applied sciences

MDPI

Article

Communication-Efficient Secure Federated Statistical Tests from Multiparty Homomorphic Encryption

Meenatchi Sundaram Muthu Selva Annamalai, Chao Jin * and Khin Mi Mi Aung

Institute for Infocomm Research, A*STAR, Singapore 138632, Singapore
* Correspondence: jin_chao@i2r.a-star.edu.sg

Abstract: The power and robustness of statistical tests are strongly tied to the amount of data available for testing. However, much of the collected data today is siloed amongst various data owners due to privacy concerns, thus limiting the utility of the collected data. While frameworks for secure multiparty computation enable functions to be securely evaluated on federated datasets, they depend on protocols over secret shared data, which result in high communication costs even in the semi-honest setting.In this paper, we present methods for securely evaluating statistical tests, specifically the Welch's t-test and the χ^2-test, in the semi-honest setting using multiparty homomorphic encryption (MHE). We tested and evaluated our methods against real world datasets and found that our method for computing the Welch's t-test and χ^2-test statistics required 100× less communication than equivalent protocols implemented using secure multiparty computation (SMPC), resulting in up to 10× improvement in runtime. Lastly, we designed and implemented a novel protocol to perform a table lookup from a secret shared index and use it to build a hybrid protocol that switches between MHE and SMPC representations in order to calculate the p-value of the statistics efficiently. This hybrid protocol is 1.5× faster than equivalent protocols implemented using SMPC alone.

Keywords: multiparty homomorphic encryption; federated analytics

Citation: Annamalai, M.S.M.S.; Jin, C.; Aung, K.M.M. Communication-Efficient Secure Federated Statistical Tests from Multiparty Homomorphic Encryption. *Appl. Sci.* **2022**, *12*, 11462. https://doi.org/10.3390/app122211462

Academic Editors: Yunheung Paek and Sangkyun Lee

Received: 22 October 2022
Accepted: 9 November 2022
Published: 11 November 2022

1. Introduction

While an increasing amount of data from a variety of domains is being collected to fuel much of the data-driven research conducted today, the data has also become increasingly partitioned and siloed due to privacy and data ownership concerns, especially in the medical domain [1]. This partitioning severely limits the utility of the data and hinders potentially life saving research from being conducted as modern data analytics pipelines strongly depend on the amount of data available. This is especially true for statistical testing, which is an important tool for analysis of data. In fact, more than 80% of articles from the New England Journal of Medicine and Nature Medicine contained inferential statistical methods such as t-test and χ^2-test [2], thus highlighting the importance of these types of analyses.

One solution to this privacy problem is the federated analytics frameworks based on secure multiparty computation (SMPC) techniques such as Sharemind [3] and MP-SPDZ [4], which provide protocols to compute over secretly shared data. These frameworks provide a general method for any number of data owners to collaborate and analyze the combined data without leaking any information about the inputs and intermediate values through secret sharing. Such frameworks have increased in popularity over the years due to advancements in computational efficiency [5] and their support for various security assumptions ranging from semi-honest to malicious security. However, one main disadvantage that is shared by such frameworks is the communication cost that incurs, especially when computing over a large batch of data as often is the case in statistical testing.

On the other hand, homomorphic encryption (HE) is an alternative solution to the problem that requires no interactivity during computation, thus resulting in significantly lesser communication costs. Even though HE has been applied in a range of machine learning settings such as Logistic Regression [6] and even Convolutional Neural Networks [7–10], the application of HE to computations that are not well approximated by polynomials such as statistical tests that involve division has been limited due to computationally expensive bootstrapping protocols. Multiparty homomorphic encryption (MHE) has recently been thrust into the spotlight [11] as it strikes a optimal balance between SMPC and HE by limiting interactivity to a computationally lightweight collective bootstrapping protocol that has to be invoked only after the computation has reached a predefined multiplicative depth. Therefore, our main purpose of research in this paper is to optimize the performance and communication efficiency of federated anlaytics frameworks through cryptographic algorithms and protocols design based on MHE.

