Privacy-Preserving Aging Analytics

Shelly Brezner , Ran Toledo

Lecturer: Dr. Adi Akavia, Privacy-Preserving Machine Learning Laboratory

Department of Computer Science, University of Haifa

Abstract

In recent studies, it has been found that DNA methylation could be an accurate biomarker for aging in humans. The process of identifying accurate biomarkers such as this and their affecting factors could substantially increase our understanding of the aging process. Despite analytical tools to model the change of methylation and aging being relatively undeveloped, a recent line of work known as the Epigenetic PaceMaker (EPM) has developed a statistical, likelikhood-based approached to model precisely that. However, employing such a model on human subjects raises privacy concerns and if privacy is not ensured it could have a vast range of serious implications such as spikes in health insurance, job stability risk, etc. Also, the input for the EPM model relies on sensitive personal data and the model itself could also be confidential, for example; when it is a proprietary asset of a commercial entity. Therefore it is highly crucial to add a privacy component to the EPM system. In our work, we have developed a privacy-preserving protocol to securely infer epigenetic aging using chronological ages and DNA methylation arrays. Our solution builds on the EPM method [1, 2] along with privacy-preserving solutions for linear regression [3, 4].

1 Introduction

1.1 DNA Methylation

DNA is the hereditary material in humans and almost all other organisms. The information in DNA is stored as a code made up of four chemical bases: A, T, G and C. The order, or sequence, of these bases determines the information available for building and maintaining an organism. Each base is attached to a sugar molecule, and any two sugars are linked together by phosphate. CpG sites are regions of DNA where a C base is followed by a G base, separated by only one phosphate group in the linear sequence. CpG islands are regions of the genome that contain a large number of CpG repeats. In mammalian genomes, for example, CpG islands usually extend for 300–3000 base pairs. DNA contains the information needed to make all of our proteins, and RNA is a messenger that carries this information. The process of copying a segment of DNA into RNA is called transcription. In order to perform this process,

transcription factors must know where to start transcribing. Promoter sequences are DNA sequences that define where transcription of genes begins and CpG islands are often present around these promoter sequences. Methylation is the process by which methyl groups are added to the DNA molecule. When a CpG site is methylated within the promoter region of a gene, its transcription is inhibited, resulting in a loss of gene expression. DNA methylation has proven to be a remarkably accurate biomarker for human age, allowing the prediction of chronological age to within a couple of years. Owing to this, DNA methylation levels can be used to accurately estimate the age of tissues and cell types, hence forming an accurate epigenetic clock.

1.2 Epigenetic PaceMaker System

The first description of a robust epigenetic clock that worked across different tissues was presented by Steve Horvath and consisted of 353 CpG sites whose weighted sums yield an estimate of chronological age to within a couple of years. The time linearity of the Horvath clock makes it the "epigenetic parallel" of the classical evolutionary version of the molecular clock (MC). Mechanisms from molecular evolution have been adapted to the process of methylation, providing aleviation from the rigidness of the Horvath clock. Specifically, the Universal PaceMaker (UPM) allows for rate variability as opposed to the constancy imposed by the molecular clock. In the UPM approach, we assume that all sites are changing linearly with an adjusted time, which is a non-linear function of the chronological time. The CEM-UPM algorithm used in the Epigenetic PaceMaker extends the classical EM algorithm (Expectation Maximization) by portioning the parameter set into several parts when optimizing the likelihood function over the entire set is hard. The choice of partitioning is important as it determines the optimization algorithm.
The partitioning uses two fast optimization algorithms:

1. In the first step, use the same parameters as used by the linear MC model however using a simple, direct solution relying on the special structure of the UPM. This yields a significant practical and asymptotic reduction in both time and space. Also referred to as the '*site step*', in which the site-specific parameters; rate and starting state, are optimized.

2. The '*time step*' of the CEM, in which each individual's times (biological ages) are optimized based on the rate and starting states optimized in the prior step.

The UPM framework models epigenetic aging better than the molecular clock- an observation strengthened by analysis of the algorithm's classification of substantially larger inputs of individuals and methylation sites [7]. It allows sites to accelerate and decelerate jointly, accounting for non-linear trends in aging.

1.3 Privacy-Preserving Linear Regression

To enhance the efficacy of a learned Linear Regression model, prior experience in model training suggests using training data from a large and diverse set. A simple way to obtain such a training dataset is to merge data contained in "data silos" collected

