Medical Diagnostics Based on Encrypted Medical Data

Alexey Gribov[1], Kelsey Horan[1,2(✉)], Jonathan Gryak[2], Kayvan Najarian[2], Vladimir Shpilrain[1], Reza Soroushmehr[2], and Delaram Kahrobaei[1,3]

[1] The Graduate Center, The City University of New York, 365 Fifth Ave, New York, NY 10016, USA
[2] Michigan Center for Integrative Research in Critical Care, University of Michigan Ann Arbor, 2800 Plymouth Road, Ann Arbor, MI 48109, USA
khoran@gradcenter.cuny.edu
[3] University of York, Heslington, York YO10 5DD, UK

Abstract. We utilize a type of encryption scheme known as a Fully Homomorphic Encryption (FHE) scheme which allows for computation over encrypted data. Our encryption scheme is more efficient than other publicly available FHE schemes, making it more feasible. We conduct simulations based on common scenarios in which this ability is useful. In the first simulation we conduct time series analysis via Recursive Least Squares on both encrypted and unencrypted data and compare the results. In simulation one, it is shown that the error from computing over plaintext data is the same as the error for computing over encrypted data. In the second simulation, we compute two known diagnostic functions over publicly available data in order to calculate computational benchmarks. In simulation two, we see that computation over encrypted data using our method incurs relatively lower costs as compared to a majority of other publicly available methods. By successfully computing over encrypted data we have shown that our FHE scheme permits the use of machine learning algorithms that utilize polynomial kernel functions.

Keywords: Clinical decision support · Data mining · Machine learning · Privacy preserving classifier · Encryption

1 Introduction

The field of medical informatics utilizes data mining algorithms that should be built on diverse databases in order to obtain generalizable results [36]. Training, validating, and testing a computational hypothesis typically requires access to large sample datasets that adequately represent any variation within the relevant population [29]. Ideally, data from separate entities, i.e., multiple hospitals and healthcare systems, would be integrated to en-sure an accurate sample. It is evident that healthcare systems vary in location, specialty, and patient care protocols; patients at any particular hospital represent only a subset of the population

A. Compagnoni et al. (Eds.): BICT 2019, LNICST 289, pp. 98–111, 2019.
https://doi.org/10.1007/978-3-030-24202-2_8

and therefore data from a single hospital may contain only a subset of potential illnesses and injuries. Any computational analysis meant to produce diagnostic or prognostic clinical decision support needs to be validated on data collected from multiple healthcare systems. However, ownership and privacy issues greatly limit the possibility of creating such large, comprehensive databases.

Due to the sensitivity of medical data, federally enforced privacy regulations such as Health Insurance Portability and Accountability Act (HIPPA) [7] place firm constraints on data sharing. Therefore, creating or obtaining diversified medical datasets without violating privacy regulations is challenging. Anonymization, the systematic removal of potential patient identifiers, is often seen as a potential solution. However, it has been shown that deanonymization is a relatively simple task. The ease of deanonymization is exemplified by the re-identification of the Governor of Massachusetts within an anonymized health database [11], as well as many other instances. These privacy concerns suggest that medical data should only be released in encrypted format. Thus far, for medical data, there has been a trade-off between utility and security; a useful patient database that allows for computation is insecure, whereas a secure patient database is practically useless for computation and research. This complication stems from the lack of existing technologies that are capable of supporting statistical analysis or machine learning methods on encrypted data. In order to make comprehensive, secure and publicly useful databases possible, the medical community needs to adopt an approach that would allow data analytics on shared encrypted databases while respecting federal privacy restrictions.

The cryptographic community has recently developed fully homomorphic encryption (FHE) schemes, which admit such secure computation. The fully homomorphic property of an encryption function E, can be defined as follows: for any a, b, one has that

$$E(a+b) = E(a) + E(b) \text{ and } E(ab) = E(a)E(b).$$

This implies that for any polynomial function $F(x_1, x_2, \ldots, x_n)$ one has the following property:

$$E(F(x_1, x_2, \ldots, x_n)) = F(E(x_1), E(x_2), \ldots, E(x_n))$$

In other words, computing on an encrypted database yields essentially the same results as computing on an unencrypted database. In this paper we propose a secure machine learning approach based on FHE. This allows researchers and hospitals to run a family of machine learning methods on encrypted data. We offer a solution to the privacy standoff in the case that the desired machine learning algorithm uses polynomial kernel functions.