1.1. Related Work

MP-SPDZ. The MP-SPDZ framework provides a wide range of SMPC protocols, in which we are particularly interested in the SPDZ2K protocol (in semi-honest mode) that is commonly used as the standard protocol for SMPC-based frameworks. We will also compare our MHE-based methods with SPDZ2K-based methods in this paper;

Sharemind. Sharemind is another SMPC-based framework similar to MP-SPDZ which implements protocols for secure computation over secret shared data. While there has been prior work done to specifically implement the t-test and χ^2-test amongst various other statistical tests on the Sharemind framework [12], a thorough evaluation that includes communication costs was not presented. Furthermore, the t-test took a long time (2.75 min) to finish even with moderately small number of records (2000). We will show that our MHE-based methods have more superior performance compared with SMPC-based methods;

FAMHE. FAMHE is a Federated analytics framework based on multiparty homomorphic encryption [11], which implements secure genome wide association study (GWAS) and secure calculation of the Kaplan–Meier survival analysis. In their paper, they mention a secure division protocol from MHE supposedly using polynomial approximations, which may lose accuracy. Additionally, they present a method for p-value calculation in the GWAS implementation, but since the distribution was assumed to be standard normal, this computation was approximated using polynomials and was directly calculated. Different from their approaches, we design more accurate secure division and P-value computation protocols, instead of using polynomial approximations;

STAR. The STAR system [13] presents methods for computing the t-test and χ^2-test among various other statistical tests using a hybrid MHE-SMPC approach to bypass secure division on encrypted data. The numerators and denominators for the statistics were calculated using MHE and converted into SMPC secret shares for the secure division to be done using SMPC. However, the encryption to share protocol was only simulated and the p-value computation was done in the clear. On the contrary, our system is end-to-end secure as the entire process including both statistical value computation and p-value computation is done in the encrypted domain, without the decryption of any intermediate data;

Fed-χ^2. The Fed-χ^2 [14] protocol recasts the χ^2-test as a second frequency moment estimation problem and uses secure aggregation and stable projections to reduce the dimensionality of the contingency table and by extension the communication cost of the overall protocol. However, the marginal statistics were leaked and no empirical communication cost data was presented. Our work differs from theirs on the design of more efficient and secure MHE-based computation protocols, which enables more efficient statistical test computations on encrypted data.

1.2. Our Contributions

This paper focuses on federated settings where the dataset is partitioned amongst collaborating institutions that wish to perform statistical tests securely. To that end, we

present communication efficient methods for computing the test statistics that are more efficient in terms of communication and runtime compared to an equivalent SMPC-based method. An overview of the system architecture is shown in Figure 1.

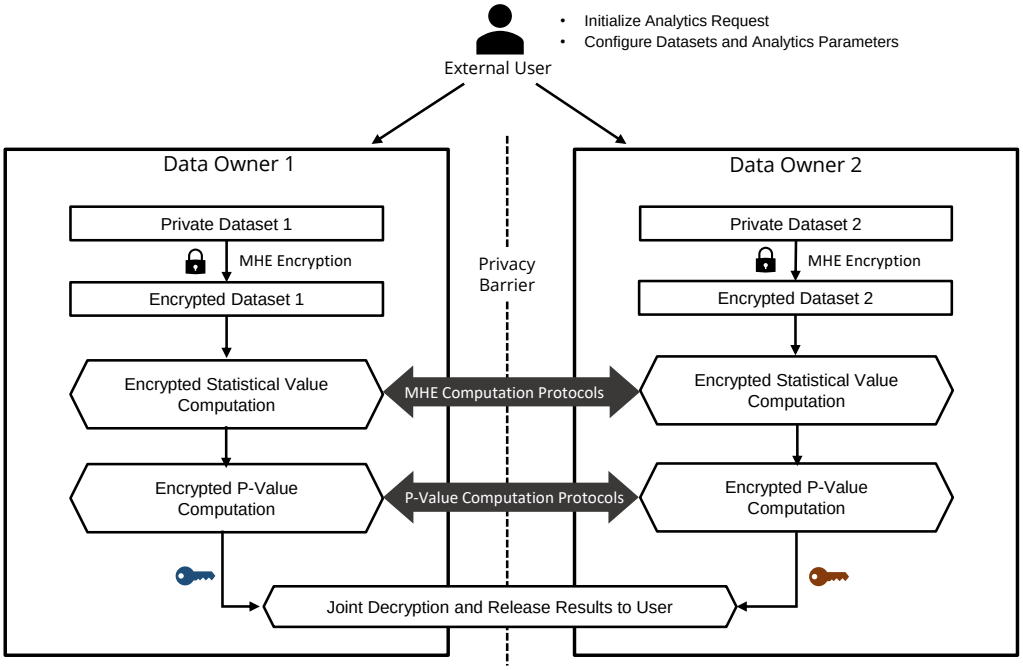

Figure 1. Federated analytics system architecture.

As a prerequisite, we first present a new method for the secure division fully using MHE. We find that even for a relatively moderate batch of divisions, our MHE-based method is far more efficient in terms of communication and outperforms equivalent SMPC-based methods in terms of runtime.

We then use this secure division method to compute the Welch's t-test and χ^2-test statistics on real world datasets and find that our method remains highly accurate while requiring far less communication than the SMPC method. This communication efficiency is leveraged to result in a significant speedup in runtime for our MHE-based method.