by different entities. In many applications, the data points in these data silos encode sensitive information and are possibly collected by multiple distrustful entities. These entities often cannot (due to legal obligations) share the private data contained in their silos, hence making collaborative analysis on joint data impossible. The challenge is training a linear regression model on joint data that must be kept confidential and/or is owned by multiple distrustful parties. The paper [3] proposes an efficient solution in the two-server model commonly used by previous works on privacy-preserving machine learning, where no party needs to be trusted to handle the data in the clear. In this model, the computation of the model is outsourced to two non-colluding third parties. First phase: collecting private encrypted data from many possibly many data-owners. Second phase: the two third-parties engage in the computation of the model itself. The system described is based only on a simple cryptographic primitive that can be implemented via efficient constructions – a linearly-homomorphic (partial) encryption scheme (LHE). Such an encryption scheme enables computing the sum of encrypted messages. The system achieves practical performances when implemented using a standard encryption scheme such as Pailler's Cipher. Experiments described in [3] demonstrate that for many real scenarios, LHE is all that is needed to privately, yet efficiently, train a ridge or linear regression model on distributed data. The system guarantees confidentiality for the data silos of each research group involved. In the second phase, to compute the ridge regression model which is a solution of a linear system of the form $Aw = b$ (Matrix A and vector b are encrypted and must be kept private and the solution w^* is the model) the system designed a two-party protocol that solves this equation using only the linear homomorphic property of the underlying encryption scheme. This allows to boost the overall performance, and in particular, to considerably reduce the communication overhead. Gradient based solutions for solving the model use iterative algorithms and present the problem of estimating the number of iterations t which is either fixed or chosen adaptively based on the dataset. This can be infeasible in the privacy-preserving setting and the solution that the system designed in the paper for solving $Aw = b$ does not present this issue.

2 Preliminaries

2.1 CEM-UPM Algorithm

2.1.1 Parameters

- Model has m individuals (indexed by j) and n methylation sites (indexed by i)
- t_j is a time period representing individual j's age
- s_i is a methylation site that undergoes changes throughout the lifespan of an individual
- r_i is the change rate of methylation site i
- s_i^0 is the initial (starting) methylation level of site i
- $s_{i,j}$ measures the methylation level at sites$_i$, in individual j at time t_j

Under the molecular clock model (rate is constant over time), we expect: $s_{i,j} = s_i^0 + r_i t_j$. The observed value after noise addition is:$s_{i,j} = s_i^0 + r_i t_j + \varepsilon_{i,j}$, however this

model assumes that $\varepsilon_{(i,j)} \sim N(0, \sigma^2)$ and is therefore neglectible. Given the input matrix $\hat{S} = [\hat{s}_{i,j}]$ holding the observed methylation level at site s_i of individual j, the goal is to find the maximum likelihood values for the variablesr_i and s_i^0 for each site i $(1 \leq i \leq n)$. Maximizing the model's likelihood is equivalent to minimizing the residual sum of squares (RSS) function: $\sum_{i \leq n} \sum_{j \leq m} (s_{i,j} - (s_i^0 + r_i t_j))^2$

It is important to note that in contrast to the MC, under the UPM model sites may change their rate at any point in life (arbitrarily and independently of their counterparts in other individuals) and when this happens, all sites of that individual change their rate proportionally, meaning the ratio $r_i / r_i^{'}$ is constant between any two sites $i, i^{'}$ at any individual j at all times. It is equivalent to extending the age of an individual j by the same proportion of the rate change. This new age is denoted as the individual's *epigenetic age* as opposed to their *chronological age*. In conclusion, under this model we have another variable, the age: t_j.

2.1.2 Methods

For the MC model, the solution is given by the vector $\tilde{\beta}$ in the following equation : $\tilde{\beta} = (X^T X)^{-1} X^T y$ [2], where y is a vector holding the observed methylation values - in our case the entries of $\tilde{S}$, and X is a $(2n \times nm)$ matrix over the variable's coefficients in the problem (*see Figure 1*). $\tilde{\beta}$ is a $2n$ vector in which the first n variables represent r_i's variants and the second n variables are the s_i^0 variants.

$$X = \left[\begin{array}{cccc|cccc}
t_1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\
t_2 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\
 & & \vdots & & & \vdots & & \\
t_m & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\
0 & t_1 & \cdots & 0 & 0 & 1 & \cdots & 0 \\
0 & t_2 & \cdots & 0 & 0 & 1 & \cdots & 0 \\
 & & \vdots & & & \vdots & & \\
0 & t_m & \cdots & 0 & 0 & 1 & \cdots & 0 \\
 & & \vdots & & & \vdots & & \\
 & & \vdots & & & \vdots & & \\
0 & \cdots & 0 & t_1 & 0 & \cdots & 0 & 1 \\
0 & \cdots & 0 & t_2 & 0 & \cdots & 0 & 1 \\
 & & \vdots & & & \vdots & & \\
0 & \cdots & 0 & t_m & 0 & \cdots & 0 & 1
\end{array}\right]$$

Figure 1: The $mn \times 2n$ matrix X used to compute the MC model

The standard algebraic implementation of the above equation is computationally heavy (even moreso on ciphertexts), as it requires the multiplication of a tremendous amount of values. In [9], it is shown that a closed form solution exists that can solve this equation in $O(nm)$ time and space:

