Secure Genotype Imputation Using Homomorphic Encryption

Junwei Zhou[a,*], Botian Lei[a], Huile Lang[a], Emmanouil Panaousis[b], Kaitai Liang[c], Jianwen Xiang[a]

[a]*School of Computer Science and Technology, Wuhan University of Technology, China*
[b]*Department of Computing & Information Systems, University of Greenwich, London*
[c]*Cybersecurity Group, EMCS, Delft University of Technology, Netherlands*

Abstract

Genotype imputation estimates missing genotypes from the haplotype or genotype reference panel in individual genetic sequences, which boosts the potential of genome-wide association and is essential in genetic data analysis. However, the genetic sequences involve people's privacy, confirming an individual's identification and even disease information. This work proposes a secure genotype imputation model, which uses a linear regression model and the homomorphic encryption scheme over ciphertext to impute missing genotypes. The inference model is trained with float plaintext parameters, which are round into integers to avoid high complexity homomorphic evaluation on float number operations without bootstrapping operations. Even though the rounding parameters in the inference model are not the same as those in the trained model, We find that it will no effect on the outcome of the homomorphic prediction. Thus, a high-efficiency genotype imputation inference model over the ciphertext is obtained while keeping the high-security level. The simulation results indicate that the accuracy of the secure inference model is almost the same as the original model trained on float parameters. The secure inference model's accuracy is 98.6% for a single genotype.

Keywords: Privacy-preserving, Homomorphic encryption, Genotype imputation, Privacy computing, Genetic security.

[*]Corresponding author

Email addresses: junweizhou@msn.com (Junwei Zhou), botianlei@qq.com (Botian Lei), hllang@qq.com (Huile Lang)

1. Introduction

DNA sequencing is an indispensable part of medical diagnosis and treatment, biotechnology, and genetic data analysis [1]. It can help researchers understand and study the relationship between diseases [2], distant ancestors [3], and genes. For example, the genome-wide association study (GWAS) [4] aims to study the relationship between human diseases and complex traits. However, due to individual genetic variation or genetic testing technology issues, some genotypes in genetic sequencing are missing or low-quality. The missing genotypes can affect the genetic information integrity, thereby affecting downstream analysis such as GWAS [5]. Genotype imputation uses the association between genetic variations to predict those missing or low-quality genotypes [6]. It is a foundational step of gene analysis, with a wide range of practical applications. It is now widely used in GWAS to find new risk alleles, obtain high-resolution views in fine positioning to increase the possibility of identifying causal variants and integrate Meta-analysis of research on different platforms. Currently, the state-of-the-art plaintext genotype imputation methods include IMPUTE2 [7], Minimac3 [8], and Beagle [9] and fastPHASE [10]. IMPUTE2 uses sophisticated recombination maps and dense genotype reference panels to impute missing genotypes in the research dataset. Minimac3 divides the genome into contiguous blocks and iterates only the unique haplotypes in each genome block. It uses a reversible mapping function to reconstruct the state space used by IMPUTE2 accurately. Beagle uses the Li and Stephens haplotype frequency models with highly reduced model state space to interpolate the phased haplotypes. FastPHASE is a flexible method that allows the "blocky" pattern of linkage disequilibrium (LD), and the LD gradually decreases with distance. These methods are applied based on plaintext gene datasets to impute missing genotypes and cannot secure the gene data. The genetic data is sensitive, where a sequence larger than 75 single-nucleotide polymorphisms (SNPs) array can confirm an individual's identification [11]. A genetic sequence can reveal human ancestry, relatives, and disease type, which involves privacy concerns.

Nowadays, the pressure brought by the increasing genetic data and the huge amount of imputation computations has made people turn their attention to convenient cloud service providers [12] for imputation. However, the genetic data is stored in plaintext in the cloud, and anyone who has access to the cloud platform may obtain these plaintext data. Even the semi-trusted cloud platforms may steal the users' genetic data motivated by benefits. The privacy concerns prohibit people from trusting the third-party platform to process their genetic data [13, 14].

The homomorphic encryption (HE) [15, 16] allows computing over encrypted data directly without revealing data. It provides mathematically provable security guarantees for protecting genotype data while performing imputations in an untrusted cloud platform. We note that other cryptologies like multiparty computation (MPC) [17, 18] require interaction between multiple parties that hold the data, and the cloud platform performs imputation operations. Although the functional performance of MPC-based methods is impressive, they may cause problems such as network latency and high bandwidth usage. Thus, we rely on the HE scheme to secure the gene data and perform the imputation evaluation over encrypted genetic data. However, the HE scheme limits the computational circuit depth or requires time-consuming bootstrapping operations to refresh the ciphertext, making the HE-based applications computationally inefficient. The main challenge of the HE-based genotype imputation is to achieve efficient and accurate genotype imputation under the premise of ensuring security.

To explore the practical feasibility of the cryptographic methods for genotype imputation, IDASH organized the genotype imputation track in iDASH2019 Genomic Privacy Challenges. The participating teams focused on the three most advanced HE cryptosystems, which are namely Brakerski/Fan-Vercauteren (BFV) [19], Cheon-KimKim-Song (CKKS) [20], and fast fully homomorphic encryption over the torus (TFHE) [21]. The highest accuracy of the participating teams was 95.5% with a 128-bit security level from the HE standardization workshop paper [22]. Since the imputation accuracy of these secure genotype imputation methods is lower than those of plaintext methods. We need higher accuracy to improve the downstream analysis, such as GWAS.

In this work, we propose a fast and secure genotype imputation [23] inference model based on the TFHE [21]. The main contributions are as follows:

- Based on the iDASH Secure Genome Analysis Challenge 2019 dataset, we provided several secure genotype imputation models of a single variant, exhibiting the connection between the tag and target variants. Simulation results indicate that the accuracy for a single genotype of a variant of 1000 individuals reached 98.6%.

- We used an LWE-based linear regression model without bootstrapping or key-switching operation, improving genotype imputation speed. Our experimental results show that the imputation time for 1000 individuals' single genotype is approximately 0.269 seconds.

- We found that the rounding error in the imputation vanished the noise brought by HE encryption. The obtained secure genotype imputation model has a similar performance to the original plaintext model.