Lastly, we design and implement a novel protocol to perform privacy-preserving table lookup from a secret shared index. This is used to calculate the p-value of the Welch's t-test for which both the statistic and degree of freedom contain private information, and polynomial approximations are hard to craft. This is a similar yet different problem from that of private information retrieval (PIR) [15] where the table is held by the server and the index is held by the client. In order to compute the secret shared index from the encrypted statistic and degree of freedom, we design and implement another protocol that switches between the MHE and SMPC representations—specifically one that preserves the packing and "real" number encoding of the Cheon-Kim-King-Song variant of MHE [16].

2. Preliminaries

2.1. Multiparty Homomorphic Encryption

In this paper, we utilize the Cheon-Kim-Kim-Song [16] variant of the MHE scheme presented in Ref. [17], which enables secure computations to be performed over "real" numbers. Under this scheme, ciphertexts are encrypted under a public key with the corresponding secret key being held in a distributed manner amongst the parties. Therefore, decryption is an interactive process with each party performing some partial decryption, which is combined in the end to result in the plaintext. Additionally, the MHE scheme admits a collective bootstrapping protocol that is much more computationally efficient than standard bootstrapping techniques for the non-multiparty CKKS scheme.

The plaintext space is $R_{Q_L} = \mathbb{Z}[X]_{Q_L} / X^{\mathcal{N}} + 1$ where $Q_L = \prod_{i=1}^{L} q_i$ for q_i prime, $\mathcal{N}$ is a power of 2 and L represents the number of levels or the number of multiplications allowed before the noise in the ciphertext must be refreshed. The ciphertext space is consequently $R^2_{Q_L}$. Under this scheme, $\frac{\mathcal{N}}{2}$ values can be packed into a single ciphertext. The following functions are permitted in the CKKS variant of the MHE scheme:

- **SecKeyGen(1^λ):** Each party P_i generates its own secret key sk_i with security parameter λ;
- **ColKeyGen($\{sk_i\}$):** The parties collectively generate the collective public key pk;
- **Encode(v):** Encodes a vector of complex numbers $\mathbf{v} \in \mathbb{C}^{\mathcal{N}/2}$ as a plaintext $\bar{p} \in R_{Q_L}$;
- **Decode($\bar{p}$):** Decodes a plaintext $\bar{p} \in R_{Q_L}$ into a vector of complex numbers $\mathbf{v} \in \mathbb{C}^{\mathcal{N}/2}$;
- **Encrypt(pk, $\bar{p}$):** Encrypts plaintext $\bar{p} \in R_{Q_L}$ to ciphertext $\hat{c} \in R^2_{Q_L}$ under the public key pk;
- **ColDecrypt($\hat{c}$, $\{sk_i\}$):** The parties collectively decrypt a ciphertext $\hat{c} \in R^2_{Q_L}$ into a plaintext $\bar{p} \in R_{Q_L}$;
- **ColBootstrap($\hat{c}$, $\{sk_i\}$):** The parties collectively refresh the noise in the ciphertext $\hat{c} \in R^2_{Q_L}$ returning a new ciphertext $\hat{c}' \in R^2_{Q_L}$ with less noise;
- **HAdd($\hat{c}$, $\hat{c}'$):** Adds 2 ciphertexts $\hat{c}, \hat{c}' \in R^2_{Q_L}$;
- **HSub($\hat{c}$, $\hat{c}'$):** Subtracts the second ciphertext $\hat{c}'$ from the first $\hat{c}$;
- **HMul($\hat{c}$, $\hat{c}'$):** Multiplies 2 ciphertexts $\hat{c}, \hat{c}' \in R^2_{Q_L}$;
- **HRotate($\hat{c}$, k):** Assuming $\hat{c}$ encodes and encrypts $[v_1, \ldots, v_d]$, returns a ciphertext $\hat{c}'$ encoding and encrypting $[v_k, \ldots, v_d, v_1, \ldots, v_{k-1}]$;
- **HInnerSum(c):** For a ciphertext $\hat{c} \in R^2_{Q_L}$ which encodes and encrypts a vector $\mathbf{v} \in \mathbb{C}^{\mathcal{N}/2}$, compute the ciphertext that encodes and encrypts the vector $\mathbf{v}' \in \mathbb{C}^{\mathcal{N}/2}$ which has, as all its elements, the L_1 norm of the vector $\mathbf{v}$ i.e., $\mathbf{v}' \approx ||\mathbf{v}||_1 \cdot \mathbf{1}$.

Note that while technically the HAdd, HSub, and HMul operations have been defined here to operate on two ciphertexts, we abuse the notation in this paper to allow the second argument to be a plaintext (HAdd($\hat{c}$, $\bar{p}$)) or a constant (HAdd($\hat{c}$, a)).