The FHE scheme most widely accepted as theoretically secure is due to the work of Gentry [14]. This scheme was subsequently improved by Brakerski et al. [5] and the relevant software under development by IBM is available on GitHub [20]. The security of these particular solutions is given by the property of "random self-reducibility". Essentially, finding a solution to the underlying problem is about as hard on average as it is in the worst case. While this property is

indeed good evidence of theoretical security, the resulting homomorphic encryption algorithm is too inefficient to be practical.

The conflict can be stated as follows: in order to provide semantic security an encryption algorithm must be randomized, but on the other hand any homomorphism should map zero to zero; an encryption of zero cannot be zero but an encryption of zero must be zero. To resolve this, encryptions of zero are "masked" by "noise" in the aforementioned schemes. The new problem is that during computation on encrypted data this "noise" tends to accumulate and must occasionally be reduced to preserve the correctness of decryption. Therefore, the inefficiency of these schemes can be attributed to the adopted noise management solution. In the above implementations the noise reduction process is *recryption* (also known as bootstrapping), a function that takes a noisy ciphertext and produces an equivalent ciphertext with less noise. Recryption is an expensive procedure and limits both the efficiency and real-life applicability of any existing FHE solution.

There were alternative proposals for FHE following Gentry's, given by Brakerski et al., Ducas et al. and Van Dijk et al., which can be found in their respective works [6,10,35]. While some of these approaches seem conceptually simple and effective, all proposed solutions still involve "bootstrapping". In addition, a majority of the existing methods deal with encrypting Boolean circuits (i.e., AND and OR Boolean operations) as opposed to the "usual" arithmetic operations of multiplication and addition. A "leveled" scheme, an encryption scheme that is typically more efficient than FHE schemes but can only correctly compute a bounded number of operations on ciphertexts, based on adaptations of one of the works above has been developed by Fan and Vercauteren [13], an implementation of this scheme in the R-platform can be found online. In contrast to leveled schemes, our scheme is fully homomorphic and therefore does not require that any parties determine the desired function to compute prior to instantiating the encryption scheme. A further review of FHE schemes and software tools has been provided by Aslett et al. [2].

The first implementation of Gentry's FHE scheme boasted one – albeit large dimensional – bootstrapping operation in 31 min [15]. Subsequent implementations of the improved FHE schemes utilize as benchmarks the computation time for one bootstrap operation as well as computing the AES encryption circuit. The HElib [19], FHEW [10], and TFHE [8] are implementations of alternative encryption schemes. In terms of benchmarking, the fastest and most recent implementation, TFHE, reports 13 ms for the computation of one bootstrapped binary gate.

Recently, secure machine learning techniques have appeared in the literature. Many techniques implement the schemes mentioned above, but many privacy preserving classifiers circumvent FHE and instead make use of leveled homomorphic encryption schemes, garbled circuits, secret sharing, or multi-party computation. For example, Du et al. [9] develop a secure model for linear regression and two-party multivariate classification. This technique does not use FHE, nor does it provide computational benchmarks.

Various techniques attempt to use differential privacy, a tool to protect against database deanonymization, to allow for data sharing. These schemes include those provided by Zhang et al. [38] and Sarwate et al. [32]. These techniques do not implement cryptographic schemes but, if provably secure, could allow for useful patient data warehouses. Yang et al. [37] develop a framework for delegated computation, such as searching encrypted databases in the cloud, which can be used for electronic medical records. This framework, while efficient, does not implement machine learning or FHE.

Graepel et al. [16] construct low-degree polynomial versions of classification algorithms on a leveled homomorphic encryption scheme. Nikolaenko et al. [28] improve upon this performance, reducing computation time, and construct privacy preserving ridge regression using encryption and garbled circuits. This construction does not use an FHE scheme, but is a hybrid construction. Other hybrid constructions exist, such as the constructions that use garbled circuits proposed by Sadeghi et al. [31] and Evans et al. [12] or garbled circuits combined with secret sharing proposed by Lindell et al. [24], but do not implement FHE.

Bos et al. [3] also utilize a leveled homomorphic encryption scheme. The encryption scheme uses a polynomial ring as a platform to allow a client to delegate medical prediction functions to a server. In this setting the algorithm only keeps the patient data private, and allows everyone to know the function.