We trained the model with genetic data in plaintext. The genotypes of tag variants used for the inference model are encrypted into ciphertext. We set the message space of ciphertext in advance by calculating the maximum multi-sum between the trained parameters and genotype value. The homomorphic evaluation is based on the security of the LWE problem. The encryption phase uses the LWE security concept and there did not use bootstrapping operation and key-switching key. The key distribution will not cause any time costs. There is no additional overhead (except for linearly increasing input size) to extend the interpolation calculation. The rounding parameters will not affect homomorphic computation on the ciphertext and the security level.

The remaining parts of the paper are as follows. Section 2 gives the related work of secure genotype imputation. Section 3 describes the model's training process on the plaintext and the implementation of secure inference models. Section 4 introduces our experiments. The conclusion and expectations are provided in the last section.

2. Related Work

In 1978, Rivest [24] first proposed the HE scheme's assumption, which allowed various calculations over encrypted data. Nowadays, HE schemes are generally classified into three types: Partially Homomorphic Encryption (PHE) [25], Somewhat Homomorphic Encryption (SWHE) [26, 27, 28], and Fully Homomorphic Encryption (FHE) [29, 21] according to calculation depth and capacity. PHE can allow homomorphic multiplications or homomorphic additions with limited calculation depth. SWHE supports homomorphic multiplications and additions with limited calculation depth. FHE supports the arbitrary depth of any calculations on the ciphertext by using bootstrapping. The bootstrapping operation [29, 30, 31] can refresh ciphertext and reduce noise in ciphertext, which allows FHE to achieve the arbitrary depths of circuits and maintain the decryption's correctness.

The current popular HE schemes include BFV [26, 19], CKKS [20], BGV [28] and TFHE. These schemes are implemented based on the ring learning with error (R-LWE) problem [32], while TFHE is based on LWE and GSW [33] problems. Both BFV and BGV allow homomorphic calculations on vectors of finite field elements, and the CKKS scheme allows approximate homomorphic calculations on real or complex numbers.

In the era of cloud computing and machine learning, HE provides a solution to protect users' outsourced data [34, 35, 36, 37]. The user uploads encrypted data to the cloud service without decryption, and the cloud service directly performs homomorphic addition and multiplication on the ciphertext. The other computing on ciphertext can be constructed using homomorphic addition and multiplication.

HE&Genotype imputation. Advances in information technology and bioinformation have made people and institutions use third-party cloud platforms to store and process data, such as online health status monitoring [38], disease diagnosis [39, 40], and genotype imputation [41, 42]. However, once users upload their data in plaintext to the third-party platform, they will lose control of their sensitive data. Anyone who can access the third-party platform can steal users' genetic data.

Kocabas [38] proposed a secure health monitoring system (real-time monitoring of heartbeat frequency) with the FHE scheme. When the monitor system obtained the user's heartbeat frequency, it encrypted the frequency data locally using HElib library [43] and uploaded encrypted data to the cloud platform for analysis. The cloud platform analyzed the encrypted data and transmitted the encrypted result to the user. Then the user decrypted it to check the result. In this process, since the secret key was in the users' hands, the cloud platform could not obtain any information about the user's data. Thus the privacy of the user's health status was guaranteed. Meehan [39] proposed a secure model using the TFHE library to diagnose whether users had breast cancer. It guaranteed the privacy and safety of users' health status and avoided disease discrimination. Kim etc. [41] proposed to combine genotype imputation and HE scheme to protect people's genetic data without data leakage. The UTMSR team presented a fast and secure linear regression model based on BFV and CKKS schemes, and the accuracy achieved 95.4%. EPFL team applied a multinomial logistic model to impute missing genotypes based on the CKKS scheme homomorphically and got 95.5% accuracy. The Chimera team presented a TFHE-based logistic regression model, and the accuracy was 95.1%. The SNU team applied a one-hidden layer neural network with the CKKS scheme, and the accuracy was 95.0%. The models only took 380 microseconds to predict the genotype of a variant of 1000 individuals. They output each type of genotype's probability to determine the imputed genotype. Compared with the plaintext models (higher than 97.1% [41]), the accuracy of the secure models can still be improved.

We propose a secure linear regression inference model with the TFHE library based on the LWE problem to secure genetic data and efficiently impute the missing SNPs. The inference model is trained with plaintext float parameters, which are round into integers to avoid high complexity HE evaluation on float number operations. We performed detailed experiments on the time and memory requirements of the HE-based imputation model and demonstrated the feasibility of large-scale secure imputation. We found comparable performance (without decrease) in imputation accuracy with total genomic data security benefit. Our experimental results provide evidence that HE-based methods can perform efficient calculations to analyze massive genetic data.

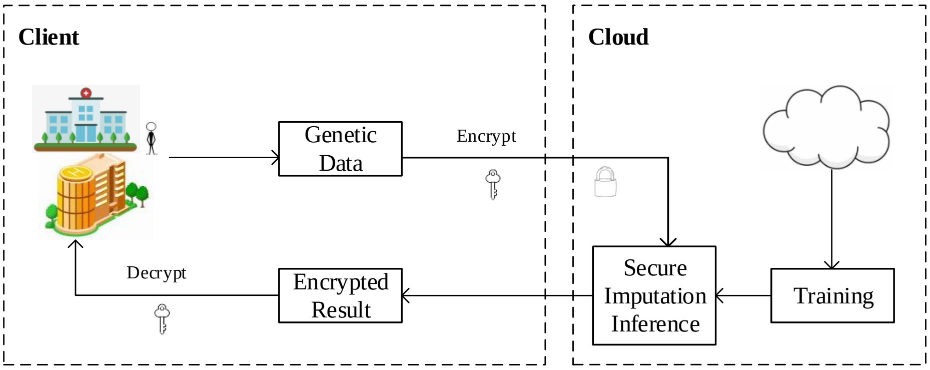

Figure 1: The overview of our privacy-preserving genotype imputation inference.

3. The Proposed Model

The secure inference model over encrypted data is shown in Figure 1. The client encrypts genetic data with a secret key and then uploads the encrypted data to the cloud service provider (CSP) for genotype imputation. CSP returns the encrypted result to the client after imputation computations. Even if an untrusted third party steals the encrypted genetic data in the entire system imputation process, he cannot obtain any information because he does not have the key. This section will introduce the models in detail, including data encryption and decryption process in client and model training and secure imputation inference model in the cloud.

3.1. Preliminaries

This subsection gives the notations, definitions of the LWE problem and the LWE encryption scheme, and homomorphic computations used in the paper.