2.2. System Overview

System Model. The setting considered is one where N parties each locally hold a partition of a global dataset wishing to collaboratively perform statistical analysis securely. For the Welch's t-test, we assume that the dataset is partitioned vertically whereas for the χ^2-test, we assume that the dataset is partitioned horizontally. Different types of partitioning for the two statistical tests were focused so as to capture real world use cases as explained in Section 4 and additionally to compare the scalability of the secure analytics solutions with respect to the size of the dataset. For example, if the horizontal federated setting was considered for the Welch's t-test, then each party can simply compute the mean and standard deviation on their local datasets and combine the results securely, thus making the size of the dataset completely irrelevant in the performance analysis.

Threat Model. We consider the semi-honest setting with a dishonest majority where a majority of parties can collude to share information and try to extract information about the other parties.

3. Communication-Efficient Secure Batch Division

An important prerequisite for computing statistical tests securely is secure division. There are two main algorithms that are used for division—Newton–Raphson and Goldschmidt. However, the Goldschmidt division (given below in Algorithm 1) is usually preferred as the multiplications in each iteration can be done concurrently [18]. Moreover, the Newton–Raphson method can only be used to calculate the inverse of the divisor with the dividend being multiplied in the end to calculate the result. The Goldschmidt method, on the other hand, is able to directly calculate the result, thus reducing the multiplicative depth, which is a crucial consideration in the context of MHE.

Goldschmidt's division algorithm begins with a "good" initial approximation and iteratively improves upon the approximation. Specifically, a "good" initial approximation

is defined as an approximation with relative error $\epsilon_0 = 1 - bw_0 < 1$ and it can be shown through induction that after t iterations, a_t has relative error $\epsilon_0^{2^t}$ [18], which makes this a very efficient method.

Algorithm 1 Goldschmidt division

Require: a, b, T, w_0 s.t. w_0 is a "good" initial approximation to $\frac{1}{b}$
Ensure: $c \approx \frac{a}{b}$
  $a_0 \leftarrow a$
  $b_0 \leftarrow w_0$
  $i \leftarrow 0$
  while $i < T$ **do**
    $a_{i+1} \leftarrow a_i w_i$
    $b_{i+1} \leftarrow b_i w_i$
    $w_{i+1} \leftarrow 2 - b_{i+1}$
    $i \leftarrow i + 1$
  end while
  return $c = a_T$

Traditionally, in a SMPC setting, the initial approximation is obtained by normalizing the denominator to the range $(0.5, 1]$, where a well known "good" initial approximation exists ($w_0 = 2.9142 - 2b$, $\epsilon_0 < 0.08578$) [18]. The normalization is done through the use of advanced operations such as Damgård et al.'s secure bit decomposition protocol [19]. However, in MHE, efficient secure bit decomposition protocols are not known to exist, which forces us to obtain the initial approximation through other means. Our solution is to relax the problem and assume that the denominator is known to exist in some arbitrary range $[b_\ell, b_u]$. This is a safe assumption in the context of statistical tests where the input variables are typically bounded, thus allowing us to compute the bounds for the denominator as was the case for both our target applications of Welch's t test and χ^2 test (see Section 4). Assuming that the denominator is in some arbitrary range, we then construct a "good" initial approximation by utilizing optimal initial approximations that minimize the relative error of the initial approximation, which have been published by Schlute et al. [20]. It can be shown that for a given denominator $b \in [b_\ell, b_u]$, the approximation $w_0 = \frac{-8}{b_\ell^2 + 6b_\ell b_u + b_u^2} b + \frac{8(b_\ell + b_u)}{b_\ell^2 + 6b_\ell b_u + b_u^2}$ minimizes the relative error $\epsilon_0 = \frac{(b_u - b_\ell)^2}{b_\ell^2 + 6b_\ell b_u + b_u^2} = \frac{b_\ell^2 - 2b_\ell b_u + b_u^2}{b_\ell^2 + 6b_\ell b_u + b_u^2} < 1$, which satisfies the constraint for a "good" initial approximation for the Goldschmidt algorithm as well.

One important consideration in utilizing this method is that since we may not have tight bounds on the denominator, the algorithm can take a large number of iterations—taking more than 10 iterations for denominators in the range $[1, 1 \times 10^4]$—unlike the SMPC method, which can finish within 2 iterations after normalization. While this theoretically results in a high multiplicative depth circuit, we leverage the lightweight collective bootstrapping protocol presented by Mouchet et al. [17] to trade off some communication cost for computational efficiency.