Assume there are 3 sites and 3 people aged 10, 15, and 30. The X matrix would be:

$$\begin{pmatrix} 10 & 0 & 0 & 1 & 0 & 0 \\ 15 & 0 & 0 & 1 & 0 & 0 \\ 30 & 0 & 0 & 1 & 0 & 0 \\ 0 & 10 & 0 & 0 & 1 & 0 \\ 0 & 15 & 0 & 0 & 1 & 0 \\ 0 & 30 & 0 & 0 & 1 & 0 \\ 0 & 0 & 10 & 0 & 0 & 1 \\ 0 & 0 & 15 & 0 & 0 & 1 \\ 0 & 0 & 30 & 0 & 0 & 1 \end{pmatrix}$$

Denote $M = X^t X = \begin{matrix} A & B \\ C & D \end{matrix}$, where $A = diag(\sum t_i^2)$, $B = C = diag(\sum t_i)$, $D = diag(m = 3)$

$$\begin{pmatrix} 1225 & 0 & 0 & 55 & 0 & 0 \\ 0 & 1225 & 0 & 0 & 55 & 0 \\ 0 & 0 & 1225 & 0 & 0 & 55 \\ 55 & 0 & 0 & 3 & 0 & 0 \\ 0 & 55 & 0 & 0 & 3 & 0 \\ 0 & 0 & 55 & 0 & 0 & 3 \end{pmatrix}$$

$N = M^{-1} = (X^T X)^{-1} \rightarrow N = \wedge * \begin{matrix} D^{-1} & B \\ C & A^{-1} \end{matrix}$, where $\wedge = \frac{1}{((\sum_{i \le m} t_i)^2 - m \sum_{i \le m} t_i^2))}$

$$\begin{pmatrix} \frac{3}{650} & 0 & 0 & -\frac{11}{130} & 0 & 0 \\ 0 & \frac{3}{650} & 0 & 0 & -\frac{11}{130} & 0 \\ 0 & 0 & \frac{3}{650} & 0 & 0 & -\frac{11}{130} \\ -\frac{11}{130} & 0 & 0 & \frac{49}{26} & 0 & 0 \\ 0 & -\frac{11}{130} & 0 & 0 & \frac{49}{26} & 0 \\ 0 & 0 & -\frac{11}{130} & 0 & 0 & \frac{49}{26} \end{pmatrix}$$

$M' = NX^T = (X^T X)^{-1} X^T$ is a $2n \times mn$ matrix. We divide M into two matrices: U and L (both $n \times mn$ matrices).

- U is constructed as follows: in each row $1 \le k \le n$ the values in indexes $(km - m + 1, \ldots, km)$ are calculated as: $D^{-1} * t_j + B$ and the rest are equal to 0
- L is constructed as follows: in each row $1 \le k \le n$ the values in indexes $(km - m + 1, \ldots, km)$ are calculated as: $C * t_j + A^{-1}$ and the rest are equal to 0

In the example above, U and L appear as illustrated below:

$$\begin{pmatrix} -\frac{1}{26} & -\frac{1}{65} & \frac{7}{130} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{26} & -\frac{1}{65} & \frac{7}{130} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{26} & -\frac{1}{65} & \frac{7}{130} \\ \frac{27}{26} & \frac{8}{13} & -\frac{17}{26} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{27}{26} & \frac{8}{13} & -\frac{17}{26} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{27}{26} & \frac{8}{13} & -\frac{17}{26} \end{pmatrix}$$

In conclusion, instead of using the X it is enough to calculate a vector N' of size $2m$ for the calculation, that is constructed from the following equation:
For each $1 \le j \le m$, the first m values are: $-m * \wedge * t_j + \sum_{i \le m} t_i * \wedge$, and the other m values are: $\sum_{i \le m} t_i * \wedge * t_j - \sum_{i \le m} t_i^2 * \wedge$

However, despite this available solution, it is not implemented in our privacy-preserving protocol due to the limitations of the encryption scheme. Unfortunately, this results in increased runtime and space complexity but does not impact the model's correctness.

In the UPM model, as noted before, we have another variable which needs to be estimated, t_j . Hence the least square problem is not linear in this case since this set of variables needs to be optimized as well. Therefore, we use a heuristic solution that provides a good result in reasonable time and for a non-trivial dataset. We partition the set of variables into two: one is the set of rates and initial (start) states, and the other is the set of times. The algorithm uses conditional expectation maximization (CEM) and optimizes each set separately by alternating between two steps. Optimize the first set by the same equation as in MC model and then optimize the second set by using the equation [1]:

$$t_j = \frac{\sum_{i \leq n} (r_i(\hat{s}_{i,j} - s_i^0))}{\sum_{i \leq n} r_i^2}$$

After optimizing both sets, we calculate the residual sum of squares (RSS) in order to determine if it is necessary to perform another iteration. This is done by calculating: $RSS = \sum_{i \leq n} \sum_{j \leq m} (\hat{s}_{i,j} - (s_i^0 + r_i t_j))^2$.

2.1.3 Complete Algorithm:

Given $\hat{S}$, δ_{CEM} and n (δ_{CEM} is the minimal improvement acceptable, n is the maximum number of iterations):

1. Apply the site step: Find the first set containing r_i and s_i^0 (rates and initial states). If the algorithm is in its first iteration then t_j is the chronological age provided by the dataset. During the proceeding iterations, use the values of t_j calculated in the previous iteration.