Notation. A vector is denoted by a bold letter and $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$ denotes the inner product between two vector $\boldsymbol{a}$ and $\boldsymbol{b}$. $||\cdot||_1$ denotes the L_1 norm of a vector, $||\cdot||_2$ denotes the L_2 norm of a vector, and $||\cdot||_\infty$ denotes the infinite norm of a vector. $\mathbb{R}$ denotes the real numbers, $\mathbb{Z}$ denotes the integers, and $\mathbb{T}$ is torus $\mathbb{R}/\mathbb{Z}$. Furthermore, given a set $\mathbb{Q}$, $\boldsymbol{a} \overset{\$}{\leftarrow} \mathbb{Q}$ represents that $\boldsymbol{a}$ is chosen uniformly and randomly from $\mathbb{Q}$.

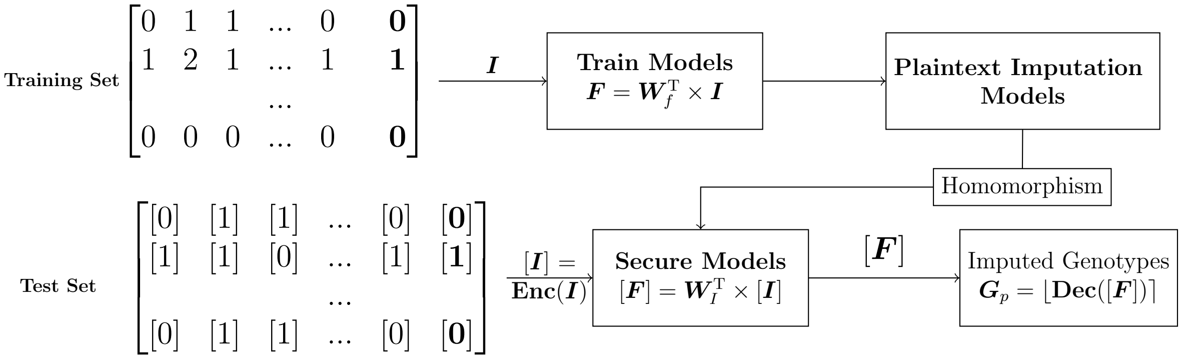

Figure 2: Training and inference on the cloud.

Learning With Errors. Regev [44] introduced the Learning With Errors (LWE) problem in 2005. Let n be a positive integer ($n \geq 1$), any vector $\boldsymbol{s} \overset{\$}{\leftarrow} \mathbb{Z}^n$, ϕ be a distribution over $\mathbb{R}$, and the noise e is sampled from distribution ϕ. For any vector $\boldsymbol{s}$, we define the LWE distribution $\mathrm{LWE}_{s,\phi}$ as $(\boldsymbol{a}, b)$, where the vector $\boldsymbol{a} \overset{\$}{\leftarrow} \mathbb{T}^n$ and $b = \langle \boldsymbol{s}, \boldsymbol{a} \rangle + e$.

Regev defined the LWE problem is: for a fixed $\boldsymbol{s} \overset{\$}{\leftarrow} \mathbb{Z}^n$, it is hard to distinguish between $\mathrm{LWE}_{s,\phi}$ and the uniform distribution over $\mathbb{T}^{n+1}$.

Regev stated that the LWE problem is as asymptotically difficult as the worst-case lattice problem.

LWE-based encryption. Define a positive integer B related to the message space. Let $m \in [-B, B]$ be an integer message. The torus is split into $2B + 1$ slices, and each slice denotes one possible integer value.

Let n denotes the security parameter, $\boldsymbol{s} \overset{\$}{\leftarrow} \mathbb{Z}^n$. ϕ is a Gaussian distribution.

Enc(m): Return $(\boldsymbol{a}, b)$, with $\boldsymbol{a} \overset{\$}{\leftarrow} \mathbb{T}^n$, and $b = \langle \boldsymbol{s}, \boldsymbol{a} \rangle + \frac{m}{2B+1} + e$, where $e \leftarrow \phi$.

Dec$(\boldsymbol{s}, (\boldsymbol{a}, b))$: Return $m = \lfloor (b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle) \times (2B + 1) \rceil$.

Homomorphic addition. Suppose two messages m_1, $m_2 \in [-B, B]$ with a randomly generated secret key $\boldsymbol{s}_1$, $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ are randomly chosen from $\mathbb{T}^n$. $\boldsymbol{c}_1 = \mathbf{Enc}(m_1)$, $b_1 = \langle \boldsymbol{s_1}, \boldsymbol{a_1} \rangle + \frac{m_1}{2B+1} + e_1$. $\boldsymbol{c}_2 = \mathbf{Enc}(m_2)$, $b_2 = \langle \boldsymbol{s}_1, \boldsymbol{a_2} \rangle + \frac{m_2}{2B+1} + e_2$, and we can get $\boldsymbol{c}_1 + \boldsymbol{c}_2 = (\boldsymbol{a}_1 + \boldsymbol{a}_2, b_1 + b_2)$.

$$\begin{aligned} \mathbf{Dec}(\boldsymbol{s}_1, \boldsymbol{c}_1 + \boldsymbol{c}_2) &= \lfloor (b_1 + b_2 - \langle \boldsymbol{s}_1, \boldsymbol{a}_1 + \boldsymbol{a_2} \rangle) \times (2B + 1) \rceil \\ &= \lfloor (m_1 + m_2) + (e_1 + e_2) \times (2B + 1) \rceil \end{aligned}$$

Suppose $m_1 + m_2 \in [-B, B]$, and the noise $(e_1 + e_2) \times (2B + 1)$ is within the controllable range, and the ciphertext is expanded in the ciphertext space after the homomorphic addition, then the decryption of homomorphic addition result will be correct with overwhelming probability: $\mathbf{Dec}(\boldsymbol{s}_1, (\boldsymbol{c}_1 + \boldsymbol{c}_2)) = m_1 + m_2$. Therefore, addition over ciphertext is homomorphic.

