \xi(m_2)) \bmod n^2$$

  which denotes that we can do an addition operation over ciphertexts.

Table 2.1 shows a partial list of homomorphic encryption schemes. As we can see, most of the traditional homomorphic encryption schemes are either additively homomorphic (Paillier [17]), or multiplicatively homomorphic (ElGamal [18]). However, such restriction to one single algebraic operation is very inconvenient for general purpose applications.

Finally, Boneh et al. [22] proposed a partially homomorphic cryptosystem in 2005 that is able to perform one multiplication and any number of additions.

Fully Homomorphic and Somewhat Homomorphic Encryption

Developing an encryption scheme that supports an arbitrary number of additions and multiplications was an open problem until 2009. In 2009, Gentry showed the first construction of fully ho-

Table 2.1: A list (partial) of homomorphic encryption schemes

Cryptosystem	Homomorphism
Goldwasser-Micali [19], Paillier [16]	Additive
RSA [20], ElGamal [21]	Multiplicative
Boneh et al. [22]	Both

momorphic encryption (FHE) scheme [23] that can do any number of addition and multiplication on encrypted data. Lately, researchers are adopting lattice cryptosystems, which leverage ring homomorphism (addition and multiplication) [24, 25]. Since addition and multiplication operations over integer ring Z_2 form a complete set of operations, this type of encryption schemes supports any polynomial-time computation on ciphertext.

To explain FHE, say ciphertext c_i is the encrypted form of plaintext m_i, where m_i and c_i are elements of a ring (the operations of the ring are: addition and multiplication). In FHE, if a function f consists of addition and multiplication in the ring, then

$$\text{decryption}(f(c_1, c_2, ..., c_n)) = f(m_1, m_2, ..., m_n)$$

Generally, f is expressed by an arithmetic circuit over GF(2) (Gallois field). This is equivalent to a Boolean circuit with exclusive OR and AND gates.

In the existing FHE schemes, a certain amount of noise needs to be introduced in the ciphertexts to ensure data confidentiality. This noise grows while performing homomorphic operations on ciphertexts. In particular, a homomorphic multiplication operation increases the size of the ciphertext abruptly. For instance, if 2 input ciphertexts have size M and N, then the output ciphertext will be of size $M + N - 1$. If the amount of noise becomes too high, then the ciphertext cannot be decrypted correctly. To perform any number of homomorphic operations, the noise of the ciphertexts needs to be reduced. As mentioned before, this can be done using a method known as

$bootstrapping$ [23], which is computationally expensive.

In use cases, where only a predetermined number of computational operations need to be done, the costly bootstrapping process can be avoided by using a somewhat homomorphic encryption (SWHE) scheme [26]. This scheme is often more practical than using FHE scheme with bootstrapping. SWHE schemes use a method called *relinearization* [27, 28] to reduce the size of the ciphertext.

The cryptosystem in [29] is a Somewhat Homomorphic Encryption (SWHE) scheme that can compute a bounded number of homomorphic functions. Other recent RLWE-based SWHE cryptosystems include BGV [30], FV [31], and YASHE [32]. While these systems are intrinsically similar, there are differences and trade-offs. Interested readers can refer to [33, 34] for more details.

In Chapters 4 and 5, we used the YASHE and FV cryptosystem respectively (other RLWE-based systems will work in a similar manner), which consists of the following functionalities:

- $KeyGen(params)$: Given the system parameters $params$ as input, $Keygen$ generates a public-private key pair and an evaluation key (pk, sk, evk).
- $Enc(pk, m)$: An encryption algorithm encrypts a plaintext message m using the public key pk.
- $Dec(sk, c)$: Let c be the encryption of plaintext m. A decryption algorithm outputs m, given private key sk and ciphertext c as input.
- $Add(c_1, c_2)$: Let c_1, c_2 be the ciphertexts for messages m_1, m_2 respectively. Given, c_1, c_2 as input, a homomorphic addition operation Add computes the encrypted sum of m_1, m_2.
- $Mult(c_1, c_2)$: Let c_1, c_2 be the ciphertexts for messages m_1, m_2 respectively. Given, c_1, c_2 as input, a homomorphic multiplication operation $Mult$ computes the encrypted product of m_1, m_2.

- $ReLin(c_{mult}, evk)$: The objective of relinearization operation $ReLin$ is to reduce the size of a given ciphertext c_{mult} back to (at least) 2. Relinearization is generally performed when the size of the ciphertext increases substantially by multiplication operations. Relinearization operation requires the evaluation key evk.

2.2 Related Works

In this section, we discuss some related works on secure genome-wide association studies and regression analysis. To the best of our knowledge, there is no prior work that focuses on securely de-identifying clinical notes from distributed sources. However, some relevant techniques are discussed in Chapter 5.

2.2.1 Secure Genome-wide Association Studies

Homomorphic Encryption based Approaches

Data privacy is a critical issue when genomic data is processed in an untrusted cloud environment. For privacy-preserving processing of genomic data, each data contributor sends data to the cloud server in encrypted form and all the computations are performed on encrypted data. An homomorphic encryption scheme allows computation on encrypted data. But existing homomorphic encryption schemes support only a limited family of computations. For instance, Smart and Vercauteren [35] proposed a fully homomorphic encryption scheme, which supports only addition and multiplication operations on encrypted data. As a result, performing complex statistical analysis for Genome-wide Association Studies (GWAS) using homomorphic encryption is challenging.

As existing homomorphic encryption schemes do not support certain operations, the cloud server can perform only the supported ones and leave further operations for executing locally. In a solution proposed by Lu et al. [36], when the cloud server receives a GWAS query from a re-

searcher, it merges the encrypted data from different contributors. Then, using homomorphic addition and multiplication operations, the cloud server calculates different values required to perform statistical tests in GWAS. However, computing the final result involves the division operation, and in this scheme, the cloud server does not support homomorphic division. So, the server sends these values to the researcher for further computation. The researcher then decrypts the values and calculates the final result locally.

Instead of sending necessary values for performing statistical tests, the cloud server can send the final results of the statistical tests by implementing homomorphic division operation. Zhang et al. [37] proposed two homomorphic division schemes (secure errorless division and secure approximation division) to conduct GWAS in the cloud server. Although a secure approximation division protocol comes with some approximation errors, it provides a good trade-off in terms of complexity and accuracy. In addition, their proposed division algorithms can be executed in a parallel fashion to reduce the number of homomorphic additions and multiplications.

Unlike Lu et al. [36], Zhang et al. [37] allows computation of final results of statistical tests for GWAS in the cloud server which facilitates complete outsourcing of GWAS to the cloud. Moreover, in the proposed approach of Lu et al. [36], the cloud server sends individual values due to the lack of a homomorphic division operation, which exposes a security threat. A researcher who receives those values can infer additional information.

Lauter et al. [38] worked on some statistical algorithms, which are widely used in GWAS. They made those algorithms to work on encrypted genomic data using homomorphic encryption. Their proposed method operates on a database containing encrypted genotype and phenotype counts in two steps. In the first step, encrypted genotype and phenotype counts are calculated using homomorphic addition operation. In the second step, outputs of the statistical algorithms are calculated from counts calculated in the first step.

Secure Count and Ranked Query: To execute count and ranked queries securely on genomic data,

Aziz et al. [39] proposed an architecture where data owners keep their data in their local databases. Their architecture uses the Paillier cryptosystem [16]. In this architecture [39], when a researcher wants to execute a count or ranked query on the integrated data, each query is executed separately in each data owner's database. Then, individual results are sent to the central server in encrypted form. The central server then performs computation on the encrypted results. Finally, the researcher receives the result of his desired query. Since most computations and query executions are performed on plaintext, and only individual encrypted results are used to formulate the final result, this process is much faster than computation on encrypted data. There are some other solutions based on similar architectures [37, 40]. These approaches are computationally secure but not suitable for real world applications due to high computational overhead.

Secure Hardware (Intel SGX) based Works

There are some recent works, which perform privacy-preserving statistical analysis on genomic data using secure hardware: PREMIX [41], PRINCESS [42].

PREMIX: In biomedical research, identification of demographic histories of patients is very important. For instance, by analyzing ethnicity information, researchers can understand whether a population is more susceptible to a certain disease or likely to benefit from a specific therapeutic intervention. In addition, understanding the individual admixture from different ancestries is also required for this purpose (to clarify the term admixture, an example may be used: Hispanics represent an admixed group among African, Caucasian, and Native American). However, many ethnicity/race identification studies are limited by sample size [43]. Aggregating data from different sources could improve the accuracy and reliability of these studies. However, sharing patients' genetic data violates the patient privacy policy.

Taking into account the above mentioned issue, PREMIX [41] introduces a solution based on Intel SGX, which enables efficient and privacy-preserving estimation of individual admixture.

PREMIX [41] can protect the confidentiality of genomic data and ancestry information of patients. Usage of strong security primitives of Intel SGX, allows PREMIX to enable multiple parties to collaborate on individual admixture estimation. PREMIX includes, a secure feature selection module based on discriminative SNP and an expectation maximization based maximum likelihood estimator to estimate individual admixture. These modules enhances the estimation accuracy and computational efficiency of the framework.

PRINCESS: As mentioned before in Chapter 1, protecting confidentiality during collaborative (cross-institutional) genomic data analysis remains a big challenge. Cross-border collaborations are more complex due to governmental policies that may restrict genomic data sharing. In this context, the major challenge is, how to perform cross-border (or, institutional) collaborative research on aggregate genomic data while satisfying legal and privacy policy constraints. Due to this complexity, recent genomic data analysis projects [40, 44] used simulated data.

PRINCESS [42] introduces an international collaboration framework for privacy-preserving analysis of rare disease genetic data that are distributed around the world. PRINCESS [42] was evaluated in a study of family-based transmission tests (TDT) in order to understand the genetic architecture of Kawasaki Disease (KD) [45].

In order to isolate sensitive genomic data analysis within a secure enclave, PRINCESS [42] uses Intel SGX. It includes a secure TDT module to study KD.

2.2.2 Secure Regression Analysis

In this section, we discuss some of the existing works that focus on regression analysis in a secure way. Table 3.1 demonstrates some of the recent works in this area in chronological order. Each of the following Subsections cover a certain secure computation scheme.

Homomorphic Encryption based Approaches

There are many applications of homomorphic encryption in privacy-preserving machine learning [46–49]. Some of these target regression analysis [48, 49]. Hall et al. [49] proposed a multiple linear regression analysis technique based on homomorphic encryption. Their main idea is, since computing regression coefficients is basically done by matrix products, a secure method can be formulated by composing secure matrix products. Another work by Bos et al. [48] demonstrates a private predictive analysis (based on logistic regression) on encrypted medical data using homomorphic encryption.

Homomorphic encryption is impractical as it comes with huge computational and storage overhead. Some practical variants of homomorphic encryption scheme have been proposed over the last few years [50–52]. However, the overhead issue still persists.

Garbled Circuit based Approaches

In the mid 80s, Yao proposed garbled circuits [61] in the context of secure two-party computation, which can compute a function f on input x without exposing anything about f or x. So, a malicious party cannot learn anything about the function f or the input x other than the result $f(x)$. It should be noted that the term *circuit* in this context means, boolean circuit.

Valeria et al. [57] implemented an evaluator for computing regression coefficient that uses linear homomorphism in the first phase to perform all the linear operations. In the second phase, it uses garbled circuit for non-linear computations since garbled circuit is much more efficient than homomorphic encryption for this purpose.

However, there are some critical issues of garbled circuits.

1. First of all, standard garbled circuits suffer from one limitation: they offer no security if used on more than one inputs. In other words, garbled circuits are not reusable. Consequently,

Table 2.2: Previous research works in secure regression analysis

Technique	Year	Regression Type	Differential Privacy	Homomorphic Encryption	Garbled Circuit	Intel SGX
Chaudhuri et al. [53]	2011	Linear	✓			
Lei et al. [54]	2011	Linear	✓			
Hall et al. [49]	2011	Linear		✓		
Zhang et al. [55]	2012	Linear, logistic	✓			
Wu et al. [56]	2012	Logistic			✓	
Valeria et al. [57]	2013	Ridge		✓	✓	
Bos et al. [48]	2014	Logistic		✓		
Wang et al. [58]	2016	Exact logistic		✓		
Shi et al. [59]	2016	Logistic			✓	
Ohrimenko et al. [60]	2016	–				✓
Our proposal	2017	Linear, logistic		✓		✓

evaluating the circuit on a new input requires a completely new garbling of the circuit.

2. Another problem with garbled circuits is that the communication complexity is proportional to the size of the circuit. This makes garbled circuits inefficient from the communication perspective [62, Page 22]. However, with homomorphic encryption, the communication complexity is much less.

   For instance, consider a scenario, where encrypted clinical data is stored in the cloud, and a researcher executes private prediction queries on this massive clinical dataset. In this case, the communication complexity of a private query is extremely high since the garbled circuit used to represent the query is proportional to the size of the dataset. On the contrary, the com-

munication complexity of such a query in homomorphic encryption scheme is proportional to the size of the encrypted response to the query.

3. Finally, garbled circuit-based techniques need complex circuit design and optimization for each particular computation. Thus, it is not very flexible.

Differential Privacy based Approaches

Solutions based on differential privacy [63] add noise to the data to preserve individual privacy.

There are also some works on differentially private regression analysis [53–55, 64]. The solution proposed by Chaudhuri et al. [53,64] is applicable only for linear regression. Lei [54] proposed another technique where in the first step, they generate a noisy histogram from the input data. Then, from the noisy histogram they generate synthetic data by preserving statistical properties of the histogram. In the final step, they use synthetic data to compute the regression results. Finally, Zhang et al. [55] proposed a solution based on functional mechanism. Instead of perturbing the results, they perturb the objective function (cost function) of the regression analysis.

Noise added by differentially private techniques reduces data utility, and makes statistical analysis very difficult. Also, differential privacy requires one trusted entity who can access the integrated dataset. In addition, in client-server architecture, where a client executes a query on the database stored in the server, differential privacy is not applicable for several types of queries [65].

Secure Hardware (Intel SGX) based Approaches

There are no secure hardware based techniques that target regression analysis (to the best of our knowledge). However, Ohrimenko et al. [60] worked on some machine learning algorithms using Intel SGX. Although, SGX is very efficient from a computation and storage point of view, the trustworthiness of SGX is yet to be fully established due to some recently proposed side-channel attacks against SGX [66–68].

Chapter 3

SAFETY: Secure gwAs in Federated Environment Through a hYbrid solution

Recent studies demonstrate that effective healthcare can benefit from using the human genomic information. For instance, analysis of tumor genomes has revealed 140 genes whose mutations contribute to cancer [69]. As a result, many institutions are using statistical analyses of genomic data, which are mostly based on genome-wide association studies (GWAS). GWAS analyze genome sequence variations in order to identify genetic risk factors for diseases. These studies often require pooling data from different sources together in order to unravel statistical patterns or relationships between genetic variants and diseases. In this case, the primary challenge is to fulfill one major objective: accessing multiple genomic data repositories for collaborative research in a privacy-preserving manner. Due to the sensitivity and privacy concerns regarding the genomic data, multi-jurisdictional laws and policies of cross-border genomic data sharing are enforced among different regions of the world.

3.1 Introduction

Rapid advancement in human genome sequencing has led us to a genomic era where human genomic data play an ever-important role in clinical research [70]. As cost-effective and efficient genome sequencing technologies are readily available, the research community can conduct experiments on different genomic data repositories for scientific discovery [71]. As a result of this massive data availability, Genome-Wide Association Studies (GWAS) are gaining popularity as they answer critical questions like susceptibility towards a disease or a physical trait by analyzing genome sequence variations in different individuals. GWAS examine genetic architecture of a disease to identify genetic risk factors associated with it. In other words, GWAS aims at finding if there is any correlation between a certain set of genes and a specific disease. Another fundamental goal of GWAS is to identify biological factors responsible for disease susceptibility in order to develop more effective diagnosis, treatment, and prevention techniques.

A larger genomic dataset is quintessential to perform any analytical study such as GWAS. Different research organizations or healthcare facilities often sequence genomes of different patients or participants for this reason. Researchers are interested in executing queries over these massive genomic datasets for unraveling new pieces of information about diseases under study. Oftentimes, the accuracy of this evaluation relies on the quantity and quality of the data used in the analysis– but a single organization often does not possess adequate genomic data (collection, processing, and storing of large-scale data is non-trivial) to perform a comprehensive or meaningful experiment. Because more data can reduce the sampling errors and improve the power of the analysis (for instance, statistical strength of GWAS increases with the quantity of data [72]), organizations tend to collect as much data as possible to meet data analysis needs.

Because sharing genomic data in plaintext possesses serious privacy implications for the participants [77, 78], in addition to the approval from an institutional review board (IRB), collaborative

Table 3.1: Previous research works in secure and privacy-preserving GWAS in chronological order

Existing Techniques	Year	Secret Sharing	Homomorphic Encryption	Differential Privacy	Intel SGX
Bogdanov et al. [73]	2014	✓			
Lauter et al. [38]	2014		✓		
Tramèr et al. [74]	2015			✓	
FORESEE [37]	2015		✓		
Simmons et al. [75]	2016			✓	
Shahbazi et al. [76]	2016	✓			
PRINCESS [42]	2017				✓
SAFETY (our proposal)	2017		✓		✓

research on shared genomic data often needs to satisfy two criteria at the same time — a) authorizing access to genomic data for research and b) preserving participants' privacy and protecting the confidentiality of their genomic information [79]. That is why strict policies regarding genomic data sharing have been enforced, and generally, these policies are different in different regions of the world. This difference in the regulations of cross-border genomic data sharing greatly impedes international research projects [80]. It is imperative to address the reality challenge with practical solutions to promote health science discoveries.

3.1.1 Existing Techniques

To ensure the security and privacy of shared genomic data, different privacy-preserving techniques (for instance, homomorphic encryption [23], garbled circuit [61], secure hardware [42], secret sharing [81], differential privacy [82] etc.) have been proposed. Some applications of these techniques in secure GWAS, are demonstrated in Table 3.1. Each of these techniques has some advantages

as they allow some functions to be computed on the data without compromising confidentiality and integrity of it [83]. For instance, homomorphic encryption allows to perform some computation (i.e. addition and multiplication) on encrypted data. Hence, encrypting the data before sharing and then executing the required computation on encrypted data is an intuitive solution. However, homomorphic encryption based solutions have significant computational and storage overhead, which makes them often impractical for real life applications [84].

Secure multiparty computation (garbled circuit and secret sharing) is also a prospective solution because of their lower computational overhead. However, garbled circuit based solutions need complex circuit design and optimization, which limit its flexibility and usability greatly. Another technique of this genre, secret sharing involves huge communication overhead. Besides, it is not suitable for client server architecture [85].

Differential privacy based techniques add noise to the data in order to protect individual privacy. But, this noise reduces data utility, and makes accurate statistical analysis much harder. In addition, differential privacy requires one trusted party who has access to the integrated dataset. This requirement is not applicable to our scenario.

Intel SGX [9, 86] is an extension to the Intel architecture which allows an application to run inside protected execution areas of CPU. Although, SGX is efficient from computation and storage perspective, the security of SGX is yet to be fully established due to the recent discovery of side-channel attacks against SGX [66].