2. Apply the time step: Find the second set (t_j) by the equation described in SECTION 2.2.2 using the parameters calculated in stage 1.

3. Calculate the residual sum of squares using the values of $(\hat{S}, t, s^0, r)$ from the previous iteration. Denote the result as RSS0.

4. Calculate the residual sum of squares using the values of $(\hat{S}, t, s^0, r)$ from the current iteration. Denote the result as RSS1.

5. If $RSS0–RSS1 > \delta_{CEM}$ and the maximum number of iterations n has not yet been reached, update the values of t, s^0, r with those from the current iteration and return to stage 1.

Before applying the algorithm, k sites are selected from the dataset ($k \leq n$), for example, the sites with the largest Pearson correlation. This is due to the noise added to methylation patterns and the Epigenetic PaceMaker's ultimate goal being to find CpG sites that change in predictable ways in order to develop an accurate estimate of biological age. The algorithm is then run solely on the values of $\hat{S}$ containing these sites, and the rest are discarded from the dataset.

2.2 Cryptographic Tools

2.2.1 Linear Regression

A linear regression learning algorithm is a procedure that on inputs $\{(x_1, y_1), ..., (x_n, y_n)\}$ where $x_i \in R^d, y_i \in R$),outputs a vector $w^* \in R^d$ such that ${w^*}^T \cdot x_i \approx y_i$ for all i. The model $w^*$is computed by minimizing $Aw^* = b$ where A is a $n \times d$ matrix which contains $x_1, \ldots, x_n$ and b is an n-sized vector containing $y_1, \ldots, y_n$.

2.2.2 Linearly Homomorphic Encryption

The privacy-preserving system in our protocol utilizes homomorphic encryption. $(M, +)$ is a finite group. A linearly homomorphic encryption (LHE) scheme for messages in M is defined by the following three algorithms:

1. **Key-generation algorithm** $(Gen(k))$: This algorithm takes an input of the security parameter k and outputs a matching pair of secret and public keys (sk, pk)
2. **Encryption algorithm** $(Enc_{pk}(m))$: A randomized algorithm that uses the public key pk to transform a message m from M into ciphertext c
3. **Decryption algorithm** $(Dec_{sk}(c))$: A deterministic algorithm which uses the secret key sk to recover the original plaintext from the ciphertext c

The LHE scheme satisfies two important properties. The first of them being the *security property*, meaning it is infeasible for any computationally bounded algorithm to gain extra information about a plaintext when given only its ciphertext and the public key pk. The latter is the *homomorphic property*; being that an operation $\bigodot$ exists such that for any a ciphertexts $c_1, \ldots, c_a$ $(c_i \leftarrow Enc_pk(m_i))$, it holds that $Dec_{sk}(c_1 \bigodot \ldots \bigodot c_a) = m_1 + \cdots + m_a$. This implies that for any $c = Enc_{pk}(m)$, $Dec_{sk}(c \bigodot \ldots \bigodot c) = a \cdot m$. Hence, LHE schemes enables computing the sum of ciphertexts and multiplication of ciphertexts and plaintexts however does not support the multiplication of two or more ciphertexts. In some cases, being able to perform only linear operations on encrypted data is not sufficient. A general solution to the problem of computing on encrypted data can be obtained via the use of *fully homomorphic encryption*, however, systems constructed using fully homomorphic encryptions are still very inefficient and more efficient solutions are available for evaluating low-degree polynomials over encrypted data functionalities (*e.g.*, privacy-preserving ridge regression[11]).

3 Theory

3.1 System Overview

Consider the setting where the training dataset is not available in the clear to the entity that wants to train the ridge regression model. The entity needs the help of the party handling the cryptographic keys in order to learn the desired model. The protocol described in [9] is designed for the following parties:

- **The Data-Owners:** m data owners $DO_1, \ldots, DO_m$ where each data owner DO_i has a dataset D_i and is willing to share it only if it is encrypted. In our

case, these might be different medical centers that each have a dataset containing information about patients' ages and methylation rates.

- **Machine-Learning Engine (MLE):** The party that wants to run a linear regression algorithm on the dataset $\mathscr{D}$ which is obtained by merging $D_i, \ldots, D_m$ however is only given access to encrypted copies of the datasets. In our case, this would be the Epigenetic Pacemaker using the CEM-UPM linear regression algorithm.
- **Crypto-Service Provider (CSP):** Takes care of initializing the encryption scheme used in the system and interacts with the MLE. CSP manages the cryptographic keys and is the only entity capable of decrypting them.

We assume that MLE and CSP do not collude and that for each pair of parties involved in the protocol there exists a private and authenticated peer-to-peer channel in which communication amongst any two players cannot be eavesdropped. The goal is to ensure that only the MLE obtains the desired model, without MLE and CSP learning any information about the private datasets D_i.