Homomorphic multiplication. Homomorphic multiplication supports the calculation between ciphertext and an integer plaintext. Let k be an integer constant in plaintext, the message $m \in [-B, B]$, $\boldsymbol{a}$ is randomly chosen from $\mathbb{T}^n$, $\boldsymbol{c} = \mathbf{Enc}(m), b = \langle \boldsymbol{s}, \boldsymbol{a} \rangle + \frac{m}{2B+1} + e$. The multiplication over ciphertext and plaintext and decryption processes are as follows.

$$k \times \boldsymbol{c} = k \times (\boldsymbol{a}, b) = (k \times \boldsymbol{a}, k \times b)$$

$$\begin{aligned} \mathbf{Dec}(\boldsymbol{s}, k \times \boldsymbol{c}) &= \lfloor (k \times b - \langle \boldsymbol{s}, k \times \boldsymbol{a} \rangle) \times (2B+1) \rceil \\ &= \lfloor k \times m + k \times e \times (2B+1) \rceil \end{aligned}$$

If $k \times m \in [-B, B]$ and the noise $k \times e \times (2B+1)$ is within the controllable range, and the expanded ciphertext size is within the ciphertext space after the homomorphic multiplications, the decryption of multiplication will succeed with overwhelming probability: $\mathbf{Dec}(\boldsymbol{s}, k \times \boldsymbol{c}) = k \times m$. Therefore, we believe that the multiplication over ciphertext and plaintext is homomorphic.

Here is a toy example where insecure parameters are used for straightforward explanation. Let's choose $n = 4, B = 12, m = 5$ and $a = (0.1, 0, 0.1, 0), s = (0, 1, 0, 1), e = 0.001$. To encrypt, we need to compute $b = \langle \boldsymbol{s}, \boldsymbol{a} \rangle + \frac{m}{2B+1} + e = 0.3001$, thus the ciphertext is $(a, b) = (0.1, 0, 0, ., 0, 0.3001)$. We can use s to decrypt and then $m = \lfloor (b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle) \times (2B+1) \rceil = \lfloor 5.0025 \rceil = 5$ can be obtained.

3.2. Client

Clients are limited by high computation capacity and genotype imputation technology; thus, they are willing to upload genetic data to a third-party platform for convenient imputation. Before clients send their data to CSP, they first encrypt the genetic data with a secret key, and CSP will impute missing genotypes over these encrypted data.

We use the subsets of genetic sequences [45] to reduce the cost of large-scale genotype imputation and enhance the power of genetic data analysis. The genotypes in the subset are called tag variants, and we study the relationship between these tag variants to impute missing or low-quality variant genotypes (called target variants).

Suppose user's genetic data in plaintext is $\boldsymbol{I}$ $(n \times 1)$, each genotype value $\boldsymbol{I}_i \in \boldsymbol{I}$ is a discrete value which is defined as 0, 1, or 2. 0 denotes homozygous reference genotype, 1 denotes heterozygous genotype, and 2 indicates homozygous alternate genotype. The genetic data $\boldsymbol{I}$ is encrypted by $\mathbf{Enc}(\cdot)$ operation into encrypted data $[\boldsymbol{I}]$, and then client inputs $[\boldsymbol{I}]$ into the secure imputation inference model.

$$\boldsymbol{I} : (1, 2, 0, 0..., 2) \xrightarrow{\mathbf{Enc}(\cdot)} [\boldsymbol{I}]([1], [2], [0], [0], ..., [2]) \tag{1}$$

3.3. Cloud Service Provider

The cloud service provider (CSP) uses the linear correlation between genetic data to train and get the model's parameters by the public genomic dataset such as 1000 genomic project [46]. CSP takes the client's encrypted genetic data as the input of the secure inference model and returns the encrypted imputed result to the client.

CSP uses the forward propagation and backpropagation [47] to train the model on a public genetic dataset. The forward propagation calculates each neuron's output, and the backpropagation updates and optimizes the weight parameters. The training process usually involves thousands of iterations. We use stochastic gradient descent, and the parameters will be optimized after a new sample is trained. CSP trains the model on the public genetic dataset and imputes missing SNPs over the encrypted data. Figure 2 illustrated the process of training and inference.

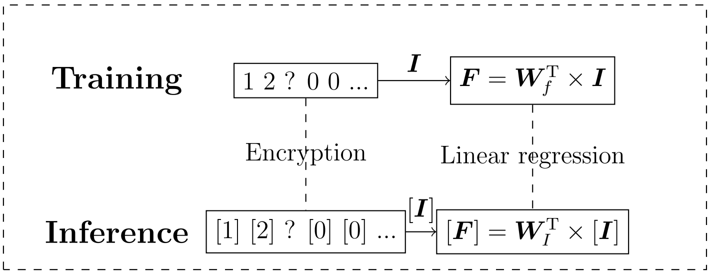
Training
1 2 ? 0 0 ...
$\boldsymbol{I}$
$\boldsymbol{F} = \boldsymbol{W}_f^{\mathrm{T}} \times \boldsymbol{I}$
Encryption
Linear regression
Inference
[1] [2] ? [0] [0] ...
$[\boldsymbol{I}]$
$[\boldsymbol{F}] = \boldsymbol{W}_I^{\mathrm{T}} \times [\boldsymbol{I}]$

Figure 3: Linear regression model. The CSP trains the model on the plaintext, rounds the model's weight, then constructs the ciphertext model to accept input ciphertexts from the client.

3.3.1. Model Training

As shown in the Figure 3, the linear regression model's input is $\boldsymbol{I}$ $(n \times 1)$, and $\boldsymbol{I}_i \in \{0, 1, 2\}$, while its output is a float vector $\boldsymbol{F}$ $(m \times 1)$. Before the model starts training, weight parameters $\boldsymbol{W}_f$ $(n \times m)$ are randomly initialized as float numbers between $(0, 1)$. In the forward propagation, $\boldsymbol{F} = \boldsymbol{W}_f^{\mathrm{T}} \times \boldsymbol{I}$.

During the backpropagation process, we use the mean-square error function as the cost function J, where $\boldsymbol{F}'$ denotes the target genotype, $\boldsymbol{F}$ is the model's output.

$$J = \frac{1}{2}(\boldsymbol{F} - \boldsymbol{F}')^2 \tag{2}$$

The stochastic gradient descent algorithm is performed according to Eq. 3, where α is the learning rate and $\boldsymbol{W}_f$ represents the weight parameters.

$$\boldsymbol{W}_f = \boldsymbol{W}_f - \alpha \frac{\partial J}{\partial \boldsymbol{W}_f} \tag{3}$$

The parameters of the model are iterative optimized until J convergence to a minimal value or the iteration number reaches the limitation. After training the model, CSP constructs a homomorphic linear regression inference model based on the trained parameters.