3.1.2 Contributions

In this chapter, we propose a hybrid framework, SAFETY, for secure execution of some popular statistical tests used in GWAS in a federated environment. Our proposed hybrid model incorporates security and efficiency of two different cryptographic schemes in a single system. More precisely, it is the first attempt to infuse homomorphic encryption with SGX to develop a secure and scalable

genomic data computation model. The experimental results clearly demonstrate that it performs consistently *irrespective of the number of data owners making it highly scalable* (see Section 3.5 for details). This hybrid model captures the essence of both techniques: ability of computing some functions on encrypted data (homomorphic encryption) and performing sophisticated mathematical operation in the secure execution area of an SGX enabled CPU. Our primary goal behind proposing a hybrid model relies on better security guarantee from existing secure computation schemes, and faster and scalable execution for any number of data owners. From our experimental results, it is evident that the proposed hybrid model provides better efficiency and security than pure secure hardware or homomorphic encryption based solutions.

SAFETY utilizes an architecture [39] to execute secure count query on federated genomic datasets. Similar federated architectures are available in literature [73, 87]. In our adopted architecture [37, 39], genomic data resides in the local premises of individual data owners in plaintext (see Figure 3.1). Data owners have their own database systems which are geographically distributed and have different policy compliance for the data usage. An overview of data representation for each data owner is shown in Table 3.2. Proper authentication allows any researcher to execute queries on their data.

Among the existing secure computation techniques, SGX is the most efficient. For instance, an implementation of SGX-based MapReduce framework [88] shows a very modest overhead of 8% to achieve read/write integrity. This is a great advantage of SGX in comparison to other secure computation schemes like garbled circuit and homomorphic encryption, which generally increase the computational overhead to a great extent. However, our proposed hybrid model is *1.7* to *4.82* times faster than SGX (see Section 3.5). This comparative efficiency increases with the number of data owners. The contributions of this work are summarized as follows:

1. We propose a hybrid cryptographic framework, SAFETY, which uses homomorphic encryption along with secure hardware features of the Intel SGX. SAFETY is not only secure and ef-

ficient, but also overcomes the limitations of solely homomorphic encryption based solutions which often come with higher computational overhead for processing higher order polynomials. In addition, SAFETY also simplifies solely SGX based solutions, which require pairwise attestation and secure key distributions between server and data owners.

2. Using SAFETY, we securely execute and evaluate some of the major functions of GWAS in federated architecture where genomic data is distributed and owned by different parties. We performed four statistical tests: Linkage Disequilibrium (LD), Hardy-Weinberg Equilibrium (HWE), Cochran-Armitage Test for Trend (CATT), Fisher's Exact Test (FET) to evaluate SAFETY over a variety of settings. Note that, our framework SAFETY can incorporate any GWAS functions (i.e., transmission disequilibrium test [42], EigenSTRAT [89], linear mixed model [75], etc.) and is not limited to the GWAS functions mentioned previously. The methodology to perform these statistical tests securely is discussed in Section 3.4.

3. SAFETY ensures that each data owner is completely unaware of the contributions from the other data owners, who are participating in the same analysis. Moreover, the final result is revealed only to the researchers without disclosing individual contribution of data owners. This allows us to preserve the privacy of the output of each data owner.

4. We conduct multiple experiments in different realistic setting in a federated environment varying the data size and the geographic locations of data owners (see Section 3.5 for details).

3.2 System Overview

In this section, we discuss the system architecture and the threat model.

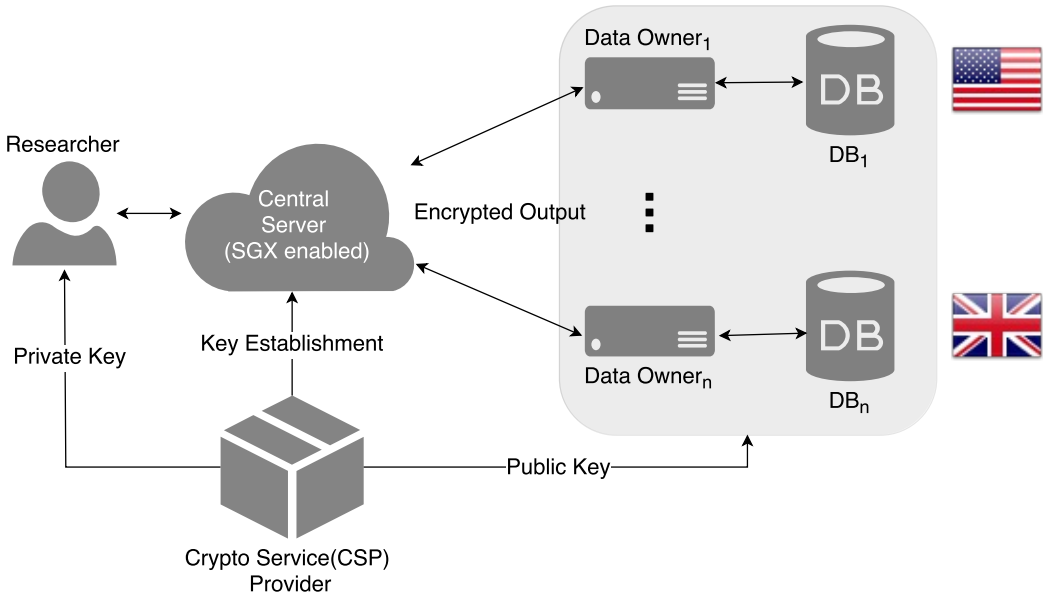


Figure 3.1: Diagram of the federated architecture where data owners are geographically distributed.

3.2.1 System Architecture

There are four entities in the proposed system architecture as shown in Figure 3.1:

- *Researchers (authorized):* Individuals or organizations who want to execute queries over genomic databases. This party sends queries to the central server and expects encrypted results of different GWAS functions.

- *Data Owners:* These parties are geographically distributed and possess databases upon which queries are performed. These data owners might be hospitals or government organizations who want to share their genomic data and have different policies regarding the data sharing. The proposed model supports any number of data owners where they can execute any aggregate query locally.

- *Crypto Service Provider (CSP):* CSP manages the cryptographic keys that will be used for

Table 3.2: Data representation in each party

		Sequence				
#	**Case**	**rs4426**	**rs4305**	**rs4630**		**Cancer**
Data Owner 1	1	CC	CT	GG	...	Negative
	2	CT	CT	AG	...	Negative
	3	CC	CT	GG	...	Negative
Data Owner 2	1	CC	CT	GG	...	Negative
	2	CT	CC	GG	...	Positive
	3	CC	CT	GG	...	Positive
Data Owner 3	1	CT	CC	AG	...	Positive
	2	CT	CT	AG	...	Negative
	3	TT	CC	GG	...	Positive
Data Owner 4	1	TT	CC	AA	...	Positive
	2	CC	CC	GG	...	Positive
	3	CC	CT	GG	...	Positive

data encryption and decryption in various steps of the system protocol. Each data owner receives a public key from the *CSP* and encrypts its output using that public key. CSP also issues the private key to an authorized researcher who can decrypt the final result.

- *Central Server:* The central server maintains communications with all other entities of the system architecture. It receives queries from the researcher, sends them to the data owners, and collects individual encrypted results from each data owner. Individual encrypted results from data owners are securely combined by the central server to compute the final result of the query with the help of homomorphic encryption and SGX.

3.2.2 Threat Model

In this chapter, our goal is to ensure the confidentiality of individual contributions and data from different geographically distributed data owners. Researchers can decrypt only the final result provided by the central server. We assume the CSP to be a trusted entity while the central server is a semi-honest entity (also known as honest-but-curious) where it follows the protocol but may attempt to derive additional information from the server logs or received defined protocols [90].

We assume that the computation (required for statistical tests of GWAS) runs in an SGX enabled central server. SGX architecture facilitates the central server to perform any computation securely on data provided by multiple data owners. We assume that the processor works properly, and is not compromised. We trust the design and implementation of SGX including all cryptographic operations performed by it.

It should be mentioned that there are some recently proposed side-channel attacks against SGX [66, 67, 91]. All of these attacks are software attack. To the best of our knowledge, there is no known physical attack against SGX. The attack proposed in [66], is a limited or controlled side-channel attack against a particular SGX based framework [92]. This is basically a page-fault attack. Another software attack proposed in [67] (known as *synchronization bugs*), is applicable only for multi-threaded applications.

Existing defense mechanisms are proposed targeting only the page-fault side-channel attacks [68, 93, 94]. However, these defense mechanisms may not prevent the future attacks. Further research is required to better understand the potential vulnerability of SGX, consequence of different side-channel attacks, and possible defense mechanisms.

Addressing these side-channel attacks against SGX, SAFETY protects the institutional privacy of the participating data owners by aggregating their local statistics without decrypting them. This approach provides a higher layer of data security.

3.3 Statistical Tests Used in GWAS

In this section, we discuss the statistical tests that we used for GWAS. These statistical tests were also discussed in some of the prior works on privacy-preserving GWAS [38, 76]. We are describing these tests here to keep the chapter self-contained.

3.3.1 Linkage Disequilibrium

Linkage disequilibrium refers to a nonrandom association of alleles at two or more loci. On the other hand, when the alleles are randomly associated they are said to be in a state of linkage equilibrium.

For example, suppose at SNP locus 1, the possible alleles are *A*, a, and at SNP locus 2, the possible alleles are *B*, b. So, there are four possible pairs of alleles *AB, Ab, aB, ab* which are called haplotypes. Let, N_{AB}, N_{Ab}, N_{aB}, and N_{ab} be the number of four haplotypes AB, Ab, aB, and ab respectively. If the total size of the population sample is N, then the population frequencies are calculated as:
$P_{AB} = \frac{N_{AB}}{N}$, $P_{Ab} = \frac{N_{Ab}}{N}$, $P_{aB} = \frac{N_{aB}}{N}$, $P_{ab} = \frac{N_{ab}}{N}$. Let,

$$P_A = \text{frequency of allele A at locus 1},$$

$$P_a = \text{frequency of allele a at locus 1},$$

$$P_B = \text{frequency of allele B at locus 2},$$

$$P_b = \text{frequency of allele b at locus 2}$$

If locus 1 and 2 are independent, we would expect to see that a haplotype, say, AB has frequency $P_A P_B$. If the frequency of haplotype AB is different from $P_A P_B$, the two loci are said to be in linkage disequilibrium. Linkage disequilibrium is defined based on following quantity:

$$D_{AB} = P_{AB} - P_A P_B \tag{3.1}$$

Here, D_{AB} is called the coefficient of linkage disequilibrium. It characterizes the extent to which two alleles are nonrandomly associated. Each pair of alleles has a specific value of D. The values of D for different pairs of alleles are constrained by the following facts,

$$P_A + P_a = 1$$

$$P_B + P_b = 1$$

$$P_{AB} + P_{Ab} + P_{aB} + P_{ab} = 1$$

The above set of equations demonstrate the underlying relationship among allele frequencies and haplotype frequencies. This relationship can be illustrated by a table (see Table 3.3).

Table 3.3: Relationship among allele frequencies and haplotype frequencies

SNP 2				Total
SNP 1	Allele	B	b	
	A	P_{AB}	P_{Ab}	P_A
	a	P_{aB}	P_{ab}	P_a
Total		P_B	P_b	1

Only one value of D is required to characterize linkage disequilibrium between two loci because $D_{AB} = -D_{Ab} = -D_{aB} = D_{ab}$. Generally, D is used without subscript.

The following table illustrates the value of haplotype frequency under linkage equilibrium and linkage disequilibrium.

P_{ij}	Linkage disequilibrium	Linkage equilibrium
P_{AB}	$P_{AB} = P_A P_B + D$	$P_{AB} = P_A P_B$
P_{Ab}	$P_{Ab} = P_A P_b - D$	$P_{Ab} = P_A P_b$
P_{aB}	$P_{aB} = P_a P_B - D$	$P_{aB} = P_a P_B$
P_{ab}	$P_{ab} = P_a P_b + D$	$P_{ab} = P_a P_b$

It can be shown that,

$$D = P_{AB}P_{ab} - P_{Ab}P_{aB} \tag{3.2}$$

The range of D depends on the allele frequencies, which makes it troublesome to use it as a measure of disequilibrium. There are two scaled-down variants: D'- measure or r^2- measure.

D'-measure: The D'-measure is given by: $D' = |\frac{D}{D_{max}}|$; where $D_{max} = \begin{cases} \min\{P_A P_b, P_a P_B\}, & \text{if} D > 0 \\ \min\{P_A P_B, P_a P_b\}, & \text{if} D < 0 \end{cases}$

r^2-measure: The r^2-measure is given by: $r^2 = \frac{D^2}{P_A P_B P_a P_b}$

The range of both D' and r^2 is $[0, 1]$. Here, 0 indicates complete linkage equilibrium and 1 indicates complete linkage disequilibrium.

Now, let us consider the data from Table 3.2. We try to determine if *rs4305* and *rs4630* are at linkage disequilibrium. Both SNPs are bi-allelic. So, there are four possible haplotypes: CA, CG, TA and TG.

rs4630			
rs4305	Allele	A	G
	C	CA	CG
	T	TA	TG

Let, total number of haplotypes be N and number of haplotypes CA, CG, TA and TG are $N_{CA},\ N_{CG},\ N_{TA}$ and N_{TG} respectively. We use subscript i to denote the contribution of i^{th} data owner. So, for n data owners,

$$N_{CA} = N_{CA_1} + N_{CA_2} + \ldots + N_{CA_n}$$

$N_{CG},\ N_{TA}$ and N_{TG} are computed similarly. Now, we can calculate the frequencies of these haplotypes. For instance,

$$P_{CA} = \frac{N_{CA}}{N}$$

With these haplotype frequencies we can compute the frequency of a particular allele at a specific locus.

$$P_C = P_{CA} + P_{CG}$$

$$P_T = P_{TA} + P_{TG}$$

$$P_A = P_{CA} + P_{TA}$$

$$P_G = P_{CG} + P_{TG}$$

So,

$$D = P_{CA}P_{TG} - P_{CG}P_{TA} \tag{3.3}$$

3.3.2 Hardy-Weinberg Equilibrium

Hardy-Weinberg Equilibrium (HWE) is a principle which states that allele and genotype frequencies in a population will remain unchanged from one generation to the next generation given that there are no evolutionary influences. A lot of assumptions are required for HWE to hold including no inbreeding, random mating, infinite population size, and no mutation, selection, or migration [95].

HWE indicates a relationship between allele frequency in parents and genotype frequency in offspring. For example, consider two alleles A and a of locus 1 (mentioned in 3.3.1). After one round of random mating, the genotype frequencies in the offspring are:

$$\left.\begin{aligned} P(\text{genotype } AA) &= P_A^2 \\ P(\text{genotype } Aa) &= 2P_AP_a \\ P(\text{genotype } aa) &= P_a^2 \end{aligned}\right\} \tag{3.4}$$

If all the genotypes of a population satisfy equation 3.4, it is said to be in HWE . It is noteworthy that the sum of all the frequencies is the binomial expansion of the square of the sum of P_A and P_a. Since $P_A + P_a = 1$, $P_A^2 + 2P_AP_a + P_a^2 = (P_A + P_a)^2 = 1$.

Table 3.4: A Punnett square demonstrating the probabilities of generating all possible genotypes at locus 1

	Females		
Males	Allele	A(P_A)	a(P_a)
	A(P_A)	AA(P_A^2)	Aa($P_A P_a$)
	a(P_a)	Aa($P_A P_a$)	aa(P_a^2)

To estimate deviation from HWE, Pearson goodness of fit test is generally used. In this test, the observed genotype counts are obtained from data, and the expected genotype counts are calculated using HWE.

Table 3.5: Pearson Goodness of Fit Test for HWE

Genotype	AA	Aa	aa	total
Observed count	n_{AA}	n_{Aa}	n_{aa}	n
Expected count	nP_A^2	$nP_A P_a$	nP_a^2	n

P_A can be calculated using, $P_A = \frac{n_{AA}}{n} + \frac{1}{2} \times \frac{n_{Aa}}{n}$.

Pearson Goodness of Fit Test for HWE is given by:

$$\chi^2 = \sum \frac{(\text{Observed} \quad \text{count} - \text{Expected} \quad \text{count})^2}{\text{Expected Count}} \tag{3.5}$$

Now, let us consider, the data from Table 3.2. We try to determine if HWE holds for *rs4305*. Possible genotypes at *rs4305* are CC, CT and TT.

Observed count for genotype CC is given by,

$$n_{CC} = n_{CC_1} + n_{CC_2} + n_{CC_3} + n_{CC_4}$$

n_{CT} and n_{TT} are calculated similarly. So, here population size is,

$$n = n_{CC} + n_{CT} + n_{TT}$$

Now, the frequency of the allele C in this population is calculated as follows:

$$P_C = \frac{n_{CC}}{n} + \frac{1}{2} \times \frac{n_{CT}}{n} \tag{3.6}$$

Therefore, the frequency of the allele T, $P_T = 1 - P_C$.

Expected counts of genotype CC, CT and TT are $nP_C^2, 2nP_CP_T$, and nP_T^2 respectively. Pearson goodness of fit test for HWE is given by:

$$\chi^2 = \frac{(n_{CC} - nP_C^2)^2}{nP_C^2} + \frac{(n_{CT} - 2nP_CP_T)^2}{2nP_CP_T} + \frac{(n_{TT} - nP_T^2)^2}{nP_T^2}$$

In this case, critical chi-square value with 1 degree of freedom (3 genotypes - 2 alleles) is 3.841 (for 0.05 significance level). If $\chi^2 < 3.841$, then HWE holds in this population.

3.3.3 Cochran-Armitage Test for Trend

Cochran-Armitage Test for Trend (CATT) is highly used in case-control studies in order to determine if an allele is associated with a disease. Such a study identifies factors that may contribute to a disease by comparing the genotypes of the individuals who have the disease (cases) with the individuals who do not have the disease (controls).

CATT can be applied when the data takes a form of $2 \times k$ contingency table. For instance, a 2×3 contingency table can be constructed with 3 genotypes vs. cases/controls as follows.

	AA	Aa	aa	Sum
Controls	N_{11}	N_{12}	N_{13}	R_1
Cases	N_{21}	N_{22}	N_{23}	R_2
Sum	C_1	C_2	C_3	N

In this table, N_{ij} represents genotype frequency, R_i is the sum of the i^{th} row and C_j is the sum of the j^{th} column. If a contingency table like this is given, CATT computes the trend test statistic as follows:

$$T = \sum_{i=1}^{3} w_i(N_{1i}R_2 - N_{2i}R_1) \tag{3.7}$$

where w_i are weights. Chi-square value is given by:

$$\chi^2 = \frac{T^2}{Var(T)} \tag{3.8}$$

where the variance of T is given by:

$$Var(T) = \frac{R_0R_1}{N}(\sum_{i=1}^{3} w_i^2C_i(N - C_i) - 2\sum_{i=1}^{2}\sum_{j=i+1}^{3} w_iw_jC_iC_j) \tag{3.9}$$

3.3.4 Fisher's Exact Test

Like CATT, Fisher's Exact Test (FET) is another statistical test which is used to analyze contingency table in order to find association. FET performs well when the sample size is small.

FET operates on a contingency table to identify any association between a variable of s different categories and a variable with t different categories.

	Group 1	Group 2	$\cdots$	Group s	Sum
Category 1	N_{11}	N_{12}	$\cdots$	N_{1s}	R_1
Category 2	N_{21}	N_{22}	$\cdots$	N_{2s}	R_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Category t	N_{t1}	N_{t2}	$\cdots$	N_{ts}	R_t
Sum	C_1	C_2	$\cdots$	C_s	N

The $p - value$ of FET is computed by the following formula:

$$p = \frac{(\prod_{i=1}^{t}(R_i))(\prod_{i=1}^{s}(C_i))}{N! \cdot \prod_{i=1,j=1}^{t,s}(N_{ij}!)} \tag{3.10}$$

3.4 Methodology

In this section, we discuss how to perform the statistical tests (LD, HWE, CATT, and FET) securely. To explain our proposed methods we use the data from Table 3.2.