3.2 Linear Regression with only LHE Protocol Description

The protocol described in [9] for linear regression over a linearly homomorphic encryption scheme follows a two-phase architecture:

Phase 1:

1. CSP generates the key pair (sk, pk), stores sk and makes pk public.
2. Each data owner DO_i sends MLE ciphertexts of the values in their dataset D_i, encrypted using the public key pk.
3. MLE uses these ciphertexts and the homomorphic property of the encryption scheme in order to obtain encryptions of A and b (coefficient matrix and vector). These encryptions are denoted as $Enc_{pk}(A)$ and $Enck_p(b)$.

Phase 2:

1. MLE uses the ciphertexts $Enc_{pk}(A)$ and $Enck_p(b)$ and private random values from the message space in order to obtain encryptions of new values called "masked data". This is referred to ask the *masking trick* and is elaborated on below.
2. These encryptions are then sent to the CSP, which decrypts and runs a given algorithm on the masked data.
3. The output of this computation is the "*masked model*", a vector $\tilde{w}$, which is then sent back from the CSP to the MLE.
4. MLE computes the output w^* from the masked model $\tilde{w}$ by using the same private random values sampled in stage 1.

We say the system is *correct* if the model computed by the MLE is equal to the model computed by the learning algorithm in the clear using the unencrypted data

as training data. We say the system is *private* if the distribution of the masked data sent by MLE to the CSP is independent of the distribution of the local inputs. In the protocols described in [9], Phase 1 is dependent on whether the dataset is horizontally or arbitrarily partitioned. However, in both cases, MLE gets the encryptions of A and b and the CSP takes care of initializing the cryptographic primitive and generates the relative keys. In our implementation, we assume that the dataset is vertically partitioned and that all datasets are shaped the same and contain data from the same methylation sites (however the observed methylation values and their corresponding sites are of course, ultimately unknown to the MLE nor the CSP). Phase 2 is realized by an interactive protocol between the MLE and the CSP; the CSP takes the encryptions of A and b from the MLE and returns the solution of the system $Aw = b$ following the pattern mentioned earlier known as the "*masking trick*":

1. The MLE samples a random invertible matrix R (size $d \times d$, containing values from the message space M) and a random vector $r \in M^d$, and uses the linear homomorphic property of the encryption scheme to compute $C = Enc_{pk}(AR)$ and $d = Enc_{pk}(b + Ar)$. The values $AR, b + Ar$ are the "*masked data*".

2. The CSP decrypts C and d and computes $\tilde{w} = C^{-1}d$. The vector $\tilde{w}$ is the "*masked model*" which is then sent back to the MLE.

3. The MLE computes the desired model as $w^* = R\tilde{w} - r$

If R and r are sampled uniformly at random, then the distribution of the masked data is independent of A and b and thus satisfying the privacy property. In our case, the CEM-UPM algorithm requires several iterations in order to optimize the values according to RSS. This requires several iterations of encryption and decryption between MLE and CSE.

4 Implementation

In our implementation, we've adjusted the CEM-UPM algorithm to be compatible with the protocol outlined in the previous chapter for linear regression over a LHE scheme. Our goal is to be able to correctly compute the Epigenetic Pacemaker's output on encrypted datasets obtained from m different data-owners using a modified version CEM-UPM algorithm . Prior to the initiation of the protocol, a limit of ℓ iterations is set.
Our privacy-preserving protocol for the Epigenetic Pacemaker is realized in the following phases:

Phase1:

This phase is almost entirely identical to the LHE linear regression protocol described above:

1. CSP generates the key pair (sk, pk) using the Paillier cryptosystem, stores sk and makes pk public.

2. Each data owner encrypts their dataset using pk and sends it to the MLE:

- It is important to note that the protocol assumes that pearson correlation was computed by the data-owners on their respective datasets prior to the encryption process and that all datasets contain data about the same agreed upon methylation sites (however which sites those are is ultimately unknown and irrelevant to the MLE). This assumption is critical in reducing runtime complexity and is also necessary due to the nature of the underlying linearly homomorphic encryption scheme.
- Another assumption mentioned earlier is that all datasets provided by the data owners are shaped alike, allowing the MLE to merge the data in negligible time.

Phase2-A: '*site step*'