Bost et al. [4] explore privacy-preserving classifiers in the form of hyperplane decision, Naive Bayes, and decision trees using delegated computation. This is based on additively homomorphic encryption schemes and garbled circuits. Our scheme has the advantage of involving lower communication complexity for a linear classifier. Eventually, our framework could be extended to include more sophisticated learning algorithms such as those addressed in this paper. Many other constructions, such as that by Aono et al. [1], merely use an additively homomorphic encryption scheme for computations such as logistic regression or statistical analysis [21,23,27].

Liu et al. [25] use the IBM implementation of FHE to delegate the computation of Support Vector Machine classification on encrypted data to a server. The paper provides benchmarks for encryption and decryption time, as well as homomorphic multiplication and addition, on a small dataset. Our scheme, as seen in our results, greatly reduces computation time. Our approach to homomorphic encryption is based on the use of mathematical rings and homomorphisms between those rings. This framework has the additional benefit of avoiding any computational overhead due to converting between "real-life arithmetic" and Boolean circuits. In this paper we avoid detailing the specifics of our encryption method, which are outlined in [17,22]. We will say that since the ciphertext ring, in which computations on encrypted data are performed, has a very simple structure all computations within our scheme are orders of magnitude more efficient than the schemes mentioned.

The advantage of using this specific FHE scheme is error-less computation with guaranteed security, in contrast to some of the non-FHE solutions provided above. The main contribution of this paper is that our FHE scheme is a fully

homomorphic scheme that is efficient enough to allow for machine learning on encrypted data. We provide simulations, error, and computational overhead of using our scheme for this purpose. While the works in this paper are generally applicable to a multitude of scenarios, we consider two specific applications for machine learning over encrypted data. These two scenarios will be simulated in the following experiments. Most of the data utilized is actual patient data; the term *simulation* refers to the simulation of both the communication between parties, and computations performed by each party.

In the first scenario, a hospital $\mathcal{H}$ has a private database of patients' data, D, which cannot be shared with an external entity. On the other hand, a research center $\mathcal{C}$ would like to use D to construct a machine learning model, for example to build a function F to calculate a risk assessment score, or perform actuarial analyses. In practice, $\mathcal{C}$ could be a research center, another hospital, a pharmaceutical company, an insurance company, or any other third party. Additionally, $\mathcal{C}$ may be reluctant to share its intellectual property, F, with $\mathcal{H}$. This scenario happens quite often within the research community. Researchers find it difficult to train generalizable models because of the lack of public data. This scenario is, in part, simulated in this paper, when we compare the error results of training a function on encrypted and unencrypted data.

In the second scenario, $\mathcal{C}$ has a collection of multivariate functions

$$F_i : X_n \to Y \quad \text{for} \quad 1 \leq i \leq k$$

representing a specific machine learning algorithm, while $\mathcal{H}$ has the inputs representing the medical data for patient $x = (x_1, x_2, \ldots, x_n)$. The set of known functions, $F = F_{i\,i=1}^{k}$ are expected to predict a diagnosis or prognosis for x, for example the chances of patient x having Parkinson's, cancer, heart attack, etc. The difference between this second scenario and the first outlined scenario is that in this scenario we assume that F is a known function (e.g., we know how to calculate Glasgow Coma Scale (GCS) using eye, verbal, and motor responses). Our goal is to apply F on both encrypted and unencrypted data and show that the output of a known function evaluation on both sets of data is the same. Outputs of these functions could be "health metrics", "severity scores" and other clinical functions typically computed as a linear combination of privacy-protected factors, such as quantitative clinical or physiological patient data with a set of weights (coefficients). In practice, these functions are often designed to receive integer values as input variables and produce an integer as an output score. There are a large number of such studies conducted for diseases such as diabetes [33]. In many such modeling tasks, particularly when designing these models as commercial products, it is highly desirable to design, test and validate the functions privately. This scenario, in part, is simulated in this paper, where we apply known classification functions to encrypted and unencrypted data.

In very general terms, a potential exchange between the hospital $\mathcal{H}$ and a center $\mathcal{C}$ goes as follows. $\mathcal{H}$ encrypts data, $x_1, x_2, \ldots, x_n$, of a particular patient x with E and sends the encrypted values $E(x_1), E(x_2), \ldots, E(x_n)$ to $\mathcal{C}$. $\mathcal{C}$ applies the private function F to the encrypted data, thereby computing

$F(E(x_1), E(x_2), \ldots, E(x_n))$. The result of this computation, by the fully homomorphic property of E, is equal to $E(F(x_1, x_2, \ldots, x_n))$. Next $\mathcal{C}$ sends this result to $\mathcal{H}$, who decrypts the value and thus recovers $F(x_1, x_2, \ldots, x_n)$. $\mathcal{H}$ then sends this decrypted value back to $\mathcal{C}$. Based on the received evaluation of F, the final plaintext message, the hospital $\mathcal{H}$ has a diagnosis or risk assessment for patient x. Thus, $\mathcal{C}$ never learns the plaintext patient data x and $\mathcal{H}$ never learns the function F, but does learn $F(x)$.