3.3.2. Secure Inference Model

As shown in Figure 2, the inference model takes the encrypted genetic data $[\boldsymbol{I}]$ $(n \times 1)$ as the input. Since the trained parameters on plaintext are float and homomorphic multiplication requires the plaintext to be an integer, we convert $\boldsymbol{W}_f$ into integer $\boldsymbol{W}_I$. We keep two numbers after the decimal point for the $\boldsymbol{W}_f$ and scale them by 100 times. We use integers to approximate the floating-point weight, without affecting the encrypted genetic data and homomorphic computation on the ciphertext, as seen from the homomorphic multiplication formula in section 3. Thus, the conversion will not reduce the complexity of the LWE-based homomorphic computation and security level.

$$\boldsymbol{W}_I = \lfloor \boldsymbol{W}_f \times 100 \rceil \tag{4}$$

$$[\boldsymbol{F}] = \boldsymbol{W}_I^{\mathrm{T}} \times [\boldsymbol{I}] \tag{5}$$

Where $i \in [1, n]$ is a integer, and $[\boldsymbol{F}]$ is encrypted imputation result. The inference of missing SNPs is shown as Eq. 5. After CSP inferences the missing SNPs, it returns back the encrypted imputation result $[\boldsymbol{F}]$ to the client. The client decrypts $[\boldsymbol{F}]$ with the secret key, then decode by scaling it by 1/100. The decoded results are the predicted genotypes of missing SNPs.

To correctly evaluate the model's multi-sum, we need to include all possible values of $\boldsymbol{W}_I^{\mathrm{T}} \times \boldsymbol{I}$ in the message space B [48], $\boldsymbol{W}_I^{\mathrm{T}} \times \boldsymbol{I} \in [-B, B]$. Otherwise, the decryption of the multi-sum will fail. CSP first chooses models to train and obtains optimized model parameters $\boldsymbol{W}_I$ after training. The maximum of I_i is 2, and CSP can get the possible maximum multi-sum of the model: $||\boldsymbol{W}_I||_1 \times 2$. As long as B satisfies the following formula, the multi-sum of the model will be in the range of $[-B, B]$, and homomorphic decryption will succeed correctly with an overwhelming possibility.

$$B \geq ||\boldsymbol{W}_I||_1 \times 2 \tag{6}$$

3.4. Analysis of Noise

With homomorphic addition and multiplication calculation, the ciphertext's noise will expand. The expanded noise will result in decryption failure if calculation times are not limited. Thus, we should limit calculation times by reducing model's input units and the size of $||\boldsymbol{W}_I||_2$. The standard deviation is used to evaluate whether ciphertext's noise is out of bounds, which is σ^2 in a fresh ciphertext. With every multiplication, the standard deviation gets larger by the square of the multiplier. When a ciphertext is multiplied by an integer p, the noise's standard deviation of obtained ciphertext will be expanded by p^2 times. To decrypt correctly (noise will not overflow), the following inequation needs to be satisfied:

$$||\boldsymbol{W}_I||_2^2 \times \sigma^2 < \frac{1}{4B} \tag{7}$$

3.5. Analysis of Parameter Rounding

Theorem 1. *Let $\boldsymbol{W}_\xi = \boldsymbol{W}_f - \boldsymbol{W}_I/100$ be difference of between the float and integer weight parameters, $\boldsymbol{F}_f = \boldsymbol{W}_f^{\mathrm{T}} \times \boldsymbol{I}$ is the original model's output, $\boldsymbol{F}_I = \boldsymbol{W}_I^{\mathrm{T}} \times \boldsymbol{I}$ is the output of model with integer weights, **Dec([F])** denotes the decrypted result of secure inference model's output. Since the prediction result on the plaintext is close to an integer, the noise generated by rounding the weight parameter is very small and will not affect the prediction accuracy.*

Proof. $\boldsymbol{W}_I$ takes the accuracy of $\boldsymbol{W}_f$'s two decimal places, the values in $\boldsymbol{W}_\xi$: $\boldsymbol{W}_\xi^i < 5 \times 10^{-3}$.

$$\begin{aligned} &\boldsymbol{W}_f^{\mathrm{T}} \times \boldsymbol{I} - (\boldsymbol{W}_I^{\mathrm{T}} \times \boldsymbol{I})/100 \\ &= (\boldsymbol{W}_I^{\mathrm{T}}/100 + \boldsymbol{W}_\xi^{\mathrm{T}}) \times \boldsymbol{I} - (\boldsymbol{W}_I^{\mathrm{T}} \times \boldsymbol{I})/100 \\ &= \boldsymbol{W}_\xi^{\mathrm{T}} \times \boldsymbol{I} \end{aligned} \tag{8}$$

$$\begin{aligned} &\boldsymbol{W}_f^{\mathrm{T}} \times \boldsymbol{I} - \mathbf{Dec}([\boldsymbol{F}])/100 \\ &= (\boldsymbol{W}_I^{\mathrm{T}}/100 + \boldsymbol{W}_\xi^{\mathrm{T}}) \times \boldsymbol{I} - \\ &\left\lfloor \boldsymbol{W}_I^{\mathrm{T}} \times \boldsymbol{I} + \boldsymbol{W}_I^{\mathrm{T}} \times \boldsymbol{e} \times (2B+1) \right\rceil /100 \\ &= \boldsymbol{W}_\xi^{\mathrm{T}} \times \boldsymbol{I} \end{aligned} \tag{9}$$

Eq. 8 represents the output difference between models on the plaintext. From the Eq. 9, We can find that the noise in the encryption is taken from a Gaussian sample centered on the input message, with the standard deviation sd. It will not affect decryption. At the same time, since the output value on the plaintext is close to an integer and the $\boldsymbol{W}_{\xi} < 5 \times 10^{-3}$ is small, which will not affect the imputation accuracy. Thus, even though the secure model's decrypted result contains noise, it does not affect the imputation result. □

4. Experiments

Our models are implemented in C++. The models are run on a PC with i7-6700 CPU and 8G RAM. This section describes the experimental dataset, parameter settings, imputation accuracy, resource usage, and time consumption. The code address: `https://github.com/tfhe-genotype-imputation/HE_genotype`.

4.1. Dataset

The simulation datasets include two datasets that come from the iDASH Secure Genome Analysis Challenge 2019, containing 2504 individuals' genetic data.

As shown in Table 1, in the "sorted_tag_SNPs_1k_genotyp- es" dataset, it includes each individual's 9764 SNPs, and the distance between two nearby genotypes is 1k. "sorted_tag_SNPs_1k_genotypes" dataset includes each individual's 1045 SNPs, and the distance is 10k. The 500 target SNPs are the missing SNPs to be imputed. In experiments, 1500 individuals are used as the training set and 1004 as the test set (3:2). The dataset can be found in the following URL[1].