1. MLE uses these encrypted datasets and the homomorphic property of the encryption scheme in order to obtain encryptions of X and y (as defined in [2])
   (a) First, MLE merges the encrypted datasets it has received from the data owners. This is easily performed due to the assumption that all datasets are shaped alike and contain data from the same methylation sites (in the dataset we've used [12] to validate the protocol, this is equivalant to merging a vertically partitioned dataset).
   (b) Then, the MLE utilizes the homomorphic property of the encryption scheme in order to compute $Enc_{pk}(X)$ and $Enc_{pk}(y)$ from the merged encrypted dataset, as defined in the CEM-UPM algorithm [2].
      - X is a $(2n \times nm)$ matrix consisting of age values. In the first iteration of our system's protocol, these ages are the encrypted values in the merged dataset. In the following iterations, the values of the predicted ages computed in the time step of the previous iteration are used to obtain this matrix. These predicted ages are unencrypted and are revealed to the MLE at the end of each iteration's time step. This is due to the fact that concealing them would require a more complex protocol involving interraction between a third-party and the CSP, and such a protocol is not covered in the paper we've based our implementation on [9].
      - The y vector contains the observed methylation values ($y = (s_{1,1}, \ldots, s_{n,m} | s_{i,j} \in \hat{S})$) and remains constant and encrypted throughout all iterations of the protocol.
2. MLE uses the ciphertexts $Enc_{pk}(X)$ and $Enc_{pk}(y)$ and private random values in order to obtain encryptions of new values called "*masked data*":
   (a) MLE samples a random **invertible** matrix R and a random vector r consisting of values in the range $[0, \delta]$
   (b) MLE uses the homomorphic property of the encryption scheme in order to compute $C = Enc_{pk}(XR)$ and $d = Enc_{pk}(Xr + y)$ (these computations consist solely of multiplying ciphertexts and plaintexts)
3. These encryptions are then sent to the CSP:

(a) The CSP decrypts C and d and obtains $C' = Dec_{sk}(C)$, $d' = Dec_{sk}(d)$ (the data in C',d' is still masked and hence not revealed to the CSP)

(b) The CSP computes $(C'^T C')^{-1} C'^T d$ (this is equivalant to solving $\tilde{\beta}$ in the equation given for solving the MC model)

4. This computation's output is the "*masked model*", a vector $\tilde{w}$, which is then sent back from the CSP to the MLE

5. MLE computes the output w^* from $\tilde{w}$ by unmasking the data: $w^* = R\tilde{w} - r$

Phase2-B: '*time step*'

1. MLE uses the output w^* from Phase 2-A to retrieve the rates and initial states. w^* is obtained by decrypting and unmasking $\tilde{\beta}$, which, as mentioned earlier, is a $2n$-sized vector in which the first n variables represent the rates (r_i) and the second n variables are the initial states (s_i^0).

2. MLE uses these rates and initial states, and the encrypted datasets to calculate the encrypted t vector (ages vector):

   (a) First, MLE calculates the squared sum of rates: $SSM = \sum r_i^2$ once (this sum is unencrypted and is not dependent on the values of $\hat{S}$)

   (b) Then each $Enc_{pk}(t_j) \in Enc_{pk}(t)$ is computed: $t_j = \sum_{i \leq n} \frac{(r_i(\hat{s}_{i,j} - s_i^0))}{SSM}$

3. MLE uses the ciphertexts $Enc_{pk}(t)$ and private random values in order to obtain encryptions of new values called "*masked data*":

   (a) MLE samples a random **invertible** matrix R consisting of values in the range $[0, \delta]$

   (b) MLE uses the homomorphic property of the encryption scheme in order to compute $Enc_{pk}(Rt)$ (this computation only requires multiplying ciphertexts and plaintexts)

4. This encrypted and masked vector is then sent to the CSP, which decrypts it.

5. The output is a vector t', which is then sent back from the CSP to the MLE

6. MLE computes the output from t'. This is done by computing $R^{-1}t'$.

7. This output represents the predicted ages at the end of the current iteration. The ages from the current iteration are then replaced by these predicted ages and another iteration of the site and time steps is run (if the iteration limit has not yet been reached).

Note that in our implementation, in contrast with the original CEM-UPM algorithm, the protocol does not require computing the residual sum of squares at the end of each iteration. This is because after running the CEM-UPM algorithm on unencrypted datasets, we've concluded that no more than a few iterations are necessary in order to reach the final model (the predicted epigenetic ages). If we were to compute the RSS, then the minimal accepted improvement, δ_{CEM}, would need to be set extremely low in order for the RSS improvement to be tolerated after several iterations (*see Table 1*). Therefore, and due to our interest in reducing the overall runtime and the

computation complexity over the LHE scheme, we deemed this step to be relatively unnecessary in the protocol and instead decided to define a fixed number of iterations ($\ell = 4$) for the protocol to run.

RSS improvement = $RSS0 - RSS1$

	2nd Iteration	3rd Iteration	4th Iteration	5th Iteration	10th Iteration
5 Sites	0.0790	0.0019	$4.362e^{-8}$	$5.921e^{-11}$	$-4.440e^{-15}$
10 Sites	0.1915	0.0071	$3.437e^{-7}$	$4.694e^{-10}$	$5.329e^{-15}$
20 Sites	0.2519	0.0086	$3.872e^{-7}$	$5.328e^{-10}$	$1.598e^{-14}$
50 Sites	0.5092	0.0275	$4.279e^{-6}$	$1.257e^{-8}$	$2.309e^{-13}$
900 Sites	8.1140	0.2978	$1.850e^{-5}$	$3.214e^{-8}$	$-1.818e^{-12}$

Table 1: RSS improvement throughout several iterations when running CEM-UPM on a dataset consisting of 675 individuals