2 Encryption Overview

We first give a general description of FHE and its relevant terminology. The term *plaintext* refers to unencrypted data, whereas *ciphertext* refers to encrypted data. For our applications all data will be integers. Additionally, in order to enable the encryption process to select elements of the plaintext space uniformly at random, we require that the plaintext space be finite. This randomness is required for security. In the case that actual medical data is not measured in integers, we can merely "re-scale" the measurement by multiplying the value with a sufficiently large integer, to ensure that this property holds. Thus, we can guarantee that the set of plaintexts, $\mathbb{Z}_p$, will be the ring of integers modulo p. The prime number p here should be large enough that all plaintexts would be integers much less than p. Specific implementation details can be found in supplementary materials, [17,22]. To keep this paper relatively self-contained we describe our general encryption scheme below. Although certain details of the scheme are provided, the FHE scheme will essentially be implemented as a black box encryption function for the duration of this work.

- Plaintexts are elements of the ring $\mathbb{Z}_p$. We start by embedding $\mathbb{Z}_p$ into a larger ring R, which is a direct sum of several copies of $\mathbb{Z}_p$. We denote this embedding by α and the inverse map by β. The ring R can be public but both α and β are private; α is a part of the private encryption key, while β is a part of the private decryption key.
- Ciphertexts are elements of a ring S, such that $R \subset S$ is a subring of S. In our scheme S is, again, a direct sum of several copies of $\mathbb{Z}_p$. The ring S is public but contains a private ideal I such that $S/I = R$.
- Encryption is given by $E(u) = u + E(0)$, for an element $u \in R$, where $E(0)$ is a random element of the private ideal I. This encryption function is a homomorphism; the additive property is clear. For multiplication we have, for some $j_1, j_2, j_3 \in I$, that

$$E(u)E(v) = (u + j_1)(v + j_2) = uv + j_1u + uj_2 + j_1j_2 = uv + j_3 = E(uv)$$

- Decryption is computed with a private decryption key, a map $\rho : S \to R$ that takes every element of the ideal I to 0, followed by the map β from R to $\mathbb{Z}_p$.

Here is a diagram to visualize our general scheme:

$$\mathbb{Z}_p \xrightarrow{\alpha} R \xrightarrow{E} S \xrightarrow{\rho} R \xrightarrow{\beta} \mathbb{Z}_p$$

It should be noted that the number of distinct fully homomorphic embeddings, α, from $\mathbb{Z}_p$ into R is quite large, even if R is a direct sum of only a few copies of $\mathbb{Z}_p$. To explain this we first mention that a map $\alpha : 1 \to \alpha(1)$ extends to an embedding of $\mathbb{Z}_p$ into R if and only if $\alpha(1)$ is an idempotent of R, i.e., $\alpha(1)^2 = \alpha(1)$. When R is a direct sum of n copies of $\mathbb{Z}_p$, then it has 2^n idempotents, so there are 2^n different embeddings of $\mathbb{Z}_p$ into R. Similarly, a fully homomorphic encryption function $E : R \to S$ is an embedding of rings. Therefore, a similar argument applies to counting embeddings of R into S.

From the security point of view, it is known that all FHE schemes have a theoretical vulnerability [18]. In our scheme, the ideal I used for encryptions of zero is finite dimensional and therefore accumulating sufficiently many encryptions of zero may give an adversary the ability to recover I. Fortunately, the recovery of I is not a security threat from a practical point of view because I is not a part of the decryption key; the decryption key consists of the maps ρ and β. If an adversary has recovered the ideal I they may be able to recognize all encryptions of zero, but in real-life scenarios encryptions of zero are not frequently transmitted. We note that if a plaintext is "close" to zero, which can be the case in a medical database, the corresponding ciphertext does not have to be "close" to zero; the proposed encryption function does not preserve any metric and therefore does not preserve distance from zero. In fact, it is easy to see that any encryption function which does preserve such a metric cannot possibly be secure.