Additionally, by leveraging the Single Instruction Multiple Data (SIMD) property of MHE, our method allows us to compute an entire batch of divisions in one go, thus resulting in much lower communication costs compared to performing a similar batch of divisions in SMPC. Please see a full discussion based on the experimental results on the tradeoffs of this method as compared to the standard SMPC method in Section 6. The full secure division algorithm based on MHE is presented below in Algorithm 2.

Algorithm 2 Homomorphic division (HDiv)

Require: Encryptions $\hat{a}, \hat{b}$, Range of denominator b_ℓ, b_u, Number of iterations T
Ensure: Decode(ColDecrypt($\hat{c}$)) $\approx \frac{a}{b}$

$\hat{w}_0 = \text{HAdd}(\text{HMul}(\frac{-8}{b_\ell^2+6b_\ell b_u+b_u^2}, \hat{b}), \frac{8(b_\ell+b_u)}{b_\ell^2+6b_\ell b_u+b_u^2})$
$\hat{a}_0 \leftarrow \hat{a}$
$\hat{b}_0 \leftarrow \hat{w}_0$
$i \leftarrow 0$
while $i < T$ **do**
  $\hat{a}_{i+1} \leftarrow \text{HMul}(\hat{a}_i, \hat{w}_i)$
  $\hat{b}_{i+1} \leftarrow \text{HMul}(\hat{b}_i, \hat{w}_i)$
  $\hat{w}_{i+1} \leftarrow \text{HSub}(2, \hat{b}_{i+1})$
  if $\hat{w}_{i+1}$ will exceed mult. depth **then**
    ColBootstrap($\hat{a}_{i+1}$)
    ColBootstrap($\hat{b}_{i+1}$)
    ColBootstrap($\hat{w}_{i+1}$)
  end if
  $i \leftarrow i+1$
end while
return $\hat{c} = \hat{a}_T$

4. Secure Federated Statistical Tests

4.1. Welch's t-Test

For the secure federated *t*-test, we focus on a problem setup where a global dataset is split vertically amongst two clients. One of the clients holds the attributes of all samples in the dataset whereas the other client holds the classes that each sample belongs to. This is a practical setting especially in the healthcare industry where one entity has access to a patient's genotype information (attributes), whereas another entity has access to a patient's phenotype information (classes) and these entities collaborate to analyze their combined data, as can be seen from Figure 2. One example of such an analysis is Polygenic Risk Score Validation [21], which refers to the problem of verifying that genetic risk scores developed on external populations (e.g., Caucasian) remain applicable to local populations (e.g., East Asian). A common statistical test employed to do such testing is the *t*-test, which is a test that compares the means of two groups.

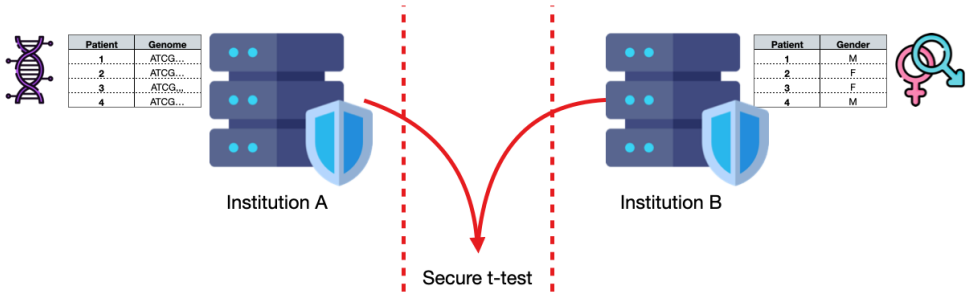

Figure 2. Federated setting for Welch's *t*-test. Institution A owns the genomic data, and institution B owns the phenotype data, on the same set of patient IDs. They jointly do the federated *t*-test without revealing any private data information to each other.

Formally, there are two clients. Client 1 has a vector of values $\mathbf{v} = [v_1, \ldots, v_N]$ such that $v_i \in [0,1]$ and client 2 has a vector of classes $\mathbf{c} = [c_1, \ldots, c_N]$ s.t. $c_i \in \{0,1\}$. Even though, mathematically, v_is do not need to be bounded, in the context of healthcare, these values are usually normalized [22], thus making the assumption practical. If $c_i = 0$, then v_i belongs to group 1 and if $c_i = 1$, then v_i belongs to group 2. The number of users in group 1 and group 2 are represented as n_1 and n_2 and are assumed to be publicly known. The task

is to then calculate the t statistic $t = \frac{\bar{X}_1 - \bar{X}_2}{s_\Delta}$ securely where $s_\Delta = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$ and $\bar{X}_i$ and s_i are the mean and standard error of samples in group i. Additionally, the degree of freedom $\frac{(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2})^2}{\frac{(s_1^2/n_1)^2}{n_1 - 1} + \frac{(s_2^2/n_2)^2}{n_2 - 1}}$ is also calculated securely so that significance testing can be done. In order to avoid having to calculate the square root securely, we instead calculate $t^2 = \frac{(\bar{X}_1 - \bar{X}_2)^2}{s_\Delta^2}$ and leave the square root calculation to be done in cleartext without any additional leakage of information.