5 Empirical Evaluation

5.1 Accuracy

In order to validate the accuracy of our privacy-preserving solution for the Epigenetic Pacemaker, we tested our protocol on encrypted datasets containing various amounts of methylation sites and participants. Our results were compared to running the CEM-UPM algorithm on the same unencrypted datasets. For example, when running both the CEM-UPM algorithm and our privacy-preserving protocol on the same dataset consisting of 5 methylation sites and a total of 675 individuals combined from 5 different data-owners, the mean error between the results was a mere 0.00551 (see *Figure 2*)

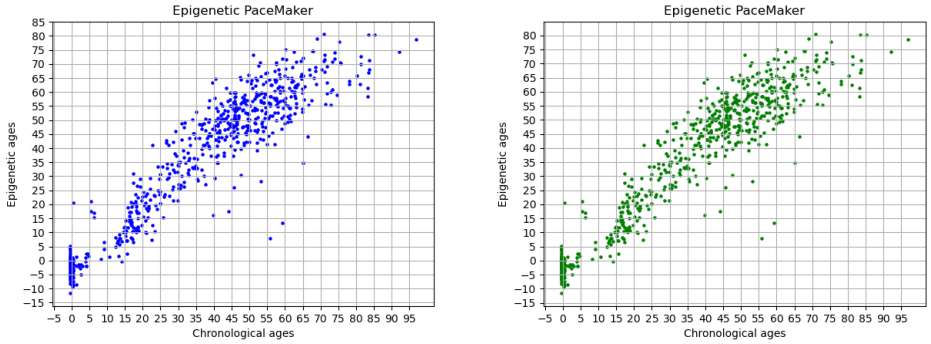

Figure 2: Result of running our privacy-preserving EPM protocol compared to those when running the CEM-UPM algorithm on the same unencryted dataset

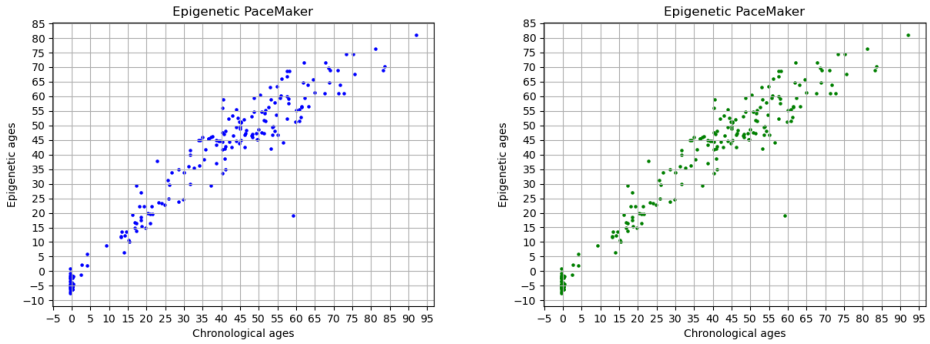

Figure 3: Results of encrypted vs. unencrypted computation of the epigenetic ages of 200 individuals on a dataset consisting of 20 methylation sites (mean error = 0.04614)

Running the complete CEM-UPM algorithm on the entire unencrypted dataset we've acquired (1000 sites, 675 individuals) yields the following result:

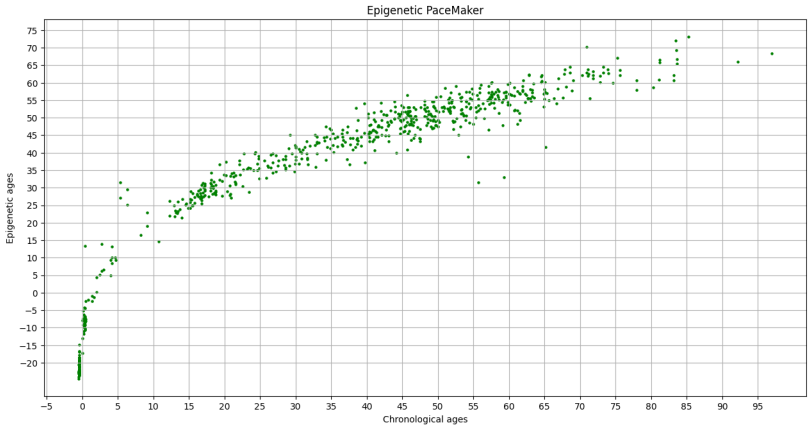

Figure 4: CEM-UPM algorithm on 1000 methylation sites, 675 individuals

5.2 Runtime

5.2.1 Privacy-Preserving Protocol vs. Unencrypted Algorithm

To examine the runtime of the privacy-preserving protocol we developed, we tested its performance on datasets of different sizes and compared it to that of the algorithm when computed on the same datasets in the clear (plaintexts). The runtime of these several dataset sizes for testing the protocol are depicted in *Table 2*.