Our encryption scheme is completely secure against Ciphertext-Only Attack (COA). This means that a "hacker" who retrieves any encrypted portion of a database, regardless of the size, has only a negligible probability of correctly decrypting any portion of the database. COA security is fairly easy to achieve with a private-key non-homomorphic encryption. However, this property becomes very nontrivial for FHE. One of the main reasons we are able to achieve this property with our FHE scheme is the incredibly large number of fully homomorphic embeddings of $\mathbb{Z}_p$ in our public ring S. This implies that there are many different ways to decrypt any given ciphertext, but only one decryption is correct. Therefore, the probability to decrypt correctly is $\frac{1}{M}$, where M is the total number of possible decryptions. If M is large enough this probability is negligible. Our suggested parameters, which yield this security guarantee, have M on the order of 2^{128}.

3 Simulations

A machine learning solution should not only calculate the essential functions on encrypted patient data but also preserve privacy. In this paper, we focus on publicly available databases and high impact functions. First, we have conducted time series analysis on both synthetic and real heart rate data. Second, we have chosen to calculate known functions on Diabetes data, as well as a known predictive function on Parkinson's data. To illustrate the correctness and efficiency of our FHE scheme we perform experiments on this data. We report both error and computational overhead in the Results Section of the paper.

In the first simulation we apply a function, e.g., develop a model, on both encrypted and plain-text datasets to estimate/predict an output for each dataset and show that there is no difference between the outputs. We perform linear time-series analysis and fit a model to the encrypted data to understand the underlying structure and perform forecasting, monitoring and so on. Here, we use Recursive Least Squared (RLS) as the F function.

In the second simulation we apply a known classification function on encrypted data and measure the computational overhead. We focus on publicly available medical data for Parkinson's and diabetes, in order to illustrate the efficacy of our scheme on medical data.

3.1 Experiment One

In this simulation we play the role of an external entity $\mathcal{C}$: we are given an encrypted database D and train a function F on the encrypted data. As an example, an assessment function can tell whether a patient has a risk of heart attack, based on his/her medical data such as age, blood pressure, heart rate, etc. With the proposed encryption method, we can obtain an assessment function from encrypted data when we are given the relevant formula for obtaining that function from unencrypted data.

We perform linear univariate time series analysis on two separate databases using the RLS algorithm. Time series data can allow for high level medical analysis, including features calculated from variations in heart rate, heart rate variability, blood pressure, etc. The RLS algorithm implemented in this section is fairly straightforward.

Synthetic Data. The first database (DB 1) consists of synthetic data, generated according to a known distribution. The input signal is $x(n)$ and we generated the output signal, $y(n)$, via an equation presented in [26]:

$$y(n) = \sum_{k=0}^{N} h(k)x(n-k)$$

where $h(k) = 1/(k+1)$. Note that these parameter values were chosen arbitrarily: any values of N would yield similar results. We have applied the RLS algorithm to both the encrypted and unencrypted data. We then measured the mean squared error between the actual, known function value and our estimated function value after the 1000 iterations. The goal, of course, is to have minimal error, thereby indicating the feasibility of running the RLS function on encrypted data. Table 1 contains the error results for this database as well as the time difference between plaintext and encrypted data computations for one output.

Santa Fe Time Series Data. Our second database (DB 2) was time-series data from the Santa Fe Time Series Competition [30]. For simplicity, only Heart

Rate Variation (HRV) over time was considered. We applied RLS to both the unencrypted and encrypted heart rate data of patients. We measured the error between the actual heart rate and the one predicted by our function on both unencrypted and encrypted data. To reiterate, the error between our function's prediction and actual heart rate is reported in both situations. Because this simulation involves actual prediction, we hope for the error values on encrypted and unencrypted data to be as close as possible. The given measurements of the heart rate were in the range $[70.00, 100.00]$. Table One contains the results of our tests.

3.2 Experiment Two

In this simulation we have run several statistical tests on real-life databases encrypted by our method, including diabetic data [33] and Parkinson's data [34]. This simulation has real-life applications. For example, a hospital $\mathcal{H}$ has a private database D of patients' medical/clinical data. At the same time, a research center $\mathcal{C}$ has statistical tools that could help $\mathcal{H}$ assess risk. We therefore calculate this classification function on the following databases while maintaining the privacy of both the model and the data. Note that this scenario does not involve any training, merely homomorphic function evaluation on ciphertexts. We compute the same known polynomial function on encrypted and unencrypted data. We measured the time for encryption and decryption of the data for one patient as well as for the whole database. The execution time of a linear function was captured for each encrypted dataset. Table 2 provides the results of simulation.