In our method, we leverage the SIMD packing of MHE schemes to pack $\mathbf{v}$ and $\mathbf{c}$ into a single ciphertext and calculate the numerators and denominators of the t^2 and degree of freedom values. We then use our secure batch division protocol presented in Section 3 to perform the two divisions simultaneously before decrypting and releasing the results.

Note that since $v_i \in [0, 1]$, $s_1^2, s_2^2 \in [0, 1]$, and the denominator of t^2 is bounded in the range $[0, \frac{1}{n_1} + \frac{1}{n_2}]$. Meanwhile, the denominator of the degree of freedom is bounded in the range $[0, \frac{1}{n_1^2(n_1 - 1)} + \frac{1}{n_2^2(n_2 - 1)}]$. This information is used in our secure batch division protocol to obtain the initial approximation to the denominator.

4.2. χ^2-Test

For the secure federated χ^2-test, we follow a similar problem setup as the one presented by Wang et al. [14] where the global dataset is split horizontally amongst clients who collaborate to perform the χ^2-test to calculate the correlation between two attributes in the dataset, which can be seen from Figure 3.

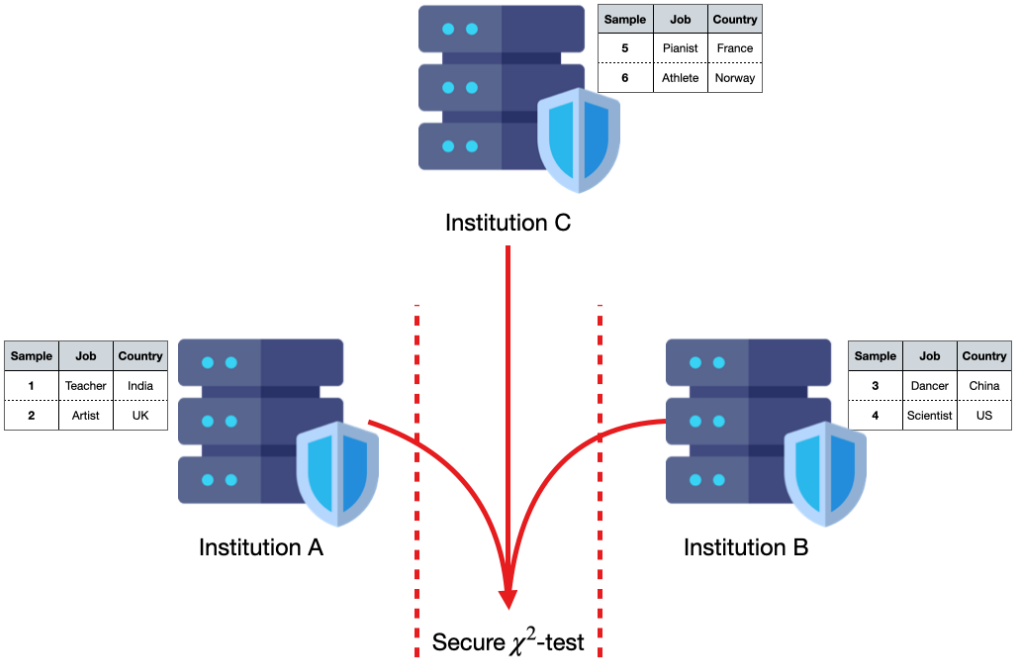

Figure 3. Federated setting for χ^2-test. All the institutions own the same type of data, but for different sample IDs. They jointly do the federated analytics on all the samples without revealing any private data information to each other.

Formally, there are n clients, and client i has a horizontal partition of the global dataset $\mathcal{D}_i = \{(x, y, v_{xy}^{(i)})\}$ where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ are categories of the contingency table and $v_{xy}^{(i)} \in \mathbb{Z}_{\geq 0}$ is the observed number of samples in category x and y in the client's local contingency table. The global dataset is $\mathcal{D} = \{(x, y, v_{xy}) : v_{xy} = \sum_{i \in [n]} v_{xy}^{(i)}\}$. For the

purposes of the χ^2-test, the data in the contingency table is assumed to be discrete and marginal statistics are represented as $v_x = \sum_{y \in \mathcal{Y}} v_{xy}$, $v_y = \sum_{x \in \mathcal{X}} v_{xy}$ and $v = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} v_{xy}$. Lastly, the expectation of v_{xy} if x and y are not correlated is defined a $\bar{v}_{xy} = \frac{v_x * v_y}{v}$. The task is to then calculate the statistic $s_{\chi^2}(\mathcal{D}) = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \frac{(v_{xy} - \bar{v}_{xy})^2}{\bar{v}_{xy}}$ securely from the local contingency tables $\mathcal{D}_i$.