As mentioned earlier, we consider the use of Intel SGX in two ways — *1*) Hybrid approach: using Intel SGX along with homomorphic encryption *2*) Secure hardware approach: using only Intel SGX. SAFETY is based on the hybrid approach.

3.4.1 Hybrid Approach (SAFETY)

Suppose, there are total n number of data owners ($D_1, D_2, ..., D_n$) connected in the federated environment where a researcher wants to execute a statistical query. The query result should follow or represent as if the query is being executed on the combined dataset. Here, each data owner will have their own individual outputs. For example, data owners $D_1, D_2, \ldots, D_n$ will have outputs $x_1, x_2, \ldots, x_n$ respectively. These outputs can be haplotype or genotype counts (encrypted) for a specific SNP locus based on the query from a researcher.

These outputs are encrypted by the public keys provided beforehand by the CSP. Data owners get the public keys from the CSP before any computation. The data owners generate their encrypted outputs $c_1, c_2, \ldots, c_n$ (from $x_1, x_2, \ldots, x_n$) using the public keys provided by the CSP and send them to the central server for further computation.

The central server then performs homomorphic addition on the individual encrypted outputs $c_1, c_2, \ldots, c_n$ with the Paillier cryptosystem [16]. After homomorphic addition, it hands over the total encrypted counts to Intel SGX for further computation required to perform different statistical tests like LD, HWE, CATT, and FET. Then, the total counts are decrypted inside the enclave, and further computation is also performed inside the enclave where no untrusted application can access these data. The sequence diagram of this protocol is shown in Figure 3.2.

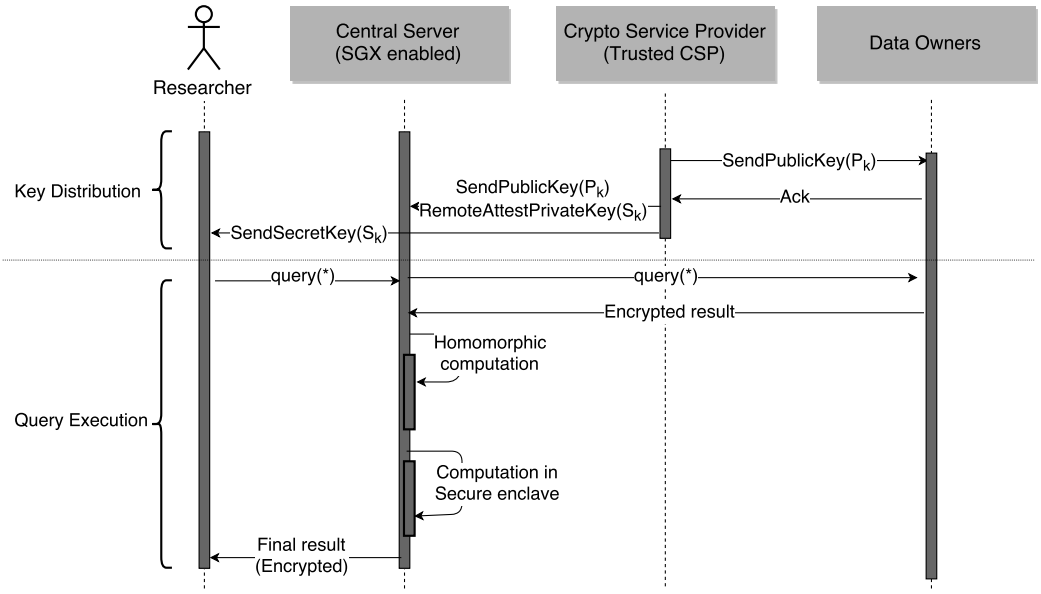


Figure 3.2: Sequence diagram for the hybrid approach.

It is noteworthy that due to the use of homomorphic addition operations, the number of decryptions required to perform statistical tests is greatly reduced (shown in Table 3.10). Also the individual contributions from the data owners are secured since their values are encrypted. Figure 3.3 demonstrates the use of homomorphic encryption and Intel SGX in a hybrid architecture.

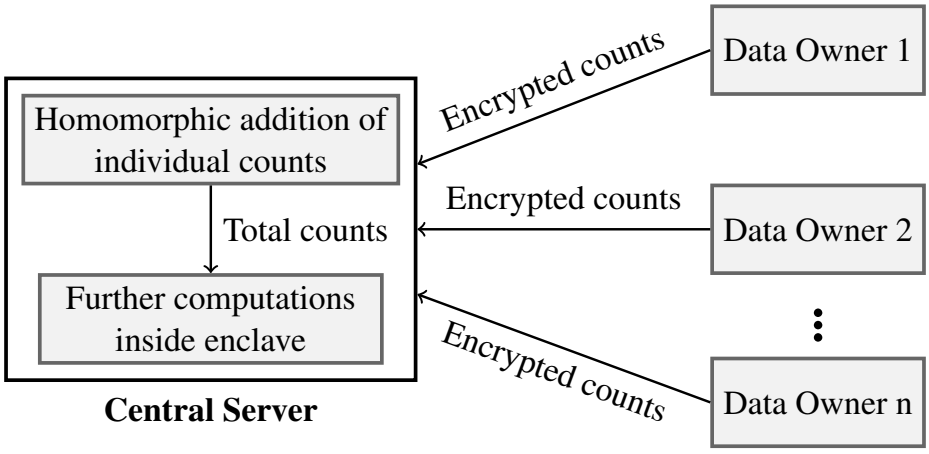


Figure 3.3: Usage of homomorphic addition in our framework.

3.4.2 Secure Hardware Approach

In secure hardware approach, after receiving individual outputs from different data owners, the central server decrypts them inside the enclave and performs further computation on plaintext. The fundamental difference between a hybrid approach and a secure hardware approach is, not using the homomorphic addition on ciphertext. Since in this approach, all the individual homomorphically encrypted outputs need to be decrypted, the computational overhead is quite large.

In the following subsections, we discuss the methods for securely performing LD, HWE, CATT, and FET according to the hybrid approach. We use the data from Table 3.2 to explain the methods.

3.4.3 Secure Linkage Disequilibrium (LD)

A sample query from a researcher regarding LD may look like: *Are rs4305 and rs4630 at linkage disequilibrium*?

Both SNPs are bi-allelic. So, there are four possible haplotypes: CA, TA, CG, and TG.

Each data owners send their haplotype counts, which are encrypted by Paillier cryptosystem [16]. For instance, data owner 1 sends $N_{CA_1} = E(1)$ where the count of haplotype *CA* is 1 and E is the encryption function. After receiving the encrypted counts of *CA* from all the data owners, the central server performs homomorphic addition operation on them to obtain the total encrypted count for *CA*. For n data owners,

$$N_{CA} = N_{CA_1} + N_{CA_2} + \ldots + N_{CA_n}$$

Similarly, total count for TA, CG, and TG are computed. Then the central server instantiates a secure enclave and provisions these encrypted values as input there. As the decryption key (private key) is sealed by the enclave, it can decrypt the counts and calculate the haplotype frequencies. The haplotype frequencies are calculated in the enclave to avoid division of encrypted numbers, which

is expensive even in fully homomorphic encryption. Finally, the coefficient of the LD is computed, and the researcher gets the result of her query from this. We discuss the detailed procedure for computing the LD coefficient in Subsection 3.3.1.

3.4.4 Secure Hardy-Weinberg Equilibrium (HWE)

A sample query regarding HWE is: *Does HWE holds at SNP rs4305*?

Possible genotypes at SNP rs4305: CC, CT, and TT. Each data owner will send their individual count for CC, CT, and TT genotypes. After receiving these encrypted genotype counts from all data owners, the central server performs homomorphic addition operation using Paillier cryptosystem [16] to obtain total encrypted counts for corresponding genotypes.

Now, all the counts are decrypted inside the enclave to calculate the frequencies P_C and P_T. P_C is calculated using, $P_C = \frac{n_{CC}}{n} + \frac{1}{2} \times \frac{n_{CT}}{n}$. Then, $P_T = 1 - P_C$.

Further discussion is available in Subsection 3.3.2.

3.4.5 Secure Cochran-Armitage Test for Trend (CATT)

A typical query from researcher regarding CATT is: *Determine if CATT can be inferred at rs4426?* Possible genotypes at SNP rs4426 are: CC, CT, and TT. For cases and controls (Cancer positives and negatives respectively), all the data owners send their encrypted genotype counts for both categories to the central server. Homomorphic addition operations are performed to calculate row total and column total using Paillier cryptosystem [16]. A contingency table needs to be constructed, which is described in Subsection 3.3.3. This table is then sent to the enclave where all these row totals and column totals are decrypted for further computation.

3.4.6 Secure Fisher's Exact Test (FET)

Like CATT, FET also operates on a contingency table. So, for FET, data flow is similar to CATT. Here, the $p - value$ is calculated in enclave after securely aggregating the individual encrypted inputs from the data owners. Please see Subsection 3.3.4 for further discussion.

3.4.7 Pre-computation Table for GWAS

As we have seen, all the statistical tests mentioned before (LD, HWE, CATT, and FET) require processing data in a tabular format. Data owners can keep their data in this format. Consequently, when the central server requests for data, data owners can respond readily. It is noteworthy that each data owner has to build the table only once. Thus, pre-computation of the table enhances the efficiency of SAFETY.

Table 3.6 represents the pre-computation table of data owner 1 for performing HWE, CATT, and FET at rs4426.

Table 3.6: Sample pre-computation table for HWE, CATT, and FET

CC		**CT**		**TT**	
Case	**Control**	**Case**	**Control**	**Case**	**Control**
2	0	1	0	0	0
2		1		0	

Since performing LD involves two SNP loci, a different pre-computation table is required. Table 3.7 represents pre-computation table of data owner 1 for performing LD at rs4305 and rs4630.

Table 3.7: Sample pre-computation table for LD

	rs4630		
rs4305	Allele	A	G
	C	CA (1)	CG (2)
	T	TA (0)	TG (3)

3.5 Experimental Results

In this section, we extensively evaluate the aforementioned hybrid approach and secure hardware based approach in a federated environment using Amazon cloud and demonstrate their applicability in a real world setting. Our proposed framework SAFETY is based on a hybrid approach where we use SGX along with homomorphic encryption. However, the secure hardware based approach uses only SGX.

3.5.1 Experimental Setting

In our experimental setup, the researcher, CSP, and central server were located in Manitoba, Canada. Our central server was hosted on a machine with Intel Core i7-6700 (3.40 GHz) processor and 8 GB memory. However, we emulate data owners in different locations of the world to evaluate the propriety of our proposed framework in a real world environment. We used ***Amazon EC2 cloud servers*** having the same configuration for all data owners. Table 3.8 shows the location, IP address, and the latency of these servers used in our experiment. In our experiments, we used 80 bit security (size of the public key is 1024 bits) on the public-key cryptosystem. The security can be improved by increasing the key length.

We performed four experiments with different settings. The number of data owners was different for different experiments which allowed us to evaluate the scalability of both methods. Table 3.9

Table 3.8: Server locations and average latency

Server Location	**IP Address**	**Network Latency (ms)**
Canada (Manitoba)	130.179.30.133	<1
USA (Oregon)	52.32.83.223	37
London	52.56.65.221	105
Seoul	52.78.100.194	170
Sydney	54.206.67.251	233

shows different settings used in the experiments. For instance, in experiment 1, two data owners were in USA and Canada while in experiment 4, five data owners were residing in all the locations mentioned in Table 3.8. Experiments were performed using synthetic data which were generated according to the allele frequency of CHB, CHS, JPT and MXL populations from *1000genomes* dataset (August 2010 Release) [96].

Table 3.9: Location of different data owners in different experimental settings

Exp. #	**Canada**	**USA**	**London**	**Seoul**	**Sydney**
Exp. 1	✓	✓	✗	✗	✗
Exp. 2	✓	✓	✓	✗	✗
Exp. 3	✓	✓	✓	✓	✗
Exp. 4	✓	✓	✓	✓	✓

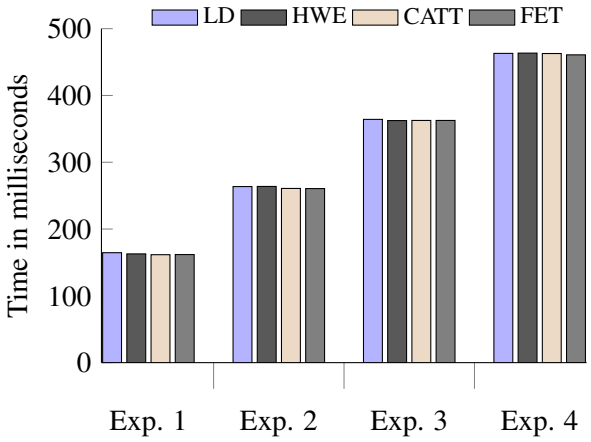


Figure 3.4: Experimental results for plaintext.

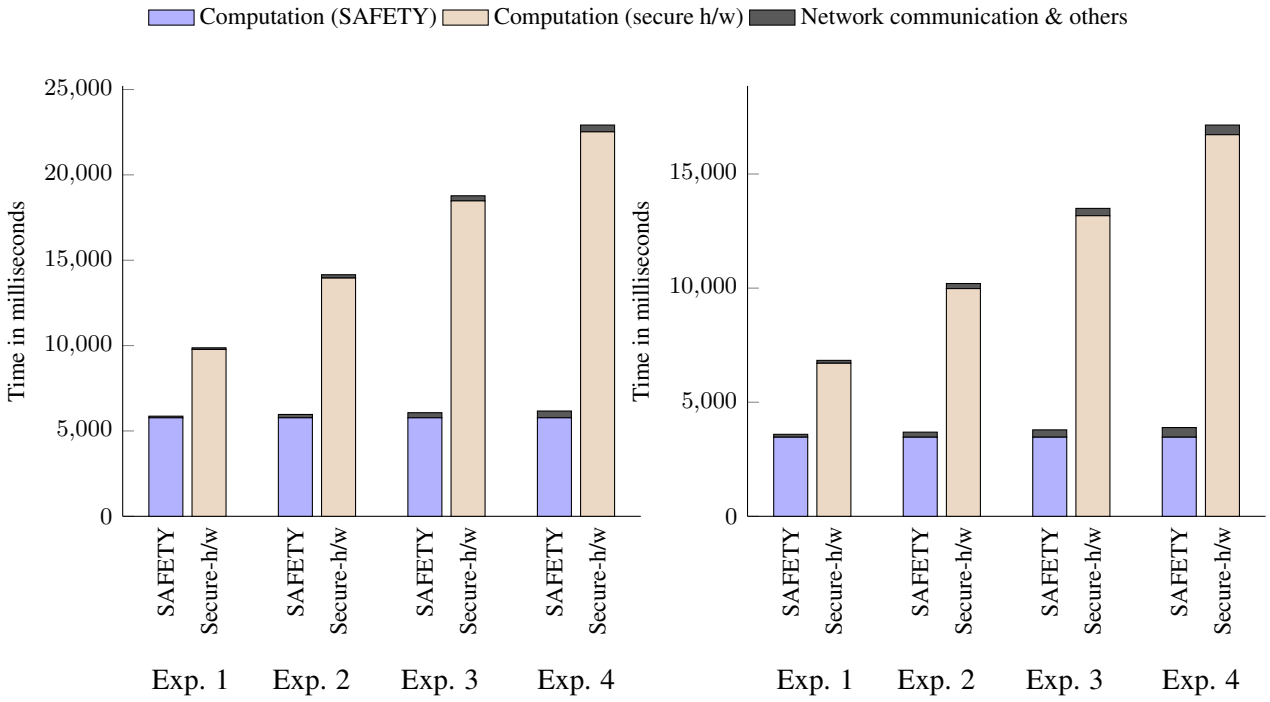
3.5.2 Results and Analysis

Prior to analyzing the running times of our proposed methods, we evaluate the required time to compute the four statistical tests on plaintext (i.e. without any security protection). We calculate the exact results for the GWAS calculations without losing any accuracy.

In Figure 3.4, we show the running time (in milliseconds) for performing the four statistical tests on plaintext. We observed that in any single experimental setup, the running time is almost the same for all the statistical tests. However, running times for different experiments are different because different experiments involve different numbers of data owners (as shown in Table 3.9). As a result, higher communication overhead is added in these experiments. For instance, experiment 2 involves more data owners than experiment 1, which yields more communication overhead and results in greater running time.

The running time for computing LD on ciphertexts is shown in Figure 3.5(a). Here, the running time is decomposed into communication overhead in the network and time required for secure computation of the method. It is noteworthy that,

$$\text{Communication overhead} \propto \text{Number of data owners}$$



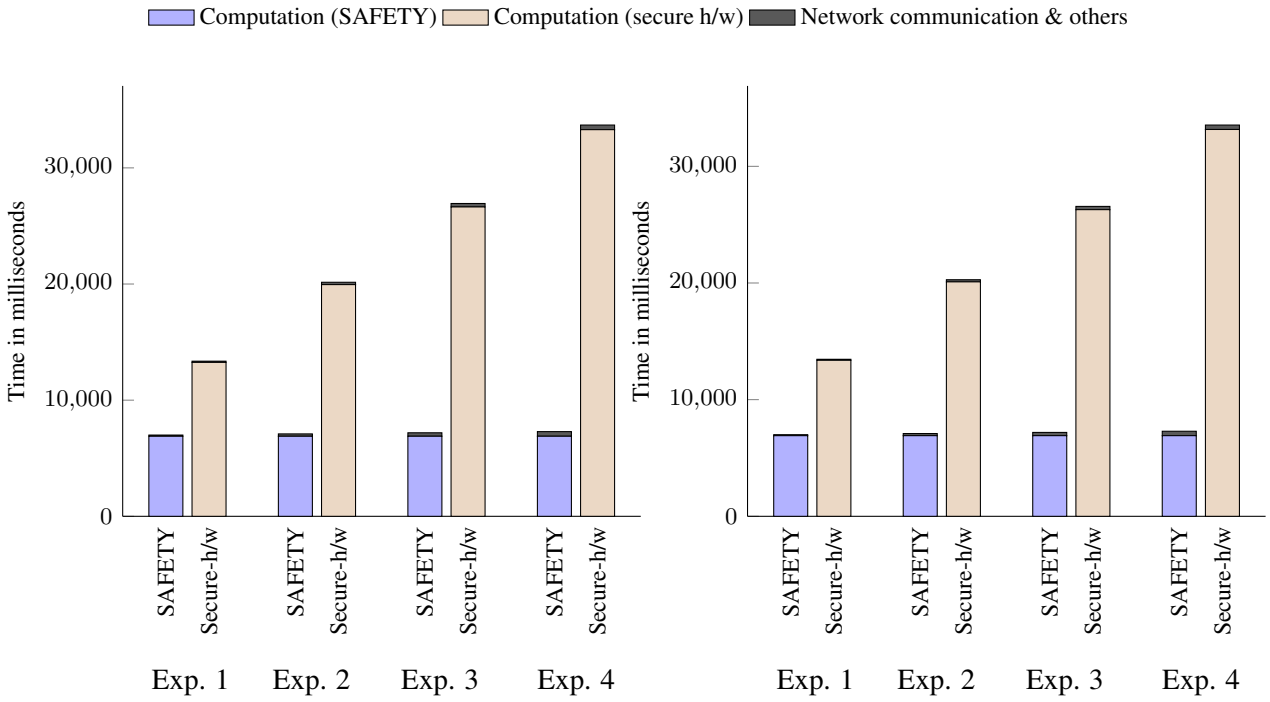
(a)Experimental results for LD

(b)Experimental results for HWE

Figure 3.5: Experimental results for LD and HWE. (a) and (b) compare computation time and communication costs of conducing Linkage Disequilibrium (LD) and Hardy-Weinberg Equilibrium (HWE) test using SAFETY and purely secure hardware approach.