Runtime (in seconds)

Individuals	25	50	100	200	500
With Encryption	18.5402	40.4321	85.8234	176.9378	655.0960
Unencrypted	0.0156	0.0156	0.0312	0.0624	0.1406

Table 2: Approximate runtime of our privacy-preserving protocol in comparison with running the CEM-UPM algorithm on the same unencrypted dataset consisting of 10 methylation sites

It is evident that running the protocol on encrypted datasets increases runtime - the speed of the privacy-preserving system is up to 2,000 times slower than that of the algorithm when it was run on the same unencrypted dataset. This is of course, owing to to the masking/unmasking process and the decryption overhead in each iteration of the protocol. Therefore, this excess runtime remains fairly understandable and is relatively reasonable when considering the amount of ciphertexts used in the protocol and the computation overhead when using a LHE scheme.

5.2.2 Naive Approach vs. Closed-form

As seen in *Table 3*, when running the closed form solution on unencrypted datasets of different sizes instead of the naive approach, we see a significant decrease in runtimes (this is also detailed in [9])

Runtime (in seconds)					
Individuals	25	50	100	200	400
Naive Approach	31.258	62.563	126.483	257.756	936.563
Closed Form	0.1405	0.3592	0.6320	1.3127	1.8969

Table 3: Runtime of the CEM-UPM algorithm on a dataset consisting of 500 methylation sites

6 Conclusions & Discussion

Utilizing a linearly homomorphic encryption scheme, we've developed a privacy-preserving protocol for the Epigenetic Pacemaker which provides an accurate, encrypted solution to the privacy concerns raised by running the CEM-UPM algorithm on plaintext datasets. We've shown that our solution provides accurate results with minimal errors and completes its runtime within a relatively reasonable time frame in comparison to running the algorithm on unencrypted data.

As a future research direction, a more secure approach could be explored such as concealing the age values from the Machine Learning engine at the end of each time step. The MLE is not exposed to the unencrypted ages in the datasets and therefore these predicted epigenetic ages are relatively meaningless, however revealing them at the end of each iteration prior to the end of the protocol could still be considered a breach of privacy. Furthermore, the MLE is also exposed to the unencrypted rates and initial states at the end of the site step in each iteration, instead of being exposed to them only when the protocol has concluded. Re-encrypting their values after each site step would require a more complex encryption scheme (because computing the values of the t vector for the time step would require multiplying two ciphertexts, which is not possible over a LHE scheme) or involving third-party interaction with the CSP and MLE.

Another option for future research would be to implement the closed form solution to the MC model in our privacy-preserving protocol (in the *site step* phase). Being able to implement this solution instead of the naive approach that we've used due to the limitations of the underlying encryption scheme would substantially decrease the protocol's runtime and allow for it to be run on encrypted datasets consisting of greater scales of methylation sites and individuals.

References

[1] S. Snir, B.M. vonHoldt, M. Pellegrini. A Statistical Framework to Identify Deviation from Time Linearity in Epigenetic Aging. PLOS Computational Biology. 12(11): e1005183. 2016.

[2] C. Farrell, S. Snir and M. Pellegrini. The Epigenetic Pacemaker - modeling epigenetic states under an evolutionary framework. Bioinformatics . 2020.

[3] Giacomelli I, Jha S, Joye M, Page CD, Yoon K. Privacy-preserving ridge regression with only linearly-homomorphic encryption. In International Conference on Applied Cryptography and Network Security 2018 Jul 2 (pp. 243-261). Springer, Cham. slides paper GJJPY18privateml.pdf (marcjoye.github.io)

[4] Adi Akavia, Hayim Shaul, Mor Weiss, Zohar Yakhini, Linear-Regression on Packed Encrypted Data in the Two-Server Model. In Proceedings of the 7th ACM Workshop on Encrypted Computing & Applied Homomorphic Cryptography, pp. 21-32.

[5] Barbosa, M., Catalano, D., Fiore, D.: Labeled homomorphic encryption: Scalable and privacy-preserving processing of outsourced data. In: Computer Security ESORICS 2017. LNCS, vol. 10492, pp. 146-166. Springer (2017)

[6] C. Farell: EPM Tutorial - Epigenetic PaceMaker [epigeneticpacemaker.readthedocs.io/]

[7] Fouque, P.A., Stern, J., Wackers, J.G.: Cryptocomputing with rationals. In: Financial Cryptography. LNCS, vol. 2357, pp. 136{146. Springer (2002)

[8] Wang, P.S., Guy, M.J.T., Davenport, J.H.: P-adic reconstruction of rational numbers. ACM SIGSAM Bulletin 16(2), 2{3 (1982)

[9] S. Snir: Epigenetic pacemaker: closed form algebraic solutions. BMC Genomics 2020, 21(Suppl 2):257

[10] National Library of Medicine: What is DNA? [https://medlineplus.gov/genetics/understanding/basics/dna/]

[11] K. Janitz, M. Janitz: Handbook of Epigenetics. Chapter 12- Assessing Epigenetic Information. 2011, pp. 173-181

[12] YourGenome: What is the 'Central Dogma'? [https://www.yourgenome.org/facts/what-is-the-central-dogma/]