Diabetic Database. The diabetic data (D) represents 10 years (1999–2008) of clinical care at 130 US hospitals and integrated delivery networks. The publically available input data contains many parameters on over $70,000$ patients, including demographic and clinical values. Each patients data includes over 50 features, representing patient and hospital outcomes. The predetermined classification function takes these recorded values, such as race, age and time of stay in hospital, and outputs a re-admission prediction.

Parkinson's Database. The Parkinson's data (P) encompasses a range of biomedical voice measurements from 42 patients with early-stage Parkinson's disease, recruited for a six-month trial of remote symptom progression monitoring. These recordings were automatically captured in the patients' homes. Parameters include demographic and clinical data, as well as features calculated from the recordings such as jitter and shimmer. The known function under consideration takes the patient data, including the dysphonia measurements, as input and out- puts the Pitch Period Entropy (PPE). The PPE value is highly correlated with the progression of Parkinson's disease. The fixed function under consideration is a linear combination of certain attributes in patient vector $x = \langle x_0, x_1, x_2, \ldots, x_n \rangle$, with precomputed coefficients $a_k : 1 \leq k \leq n$ taking the following form:

$$\text{PPE} = x_0 + \sum_{k=1}^{18} a_k x_k$$

4 Results

Tables 1 and 2 report the findings of the simulations. Table 1 reports the size of the RLS window, N, the computation error over plain-text and encrypted data, as well as the difference between these errors and the computational time. Table 2 reports the size of each database, the time to encrypt and decrypt both a single patient and the database, as well as the cost of function evaluation.

Table 1. Experiment One: RLS on Ciphertexts and Plaintexts. This table provides a comparison of RLS applied on both databases, over encrypted and unencrypted data

Database	N	Plaintext error	Encrypted error	Error difference	Time
DB 1	3	3.26×10^{-4}	3.29×10^{-4}	3×10^{-6}	0.004 ms
DB 1	9	2.95×10^{-4}	3.02×10^{-4}	7×10^{-6}	0.009 ms
DB 2	3	1.264	1.269	0.005	0.005 ms
DB 2	9	1.124	1.129	0.005	0.01 ms

In this scope, the importance of these results lies not in the error values themselves but in the proximity between the error values. If the error values are close, this implies that RLS behaves nearly identically on ciphertexts and plaintexts. The closer the error values are the less error can be attributed to the encryption scheme. Ideally, these error values would be the same but, because we must scale all plaintext values to integers prior to encryption, we introduce rounding error. In other words, the difference between the encrypted and unencrypted error values is due to rounding error generated by rescaling the real numbers to integer values. The column "Encrypted Error" in Table 1 reports the error from the recursive function. This computation includes embedding all data into the integers modulo p via a common scaling integer, a power of ten, encrypting, computing, decrypting, and dividing by the scaling integer.

Clearly, the accuracy of the assessment function on the synthetic data is practically the same for unencrypted and encrypted data. This shows the ability of the encryption function to handle RLS training while maintaining correctness. It can be seen that the difference in error over encrypted and unencrypted data for both the synthetic and real situations is negligible. This illustrates that basic machine learning models, those that utilize the proper arithmetic operations, can be trained on ciphertexts while maintaining correctness. Time series analysis is a fundamental aspect of medical data analytics and these experiments have shown that our FHE scheme permits the necessary computations for such analytics.

Note that the difference in error does not scale with N, the RLS window size. This shows that the error introduced by increasing the number of variables involved in computation is not expected to generate significant error. Unavoidably, encrypted computation time is larger than unencrypted computation time. While this computational overhead will increase with the number of patients, this scheme is feasible for performing computation on encrypted data. Table One shows that computation time with our method is still quite practical. This is not the case with alternative FHE implementation methods. See, for example, the work done by Ducas and Micciancio [10], which achieves a single bootstrapped NAND computation in 0.69 s. Additionally, work provided by Aslett et al. [2] claim a single scalar addition at 0.003 s and a single scalar multiplication at 0.084 s even using high performance computers.

Table 2. Experiment Two: Known functions computed on publicly available data.