SAFETY requires 5,770 ms to compute the LD coefficient for two data owners, which is 1.7 times faster than the secure hardware approach. The rest of the time is due to the communication overhead in the network. Figure 3.5(b), 3.6(a), and 3.6(b) illustrate the experimental results for performing HWE, CATT, and FET respectively on ciphertexts. Experimental results illustrate that SAFETY is much faster than solely secure hardware based approach. For instance, for HWE, SAFETY is 1.93, 2.87, 3.8, and 4.82 times faster than solely secure hardware based approach in Experiment 1, 2, 3, and 4 respectively (see Figure 3.5(b)). It is noteworthy that SAFETY and the secure h/w approach both utilize the asymmetric encryption and decryption.

The experimental results demonstrate that the performance of the secure hardware approach



(a)Experimental results for CATT

(b)Experimental results for FET

Figure 3.6: Experimental results for CATT and FET. (a) and (b) compare computation time and communication costs of conducing Cochran-Armitage Test for Trend (CATT) and Fishers Exact Test (FET) using SAFETY and purely secure hardware approach.

does not scale well with the number of data owners. As the number of data owner increases, the running time of secure hardware approach increases rapidly. On the contrary, the hybrid approach (SAFETY) performs consistently irrespective of the number of data owners due to its hybrid properties (homomorphic addition followed by computation inside the enclave). In this case, only the communication overhead increases which is very small considering the total running time.

Another important analysis regarding the methods is the number of decryptions needed for any statistical tests. It is evident that LD requires more time than HWE while CATT and FET require more time than the other two. The reason behind this is, the time required to perform a statistical test is proportional to the number of decryptions required. Moreover, for secure hardware approach,

all the individual contributions of data owners need to be decrypted inside the secure enclave. As the number of the data owner increases, the number of decryptions also increases which results into higher running times. Table 3.10 demonstrates how the required number of decryptions increases with the number of data owners.

Table 3.10: Number of decryptions required to perform a statistical test for different number of data owners

Test	**Number of data owners**					
	One		Three		Five	
	Hybrid	Secure h/w	Hybrid	Secure h/w	Hybrid	Secure h/w
LD	4	4	4	12	4	20
HWE	3	3	3	9	3	15
CATT	6	6	6	18	6	30
FET	6	6	6	18	6	30

3.6 Discussions

In this section, we discuss some of the other security and privacy concerns regarding the secure computation of GWAS in our hybrid model.

3.6.1 Query Privacy

In the proposed methods, we do not consider the query privacy of the researcher. In other words, we consider the queries from researchers to be public and data owners, central servers know the targeted position (locus) from the researcher. This issue can be resolved by some of the query privacy or private information retrieval techniques [97–99].

3.6.2 Output Privacy

SAFETY does not guarantee the privacy of the final result as that only gets decrypted by researcher. We are aware that there are some differential privacy based approaches [75, 100, 101], those address this issue and generate differentially private outputs for GWAS. However, as we consider this researcher to be semi-honest, this issue is beyond the scope of the chapter.

3.6.3 Assumption Regarding CSP

In our threat model, we assumed that the Crypto Service Provider (CSP) is a trusted entity, and does not collude with any other party. In other words, if the CSP leaks the secret keys, the proposed framework will not be deemed secure.

To ensure the security and integrity of the CSP, we should employ good security practices (software security, OS security etc.). However, if it is not the case, then our proposed system will fall apart. Our situation is similar to the public-key system, where there are CAs (they are assumed to be trusted).

3.6.4 Consideration of Symmetric Cryptography

We are not using symmetric cryptography (like AES) for a couple of reasons:

- **Achieving randomized encryption (initialization vector management issue):** One major drawback of using any symmetric cryptography scheme, (i.e., AES) is achieving randomized or probabilistic encryption. This randomness can be introduced by choosing initialization vectors which needs to be managed by the central server or CSP for multiple data owners. However, SAFETY is based on homomorphic encryption whose encryption is probabilistic by definition, which reduces the burden of managing initialization vectors.

- **Risk of individual contribution leakage:** One of the major concerns in addressing the se-

curity of the federated environment is hiding the individual contributions from data owners. As we perform the additions over encrypted data, these contributions are never revealed. As a result, in our proposed framework the possibility of such leakage is highly unlikely.

- **Requires n remote attestations for n data owners (key distribution problem):** Symmetric cryptography schemes like AES require key distribution/setup with every data owners which results in many network communications, which might be prone to attack. On the contrary, our proposed framework is based on public-key cryptography where the data owners use a public key to encrypt their data published by the CSP. As a result, key distribution is much simpler and our framework incurs less communication overhead.

3.7 Conclusion

Homomorphic encryption and Intel SGX have their own strengths to utilize. Homomorphic encryption can perform some computational operations without decrypting the ciphertext, and Intel SGX can perform any secure computation efficiently after decrypting ciphertext. However, a hybrid model where homomorphic encryption and secure hardware are used in appropriate use cases provides a good trade-off in terms of efficiency and computational support for secure statistical analysis. The outstanding performance of SAFETY attests this hypothesis.

Recently, some data analytics and machine learning applications [60, 88, 102] have adopted Intel SGX for secure computation. However, little work has been done using Intel SGX for analyzing genomic data. We think, using secure and efficient computation capabilities of Intel SGX to analyze genomic data is very promising for healthcare and medical research.

Chapter 4

Secure and Efficient Regression Analysis Using a Hybrid Cryptographic Framework

Machine learning is an effective data-driven tool that is being widely used to extract valuable patterns and insights from data. Specifically, predictive machine learning models are very important in healthcare for clinical data analysis. The machine learning algorithms that generate predictive models often require pooling data from different sources to discover statistical patterns or correlations among different attributes of the input data. The primary challenge is to fulfill one major objective: preserving the privacy of individuals while discovering knowledge from data.

Existing secure computation schemes are not suitable for processing the large-scale data that is used in cutting-edge machine learning applications.

4.1 Introduction

Machine learning algorithms are now being widely used in many applications to uncover deep and predictive insights from datasets that are large scale and diverse. For instance, building predictive models from biomedical data is very important in biomedical science. Such predictive models can identify genetic risk factors for a specific disease under study and can guide medical treatment. For instance, Tabaei and Hermana formulated a predictive equation to screen for diabetes [103].

Machine learning thrives on growing datasets. In most of the cases, the more data fed into a machine learning system, the more it can learn and offer the potential to make more accurate prediction. It is often known as "data never hurt in machine learning", as insufficient information cannot lead to powerful learning systems. In the context of health care, building an accurate predictive model depends on the quality and quantity of aggregate clinical data, which come from different hospitals or health care institutions. Consequently, in a real-world scenario, machine learning applications use data from several sources, including genetic and genomic, clinical, and sensor data. Day by day, many new sources of data are becoming available for instance, data from cell phones [104], wearable sensors [105], and participatory sensing applications [106]. For instance, there are wearable sensing frameworks that collect sensing information regarding heart rate, body temperature, caloric expenditure, etc, to train machine learning models. These models are then used for predictive analysis [105].

Data collection, storage, and processing power of a single institution is not always adequate to handle the large-scale data used in cutting-edge machine learning applications. For rare diseases, individual institutions oftentimes do not have sufficient data to calculate a model to achieve sufficient statistical power. Therefore, data sharing among multiple institutions is required. However, sharing sensitive biomedical data (clinical or genomic) exposes many security and privacy threats [107]. In case of data breach, there is a risk of sensitive personal information leakage. Therefore, in addi-

tion to addressing the fundamental goal of information retrieval, privacy-preserving learning also requires the learning algorithm to protect the confidentiality of the sensitive records of individuals.

In this chapter, we concentrate on secure and efficient computation for a fundamental technique used in numerous learning algorithms called $regression$ (see Methods). Regression analysis identifies the correlation among different attributes based on input data. Given a number of high-dimensional data points, regression analysis generates a best-fit line or curve through these points. To evaluate the fit, the value of a target attribute is predicted, which is associated with the given values of input. For instance, the input variables can be an individual's age, weight, sex, body mass index, and glucose level, while the output can be the likelihood to develop diabetes. Although regression analysis is widely used in practice, little work has been done in privacy-preserving regression analysis over a distributed dataset. Our objective was to perform the required computation for regression analysis without exposing any other information of user data.

4.1.1 Existing Techniques

To ensure the security and privacy of the sensitive data used in learning algorithm, different techniques (eg, garbled circuit [61], homomorphic encryption [23], differential privacy [63], and secure hardware [9]) have been adopted. But each of these techniques has certain shortcomings (eg, computational overhead, communication overhead, storage overhead, reduced data utility, and approximation error), which make these techniques difficult to use in real-world applications.

Wu et al. developed a framework, grid binary logistic regression (GLORE) [56], for developing a binary logistic regression model where data is distributed across different data owners. In their proposed approach, instead of sharing patient records, data owners send intermediary results to a central entity. These intermediary results are then used to build a prediction model without sharing patient-level data. However, in their approach, the intermediary results are exchanged in plaintext.

If the data size of a data owner is small, then sharing the intermediary results might compromise privacy.

Later, Shi et al. incorporated secure multiparty computation in GLORE. Their proposed framework, secure multiparty computation framework for grid logistic regression (SMAC-GLORE) [59], protects the confidentiality of intermediary results beside the patient data. However, SMAC-GLORE cannot handle numbers outside of a predefined range, and it does not scale well (e.g, it cannot efficiently handle data with more than 10 covariates). In addition, it uses a Taylor series approximation approach to evaluate the logit function. This approximation causes precision loss in the final output.

4.1.2 Regression Using a Hybrid Framework

There are two obvious but suboptimal solutions in terms of security and efficiency. Existing fully homomorphic encryption (FHE) techniques [23] provide rigorous security, but these solutions are not efficient. In existing homomorphic encryption schemes, with subsequent homomorphic operations, the noise (and size) of the ciphertext grows substantially, which increases computational and storage overheads to a great extent. There are some operations to reduce the size and noise of the ciphertext: bootstrapping [108] and relinearization [109]. However, these operations are very expensive from the computational point of view. Our proposed framework does not use these expensive operations at all, which enhances the efficiency of the framework greatly.

On the contrary, Software Guard Extensions (Intel SGX)-based solutions are very efficient but have some security concerns resulting from the recent discovery of side-channel attacks against SGX [66]. We developed our method so that only intermediary results, not individual records, are decrypted inside the secure hardware. Hence, a successful adversary would be unable to compromise the privacy of an individual.

Our proposed hybrid framework uses both techniques and provides a good trade-off in terms of security and efficiency.

4.1.3 Contributions

In this chapter, we propose a hybrid cryptographic framework for secure and efficient regression analysis (both linear and logistic). Our proposed framework leverages the best features of two secure computation schemes: somewhat homomorphic encryption (SWHE) and secure hardware (Intel SGX). In this framework, data resides at the data owners end. We assumed that data is horizontally partitioned, where all the records share the same attributes. Inspired by GLORE [56], we formulated the regression problem as decomposable parts. Data owners compute these decomposable intermediary results locally. Then, after encrypting these local results using homomorphic encryption, they send the encrypted intermediary results to an SGX-enabled central server. The central server now combines the intermediary results using a homomorphic addition operation. Then, these aggregate encrypted intermediary results are passed to the secure hardware hosted at the central server. Here, the aggregate intermediary results are decrypted and further computation is performed on plaintext. These computations involve matrix inversion and division, which are hard to handle in existing homomorphic encryption schemes. Finally, model coefficients are computed inside the secure hardware.

We summarize our contributions as follows:

(1) We address the limitations of existing secure computation schemes and propose a hybrid secure computation model for performing regression analysis over distributed data, which is more efficient and robust.

(2) We designed the framework in such a way that no homomorphic multiplication is necessary, which is an expensive operation. In addition, we do not need any bootstrapping or relinearization operation.

(3) In our proposed approach, a significant portion of computation is performed at the data owners' end on plaintext. In computation at a central server, after homomorphic addition operations, further computation is performed inside secure hardware on plaintext. Since most of the operations are

performed on plaintext, our proposed approach is very efficient. In addition, due to avoiding any kind of approximation technique, our proposed method does not introduce any precision loss in the final output.

4.2 Methods

4.2.1 System architecture

Our proposed framework has three main entities:

- *Data Owners:* These parties are geographically distributed and possess databases. Data can come from variety of sources including cell phones, wearable sensors, and relational databases. Data owners send encrypted intermediary results to the central server so that it can analyze the combined dataset.

- *Key Manager:* This generates and distributes the cryptographic keys that will be used for data encryption and decryption in different stages of our proposed framework. Each data owner gets a public key from the key manager and uses it for encrypting data.

- *Central Server:* The central server maintains communication with all the other entities of the framework. It receives data from the data owners, and computes the final result using SWHE and secure hardware.

4.2.2 Threat Model

In proposing this framework, our goal was to guarantee the confidentiality of data provided by different data owners. We assume that the central server is a semihonest party (also referred to as honest-but-curious), where it obeys the system protocol but may try to infer sensitive information by analyzing the system logs or received information [90].

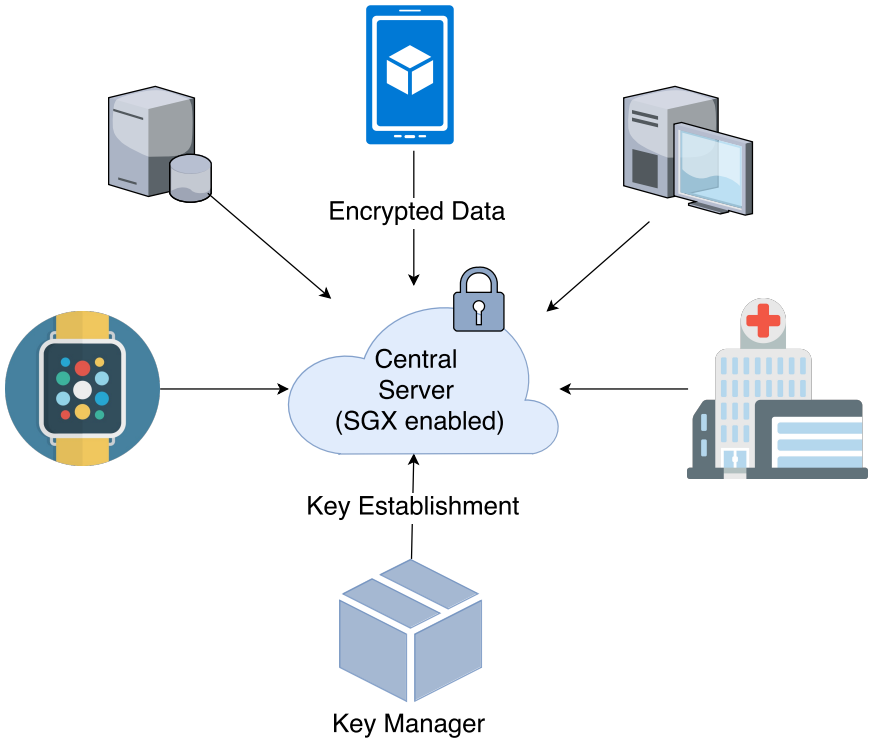


Figure 4.1: Block diagram of the system architecture.

We assume that the computation runs in an SGX-enabled central server. SGX architecture enables the central server to perform any computation securely on data provided by different data owners. We assume that the processor of the central server works properly and is not compromised. We trust the design and implementation of SGX and the cryptographic operations performed by it.

In general, side-channel attacks against SGX can be classified into two categories: physical attacks (where the attacker has physical access to the machine) and software attacks (these are launched by any malicious software running in the same machine) [110]. There has been no known successful physical attack against SGX. However, it is possible to exploit a type of software attack known as a synchronization bug [67]. Synchronization bugs are possible to exploit because an untrusted operating system can manipulate the thread scheduling of enclaves. However, it is only applicable for multithreaded applications, whereas our application is single threaded.

There is another type of well-known software attack, which is called a page-fault attack [66]. As the page tables are maintained in the operating system kernel and operated by the untrusted system software, page table entries can be manipulated to attack enclaves. But, since enclave pages are permission protected, malicious system software cannot compromise their integrity by manipulating them. However, Xu et al. [66] showed that, by clearing the present flag in the corresponding page table entries, the malicious software can generate traces of page access from the enclave. Although an adversary can observe access to different enclave pages, enclave memory can be treated as private at page-level granularity (4 kB) [93]. In other words, a different access to an enclave page is indistinguishable to an adversary. Further research is required to better understand the gap between the potential vulnerabilities of SGX and proposed defense mechanisms. Most of the existing defense mechanisms have been developed to address the page-fault side-channel attacks [68, 93, 94]. However, these mechanisms may not be effective for future attacks. Keeping these attacks in mind, we developed our framework to protect institutional privacy by combining the local inputs of participating institutions without decrypting them, therefore providing a higher layer of protection without introducing too much computational overhead.

We did not consider the aspects of adversarial machine learning through obtained outputs. Adversarial parties may try to infer sensitive attributes of data by model inversion attacks [111, 112].

4.2.3 Linear Regression

Suppose we are given a set of paired observations (x_i, y_i), for $i = 1, 2, ..., n$, and we want to generate the best-fit straight line for these points. This straight line is given by,

$$y = \beta_1 + \beta_2 x, \text{ for some } \beta_1, \beta_2 \tag{4.1}$$

The purpose is to explain the correlation between variable y and x. To evaluate the fit, the value of y is predicted that is associated with a given value of x. In the literature, y is called *the variable to*

be explained (or the *dependent* variable) and x is called the *explanatory variable* (the *regressor*, the *covariate*, or the *independent* variable) [113, Page 79].

Consider, the following simple linear regression model:

$$y = \beta_1 + \beta_2 x + \epsilon \tag{4.2}$$

Here, ϵ is the error we make in predicting y. For, $i = 1, ..., n$, we obtain n equations:

$$y_1 = \beta_1 + \beta_2 x_1 + \epsilon_1$$

$$y_2 = \beta_1 + \beta_2 x_2 + \epsilon_2$$

$$\vdots$$

$$y_n = \beta_1 + \beta_2 x_n + \epsilon_n$$

We can formulate this regression model using matrices.

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

In this way, the simple linear regression function can be represented by a short and simple equation:

$$Y = X\beta + \epsilon \tag{4.3}$$

The linear regression model with several explanatory variables is known as ***multiple linear regression***. This is given by:

$$y_i = \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \beta_k x_{ki} + \epsilon_i \ (i = 1, 2, ..., n) \tag{4.4}$$

Here, $x_{1i} = 1$ for $i = 1, 2, ..., n$. The function of 4.4 can also be expressed in matrix form which is more convenient.

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, X = \begin{bmatrix} 1 & x_{21} & \dots & x_{k1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{2n} & \dots & x_{kn} \end{bmatrix}, \beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \epsilon = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

It is noteworthy that equation 4.3 is also applicable for multiple linear regression.

Using ordinary least square estimate technique, we can show that (for details, see [113, Page 79]),

$$\beta = (X^T X)^{-1} X^T Y \tag{4.5}$$

Secure linear regression over distributed data: Each data owner D_i computes $X_i^T X_i$ and $X_i^T Y_i$ locally on plaintext. D_i then encrypts $X_i^T X_i$ and $X_i^T Y_i$ using homomorphic encryption. After receiving these intermediary results from all of the data owners, the central server then adds these using homomorphic addition operation to construct $X^T Y$ and $X^T X$. Further computation is performed inside the enclave after decryption. Our secure linear regression algorithm is illustrated in Algorithm 1.

$$\begin{aligned} X^T X &= \sum_{i=1}^{n} X_i^T X_i \\ X^T Y &= \sum_{i=1}^{n} X_i^T Y_i \end{aligned} \tag{4.6}$$

Algorithm 1 Secure Linear Regression

Input: Each data owner D_i provides encrypted $X_i^T X_i$ and $X_i^T Y_i$

Output: The model parameters(β)

1: Perform homomorphic addition over $X_i^T X_i$ for each data owner i.