Table 1: Dataset

Distance	**Dataset**	**Tag SNPs**	**Target SNPs**
1k	sorted_tag_SNPs_1k_genotypes	9764	500
10k	sorted_tag_SNPs_10k_genotypes	1045	

4.2. Parameters

The homomorphic evaluation is based on the security of the LWE problem. The setting followed the notations of [21]. The encryption phase uses LWE security notions with no bootstrapping operations and no key-switching

[1] http://www.humangenomeprivacy.org/2019/competition-tasks.html

key. We estimated the security level from the attack models by the LWE estimator from [49] that computes the computational costs of state-of-art (R)LWE attack algorithms. We employed the LWE estimator to estimate hardness for the standard deviation $sd(2^{-25})$ and dimension $n(1024)$ and get an estimated 130-bits of security. The parameters related to the LWE estimator are the following:

- Ciphertext dimension: $n = 1024$;
- Noise standard deviation: $sd = pow(2., -25)$;
- Noise rate: $alpha = sqrt(2 * pi) * stdev$;
- Compatibility: $q = pow(2., 32)$;
- The attacker can use any number of samples: $m = oo$

Inspired by the methods in [41], we conducted experiments on models of increasing input sizes. In the dataset of the different distances between variants, the models include 10, 30, and 70 tag SNPs for a single target SNP, and we represent the models as "$10 \rightarrow 1$", "$30 \rightarrow 1$", and "$70 \rightarrow 1$" models.

Finally, we calculated the message space: $B = ||\boldsymbol{W}_I^{\mathrm{T}}||_1 \times 2$, $\boldsymbol{W}_I$ denotes weight parameters of each model. In the experiment, we set $B = 700$, which is slightly smaller than the calculated value, and we found that it did not affect the results of the homomorphic evaluation.

4.3. Imputation Accuracy

As shown in Table 2, we give the accuracy of models based on the HE scheme. Our secure linear regression reference model is similar to the UTMSR's model in [41], logistic regression models in CHIMERA and EPFL. But we select the most appropriate nearby genotypes for each missing gene. Experimental results show that our "30→1" model is about 3% higher than their model on accuracy. Unlike one-hidden layer neural network in SNU, we applied a more straightforward structure, and the results show our model is 1%-3% higher than theirs.

We construct three models on two datasets to illustrate our models' performance under different input sizes and security levels. We refer to the original models with float weights as float plaintext models, the model with integer weights as integer plaintext models, and the models on ciphertext as secure inference models for convenience.

In Table 3 and Table 4, the second value refers to the accuracy of float plaintext models, and the third value refers to the integer plaintext models' accuracy, the last value refers to the accuracy gap between float plaintext models and secure inference models. The tables show that parameter rounding operation does affect the imputation accuracy. On the 1k dataset, the accuracy of the models with integer parameters is about 0.1% lower than

Table 2: Imputation accuracy of homomorphic models

Models	**Size**	**Accuracy**
UTMSR[41]	32→1	95.40%
EPFL[41]	-	95.50%
CHIMERA[41]	45→1	95.10%
SNU[41]	24→1	95.00%
Ours	10→1	**96.66**%
	30→1	**98.55**%
	70→1	**98.60**%

float plaintext models. On the 10k dataset, the accuracy difference is about 0.2%. As seen from Table 3 and Table 4, we can find that the accuracy of the secure inference model is close to that of the integer plaintext model. There is a small improvement because some results of the plaintext model are close to the middle of the label, which becomes correct after adding the noise of the rounding weights.

Table 3: Test accuracy on 1k dataset

Input→Output	Float	Integer	Security (bits)	Ciphertext	Gap
10→1	96.74%	96.65%	80	96.65%	0.09%
			130	96.65%	0.09%
30→1	98.57%	98.55%	80	98.55%	0.02%
			130	98.55%	0.02%
70→1	98.65%	98.60%	80	98.60%	0.05%
			130	98.60%	0.05%

Table 4: Test accuracy on 10k dataset

Input→Output	Float	Integer	Security (bits)	Ciphertext	Gap
10→1	87.17%	86.87%	80	86.87%	0.30%
			130	86.87%	0.30%
30→1	88.48%	88.29%	80	88.29%	0.19%
			130	88.29%	0.19%
70→1	88.40%	88.52%	80	88.52%	0.12%
			130	88.52%	0.12%

On the 1k dataset, the secure model's imputation accuracy for a single genotype achieved 98.6%. The ”70→1” homomorphic model maintained the highest accuracy and is slightly 0.04% higher than the "30→1" homomorphic model. Furthermore, on the 10k dataset, the "70→1" homomorphic model is slightly higher than the "30→1" homomorphic model. The experimental results show that the number of nearby genotypes influences imputation accuracy. The more nearby genotypes (larger input size), the higher the accuracy.

Under the same model size, the imputation accuracy of the model with a nearby genotype distance of 1k is almost 10% higher than that of the model with 10k. The experimental results show that the accuracy of the inference model is the same under the security level of 130 bits and 80 bits.

4.4. Resource Usage

With 80 bits or 130 bits security level, each genotype (0, 1, or 2) in the clear takes 2 bits, and each LWE ciphertext takes 8 bytes (64 bits). Therefore, the storage of ciphertext is 32 times that of plaintext.

As shown in Table 5, the memory usage of the proposed scheme is larger than that in the [41]. Our scheme required less than 0.39 gigabytes.

Table 5: Memory Usage

Models	**Size**	**Memory** (gigabytes)
UTMSR	32→1	0.03
CHIMERA	45→1	0.02
SNU	24→1	0.13
EPFL	-	0.06
Ours	10→1	**0.26**
	30→1	**0.30**
	70→1	**0.39**

4.5. Time Consumption

We divided homomorphic evaluation into three processes: encryption, homomorphic calculation, and decryption. Table 6 refers to each process's time consumption per 1000 individuals with a different security level. We can see from the table that the encryption operation consumes the most time, accounting for more than 95% of the total time. The whole homomorphic calculation did not use bootstrapping operations and key-switching keys, so the homomorphic calculation only spent about 5% of the total time. The decryption step took the least time. The experimental results also show that each step's time consumption has a linear relationship with the model's input size.

In addition, Table 6 shows that the secure inference model with 80 bits security level consumes nearly 25% less time than the model with 130 bits security level.