One key difference between the formulation of Wang et al. and our formulation is that they assume that the marginals $\{v_x\}_{x \in \mathcal{X}}$ and $\{v_y\}_{y \in \mathcal{Y}}$ are non-private information once they have been securely aggregated and therefore can be made public. However, as much of the literature in the area of differential privacy recognizes marginals (which are basically histograms) as private information [23], we make no such assumption and arrive at a solution that is truly end to end secure.

In our method, we leverage the SIMD packing of MHE schemes to pack the entire contingency table into a single ciphertext and utilize the secure batch division protocol presented in Section 3 to calculate all of the $\frac{(v_{xy} - \bar{v}_{xy})^2}{\bar{v}_{xy}}$ terms simultaneously. Finally, we calculate the inner sum of the resulting ciphertext to calculate the $s_{\chi^2}(\mathcal{D})$ statistic securely.

It is to be noted that in this case, we can clearly see that $\bar{v}_{xy}$ must be in the range $[0, N]$ where N is the total number of samples, which is crucial in obtaining the initial approximation in our secure batch division protocol.

5. *p*-Value Computation

In order to compute the p-value of the Welch's t-test from the encrypted t^2 and degree of freedom values, we design and implement two sub protocols. The first sub protocol converts the encrypted t^2 and degree of freedom values in the CKKS variant of MHE into secret shared data. The secret shared values are then converted into a secret shared index of a p-value lookup table by evaluating a simple rounding function using SMPC. Note that this conversion cannot be fully done in MHE as it requires the rounding operation necessary to convert "real" numbers into integers, which has not been efficiently implemented in MHE yet. The second protocol takes in the secret shared index and uses the rotation operation in MHE (**HRotate**) in order to perform a table lookup on the p-value lookup table.

5.1. MHE to SMPC Protocol

In Ref. [17], Mouchet et al. present an encryption to secret share (Enc2Share) protocol for the BFV scheme and use it for multiplication triple generation. In the protocol, each party P_i (other than P_1) samples its own secret share $M_i \in R_t$ (where R_t is the plaintext space of the BFV scheme) and sends a partial decryption of the ciphertext with its own secret share subtracted to P_1. P_1 fully decrypts the partial decryptions to get $M_1 = m - \sum_{i=2}^{N} M_i$ where m is the plaintext and N is the number of parties.

While this method works for integer-based HE schemes such as BFV where the plaintext space is small, allowing M_i to be easily sampled uniformly from R_t, in approximate arithmetic schemes such as CKKS, the size of the plaintext space presents a challenge to the Enc2Share protocol. Nevertheless, a solution is presented in the lattigo library that leverages the noise smudging approach of Asharov et al. [24] to construct secret shared polynomials in the plaintext space, whose sum is the target plaintext but are statistically indistinguishable from random polynomials.

However, this results in secret shared polynomials that have very large coefficients, and it is not clear how such a protocol can be used in practice to switch between the ciphertext encrypting a vector of real numbers $[v_1, v_2, \ldots, v_d]$ and a vector of secret shared fixed-point real numbers $[[v_1], [v_2], \ldots, [v_d]]$. A naive method for doing so would be to sum the plaintext polynomials and perform the decoding fully using SMPC, which would result in huge communication costs due to the large degree of the plaintext polynomials.

Instead, in our method, each party P_i instead decodes the polynomials first using the Fast Fourier Transform but using high precision arithmetic (128 bit floating point) to arrive at a vector of floating point numbers $[v_1^{(i)}, \ldots, v_d^{(i)}]$. We note that since the smudging lemma

was used in the generation of the secret shares, each element in the vector can in fact be summed across the parties to arrive at the expected value, i.e., $\forall k\ v_k = \sum_{i=1}^{N} v_k^{(i)}$. We can then continue our computation in SMPC by taking in the necessary $v_k^{(i)}$s as "real" valued inputs from each party and adding them up by using SMPC to arrive at the secret shared representation of the original encryption.