2: Perform homomorphic addition over $X_i^T Y_i$ for each data owner i.

3: Send $X^T Y$ and $X^T X$ to enclave.

4: Inside the enclave decrypt, encrypted $X^T Y$ and $X^T X$.

5: Inside the enclave, compute $(X^T X)^{-1}$.

6: Finally, compute β inside the enclave .

Figure 4.2 illustrates the sequence diagram of our proposed method. At first, the key manager establishes the public key and private key. The private key is sent to the central server securely using remote attestation. The data owners then encrypt their data with the public key, and send the

encrypted data to the central server. Finally, the central server computes the model parameters.

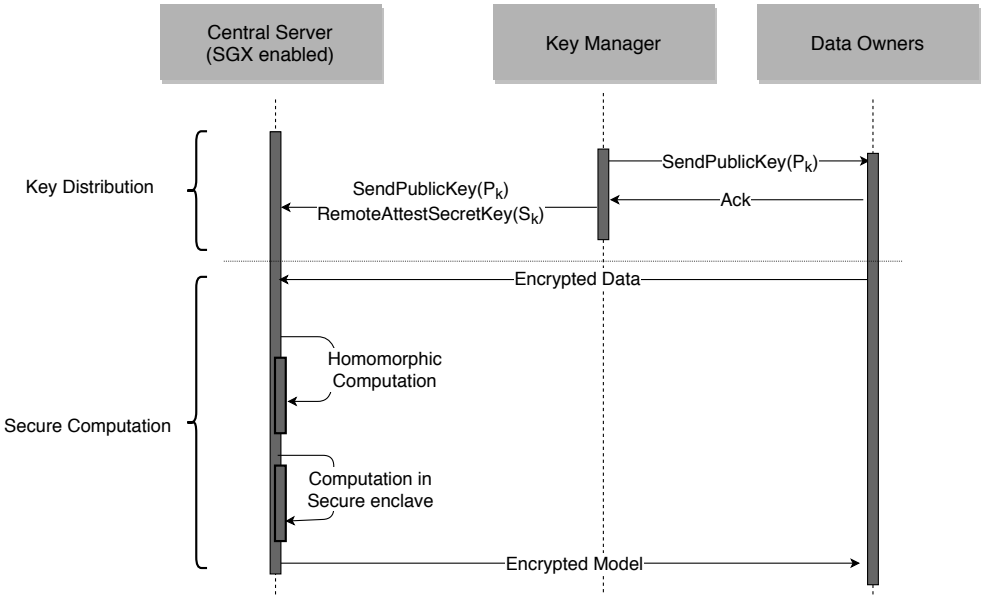


Figure 4.2: Sequence diagram of our proposed framework.

4.2.4 Logistic Regression

Logistic regression extends the principles of multiple linear regression to the case where the dependent variable y is binary (either 0 or 1). Like multiple linear regression, the independent variables can be categorical or continuous.

Instead of modeling the dependent variable directly, logistic regression models the probability of the dependent variable. Logistic regression uses the equation of linear regression (Equation 4.4). But, in that equation the value of the dependent variable can fall outside [0, 1]. Therefore, a nonlinear transformation is used, which is called *logit transformation*. The logit function takes any value x

and maps it into a value between 0 and 1. Logit function is given by: $\text{logit}(x) = \log\frac{x}{1-x}$.

$$\begin{aligned} &\text{logit}(P(y=1|x_1, x_2, ..., x_k)) \\ &= \log\frac{P(y=1|x_1, x_2, ..., x_k)}{1 - P(y=1|x_1, x_2, ..., x_k)} \\ &= \beta_1 + \beta_2 x_2 + ... + \beta_k x_k \end{aligned} \tag{4.7}$$

Therefore,

$$\begin{aligned} &P(y=1|x_1, x_2, ..., x_k) \\ &= \frac{exp(\beta_1 + \beta_2 x_2 + ... + \beta_k x_k)}{1 + exp(\beta_1 + \beta_2 x_2 + ... + \beta_k x_k)} \end{aligned} \tag{4.8}$$

where $\beta_1, \beta_2, ...\beta_k$ are unknown constants analogous to the multiple linear regression model. Probability$(y = 1|x_1, x_2, ..., x_k)$ denotes the probability that input$(x_1, x_2, ..., x_k)$ belongs to default class $(y = 1)$.

Logistic regression models are generally fit by maximum likelihood by using the conditional probability of y given x. Here, the Newton-Raphson method is used to solve the coefficients.
Let, X represents the matrix of x_i values, Y represents the vector of y_i values, , P be the vector of fitted probabilities with ith element $p(x_i; \beta^{old})$ and W be a $n \times n$ diagonal matrix of weights with ith diagonal element $p(x_i; \beta^{old})(1 - p(x_i; \beta^{old}))$. Then a Newton step is as follows:

$$\begin{aligned} \beta^{new} &= \beta^{old} + (X^T W X)^{-1} X^T (Y - P) \\ &= (X^T W X)^{-1} X^T W (X\beta^{old} + W^{-1}(Y - P)) \\ &= (X^T W X)^{-1} X^T W z \end{aligned} \tag{4.9}$$

In the second and third steps, the Newton step is expressed as a weighted least squares step, with the response $z = X\beta^{old} + W^{-1}(Y - P)$. This method is also known as *Iteratively Reweighted Least Squares (IRLS)*, since each iteration solves the weighted least square problem (see Friedman et al. [114] for details):

$$\beta^{new} \leftarrow \arg\min_{\beta}(z - X\beta)^T W (z - X\beta) \tag{4.10}$$

In practice, the W matrix is not computed explicitly because its size could be huge. If we have 1,000 rows of training data, matrix W would have 1,000,000 cells. For this reason, direct matrix operations with W may be very inefficient. Notice the beta update equation (Equation 4.9) has a term WX, which means the matrix product of W and X. Because most of the values in W are zero, most of the matrix multiplication terms are also zero. This allows W times X to be computed directly from P and X, without explicitly constructing W. Several of the math references that describe IRLS with the Newton-Raphson algorithm for logistic regression use the symbol $\widetilde{X}$ (X tilde) for the product of W and X.

$$\beta^{new} = \beta^{old} + (X^T \widetilde{X})^{-1} X^T (Y - P) \tag{4.11}$$

Secure logistic regression over distributed data: Each data owner D_i computes $X_i^T \widetilde{X}_i$ and $X_i^T(Y_i - P_i)$ locally on plaintext. D_i then encrypts $X_i^T \widetilde{X}_i$ and $X_i^T(Y_i - P_i)$ using homomorphic encryption. After receiving these intermediary results from all the data owners, the central server then adds these using homomorphic addition operation to construct $X^T \widetilde{X}$ and $X^T(Y - P)$. Further computation is performed inside the enclave after decryption. After computing β, the central server sends β to all of the data owners. For the next iteration, data owner i computes $X_i^T \widetilde{X}_i$ and $X_i^T(Y_i - P_i)$ using new β (received from the central server), and send these intermediary results to the central server. The central server then updates β using newly received $X_i^T \widetilde{X}_i$ and $X_i^T(Y_i - P_i)$. In this way, iterations continue until parameters converge. Our secure logistic regression algorithm is illustrated in Algorithm 2.

$$\begin{aligned} X^T \widetilde{X} &= \sum_{i=1}^{n} X_i^T \widetilde{X}_i \\ X^T(Y - P) &= \sum_{i=1}^{n} X_i^T (Y_i - P_i) \end{aligned} \tag{4.12}$$

Algorithm 2 Secure Logistic Regression

Input: Each data owner D_i provides encrypted $X_i^T \widetilde{X}_i$ and $X_i^T(Y_i - P_i)$, β is initialized to an all-zero vector

Output: The model parameters

1: Receive encrypted $X_i^T \widetilde{X}_i$ and $X_i^T(Y_i - P_i)$ from each data owner D_i

2: Perform homomorphic addition over $X_i^T \widetilde{X}_i$ for each data owner D_i

3: Perform homomorphic addition over $X_i^T(Y_i - P_i)$ for each data owner D_i

4: Send encrypted $X^T \widetilde{X}$ and $X^T(Y - P)$ to enclave.

5: Inside the enclave, decrypt $X^T \widetilde{X}$ and $X^T(Y - P)$

6: Update $\beta^{new} = \beta^{old} + (X^T \widetilde{X})^{-1} X^T(Y - P)$

7: If the stopping criteria is satisfied then stop; otherwise, send β to each data owner and go to step 1

4.2.5 Implementation

We developed our proposed framework using C++. For SWHE, we used the *Simple Encrypted Arithmetic Library* (SEAL) [27]. SEAL is an easy-to-use homomorphic encryption library, with no external dependencies. There is another homomorphic encryption framework called HElib [115], but we choose to use SEAL for its simplicity.

4.3 Results

4.3.1 Experimental Settings and Dataset

We performed experiments in a machine with Intel Core i7-6700 (3.40 GHz) processor and 8 GB memory. We used Intel SGX software development kit version 1.7. We simulated 2 data owners and the central server in this machine. Table 4.1 shows the SEAL parameters.

Table 4.1: Parameters used for Simple Encrypted Arithmetic Library.

Parameter	Value
Polynomial modulous	$x^{1024} + 1$
Plaintext modulous	$1 << 8$
Decomposition bit count	32
No. of coefficients reserved for fractional part	64

We performed experiments using Haberman's Survival Dataset from UCI Machine Learning Repository [116] and the Longitudinal Low Birth Weight Study dataset from Hosmer and Lemeshow [117]. The records of the datasets were evenly distributed between the 2 data owners.

Table 4.2 demonstrates the datasets with their sizes.

Table 4.2: Size of datasets used for experiments.

Dataset	No. of instances	No. of features
Haberman	270	3
Birth Weight	488	8

4.3.2 Results and Analysis

Table 4.3 shows the experimental results. For SWHE, most of the computation time was due to homomorphic operations. Our proposed framework avoided expensive homomorphic multiplication by transferring the later phase of computation to the secure hardware. In addition, we needed to decrypt only the intermediary results, not every individual attribute value. Consequently, our proposed framework was more efficient than solely secure hardware (SWHE)-based technique (where every individual attribute needs to be decrypted) and the SWHE-based technique (which involves

Table 4.3: Experimental Results

Dataset	Linear regression				Logistic regression			
	Plaintext	Proposed method	Secure h/w (SWHE)	Secure h/w (AES)	Plaintext	Proposed method	Secure h/w (SWHE)	Secure h/w (AES)
Haberman	6 ms	8.991 s	259.908 s	4.30 s	171 ms	27.037 s	264.669 s	4.65 s
Birth Weight	25 ms	39.382 s	880.228 s	8.54 s	886 ms	162.544 s	904.718 s	8.64 s

many expensive homomorphic multiplication and relinearization operations). Table 4.3 does not report the results for the SWHE-based technique. However, according to our empirical results, it took more than 2 hours for the Haberman dataset and more than 17 hours for the Low Birth Weight Study dataset for both kinds of regression analyses.

We want to emphasize that, although the secure hardware (Advanced Encryption Standard [AES]) method is faster, state-of-the-art attack models targeting SGX show that solely secure hardware-based approaches might expose data from participating institutions to potential attackers (as explained above). Our method, although a little bit slower, preserves such institutional privacy by combining the local inputs without decrypting them; therefore, it offers a stronger security guarantee without imposing too much computation or storage cost. In this way, our proposed hybrid model provides a good trade-off in terms of security and efficiency.

Table 4.4 shows the storage overhead of the solely secure hardware-based approach. For SWHE, times required to encrypt the datasets were 4.37 minutes for the Haberman dataset and 18.46 minutes for the Low Birth Weight Study dataset. For AES, times required to encrypt the datasets were 14 milliseconds for the Haberman dataset and 38 milliseconds for the Low Birth Weight Study dataset.

Table 4.4: Storage Overhead for the Secure Hardware approach

Overhead before and after encryption	Dataset	
	Haberman	Low Birth Weight Study
Before encryption (KB)	3.8	28
After encryption (SWHE) (MB)	30.3	123
After encryption (AES) (MB)	36	143

4.4 Discussions

4.4.1 Comparison with Prior Work

There is a homomorphic encryption-based implementation of linear regression [118], which required 2 days to compute on a dataset containing 51,000 input vectors of 22 features with a key size of 1024 bits. That matrix inversion procedure took 1 day to complete because matrix inversion is a very expensive computational task in homomorphic encryption. In our proposed method, we performed matrix inversion on plaintext in secure hardware, which is much more efficient.

Hall et al. [118] proposed an iterative matrix inversion algorithm, which introduces approximation errors when a fixed number of iterations is used. Their method offers a low accuracy of 10^{-3}. Precision can be slightly improved by choosing greater values for the 2 constants used by their method. However, this would require a larger public key, which would introduce significant computation overhead. In contrast, in our proposed method, there is no approximation error: the regression coefficients are completely identical to the plaintext results.

4.4.2 Security Discussions

In the Methods (Threat Model subsection), we discussed the security of SGX, specifically different side-channel attacks on SGX, and how we treat those attacks in our proposed framework. Address-

ing these attacks, we developed our framework in such a way that it can protect institutional privacy by combining the local inputs of participating institutions without decrypting them. This approach provides a higher layer of security without imposing too much computational cost.

In our proposed method, only intermediate values (eg, X^TY, X^TX) are decrypted inside secure hardware. Even if the hardware is compromised (or, in case of a side-channel attack), it is not possible to retrieve any sensitive attribute from those intermediary results. Hence, our proposed hybrid model not only achieves good performance but also guarantees stronger security than the solely SGX-based techniques. Dowlin et al. [27] and Pass et al. [13] discussed the security of SEAL and Intel SGX further.

A symmetric cryptosystem like AES requires n remote attestations to distribute the key to n data owners, which results in much more network communication, which might be prone to attack. In contrast, our proposed framework relies on public-key cryptography, where the data owners use a public key to encrypt their data published by the key manager. In this way, our proposed method reduces the attack surface of the system model, makes key distribution much simpler, and avoids additional communication overhead.

4.4.3 Limitations

There are some limitations of our proposed framework.

First, we did not consider the issue of model privacy. Several works based on differential privacy have addressed inference attacks (eg, model privacy [119]). These solutions are complementary to our proposed method and can be readily incorporated into a single framework.

Second, the central server of our proposed method must be SGX-enabled; that is, it must use an Intel processor of sixth generation or later.

Third, since computing coefficients for logistic regression require multiple iterations, all parties must be synchronized until coefficients converge. However, linear regression does not require mul-

tiple iterations. So, in this case, parties can be offline just after sending their intermediary results.

4.4.4 Generalizability

Others have addressed training machine learning models (eg, support vector machines [120]) over distributed data [121, 122]. Our proposed method can be easily applied to this kind of technique.

4.4.5 Cost of Deployment

The Intel SGX feature is available in all Intel Skylake and KabyLake processors. The price of an Intel Skylake or Kaby Lake processor is identical to that of processors from other vendors (having similar configuration). Price ranges from US$42 to US$1207 depending on configuration [123]. Recently, Microsoft started using SGX-capable servers in their Azure confidential computing service [124]. Azure confidential computing is offering the developers the ability to develop applications on top of Intel SGX software development kit. Apparently, there will be no significant additional charge for using this service.

4.5 Conclusion

In this age of big data, data need to be analyzed to uncover valuable insights and patterns. But this kind of analysis poses a threat to individual privacy, since data often contain sensitive information. In this chapter, we address this data security and privacy issue and propose a hybrid cryptographic framework to overcome the limitations of the existing cryptographic techniques. We think that secure hardware assisted predictive analysis of biomedical data is very promising for healthcare and medical research.

In future work, we will investigate the applicability of our proposed method to other learning algorithms such as neural networks, support vector machines, and decision trees.

Chapter 5

A Secure and Efficient Framework for Distributed Clinical Notes De-identification

Medical data sharing is a big challenge in biomedicine, which often hinders collaborative research. Due to privacy concerns, clinical notes often cannot be directly shared. Many effort has been dedicated to de-identifying clinical notes but it is still very challenging to accurately detect and scrub all sensitive elements from notes in an automatic manner. An alternative approach is to remove sentences that might contain sensitive terms related to personal information. A previous study introduced a frequency-based filtering approach that removes sentences containing low frequency bigrams to improve the privacy protection without significantly decreasing the utility. Our work extends this method to consider notes from distributed sources with security and privacy considerations. We developed a novel secure protocol based on private set intersection and secure thresholding to identify uncommon and low-frequency terms, which can be used to guide sentence filtering.

5.1 Introduction

Clinical notes represent an indispensable component of electronic health records (EHRs), which contain important information (such as symptoms and medical history) that structured data might not cover. Sharing clinical notes can promote research, improve healthcare services, and contribute to clinical decision support [125, 126]. However, it has been a very challenging task to de-identify such data to mitigate the privacy risks. Due to the unstructured nature of notes, de-identification is not as straightforward as the structured data. To satisfy the privacy regulations of Health Insurance Portability and Accountability Act (HIPAA) [127], we can remove the Protected Health Information (PHI) defined in the HIPAA safe harbor method [128]. Traditionally, this is done through the detection and scrubbing of 18 specific categories of PHIs including name, social security number, dates, etc. Many efforts have been devoted in this direction including both the manual and the automatic approaches. Manual approaches to identify PHI are prone to mistakes (Neamatullah et al. [129] shows the recall of 14 clinicians to detect 130 clinical notes varied from 0.63 to 0.94) and they are also expensive (e.g., $50/hour to read and label 20k words/hour in de-identifying MIMIC II database [130]). Automated algorithms can save time and reduce the human review efforts. Early systems used rule or template based approaches to match and detect PHI [131, 132]. Berman [133] developed a concept matching algorithm that steps through confidential pathology text to replace medical terms matching standard nomenclature code with a synonymous term while keeping the high frequency "stop words" intact. However, the system blocks too much and has a high false positive rate, making the outputs hard to read [129]. A similar method was proposed by Finley et al. in which words that appear in a list of patient names and do not appear in the Unified Medical Language System (UMLS) Metathesaurus are removed [134]. This method was successfully applied to de-identify distributed semantic models. Scrub [135] used a template-based approach to match components of high privacy risk, which are then removed, generalized, or replaced with

made-up ones. This method can get rid of explicit personally-identifiable information but it does not handle combinations of fields and the results might still be matched or linked to the identities of individuals [136].

Other researchers also treated text de-identification as a Named Entity Recognition (NER) problem and tried to solve it with machine learning models [137]. Szarvas et al. used decision tree to take into consideration of various features (length, frequency, etc.) to detect PHIs [138]. Several groups [129, 139, 140] developed methods based on Support Vector Machine (SVM) to classify sensitive attributes based on Part-of-speech (POS) inputs. Another popular framework is to utilize conditional random fields (CRF), an extension of logistic regression that considers correlations in the sentence to predict PHIs [141–143]. Latest methods in this direction [144] using deep learning approaches have reported improved performance in detecting PHIs but the model requires careful tuning of parameters for each dataset, which makes it hard to be portable for collaborative research.