Table 6: Time consumption of each process.

Security (bits)	**Input** → **Output**	**Enc**(s)	**Calculation**(s)	**Dec**(s)
80	10 → 1	0.1941	0.00237	0.00069
130		0.2438	0.00298	0.00075
80	30 → 1	0.5788	0.00619	0.00069
130		0.7323	0.0078	0.00076
80	70 → 1	1.3583	0.0139	0.00072
130		1.6967	0.0175	0.00076

Table 7 describes the total time consumption of homomorphic models with the security level of 130 bits. The test dataset has 1004 individuals. In the "10 → 1" model, the homomorphic evaluation time for each variant of 1004 individuals is approximately 0.269 seconds, the "30 → 1" model is about 0.78 seconds per variant per 1004 individuals, and the "70 → 1" model is approximately 1.71 seconds. The experimental results show that the secure linear model with 30 tag SNPs as the model's input for a single genotype shows the most balanced performance in terms of timing and imputation accuracy.

Table 7: Total time consumption.

Input→ **Output**	**Total Time** (s)	**Time** (ms / variant)
10 → 1	0.269	0.268
30 → 1	0.78	0.777
70 → 1	1.71	1.708

Since there are 1004 individuals in the test dataset with the same calculation in homomorphic linear regression, the genes with the same SNP's identifier of different individuals can be packaged into one ciphertext, which can calculate repeated operations in parallel. As a result, the time consumption will be significantly reduced. For the prediction of the same SNP of 1004 individuals in the "30-1" model, the input changed from 1004×30 LWE ciphertexts to 30 TLWE ciphertexts, and the output with 1004 LWE ciphertexts are packaged in one TLWE ciphertext. Table VIII shows that packaging multiple inputs into a single ciphertext can reduce the time consumption of our model nearly 300 times, and the accuracy remains the same.

Table 8: Average time consumption of each process for each SNP after data packaging.

Security (bits)	**Input** → **Output**	**Enc**(ms)	**Calculation**(ms)	**Dec**(ms)
80	10 → 1	0.507	0.01	0.011
130		0.513	0.011	0.012
80	30 → 1	1.578	0.031	0.011
130		1.6	0.04	0.012
80	70 → 1	3.43	0.075	0.012
130		3.47	0.077	0.012

5. Conclusion

In this work, we propose a secure and fast linear regression inference model to impute the missing genotypes. Homomorphic encryption is time-consuming and only allows simple addition or binary gates on the ciphertext. Thus we design the secure inference model carefully to maintain high imputation accuracy and efficiency. We round the trained weights to integers and reduce the time consumption of homomorphic evaluation without affecting the security level. The secure genotype imputation inference model reduces the cost of large-scale gene sequencing and guarantees the safety of genes. If new variants must be imputed, training and homomorphic imputation are performed independently, with fast training and prediction. Since our basic model is a simple linear regression model, we consider using neural network models and activation functions to improve accuracy in the future. We also consider using ciphertext packaging technology to reduce data encryption time.

References

[1] J. Shendure, S. Balasubramanian, G. M. Church, W. Gilbert, J. Rogers, J. A. Schloss, R. H. Waterston, DNA sequencing at 40: past, present and future, Nature 550 (7676) (2017) 345–353.

[2] H. L. Allen, K. Estrada, G. Lettre, S. I. Berndt, M. N. Weedon, F. Rivadeneira, C. J. Willer, A. U. Jackson, S. Vedantam, S. Raychaudhuri, et al., Hundreds of variants clustered in genomic loci and biological pathways affect human height, Nature 467 (7317) (2010) 832–838.

[3] A. E. Locke, B. Kahali, S. I. Berndt, A. E. Justice, T. H. Pers, F. R. Day, C. Powell, S. Vedantam, M. L. Buchkovich, J. Yang, et al., Genetic studies of body mass index yield new insights for obesity biology, Nature 518 (7538) (2015) 197–206.

[4] V. Tam, N. Patel, M. Turcotte, Y. Bossé, G. Paré, D. Meyre, Benefits and limitations of genome-wide association studies, Nature Reviews Genetics 20 (8) (2019) 467–484.

[5] E. Evangelou, J. P. Ioannidis, Meta-analysis methods for genome-wide association studies and beyond, Nature Reviews Genetics 14 (6) (2013) 379–389.

[6] D. J. Schaid, W. Chen, N. B. Larson, From genome-wide associations to candidate causal variants by statistical fine-mapping, Nature Reviews Genetics 19 (8) (2018) 491–504.

[7] B. N. Howie, P. Donnelly, J. Marchini, A flexible and accurate genotype imputation method for the next generation of genome-wide association studies, PLoS genetics 5 (6) (2009) e1000529.

[8] S. Das, L. Forer, S. Schönherr, C. Sidore, A. E. Locke, A. Kwong, S. I. Vrieze, E. Y. Chew, S. Levy, M. McGue, et al., Next-generation genotype imputation service and methods, Nature genetics 48 (10) (2016) 1284–1287.

[9] B. L. Browning, Y. Zhou, S. R. Browning, A one-penny imputed genome from next-generation reference panels, The American Journal of Human Genetics 103 (3) (2018) 338–348.

[10] B. Servin, M. Stephens, Imputation-based analysis of association studies: candidate regions and quantitative traits, PLoS genetics 3 (7) (2007) e114.

[11] Z. Lin, A. B. Owen, R. B. Altman, Genomic research and human subject privacy, Science 305 (5681) (2004) 183–183.

[12] L. Zhang, Y. Cui, Y. Mu, Improving security and privacy attribute based data sharing in cloud computing, IEEE Systems Journal 14 (1) (2019) 387–397.

[13] M. Naveed, E. Ayday, E. W. Clayton, J. Fellay, C. A. Gunter, J.-P. Hubaux, B. A. Malin, X. Wang, Privacy in the genomic era, ACM Computing Surveys (CSUR) 48 (1) (2015) 1–44.

[14] B. Berger, H. Cho, Emerging technologies towards enhancing privacy in genomic data sharing, Genome biology 20 (1) (2019) 1–3.

[15] A. Acar, H. Aksu, A. S. Uluagac, M. Conti, A survey on homomorphic encryption schemes: Theory and implementation, ACM Computing Surveys (CSUR) 51 (4) (2018) 1–35.