DB	DB size MB	Patient records	Time to encrypt record	Time to encrypt DB	Time to decrypt record	Time to decrypt DB	Time to evaluate function
D	19	101767	0.02445	2488	0.18152	18472.7	1×10^{-5}
P	1	5876	0.01194	70.1	0.09798	575.7	6×10^{-6}

The results of simulation two can be seen in Table 2. With this simulation we showed that given a polynomial function, e.g., a medical diagnostic or prognostic function, both H and C can evaluate the function homomorphically on encrypted data. This allows $\mathcal{C}$ to maintain the integrity of its intellectual property, F, while still using the function to classify private data D. There is no discrepancy between the assessment, i.e., function evaluation, on each individual patient regardless of whether the function was calculated on encrypted or unencrypted data. Therefore, instead of reporting any error measurement we report the efficiency of the scheme and computation on ciphertexts by applying the same linear functions to unencrypted and encrypted data. The efficiency of the scheme is noteworthy - with this simulation we have shown that function evaluation can be performed relatively quickly using our FHE scheme. In this table it is also notable that the time to decrypt is higher than the time to encrypt. This discrepancy is due to the fact that encryption is essentially an addition, while decryption requires computing the remainder of a ciphertext modulo the ideal I via the decryption map.

5 Conclusions

We applied our method of Fully Homomorphic Encryption (FHE) to calculate medical diagnostic functions based on encrypted medical data. Our FHE scheme permits the use of machine learning algorithms that utilize polynomial kernel

functions. These computations allow for medical diagnostics to be performed on encrypted data, maintaining the privacy of patient data. We outlined example scenarios where secure machine learning could be useful within the medical community, considering the protection of patient data as well as a researchers intellectual property.

Time series analysis was performed on synthetic and real data, showing that the increase in error is negligible when operating on encrypted data. Then known classification functions were applied to public medical databases without introducing additional error that illustrates the efficiency of our encryption method.

We have shown that it is possible to train an assessment function on encrypted data provided that relevant formulas for obtaining such a function from unencrypted data are available. Our method provides very efficient computation on encrypted data, which allows us to compute any polynomial function on a single patient's encrypted data in a fraction of a second. A polynomial function can be computed on a medical encrypted database in a couple of minutes.

We have shown that the encryption scheme is efficient enough to be practical, secure and correct for linear classifiers as well as time series analysis. Theoretically, it is possible to extend the applicability of the scheme to different families of machine learning functions. Planned future work includes adapting the current encryption scheme to working with non-linear classifiers, such as Volterra systems. This generalization would show that the encryption scheme permits a larger class of machine learning functions to be computed on encrypted data.

The above implementation of the FHE function allows for efficient data mining without decryption while maintaining correctness. Therefore, it is completely feasible to consider the utilization of this encryption function for highly sensitive, private, and federally regulated medical data.

References

1. Aono, Y., Hayashi, T., Trieu Phong, L., Wang, L.: Scalable and secure logistic regression via homomorphic encryption. In: Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, pp. 142–144. ACM (2016)
2. Aslett, L.J., Esperança, P.M., Holmes, C.C.: A review of homomorphic encryption and software tools for encrypted statistical machine learning. arXiv preprint arXiv:1508.06574 (2015)
3. Bos, J.W., Lauter, K., Naehrig, M.: Private predictive analysis on encrypted medical data. J. Biomed. Inform. **50**, 234–243 (2014). Special Issue on Informatics Methods in Medical Privacy
4. Bost, R., Popa, R.A., Tu, S., Goldwasser, S.: Machine learning classification over encrypted data. In: NDSS (2015)
5. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. ACM Trans. Computat. Theor. (TOCT) **6**(3), 13 (2014)
6. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, pp. 97–106. IEEE Computer Society, Washington, DC (2011)