A recent method was proposed by Li et al. [12] to filter out rare sentences ($frequency < 3$) and sentences containing bigrams under a certain frequency threshold ($frequency < 256$). This method demonstrated good performance in obtaining sentences with almost no PHIs (evaluated by a manual review on sampled outputs) while preserving a similar *Type Unique Identity* (TUI) distribution of the original data, providing an alternative and generalizable way to obtain useful data with mitigated privacy risks. However, the method is only designed to anonymize data from a single source. In reality, collaborative research often involves more than one party and poses new challenges to conduct filtering in a global manner. In this chapter, we propose a distributed and privacy-preserving method as an extension of the single source model [12]. In our scheme, rare sentences will be filtered at each local site considering the low frequency threshold and differences in sentence patterns from site to site. Our criterion for bigram filtering is stricter by taking distributional differences of local sites into consideration. We will only keep sentences containing bigrams observed at all collaboration sites with sufficient global frequency. To develop such a global bigram-based fil-

tering method, appropriate protection needs to be enforced on private set intersection, secure count aggregation, and thresholding to ensure data confidentiality during the process.

5.2 Existing Techniques

A critical step for our distributed bigram filtering model is to find what the bigrams in common are among all collaborative sites in a privacy-preserving manner. Although there are several previous works on 2-party private set intersection, little work has been done to solve multi-party private set intersection (MPSI) problem. Earlier approaches for MPSI have some limitations. In [145], the dataset size of each party must be equal. Another approach suffers from approximation errors [146]. Some recent works have shown the feasibility of handling $n > 2$ parties [147, 148]. In these works [147,148], each data owner constructs a Bloom filter from their data (using only the words or bigrams, not the count associated with them). Data owners send the encrypted (exponential ElGamal encryption scheme) Bloom filter to a service provider. All encrypted Bloom filters are securely added by the service provider without decrypting, which results in an encrypted Integrated Bloom Filter (IBF). Then the service provider constructs a randomized n-subtraction of IBF (encrypted), where n is the number of parties. At this point, the service provider broadcasts this encrypted randomized n-subtraction of IBF to all the data owners. Finally, all data owners jointly decrypt it and compute the set intersection: if an element x is in the set intersection, the corresponding array locations in the encrypted randomized n-subtraction of IBF, where x is mapped by k hash functions is an encryption of 0; otherwise, is an encryption of random integer. Their approach [147, 148] demonstrated good performance for set sizes range from 64 to 16,384. However, their approach may not scale well with millions of records, which is common in real world applications. With a much larger set, to reduce the probability of false positives, the size of the Bloom filter should be large enough compared to the number of items to be inserted in it. In their approach,

runtime is dominated by the encryption and decryption of Bloom filter. Constructing, encrypting, and transferring such large Bloom filters (that can deal with millions of records with a minimal probability of false positives) will introduce huge computation and communication overhead. Our problem specification is different from the works on private set intersection mentioned here, which do not involve any secure thresholding operations. We are describing these works just to give an overview of state-of-the-art solutions of the related problems. To the best of our knowledge, there is no secure protocol for sensitive information filtering that combines private set intersection and secure thresholding.

5.3 System Overview

We developed a secure and privacy-preserving framework for bigram-based filtering to simultaneously meet two goals: *multiparty private set intersection* and *secure thresholding*.

5.3.1 Architecture and Entities

There are three types of entities in the system.

- *Data owner*: Data owners might be any hospital, clinical research facility, or federal (or, provincial) health science institute that possess clinical datasets. Our proposed system supports any number of data owners.

- *Crypto Service Provider (CSP)*: Cryptographic Service provider manages public and private keys. CSP also manages salt for hashing (refer to Security Analysis, Security of Hashing for more details). Each data owner receives a public key and a private key from the CSP. Data owners use this public key to encrypt their data (count of bigram), and use private key to decrypt the encrypted response from the central server.

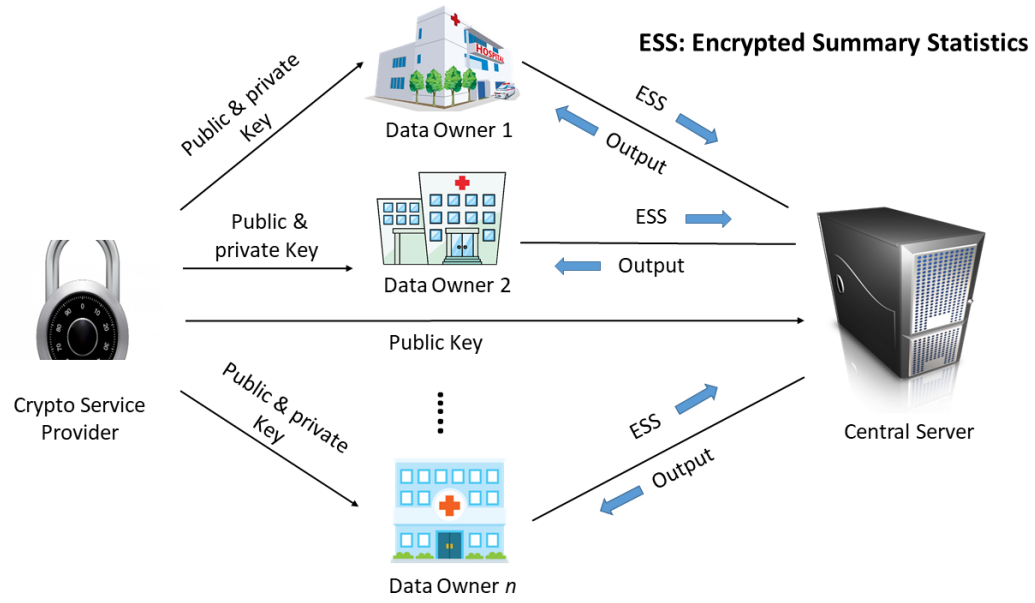


Figure 5.1: Block diagram of the system architecture. Only encrypted summary statistics are delegated to the central server to conduct the bigram filtering, which returns to individual data owners with encrypted bigrams that are both common and frequent enough in a global manner.

- *Central Server*: The central server coordinates the system protocol. It maintains communications with all other entities of the system. It receives encrypted data (hash and encrypted count of bigram) from the data owners, performs computations locally, and finally sends the encrypted result to the data owners.

5.3.2 Threat Model

In this work, our goal is to ensure each data owner knows the thresholded set intersection as a result of the protocol. Data owners should not know the elements of other data owners' dataset (elements that are not in the intersection). We consider the central server as a semi-honest party (also known as honest-but-curious). It follows the protocol but may attempt to scoop additional information from the server logs or received messages. We also assume that the data owners do not collude.

5.3.3 Problem Specification

The objective of this study is to identify the globally frequent common bigrams of participating parties based on a threshold value. In the first phase of the system protocol, all the parties jointly identify the common bigrams. Then, data owners send counts of the common bigrams to the central server (please see Detailed System Protocol). Consider the example of table 5.1. Here, data owner A sends E(count of bigram Flu-fever = 10), E(count of bigram Cancer-pain = 15), and E(count of bigram Diabetes-glaucoma = 20), where E denotes an encryption algorithm. The central server performs addition over the encrypted bigram counts after receiving counts from all data owners. If the total count for a specific bigram is less than or equal to a predetermined threshold, then that bigram is considered privacy-sensitive, and this information can be used to guide sentence filtering of clinical notes. The intuition behind this filtering is: the more potentially identifying a bigram is, the rarer it will be.

Table 5.1: Identification of globally infrequent bigrams

Data Owner	Frequency of the bigram Flu-fever	Frequency of the bigram Cancer-pain	Frequency of the bigram Diabetes-glaucoma
A	10	15	20
B	20	15	10
C	5	15	25
Total	35	45	55

Let us consider the data of the above table. Assume, the threshold value is 40. Since total count of Flu-fever (35) is less than the threshold value (40), it will be considered privacy-sensitive.

5.4 Background

5.4.1 Ciphertext Packing

The considerable computational overhead of homomorphic encryption results from the large ciphertexts. As homomorphic operations have to operate on these large ciphertexts, they can be quite slow. The primary solution to deal with this issue is to work with packed ciphertexts, which refer to the ciphertexts that encrypt a vector of plaintext values [149, 150]. Homomorphic operations can be performed on these vectors component-wise in a Single Instruction, Multiple Data (SIMD) manner. Depending on the memory allowance, this mechanism can significantly boost the performance due to parallelization.

Consider the plaintext elements in a polynomial quotient ring $m \in R_t = Z_t/(X^n + 1)$ and ciphertext elements in $R_q = Z_q/(X^n + 1)$. Here, q and t are positive integers ($q > t, q > 1$, see [31]), Z_q represents the set of integers $(-\frac{q}{2}, \frac{q}{2}]$, and $X^n + 1$ is an irreducible polynomial of degree n. Using ciphertext packing, we can encrypt n plaintext values in a single ciphertext for a single instruction execution. Since a packed ciphertext is essentially the same as a standard ciphertext, the basic homomorphic operations still work, for instance, homomorphic addition by adding ciphertexts. Ciphertext packing thus facilitates SIMD-type homomorphic computation, which is capable of computing the same function over many inputs at once. The usage of ciphertext packing in our proposed framework is elaborated in Detailed System Protocol. We apply ciphertext packing to minimize both computational and communication overhead. The data owners groups their counts of bigrams into vectors of length n, encrypt them, and send $\texttt{Cardinality of Sets}/n$ ciphertexts to the central server (see Detailed System Protocol). Then the packing mechanism allows the central server to perform computation on n items simultaneously, which results in n-fold improvement in computation and communication both. In our case, n is equal to 4096, which results to a significant time cost reduction over the naive homomorphic encryption method.

5.4.2 Hash Functions

Hash functions are one of the fundamental cryptographic primitives. Hash functions can compute a digest of a given message, which is a fixed-length bit string. For a given message, the message digest (also known as hash value or hash) can be considered as an unique representation of that message. In this work, we have used SHA-256, which is a member of Secure Hash Algorithm (SHA) family. The length of message digest for SHA-256 is 256 bits [151]. Security of hashing is discussed in detail in Security Analysis, Security of Hashing.

5.5 Detailed System Protocol

At the system initialization phase, data owners receive public and private keys from the CSP. Also, the central server receives only the public key. Then, each data owner sends the hashes of bigrams to the central server. After receiving the hashes from each data owner, the central server computes the intersection of the hashes. Then, the central server sends the elements of this intersection to data owners. Figure 5.2 shows the flow diagram of our proposed protocol.

Upon receiving the hashes from the central server, data owners encrypt the local frequency of the intersected bigrams by using ciphertext packing technique. To do so, they follow the order received from the central server. Figure 5.3 illustrates this technique for a data owner and indicates the difference with naive homomorphic encryption approach. After encrypting the counts, data owners send the packed ciphertexts to the central server, where the encrypted global frequency will be computed. After receiving the ciphertexts, the central server performs a homomorphic addition operation on these packed ciphertexts. So, at the end of this addition process, the resulting output looks like the Table 5.2. Here, E represents the encryption function.

In Table 5.2, $E(C11)$ denotes the encrypted count of bigram B1 contributed by data owner 1. $E(C12)$ denotes the encrypted count of B1 contributed by data owner 2, $E(C13)$ denotes the en-

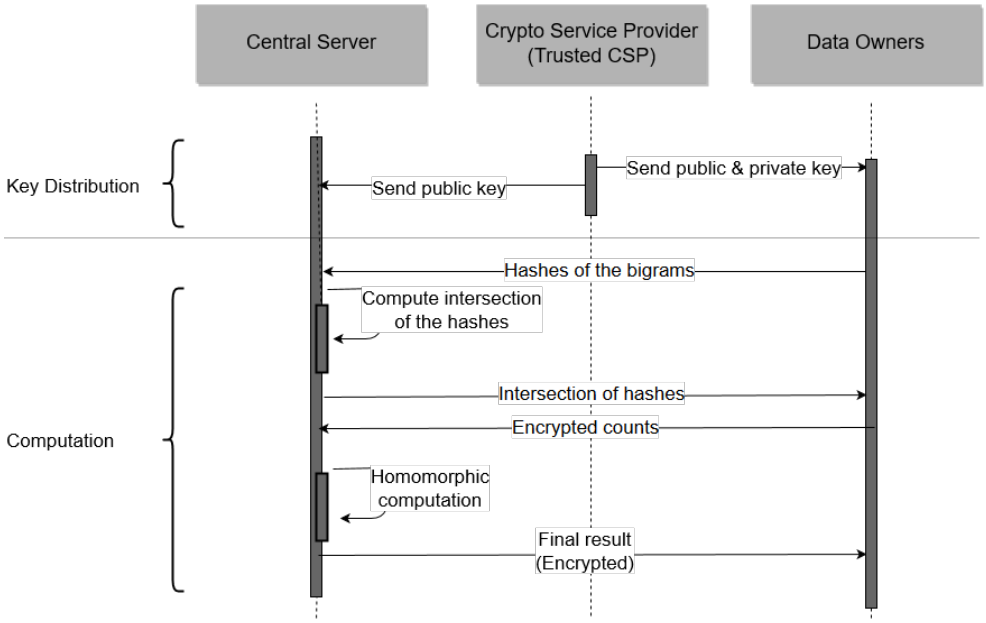


Figure 5.2: Flow diagram for the proposed system protocol. The order of the execution runs in a top down manner in key distribution and computation phases.

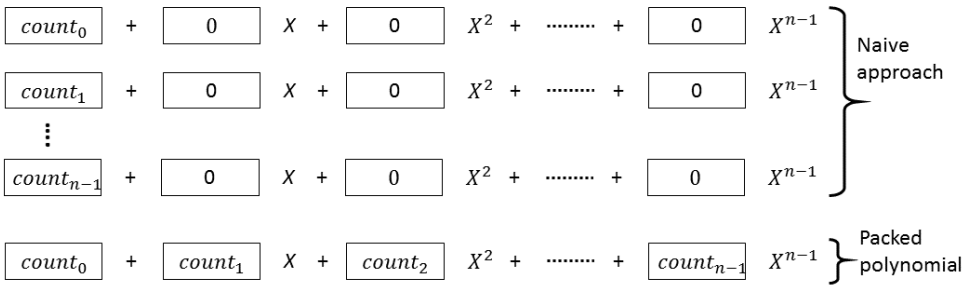


Figure 5.3: Usage of ciphertext packing in our proposed method. Here, n is the degree of the polynomial, which indicates the number of slots for parallel computing.

crypted count of B1 contributed by data owner 3, and so on.

Now we need to meet the thresholding requirement for the sum of homomorphically encrypted counts. For each of the records we check the following inequality.

$$E(C11) + E(C12) + E(C13) + \cdots > \text{threshold}$$

Table 5.2: Secure count aggregation at central server.

Bigram	**Encrypted Global Frequency**
B1	$E(C11) + E(C12) + E(C13) + \ldots$
B2	$E(C21) + E(C22) + E(C23) + \ldots$
B3	$E(C31) + E(C32) + E(C33) + \ldots$
$\vdots$	$\vdots$

Solving this problem involves both addition and comparison. It is known that in arithmetic circuits, addition is cheap but comparison is not trivial. To avoid the comparison operation in the arithmetic circuit, we formulate the problem in the following way,

$$E(C11) + E(C12) + E(C13) + \cdots - \text{threshold}$$

After performing the above mentioned homomorphic operation, the central server sends to the data owners $r * (E(C11) + E(C12) + E(C13) + \ - \text{threshold})$, where r is a random number drew by the central server. After decrypting it, if a data owner gets a random negative number (or zero), she will understand that the sum of counts of the corresponding record is less than (or equal to) the threshold. Similarly, if a data owner gets a random positive number, she will understand that the sum of counts of the corresponding record is greater than the threshold.

Multiplying every coefficient of the resulting ciphertext by same random number may expose some additional information about other data owners' counts. So, we multiply the resulting ciphertext with a random polynomial, all of whose coefficients are randomly generated.

Although polynomial addition and subtraction are coefficient-wise by nature, polynomial multiplication in R_t(and R_q) is a convolution product of the coefficients. An effective technique to transform convolution product into coefficient-wise product in polynomial ring is the Number-Theoretic Transform (NTT). NTT is a specialization of Fourier transform for finite rings. One important prop-

erty of NTT is that it works in the same ring as lattice cryptosystems do. Therefore, NTT can be used to improve the efficiency of the polynomial operations. NTT is commonly used in ciphertext space to speed up polynomial multiplication [152]. To ensure that the products in the ciphertext space be translated into coefficient-wise products in plaintext space, we perform an inverse-NTT operation to plaintext before encryption and a NTT operation after decryption. The NTT operation does not have any effect on addition and subtraction, as they are linear in nature.

5.6 Experimental Settings

5.6.1 Dataset

We have used the MIMIC-III (Medical Information Mart for Intensive Care), an openly available dataset comprising of de-identified health data associated with ˜40k critical care patients [153]. To be specific, we used `NOTEEVENTS` table of this database, which contains de-identified clinical notes including nursing and physician notes, and reports on ECG, radiology, and discharge summary. There are 2,083,180 rows in `NOTEEVENTS` table.

5.6.2 Dataset Preprocessing

The `text` column of `NOTEEVENTS` table represents the contents of the clinical notes. At first, we remove the stop words [154] from the entries of this column. Stop words [154] are frequently occurring words in English, which are semantically unimportant. We also removed any standalone symbol/character, numerical values (for instance, heart rate, arterial blood pressure, respiratory rate, or dose of a prescribed drug) including temporal expressions like 4:10 AM, 9:50 PM etc.

5.6.3 Evaluation Environment

Experiments were performed on Google Compute Engine (GCE) and Amazon EC2 cloud server. GCE is a cloud computing service that provides virtual machines running in Google's data centers. In GCE, we used a `n1-standard-8` machine with Ubuntu 16.04.3 LTS. For Amazon EC2, the configuration was `r3.xlarge` with Ubuntu 16.04.2 LTS. The central server was hosted in Amazon EC2 and the CSP and the data owners were hosted in GCE. Each entity of the system architecture communicated with others through TCP.

5.6.4 Implementation

To hash the words, SHA-256 (OpenSSL version 1.0.2g) was used. To encrypt the bigram counts, we used FV scheme [31]. For FV implementation, we choose NFLlib [155]. NFLlib [155] is an efficient and scalable C++ library for ideal lattice cryptography. This library is much faster than NTL [156] and FLINT [157] due to its algorithmic and programming optimizations.

The computation and communication tasks were processed in parallel whenever possible. We used OpenMP for this purpose.

5.7 Experimental Results and Discussions

It is evident from the methodology of our proposed method that the runtime mostly depends on the cardinality of the intersection of the sets and the number of data owners. We evaluated our proposed method in terms of these two factors. Tables 5.3 and 5.4 show the experimental results. These tables report computation time for intersecting hashes, encryption, homomorphic operation, decryption, and network communication costs. However, the total time reported here does not include cost for system initialization, for instance, reading and parsing configuration file, reading input data file, TCP socket setup and shutdown etc.

Table 5.3: Experimental results for different cardinality of intersection of sets. In the five different settings, cardinality is increased by 1% of the entire dataset. Here, the number of data owners is a constant (3). The numbers are in seconds.

Cardinality of intersection	**Intersecting hashes (s)**	**Encryption (s)**	**Homomorphic operation (s)**	**Decryption (s)**	**Network Comm. (s)**	**Total Time (s)**
1,515,520 (∼10%)	4.63	8.11	55.43	6.73	0.48	75.38
1,667,072 (∼11%)	4.69	8.92	61.19	7.06	0.52	82.38
1,818,624 (∼12%)	4.98	9.70	66.63	7.88	0.54	89.73
1,970,176 (∼13%)	5.07	10.97	72.21	8.49	0.59	97.33
2,121,728 (∼14%)	5.20	11.32	77.65	9.34	0.60	104.11

5.7.1 Communication Cost

The total number of bigrams was about 15 million. These were equally distributed among three data owners for the experiments shown in Table 5.3. Each data owner was given 4 million disjoint bigrams along with common bigrams as shown in Table 5.3. For the five different settings, the sizes of encrypted data for each data owner were 46.3, 51, 55.6, 60.2, and 64.8 MB respectively. The sizes of the files containing hash (for each data owner) were 341, 351, 360, 370, and 379 MB respectively.