[16] G. Gürsoy, E. Chielle, C. M. Brannon, M. Maniatakos, M. Gerstein, Privacy-preserving genotype imputation with fully homomorphic encryption, Cell Systems 13 (2) (2022) 173–182.

[17] R. Bost, R. A. Popa, S. Tu, S. Goldwasser, Machine learning classification over encrypted data, Cryptology ePrint Archive (2014).

[18] P. Mohassel, Y. Zhang, Secureml: A system for scalable privacy-preserving machine learning, in: 2017 IEEE symposium on security and privacy (SP), IEEE, 2017, pp. 19–38.

[19] Z. Brakerski, Fully homomorphic encryption without modulus switching from classical GapSVP, in: Annual Cryptology Conference, Springer, 2012, pp. 868–886.

[20] J. H. Cheon, A. Kim, M. Kim, Y. Song, Homomorphic encryption for arithmetic of approximate numbers, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2017, pp. 409–437.

[21] I. Chillotti, N. Gama, M. Georgieva, M. Izabachène, TFHE: fast fully homomorphic encryption over the torus, Journal of Cryptology 33 (1) (2020) 34–91.

[22] M. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, et al., Homomorphic encryption standard, in: Protecting Privacy through Homomorphic Encryption, Springer, 2021, pp. 31–62.

[23] D. Michie, D. J. Spiegelhalter, C. Taylor, et al., Machine learning, Neural and Statistical Classification 13 (1994) (1994) 1–298.

[24] R. L. Rivest, L. Adleman, M. L. Dertouzos, et al., On data banks and privacy homomorphisms, Foundations of secure computation 4 (11) (1978) 169–180.

[25] E. L. Cominetti, M. A. Simplicio, Fast additive partially homomorphic encryption from the approximate common divisor problem, IEEE Transactions on Information Forensics and Security 15 (2020) 2988–2998.

[26] J. Fan, F. Vercauteren, Somewhat Practical Fully Homomorphic Encryption., IACR Cryptology ePrint Archive 2012 (2012) 144.

[27] H. Chen, I. Iliashenko, K. Laine, When HEAAN Meets FV: a New Somewhat Homomorphic Encryption with Reduced Memory Overhead., IACR Cryptol. ePrint Arch. 2020 (2020) 121.

[28] Z. Brakerski, C. Gentry, V. Vaikuntanathan, (Leveled) fully homomorphic encryption without bootstrapping, ACM Transactions on Computation Theory (TOCT) 6 (3) (2014) 1–36.

[29] C. Gentry, S. Halevi, Implementing gentry's fully-homomorphic encryption scheme, in: Annual international conference on the theory and applications of cryptographic techniques, Springer, 2011, pp. 129–148.

[30] M. Van Dijk, C. Gentry, S. Halevi, V. Vaikuntanathan, Fully homomorphic encryption over the integers, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2010, pp. 24–43.

[31] I. Chillotti, N. Gama, M. Georgieva, M. Izabachene, Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds, in: international conference on the theory and application of cryptology and information security, Springer, 2016, pp. 3–33.

[32] Z. Liu, H. Seo, S. S. Roy, J. Großschädl, H. Kim, I. Verbauwhede, Efficient ring-LWE encryption on 8-bit AVR processors, in: International Workshop on Cryptographic Hardware and Embedded Systems, Springer, 2015, pp. 663–682.

[33] C. Gentry, A. Sahai, B. Waters, Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based, in: Annual Cryptology Conference, Springer, 2013, pp. 75–92.

[34] I. Chillotti, M. Joye, P. Paillier, New challenges for fully homomorphic encryption, in: Privacy-Preserving Machine Learning (PPML-PriML 2020) NeurIPS 2020 Workshop, 2020.

[35] S. Hong, J. H. Park, W. Cho, H. Choe, J. H. Cheon, Secure multi-label tumor classification using homomorphic encryption, Research Square (2021).

[36] H. V. L. Pereira, Bootstrapping fully homomorphic encryption over the integers in less than one second, in: IACR International Conference on Public-Key Cryptography, Springer, 2021, pp. 331–359.

[37] I. Chillotti, M. Joye, P. Paillier, Programmable bootstrapping enables efficient homomorphic inference of deep neural networks., IACR Cryptol. ePrint Arch. 2021 (2021) 91.

[38] O. Kocabas, T. Soyata, Utilizing homomorphic encryption to implement secure and private medical cloud computing, in: 2015 IEEE 8th International Conference on Cloud Computing, IEEE, 2015, pp. 540–547.

[39] A. Meehan, R. K. Ko, G. Holmes, Deep learning inferences with hybrid homomorphic encryption, 2018.

[40] A. Wood, K. Najarian, D. Kahrobaei, Homomorphic encryption for machine learning in medicine and bioinformatics, ACM Computing Surveys (CSUR) 53 (4) (2020) 1–35.

[41] M. Kim, A. O. Harmanci, J.-P. Bossuat, S. Carpov, J. H. Cheon, I. Chillotti, W. Cho, D. Froelicher, N. Gama, M. Georgieva, et al., Ultrafast homomorphic encryption models enable secure outsourcing of genotype imputation, Cell systems 12 (11) (2021) 1108–1120.

[42] N. Dokmai, C. Kockan, K. Zhu, X. Wang, S. C. Sahinalp, H. Cho, Privacy-preserving genotype imputation in a trusted execution environment, bioRxiv (2021).

[43] S. Halevi, V. Shoup, HElib-An Implementation of homomorphic encryption, Cryptology ePrint Archive, Report 2014/039 (2014).

[44] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, Journal of the ACM (JACM) 56 (6) (2009) 1–40.

[45] S. Das, G. R. Abecasis, B. L. Browning, Genotype imputation from large reference panels, Annual review of genomics and human genetics 19 (2018) 73–96.

[46] N. Siva, 1000 genomes project, Nature biotechnology 26 (3) (2008) 256–257.

[47] J. Li, J.-h. Cheng, J.-y. Shi, F. Huang, Brief introduction of back propagation (BP) neural network algorithm and its improvement, in: Advances in computer science and information engineering, Springer, 2012, pp. 553–558.

[48] F. Bourse, M. Minelli, M. Minihold, P. Paillier, Fast homomorphic evaluation of deep discretized neural networks, in: Annual International Cryptology Conference, Springer, 2018, pp. 483–512.

[49] M. R. Albrecht, R. Player, S. Scott, On the concrete hardness of learning with errors, Journal of Mathematical Cryptology 9 (3) (2015) 169–203.