7. Centers for Disease Control and Prevention: HIPAA privacy rule and public health. Guidance from CDC and the US department of health and human services. MMWR Morb. Mortal. Wkly. Rep. **52**(Suppl. 1), 1–17 (2003)
8. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Faster fully homomorphic encryption: bootstrapping in less than 0.1 seconds. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10031, pp. 3–33. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53887-6_1
9. Du, W., Han, Y.S., Chen, S.: Privacy-preserving multivariate statistical analysis: linear regression and classification. In: Proceedings of the 2004 SIAM International Conference on Data Mining, pp. 222–233. SIAM (2004)
10. Ducas, L., Micciancio, D.: FHEW: bootstrapping homomorphic encryption in less than a second. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 617–640. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_24
11. El Emam, K., Jonker, E., Arbuckle, L., Malin, B.: A systematic review of re-identification attacks on health data. PloS One **6**(12), e28071 (2011)
12. Evans, D., Huang, Y., Katz, J., Malka, L.: Efficient privacy-preserving biometric identification. In: Proceedings of the 17th conference Network and Distributed System Security Symposium, NDSS (2011)
13. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive 2012, 144 (2012)
14. Gentry, C., Boneh, D.: A Fully Homomorphic Encryption Scheme, vol. 20. Stanford University, Stanford (2009)
15. Gentry, C., Halevi, S.: Implementing Gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_9
16. Graepel, T., Lauter, K., Naehrig, M.: ML confidential: machine learning on encrypted data. In: Kwon, T., Lee, M.-K., Kwon, D. (eds.) ICISC 2012. LNCS, vol. 7839, pp. 1–21. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37682-5_1
17. Gribov, A., Kahrobaei, D., Shpilrain, V.: Practical private-key fully homomorphic encryption in rings. Groups Complex. Cryptol. **10**(1), 17–27 (2018)
18. Grigoriev, D., Ponomarenko, I.: Homomorphic public-key cryptosystems over groups and rings. arXiv preprint cs/0309010 (2003)
19. Halevi, S., Shoup, V.: Algorithms in HElib. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 554–571. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_31
20. Halevi, S., Shoup, V.: Helib (2014). HELib: https://github.com.shaih/HElib
21. Hall, R., Fienberg, S.E., Nardi, Y.: Secure multiple linear regression based on homomorphic encryption. J. Off. Stat. **27**(4), 669 (2011)
22. Kahrobaei, D., Lam, H., Shpilrain, V.: System and method for private-key fully homomorphic encryption and private search between rings. Patent US20170063526, 25 August 2017
23. Lauter, K., López-Alt, A., Naehrig, M.: Private computation on encrypted genomic data. In: Aranha, D.F., Menezes, A. (eds.) LATINCRYPT 2014. LNCS, vol. 8895, pp. 3–27. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16295-9_1
24. Lindell, P.: Privacy preserving data mining. J. Cryptol. **15**(3), 177–206 (2002)
25. Liu, F., Ng, W.K., Zhang, W.: Encrypted SVM for outsourced data mining. In: 2015 IEEE 8th International Conference on Cloud Computing (CLOUD), pp. 1085–1092. IEEE (2015)

26. Ljung, L.: System Identification: Theory for the User. Prentice-Hall, Upper Saddle River (1987)
27. Naehrig, M., Lauter, K., Vaikuntanathan, V.: Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop, pp. 113–124. ACM (2011)
28. Nikolaenko, V., Weinsberg, U., Ioannidis, S., Joye, M., Boneh, D., Taft, N.: Privacy-preserving ridge regression on hundreds of millions of records. In: 2013 IEEE Symposium on Security and Privacy, pp. 334–348, May 2013
29. Poggio, T., Smale, S.: The mathematics of learning: dealing with data. Not. AMS **50**(5), 537–544 (2003)
30. Rigney, D.R., Goldberger, A.L., Ocasio, W., Ichimaru, Y., Moody, G.B., Mark, R.: Multi-channel physiological data: description and analysis (1993)
31. Sadeghi, A.-R., Schneider, T., Wehrenberg, I.: Efficient privacy-preserving face recognition. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 229–244. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14423-3_16
32. Sarwate, A.D., Chaudhuri, K.: Signal processing and machine learning with differential privacy: algorithms and challenges for continuous data. IEEESignal Process. Mag. **30**(5), 86–94 (2013)
33. Strack, B., et al.: Impact of HbA1c measurement on hospital readmission rates: analysis of 70,000 clinical database patient records. BioMed Res. Int. **2014** (2014)
34. Tsanas, A., Little, M.A., McSharry, P.E., Ramig, L.O.: Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests. IEEE Trans. Biomed. Eng. **57**(4), 884–893 (2010)
35. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_2
36. Wiens, J., Guttag, J., Horvitz, E.: A study in transfer learning: leveraging data from multiple hospitals to enhance hospital-specific predictions. J. Am. Med. Inform. Assoc. **21**(4), 699–706 (2014)
37. Yang, J.J., Li, J.Q., Niu, Y.: A hybrid solution for privacy preserving medical data sharing in the cloud environment. Futur. Gener. Comput. Syst. **43**, 74–86 (2015)
38. Zhang, J., Zhang, Z., Xiao, X., Yang, Y., Winslett, M.: Functional mechanism: regression analysis under differential privacy. Proc. VLDB Endow. **5**(11), 1364–1375 (2012)