For the experiments shown in Table 5.4, bigrams were distributed equally among the six data owners (3,518,464 each). The size of the encrypted data for each data owner was 46.3 MB. The size of the file containing hash (for each data owner) was 218 MB.

5.7.2 Concept Distribution Analysis

Now, we show that the proposed method is able to retain enough information for data analysis. We compare the concept distribution of clinical notes and sanitized sentence repository constructed by

Table 5.4: Experimental results for different number of data owners. The cardinality of intersection of sets was fixed (1,515,520). The numbers are in seconds.

Number of Data Owners	**Intersecting hashes (s)**	**Encryption (s)**	**Homomorphic operation (s)**	**Decryption (s)**	**Network Comm. (s)**	**Total Time (s)**
2	1.69	8.17	54.72	6.29	0.32	71.19
3	2.72	8.19	55.49	6.33	0.39	73.12
4	3.53	8.28	55.51	6.60	0.46	74.38
5	4.63	8.22	56.36	6.67	0.53	76.41
6	5.36	8.24	58.01	7.11	0.60	79.32

eliminating sentences of the clinical notes that contain low frequency bigrams (frequency less than or equal to a specified threshold). Due to the significant computations involved, we sampled 800 clinical notes for this experimentation. The results of concept distribution analysis are reported in Table 5.5. Each concept is expressed as a type unique identity (TUI) defined by UMLS [158]. The difference of the TUI distriction is not too large when the threshold is small but it gets larger at an increasing threshold. However, this is not a critical issue because we can maintain the original distribution by oversampling the filtered corpus using sentences that contain one or more TUIs. This is a standard combinatorial optimization problem but we do not explore it in this work.

5.8 Security Analysis

In this section, we analyze the security of cryptographic primitives that we used in our proposed framework.

Table 5.5: Comparison of TUI Proportion Distribution

TUI	**Original Clinical Note**	**Threshold = 1**	**Threshold = 2**	**Threshold = 4**	**Threshold = 8**	**Threshold = 16**
T007	0.2627	0.2012	0.1601	0.1421	0.0922	0.0428
T023	5.8168	4.4492	3.5281	2.9490	2.5213	2.1758
T033	7.7646	5.3959	4.847	3.6402	3.1259	2.5570
T047	7.6978	5.4338	4.8742	3.7598	3.3876	2.8825
T060	2.5509	1.8672	1.6446	1.4018	1.1242	0.9680
T074	1.5871	1.2046	1.0991	0.9302	0.8257	0.6724
T093	0.9824	0.7123	0.6594	0.5846	0.5197	0.4925
T109	4.1908	2.8163	2.7084	2.8069	2.6024	1.6447
T121	1.2840	0.8898	0.8983	0.7719	0.5971	0.6253
T170	0.7523	0.5182	0.4450	0.3165	0.2764	0.1284
T184	3.5566	2.4968	2.2498	1.8443	1.4265	0.6895
T201	1.8249	1.1075	0.9960	0.9173	0.8441	0.8437

5.8.1 Security of FV Cryptosystem

To evaluate the security of a lattice cryptosystem a widely used measure is root-Hermite factor δ. Lindner and Peikert showed a mathematical relationship between root-Hermite factor and security level λ (in bits) [159].

$$\lambda = 1.8 \times \frac{1}{log_2\delta} - 110 \tag{5.1}$$

$\frac{1}{log_2\delta}$ is given by $\frac{1}{log_2\delta} = \frac{4n(log_2 q)}{(log_2(\frac{cq}{s}))^2}$, where $c = \sqrt{\frac{ln\frac{1}{\epsilon}}{\pi}}$ and $s = \sigma\sqrt{2\pi}$. n, q, and s represent the degree of the polynomial ring, ciphertext modulus, and scale parameter of the error distribution respectively. σ denotes the standard deviation of the error distribution, and is the attacker advantage. For our experiments, we choose $n = 2^{12}, q = 2^{120}, \sigma = 3, \epsilon = 2^{-32}$. According to root-Hermite factor measure, our proposed method guarantees 142 bit security.

5.8.2 Security of Hashing

One of the primary security requirements of hash function is one-wayness: given a hash output h, it must be computationally infeasible to find an input m such that $h = H(m)$. In other words, given a message digest, an adversary cannot find out the matching message m from $H^{-1}(h) = m$. There exist some cryptanalytic attacks against one-way hashing that try to break the security properties of the hash function. Brute-force attack (also known as exhaustive search) is a type of cryptanalytic attack. Let (m, h) denote the pair of input message and output hash value, and let $M = \{m_1, m_2, ..., m_k\}$ be the message space of all possible messages m_i. Such an attack checks for every element of M if $H(m_i) == h$. If this equality holds, a possible input message is found. This type of attack is impractical for large message space. A similar one is called dictionary attack, which tries all the input messages in a pre-arranged listing, generally derived from a list of words such as in a dictionary (hence the term *dictionary attack*), which has a smaller space to search. There is a variant of dictionary attack, known as Rainbow table attack [160], which uses a a precomputed table (Rainbow table [160] that contains elements up to a certain length consisting of a limited set of characters) for reversing hash functions. This attack requires less computation time but more storage compared to brute-force attack.

Addressing the above mentioned attacks, we used salt to randomize the hashing. In cryptography, salt refers to random data that are used as an additional input to a hash function. Salt was generated by the CSP and provided to data owners before each hashing process, which makes these attacks computationally infeasible.

Another desirable property of a hash function is collision resistance. A hash function is said to be collision resistant if it is computationally infeasible to find two different inputs m_1, m_2 with $H(m_1) == H(m_2)$. It seems if the hash function has an output length of b bits, we have to check about 2^b messages. However, it turns out that an attacker needs only about $2^{\frac{b}{2}}$ messages. This is a quite surprising result, which is due to the birthday attack. This attack is based on the birthday

paradox, which is a powerful tool that is often used in cryptanalysis.

It turns out that the following real-world question is closely related to finding collisions for hash functions: how many people are needed at a party such that there is a reasonable chance that at least two people have the same birthday? Collision search for a hash function $H()$ is exactly the same problem as finding birthday collisions among party attendees. The question is how many messages $(m_1, m_2, ..., m_k)$ does an attacker need to hash until he has a reasonable chance that $H(m_i) == H(m_j)$ for some m_i and m_j that he choose.

The most significant consequence of the birthday attack is that the number of messages needed to hash to find a collision is roughly equal to the square root of the number of possible output values, i.e., about $\sqrt{2^b} = 2^{\frac{b}{2}}$. Hence, for a security level of u bit, the hash function needs to have an output length of $2u$ bit. In order to thwart collision attacks based on the birthday paradox, the output length of a hash function must be at least 128 [151]. As mentioned previously, we are using SHA-256 in this work, which has output length 256.

In 2004, collision-finding attacks against MD5 and SHA-0 were demonstrated by Xiaoyun Wang [161]. One year later, it was claimed that the attack could be extended to SHA-1 and a collision search would take 2^{63} steps, which is considerably less than the 2^{80}, achieved by the birthday attack (the output width in this case is 160 bit) [162]. In this work, we are using SHA-2 (precisely, SHA-256) against which no attacks are known to date.

5.9 Conclusion

In this chapter, we proposed a novel protocol to achieve the joint mission of private set intersection and secure thresholding for a distributed data de-identification task. We extended a previous filtering-based method to cover data from distributed sources and demonstrated the feasibility of using homomorphic encryption to develop an efficient multi-party protocol. Experimental results

show that our proposed method can simultaneously guarantee data privacy and preserve data utility for analysis. To the best of our knowledge, this is the first privacy-preserving initiative to de-identify clinical notes in a distributed environment.

Chapter 6

Conclusion

In this thesis, we have presented three different frameworks for analyzing biomedical data securely, where data is possessed by different participating parties. The proposed frameworks can securely perform genome-wide association studies, regression analysis, and clinical notes de-identification.

The first two solutions leverage trusted hardware to achieve the desired security and efficiency trade-off. On the contrary, the third work relies on an optimized variant of homomorphic encryption.

6.1 Summary

Firstly, we proposed a hybrid approach to conduct genome-wide association studies securely, where homomorphic encryption and secure hardware are used in appropriate use cases. This model offers data confidentiality, efficiency, and computational support for secure statistical analysis.

Secondly, we presented a hybrid cryptographic framework to perform regression analysis over distributed biomedical data in a secure and efficient way. This framework ensures data security and computational efficiency at the same time. Unlike prior works, it does not introduce approximation errors in the final output. Computed model parameters are identical to the plaintext results.

Finally, we presented a privacy-preserving initiative to de-identify clinical notes in a distributed environment. We developed a novel secure protocol based on private set intersection and secure thresholding to identify uncommon and low-frequency terms, which can guide sentence filtering. We extended a prior filtering-based method to consider clinical notes from distributed sources and demonstrated the applicability homomorphic encryption to develop an efficient multi-party protocol.

To summarize, the primary contributions of this thesis is the secure, efficient, and privacy preserving computation on biomedical data considering different functions.

6.2 Future Work

Shortly after Intel SGX, ARM introduced their hardware security architecture ARM TrustZone [1], which is available in AMD CPUs [2]. It would be interesting to see how ARM TrustZone performs with traditional cryptographic primitives like homomorphic encryption, garbled circuit etc.

Also, memory oblivious primitives (e.g. Oblivious RAM and data oblivious algorithms) can be incorporated into our hybrid frameworks to prevent information leakage through memory access patterns. This will minimize the vulnerability of side-channel attacks.

In addition, we intend to explore more statistical tests and machine learning algorithms, and extend the existing frameworks to perform these additional functions securely.

In general, it is an exciting future research direction to develop generic privacy enhancing technologies by leveraging state-of-the-art cryptographic primitives and recent trusted execution environments, and evaluating their compliance with different biomedical data sharing policies. This will further accelerate collaborative biomedical research including personalized medicine by minimizing the regulatory burdens aimed at protecting the privacy of the individuals.

[1] https://www.arm.com/products/security-on-arm/trustzone
[2] https://www.amd.com/en/technologies/security

Bibliography

[1] Anthony Antoniou, Paul DP Pharoah, Steven Narod, Harvey A Risch, Jorunn E Eyfjord, John L Hopper, Niklas Loman, Håkan Olsson, O Johannsson, Åke Borg, et al. Average risks of breast and ovarian cancer associated with brca1 or brca2 mutations detected in case series unselected for family history: a combined analysis of 22 studies. *The American Journal of Human Genetics*, 72(5):1117–1130, 2003.

[2] Samy Suissa, David Henry, Patricia Caetano, Colin R Dormuth, Pierre Ernst, Brenda Hemmelgarn, Jacques LeLorier, Adrian Levy, Patricia J Martens, J Michael Paterson, et al. Cnodes: the canadian network for observational drug effect studies. *Open Medicine*, 6(4):e134, 2012.

[3] Latanya Sweeney, Akua Abu, and Julia Winn. Identifying participants in the personal genome project by name (a re-identification experiment). *arXiv preprint arXiv:1304.7605*, 2013.

[4] Rui Wang, Yong Fuga Li, XiaoFeng Wang, Haixu Tang, and Xiaoyong Zhou. Learning your identity and disease from research papers: information leaks in genome wide association study. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 534–544. ACM, 2009.

[5] Suyash S Shringarpure and Carlos D Bustamante. Privacy risks from genomic data-sharing beacons. *The American Journal of Human Genetics*, 97(5):631–646, 2015.

[6] Cinnamon S Bloss. Does family always matter? public genomes and their effect on relatives. *Genome medicine*, 5(12):107, 2013.

[7] Muhammad Naveed, Erman Ayday, Ellen W Clayton, Jacques Fellay, Carl A Gunter, Jean-Pierre Hubaux, Bradley A Malin, and XiaoFeng Wang. Privacy and security in the genomic era. *arXiv preprint arXiv:1405.1891*, 2014.

[8] D J Willison. Use of data from the electronic health record for health research: current governance challenges and potential approaches. 2009.

[9] Matthew Hoekstra, Reshma Lal, Pradeep Pappachan, Vinay Phegade, and Juan Del Cuvillo. Using innovative instructions to create trustworthy software solutions. In *HASP@ ISCA*, page 11, 2013.

[10] Md Nazmus Sadat, Md Momin Al Aziz, Noman Mohammed, Feng Chen, Shuang Wang, and Xiaoqian Jiang. Safety: Secure gwas in federated environment through a hybrid solution. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2018.

[11] Md Nazmus Sadat, Xiaoqian Jiang, Md Momin Al Aziz, Shuang Wang, and Noman Mohammed. Secure and efficient regression analysis using a hybrid cryptographic framework: Development and evaluation. *JMIR medical informatics*, 6(1), 2018.

[12] Dingcheng Li, Majid Rastegar-Mojarad, Ravikumar Komandur Elayavilli, Yanshan Wang, Saeed Mehrabi, Yue Yu, Sunghwan Sohn, Yanpeng Li, Naveed Afzal, and Hongfang Liu. A frequency-filtering strategy of obtaining PHI-free sentences from clinical data repository. In *Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 315–324. ACM, September 2015.

[13] Rafael Pass, Elaine Shi, and Florian Tramer. Formal abstractions for attested execution se-

cure processors. Cryptology ePrint Archive, Report 2016/1027, 2016. `http://eprint.iacr.org/2016/1027`.

[14] Victor Costan and Srinivas Devadas. Intel sgx explained. Technical report, Cryptology ePrint Archive, Report 2016/086, 2016. https://eprint. iacr. org/2016/086.

[15] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[16] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptology,EUROCRYPT99*, pages 223–238. Springer, 1999.

[17] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT)*, pages 223–238, 1999.

[18] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory*, 31(4):469–472, 1985.

[19] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 365–377. ACM, 1982.

[20] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[21] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[22] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Theory of Cryptography Conference*, pages 325–341. Springer, 2005.

[23] Craig Gentry et al. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.

[24] C A Melchor, J Barrier, L Fousse, and others. XPIR: Private information retrieval for everyone. *on Privacy Enhancing* , 2016.

[25] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning ICML*, volume 48, pages 201–210, 2016.

[26] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 309–325. ACM, 2012.

[27] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Manual for using homomorphic encryption for bioinformatics. *Proceedings of the IEEE*, 105(3):552–567, 2017.

[28] Kim Laine and Rachel Player. Simple encrypted arithmetic library-seal (v2. 0). Technical report, Technical report, September, 2016.

[29] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pages 113–124, 2011.

[30] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference on - ITCS '12*, volume 111, pages 309–325, New York, NY, USA, 2012. ACM Press.

[31] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.

[32] Joppe W Bos, Kristin E Lauter, Jake Loftus, and Michael Naehrig. Improved security for a Ring-Based fully homomorphic encryption scheme. In *IMA Int. Conf.*, pages 45–64, 2013.

[33] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *arXiv preprint arXiv:1704.03578*, 2017.

[34] Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Jeffrey Hoffstein, Kristin Lauter, Satya Lokam, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. White paper about security of homomorphic encryption.

[35] Nigel P Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *International Workshop on Public Key Cryptography*, pages 420–443. Springer, 2010.

[36] Wen-Jie Lu, Yoshiji Yamada, and Jun Sakuma. Privacy-preserving genome-wide association studies on cloud environment using fully homomorphic encryption. *BMC medical informatics and decision making*, 15(Suppl 5):S1, 2015.

[37] Yuchen Zhang, Wenrui Dai, Xiaoqian Jiang, Hongkai Xiong, and Shuang Wang. Foresee: Fully outsourced secure genome study based on homomorphic encryption. *BMC medical informatics and decision making*, 15(Suppl 5):S5, 2015.

[38] Kristin Lauter, Adriana López-Alt, and Michael Naehrig. Private computation on encrypted genomic data. In *Progress in Cryptology-LATINCRYPT 2014*, pages 3–27. Springer, 2014.

[39] Al Aziz, Md Momin, Mohammad Z Hasan, Noman Mohammed, and Dima Alhadidi. Secure and efficient multiparty computation on genomic data. In *Proceedings of the 20th International Database Engineering & Applications Symposium*, pages 278–283. ACM, 2016.

[40] Shuang Wang, Yuchen Zhang, Wenrui Dai, Kristin Lauter, Miran Kim, Yuzhe Tang, Hongkai Xiong, and Xiaoqian Jiang. Healer: Homomorphic computation of exact logistic regression for secure rare disease variants analysis in gwas. *Bioinformatics, 32(2):211–8*, 2015.

[41] Feng Chen, Michelle Dow, Sijie Ding, Yao Lu, Xiaoqian Jiang, Hua Tang, Shuang Wang. PREMIX: Privacy-preserving EstiMation of individual admixture. In *American Medical Informatics Association Annual Symposium*.

[42] Feng Chen, Shuang Wang, Xiaoqian Jiang, Sijie Ding, Yao Lu, Jihoon Kim, S Cenk Sahinalp, Chisato Shimizu, Jane C Burns, Victoria J Wright, et al. Princess: Privacy-protecting rare disease international network collaboration via encryption through software guard extensions. *Bioinformatics*, page 758, 2017.

[43] Roman Kosoy, Rami Nassir, Chao Tian, Phoebe A White, Lesley M Butler, Gabriel Silva, Rick Kittles, Marta E Alarcon-Riquelme, Peter K Gregersen, John W Belmont, et al. Ancestry informative marker sets for determining continental origin and admixture proportions in common populations in america. *Human mutation*, 30(1):69–78, 2009.

[44] Yihua Zhang, Marina Blanton, and Ghada Almashaqbeh. Secure distributed genome analysis for gwas and sequence comparison computation. *BMC medical informatics and decision making*, 15(5):S4, 2015.

[45] Chiea Chuen Khor, Sonia Davila, Willemijn B Breunis, Yi-Ching Lee, Chisato Shimizu, Victoria J Wright, Rae SM Yeung, Dennis EK Tan, Kar Seng Sim, Wang, and Jie Jin. Genome-wide association study identifies fcgr2a as a susceptibility locus for kawasaki disease. *Nature genetics*, 43(12):1241–1246, 2011.

[46] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput

and accuracy. In *International Conference on Machine Learning ICML*, volume 48, pages 201–210, 2016.

[47] Thore Graepel, Kristin Lauter, and Michael Naehrig. Ml confidential: Machine learning on encrypted data. In *International Conference on Information Security and Cryptology*, pages 1–21. Springer, 2012.

[48] Joppe W Bos, Kristin Lauter, and Michael Naehrig. Private predictive analysis on encrypted medical data. *Journal of biomedical informatics*, 50:234–243, 2014.

[49] Rob Hall, Stephen E Fienberg, and Yuval Nardi. Secure multiple linear regression based on homomorphic encryption. *Journal of Official Statistics*, 27(4):669, 2011.

[50] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Annual cryptology conference*, pages 505–524. Springer, 2011.

[51] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):13, 2014.

[52] Joppe W Bos, Kristin E Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In *IMA Int. Conf.*, pages 45–64. Springer, 2013.

[53] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

[54] Jing Lei. Differentially private m-estimators. In *Advances in Neural Information Processing Systems*, pages 361–369, 2011.

[55] Jun Zhang, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, and Marianne Winslett. Functional mechanism: regression analysis under differential privacy. *Proceedings of the VLDB Endowment*, 5(11):1364–1375, 2012.

[56] Yuan Wu, Xiaoqian Jiang, Jihoon Kim, and Lucila Ohno-Machado. Grid binary logistic regression (glore): building shared models without sharing data. *Journal of the American Medical Informatics Association*, 19(5):758–764, 2012.

[57] Valeria Nikolaenko, Udi Weinsberg, Sotiris Ioannidis, Marc Joye, Dan Boneh, and Nina Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 334–348. IEEE, 2013.

[58] Shuang Wang, Yuchen Zhang, Wenrui Dai, Kristin Lauter, Miran Kim, Yuzhe Tang, Hongkai Xiong, and Xiaoqian Jiang. Healer: Homomorphic computation of exact logistic regression for secure rare disease variants analysis in gwas. *Bioinformatics*, 32(2):211–218, 2016.

[59] Haoyi Shi, Chao Jiang, Wenrui Dai, Xiaoqian Jiang, Yuzhe Tang, Lucila Ohno-Machado, and Shuang Wang. Secure multi-party computation grid logistic regression (smac-glore). *BMC Medical Informatics and Decision Making*, 16(3):89, 2016.

[60] Olga Ohrimenko, Felix Schuster, Cedric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa. Oblivious multi-party machine learning on trusted processors. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 619–636. USENIX Association, 2016.

[61] Andrew Chi-Chih Yao. Protocols for secure computations. In *FOCS*, volume 82, pages 160–164, 1982.

[62] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[63] Cynthia Dwork. Differential privacy. In *33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006)*, volume 4052, pages 1–12, Venice, Italy, July 2006. Springer Verlag.

[64] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems*, pages 289–296, 2009.

[65] Adam Groce, Jonathan Katz, and Arkady Yerukhimovich. Limits of computational differential privacy in the client/server setting. In *Theory of Cryptography Conference*, pages 417–431. Springer, 2011.

[66] Yuanzhong Xu, Weidong Cui, and Marcus Peinado. Controlled-channel attacks: Deterministic side channels for untrusted operating systems. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 640–656. IEEE, 2015.

[67] Nico Weichbrodt, Anil Kurmus, Peter Pietzuch, and Rüdiger Kapitza. Asyncshock: Exploiting synchronisation bugs in intel sgx enclaves. In *European Symposium on Research in Computer Security*, pages 440–457. Springer, 2016.

[68] Wenhao Wang, Guoxing Chen, Xiaorui Pan, Yinqian Zhang, XiaoFeng Wang, Vincent Bindschaedler, Haixu Tang, and Carl A Gunter. Leaky cauldron on the dark land: Understanding memory side-channel hazards in sgx. *arXiv preprint arXiv:1705.07289*, 2017.

[69] Bert Vogelstein, Nickolas Papadopoulos, Victor E Velculescu, Shibin Zhou, Luis A Diaz, and Kenneth W Kinzler. Cancer genome landscapes. *science*, 339(6127):1546–1558, 2013.

[70] Wylie Burke and Bruce M Psaty. Personalized medicine in the era of genomics. *Jama*, 298(14):1682–1684, 2007.

[71] Steven E Brenner. Be prepared for the big genome leak. *Nature*, 498(7453):139, 2013.

[72] Peter M Visscher, Matthew A Brown, Mark I McCarthy, and Jian Yang. Five years of gwas discovery. *The American Journal of Human Genetics*, 90(1):7–24, 2012.

[73] Dan Bogdanov, Liina Kamm, Sven Laur, Pille Pruulmann-Vengerfeldt, Riivo Talviste, and Jan Willemson. Privacy-preserving statistical data analysis on federated databases. In *Annual Privacy Forum*, pages 30–55. Springer, 2014.

[74] Florian Tramèr, Zhicong Huang, Jean-Pierre Hubaux, and Erman Ayday. Differential privacy with bounded priors: reconciling utility and privacy in genome-wide association studies. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1286–1297. ACM, 2015.

[75] Sean Simmons, Cenk Sahinalp, and Bonnie Berger. Enabling privacy-preserving gwass in heterogeneous human populations. *Cell Systems*, 3(1):54–61, 2016.

[76] Ali Shahbazi, Fattaneh Bayatbabolghani, and Marina Blanton. Private computation with genomic data for genome-wide association and linkage studies. 2016.

[77] Yaniv Erlich and Arvind Narayanan. Routes for breaching and protecting genetic privacy. *Nature Reviews Genetics*, 15(6):409–421, 2014.

[78] Md Momin Al Aziz, Md Nazmus Sadat, Dima Alhadidi, Shuang Wang, Xiaoqian Jiang, Cheryl L Brown, and Noman Mohammed. Privacy-preserving techniques of genomic dataa survey. *Briefings in Bioinformatics*, page bbx139, 2017.

[79] Council of Canadian Academies. Accessing health and health-related data in canada : The expert panel on timely access to health and social data for health research and health system innovation. Report, Council of Canadian Academies, 2015.

[80] Erika Check Hayden. Geneticists push for global data-sharing: international organization

aims to promote exchange and linking of dna sequences and clinical information. *Nature*, 498(7452):16–18, 2013.

[81] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[82] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography*, pages 265–284. Springer, 2006.

[83] Yaniv Erlich, James B. Williams, David Glazer, Kenneth Yocum, Nita Farahany, Maynard Olson, Arvind Narayanan, Lincoln D. Stein, Jan A. Witkowski, and Robert C. Kain. Redefining genomic privacy: Trust and empowerment. 12(11):e1001983, 11 2014.

[84] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pages 113–124. ACM, 2011.

[85] Liina Kamm. *Privacy-preserving statistical analysis using secure multi-party computation*. PhD thesis, 2015.

[86] Ittai Anati, Shay Gueron, Simon Johnson, and Vincent Scarlata. Innovative technology for cpu based attestation and sealing. In *Proceedings of the 2nd international workshop on hardware and architectural support for security and privacy*, volume 13, 2013.

[87] Scott D Constable, Yuzhe Tang, Shuang Wang, Xiaoqian Jiang, and Steve Chapin. Privacy-preserving gwas analysis on federated genomic datasets. *BMC medical informatics and decision making*, 15(Suppl 5):S2, 2015.

[88] Felix Schuster, Manuel Costa, Cédric Fournet, Christos Gkantsidis, Marcus Peinado, Gloria Mainar-Ruiz, and Mark Russinovich. Vc3: trustworthy data analytics in the cloud using sgx. In *2015 IEEE Symposium on Security and Privacy*, pages 38–54. IEEE, 2015.

[89] Alkes L Price, Nick J Patterson, Robert M Plenge, Michael E Weinblatt, Nancy A Shadick, and David Reich. Principal components analysis corrects for stratification in genome-wide association studies. *Nature genetics*, 38(8):904–909, 2006.

[90] Oded Goldreich. *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2009.

[91] Sangho Lee, Ming-Wei Shih, Prasun Gera, Taesoo Kim, Hyesoon Kim, and Marcus Peinado. Inferring fine-grained control flow inside sgx enclaves with branch shadowing. *arXiv preprint arXiv:1611.06952*, 2016.

[92] Andrew Baumann, Marcus Peinado, and Galen Hunt. Shielding applications from an untrusted cloud with haven. *ACM Transactions on Computer Systems (TOCS)*, 33(3):8, 2015.

[93] Shweta Shinde, Zheng Leong Chua, Viswesh Narayanan, and Prateek Saxena. Preventing page faults from telling your secrets. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pages 317–328. ACM, 2016.

[94] Victor Costan, Ilia A Lebedev, and Srinivas Devadas. Sanctum: Minimal hardware extensions for strong software isolation. In *USENIX Security Symposium*, pages 857–874, 2016.

[95] Nan M Laird and Christoph Lange. *The fundamentals of modern statistical genetics*. Springer Science & Business Media, 2010.

[96] 1000 genomes dataset phase 1. `ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/phase1/analysis_results/integrated_call_sets/`. Online; accessed 23 December 2016.

[97] Maximilian Schneider. Private information retrieval. *Technische Berichte des Hasso-Plattner-Instituts für Softwaresystemtechnik an der Universität Potsdam*, page 60, 2014.

[98] Femi Olumofin and Ian Goldberg. Privacy-preserving queries over relational databases. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 75–92. Springer, 2010.

[99] Benjamin Fung, Ke Wang, Rui Chen, and Philip S Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys (CSUR)*, 42(4):14, 2010.

[100] Aaron Johnson and Vitaly Shmatikov. Privacy-preserving data exploration in genome-wide association studies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1079–1087. ACM, 2013.

[101] Fei Yu, Stephen E Fienberg, Aleksandra B Slavković, and Caroline Uhler. Scalable privacy-preserving data sharing methodology for genome-wide association studies. *Journal of biomedical informatics*, 50:133–141, 2014.

[102] Feng Chen, Michelle Dow, Sijie Ding, Yao Lu, Xiaoqian Jiang, Hua Tang, Shuang Wang. PREMIX: Privacy-preserving EstiMation of individual admixture. In *American Medical Informatics Association Annual Symposium*.

[103] Bahman P Tabaei and William H Herman. A multivariate logistic regression equation to screen for diabetes. *Diabetes Care*, 25(11):1999–2003, 2002.

[104] Saeed Abdullah, Elizabeth L Murnane, Mark Matthews, Matthew Kay, Julie A Kientz, Geri Gay, and Tanzeem Choudhury. Cognitive rhythms: unobtrusive and continuous sensing of alertness using a mobile phone. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 178–189. ACM, 2016.

[105] Tauhidur Rahman, Mary Czerwinski, Ran Gilad-Bachrach, and Paul Johns. Predicting about-to-eat moments for just-in-time eating intervention. In *Proceedings of the 6th International Conference on Digital Health Conference*, pages 141–150. ACM, 2016.

[106] Hossein Ahmadi, Nam Pham, Raghu Ganti, Tarek Abdelzaher, Suman Nath, and Jiawei Han. Privacy-aware regression modeling of participatory sensing data. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 99–112. ACM, 2010.

[107] Khaled El Emam, Jun Hu, Jay Mercer, Liam Peyton, Murat Kantarcioglu, Bradley Malin, David Buckeridge, Saeed Samet, and Craig Earle. A secure protocol for protecting the identity of providers when disclosing data for disease surveillance. *Journal of the American Medical Informatics Association*, 18(3):212–217, 2011.

[108] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

[109] Kim Laine and Rachel Player. Simple encrypted arithmetic Library-SEAL (v2. 0). Technical report, Technical report, September, 2016.

[110] Ben Fisch, Dhinakaran Vinayagamurthy, Dan Boneh, and Sergey Gorbunov. Iron: functional encryption using intel sgx. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 765–782. ACM, 2017.

[111] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333. ACM, 2015.

[112] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 17–32, 2014.

[113] Christiaan Heij, Paul De Boer, Philip Hans Franses, Teun Kloek, Herman K Van Dijk, et al. *Econometric methods with applications in business and economics*. OUP Oxford, 2004.

[114] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[115] Shai Halevi and Victor Shoup. Algorithms in helib. In *Advances in Cryptology–CRYPTO 2014*, pages 554–571. Springer, 2014.

[116] M. Lichman. UCI machine learning repository, 2013.

[117] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[118] Rob Hall, Stephen E Fienberg, and Yuval Nardi. Secure multiple linear regression based on homomorphic encryption. *J. Off. Stat.*, 27(4):669, 2011.

[119] Martín Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[120] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[121] Hwanjo Yu, Jaideep Vaidya, and Xiaoqian Jiang. Privacy-preserving svm classification on vertically partitioned data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 647–656. Springer, 2006.

[122] Hwanjo Yu, Xiaoqian Jiang, and Jaideep Vaidya. Privacy-preserving svm using nonlinear kernels on horizontally partitioned data. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 603–610. ACM, 2006.

[123] Santa Clara Intel Corporation. Products formerly Skylake. `http://ark.intel.com/products/codename/37572/Skylake`, 2017. [Online; accessed 2017-10-11].

[124] Russinovich M. Introducing Azure confidential computing. https://azure.microsoft.com/en-us/blog/introducing-azure-confidential-computing, 2017. [Online; accessed 2017-10-11].

[125] Dina Demner-Fushman, Wendy W Chapman, and Clement J McDonald. What can natural language processing do for clinical decision support? *J. Biomed. Inform.*, 42(5):760–772, October 2009.

[126] Stephen T Wu, Hongfang Liu, Dingcheng Li, Cui Tao, Mark A Musen, Christopher G Chute, and Nigam H Shah. Unified medical language system term occurrences in clinical notes: a large-scale corpus analysis. *J. Am. Med. Inform. Assoc.*, 19(e1):e149–56, June 2012.

[127] John J Trinckes and Jr. *The Definitive Guide to Complying with the HIPAA/HITECH Privacy and Security Rules*. CRC Press, December 2012.

[128] U S Department of Health & Human Services. Guidance regarding methods for de-identification of protected health information in accordance with the health insurance portability and accountability act (HIPAA) privacy rule. https://www.hhs.gov/hipaa/for-professionals/privacy/special-topics/de-identification/index.html, November 2015. Accessed: 2017-10-26.

[129] Ishna Neamatullah, Margaret M Douglass, Li-Wei H Lehman, Andrew Reisner, Mauricio Villarroel, William J Long, Peter Szolovits, George B Moody, Roger G Mark, and Gari D Clifford. Automated de-identification of free-text medical records. *BMC Med. Inform. Decis. Mak.*, 8:32, July 2008.

[130] M Douglass, G D Clifford, A Reisner, G B Moody, and Mark Rg. Computer-assisted de-

identification of free text in the MIMIC II database. In *Computers in Cardiology, 2004*, pages 341–344, 2004.

[131] Ozlem Uzuner, Tawanda C Sibanda, Yuan Luo, and Peter Szolovits. A de-identifier for medical discharge summaries. *Artif. Intell. Med.*, 42(1):13–35, January 2008.

[132] Bruce A Beckwith, Rajeshwarri Mahaadevan, Ulysses J Balis, and Frank Kuo. Development and evaluation of an open source software tool for deidentification of pathology reports. *BMC Med. Inform. Decis. Mak.*, 6:12, March 2006.

[133] Jules J Berman. Concept-match medical data scrubbing. how pathology text can be used in research. *Arch. Pathol. Lab. Med.*, 127(6):680–686, June 2003.

[134] Gregory P. Finley, Serguei V.S. Pakhomov, Genevieve B. Melton. Automated De-Identification of distributional semantic models. *AMIA 2016 Annual Symposium*.

[135] L Sweeney. Replacing personally-identifying information in medical records, the scrub system. *Proc. AMIA Annu. Fall Symp.*, pages 333–337, 1996.

[136] L Sweeney. Guaranteeing anonymity when sharing medical data, the datafly system. *Proc. AMIA Annu. Fall Symp.*, pages 51–55, 1997.

[137] Stephane M Meystre, F Jeffrey Friedlin, Brett R South, Shuying Shen, and Matthew H Samore. Automatic de-identification of textual documents in the electronic health record: a review of recent research. *BMC Med. Res. Methodol.*, 10:70, August 2010.

[138] György Szarvas, Richárd Farkas, and Róbert Busa-Fekete. State-of-the-art anonymization of medical records using an iterative machine learning framework. *J. Am. Med. Inform. Assoc.*, 14(5):574–580, January.

[139] Y Guo and R Gaizauskas. Identifying personal health information using support vector machines. *i2b2 workshop on łdots*, 2006.

[140] K Hara. Applying a SVM based chunker and a text classifier to the deid challenge. *i2b2 Workshop on challenges in natural language*, 2006.

[141] E Aramaki, T Imai, K Miyo, and K Ohe. Automatic deidentification by using sentence features and label consistency. *i2b2 Workshop on Challenges in*, 2006.

[142] James Gardner and Li Xiong. HIDE: An Integrated System for Health Information DE-identification. In *EDBT*, pages 254–259. IEEE, 2008.

[143] 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. *J. Am. Med. Inform. Assoc.*, 14(5):564–573, September 2007.

[144] Franck Dernoncourt, Ji Young Lee, Ozlem Uzuner, and Peter Szolovits. De-identification of patient notes with recurrent neural networks. *J. Am. Med. Inform. Assoc.*, 24(3):596–606, May 2017.

[145] L Kissner, D Song Crypto, and 2005. Privacy-preserving set operations. *Springer*, 2005.

[146] Rolf Egert, Marc Fischlin, David Gens, Sven Jacob, Matthias Senker, and Jörn Tillmanns. Privately computing Set-Union and Set-Intersection cardinality via bloom filters. In *Information Security and Privacy*, Lecture Notes in Computer Science, pages 413–430. Springer, Cham, June 2015.

[147] Atsuko Miyaji, Kazuhisa Nakasho, and Shohei Nishida. Privacy-Preserving integration of medical data. *J. Med. Syst.*, 41(3):37, March 2017.

[148] Atsuko Miyaji and Shohei Nishida. A scalable multiparty private set intersection. In *Network and System Security*, Lecture Notes in Computer Science, pages 376–385. Springer, Cham, November 2015.

[149] N P Smart and F Vercauteren. Fully homomorphic SIMD operations. *Des. Codes Cryptogr.*, 71(1):57–81, April 2014.

[150] Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in LWE-Based homomorphic encryption. In *Public-Key Cryptography – PKC 2013*, Lecture Notes in Computer Science, pages 1–13. Springer, Berlin, Heidelberg, 2013.

[151] Christof Paar and Jan Pelzl. *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer Science & Business Media, November 2009.

[152] D D Chen, N Mentens, F Vercauteren, S S Roy, R C C Cheung, D Pao, and I Verbauwhede. High-Speed polynomial multiplication architecture for Ring-LWE and SHE cryptosystems. *IEEE Trans. Circuits Syst. I Regul. Pap.*, 62(1):157–166, January 2015.

[153] Alistair E W Johnson, Tom J Pollard, Lu Shen, Li-Wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. MIMIC-III, a freely accessible critical care database. *Sci Data*, 3:160035, May 2016.

[154] Christopher Fox. A stop list for general text. *SIGIR Forum*, 24(1-2):19–21, September 1989.

[155] Carlos Aguilar-Melchor, Joris Barrier, Serge Guelton, Adrien Guinet, Marc-Olivier Killijian, and Tancrède Lepoint. NFLlib: NTT-Based fast lattice library. In *Topics in Cryptology - CT-RSA 2016*, Lecture Notes in Computer Science, pages 341–356. Springer, Cham, February 2016.

[156] V Shoup URL: http://www. shoup. net/ntl and 2015. NTL: a library for doing number theory, 2005. 2015.

[157] W B Hart Computeralgebra Rundbrief and 2013. Flint: Fast library for number theory. *wrap.warwick.ac.uk*, 2013.

[158] Martin Volk, Bärbel Ripplinger, Spela Vintar, Paul Buitelaar, Diana Raileanu, and Bogdan Sacaleanu. Semantic annotation for concept-based cross-language medical information retrieval. *Int. J. Med. Inform.*, 67(1-3):97–112, December 2002.

[159] R Lindner and C Peikert. Better key sizes (and attacks) for LWE-Based encryption. *CT-RSA*, 2011.

[160] Philippe Oechslin. Making a faster cryptanalytic Time-Memory Trade-Off. In *Advances in Cryptology - CRYPTO 2003*, Lecture Notes in Computer Science, pages 617–630. Springer, Berlin, Heidelberg, August 2003.

[161] Xiaoyun Wang, Dengguo Feng, Xuejia Lai, and Hongbo Yu. Collisions for hash functions md4, md5, haval-128 and ripemd. *IACR Cryptology ePrint Archive*, 2004:199, 2004.

[162] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full sha-1. In *Annual international cryptology conference*, pages 17–36. Springer, 2005.