5.2. Private Table Lookup

We first assume that each party P_i has a share of the secret index k_i such that the secret index $k = \sum_i^N k_i \bmod \frac{\mathcal{N}}{2}$ where there are N parties and $\mathcal{N}$ is the ring dimension of the MHE scheme. P_0 additionally has the lookup table $L = [\ell_1, \ell_2, \ldots, \ell_{\frac{\mathcal{N}}{2}}]$ such that the parties want to compute ℓ_k securely.

Firstly, P_0 rotates the table by k_0 rotations and encrypts it and sends this ciphertext to Party 1. Each subsequent party P_{i+1} then rotates the ciphertext received from P_i further by k_{i+1}. The last party, P_{N-1}, sends the resulting ciphertext to P_0, which multiplies the ciphertext by the encoding of the vector $[1, 0, \ldots, 0]$ before all parties collectively decrypt the ciphertext. The result will be the vector $[\ell_k, 0, \ldots, 0]$ since rotations in MHE happen modulo $\frac{\mathcal{N}}{2}$. Furthermore, as all but the resulting lookup value is masked in the final calculation and the index of lookup value is not leaked in any way, we can conclude that this protocol is indeed secure. Crucially, we notice that for lookup tables of size up to $\frac{\mathcal{N}}{2}$ (which in practice is usually quite large—16,384), the number of rounds of this protocol is constant compared to equivalent SMPC protocols where the number of rounds would depend on the size of the lookup table. Further analysis of the communication efficiency and runtime can be found in Section 6.4.

6. Benchmarks

Our benchmarks are run with two or three parties communicating over a LAN network with an average bandwidth of 1 Gbps. Our MHE protocol is written using the lattigo [25] library implementing the CKKS scheme [16] instantiated with the standard parameters to achieve 128-bit security for a ring size of $\mathcal{N} = 32{,}768$ ($\log Q_L = 730$, $\log scale = 40$). Batches of size $\frac{\mathcal{N}}{2} = 16{,}384$ can be processed simultaneously in this setting using the SIMD property of the CKKS scheme.

Our methods are compared with equivalent algorithms written in the MP-SPDZ framework based on the SPDZ2K protocol, which is stripped of the steps required for malicious security (also known as semi honest mode) with a statistical security of 40-bits. While we are using the 2 and 3 party settings to demonstrate the utility of our methods over traditional SMPC approaches, it is to be noted that both our methods and the SMPC approaches are flexible and can be extended to more parties.

6.1. Secure Batch Division

In comparing our secure batch division protocol, we fix three parties and focus on two settings—one consisting of a small batch of divisions ($N = 2$) and one consisting of a large batch of division ($N = 128$), which corresponds to the required number of divisions for the Welch's t-test and χ^2-test, respectively. We then linearly extrapolate the runtime and communication overhead of running the SMPC approach for a batch of 16384 divisions, as this is not practical to test.

Assuming that we have a set of numerators $a_1, a_2, \ldots, a_N$ and a set of denominators $b_1, b_2, \ldots, b_N$ that are either encrypted (for our MHE protocol) or secret shared (for the SMPC method), the task is to compute either encrypted or secret shared versions of $c_1, c_2, \ldots, c_N$ s.t. $c_i \approx \frac{a_i}{b_i}$. In order to demonstrate the robustness and relative error of our protocol, a_i, b_i are additionally randomly chosen from two ranges—a small positive range $[1 \times 10^{-4}, 1]$ and a large positive range $[1, 1 \times 10^4]$. We compare three different metrics averaged across five runs of each computation—communication overhead (MB), runtime (s), and relative error (%)—and present the results in Tables 1–3, respectively. As the com-

munication overhead and runtimes are similar in both ranges, only the results for the large range is presented.

Table 1. Communication overhead (MB) of performing a batch of N divisions where numerators and denominators are randomly chosen from $[1, 1 \times 10^4]$.

$N = 2$		$N = 128$		$N = 16{,}384$	
MHE	SMPC	MHE	SMPC	MHE	SMPC
65.5	46.5	65.5	1852	65.5	234,743

Table 2. Runtime (s) of performing a batch of N divisions where numerators and denominators are randomly chosen from $[1, 1 \times 10^4]$.

$N = 2$		$N = 128$		$N = 16{,}384$	
MHE	SMPC	MHE	SMPC	MHE	SMPC
7.54	0.73	7.54	9.75	7.54	1232

Table 3. Relative error (%) of performing secure batch division where numerators (a_i) and denominators (b_i) are randomly chosen from various ranges.

$a_i, b_i \in [1 \times 10^{-4}, 1]$		$a_i, b_i \in [1, 1 \times 10^4]$	
MHE	SMPC	MHE	SMPC
0.0316	0.0126	0.0424	0.00294

We can see that while the SMPC method requires less communication and is 10× faster when the number of divisions is low, even when the number of divisions is moderate, it requires 30× more communication than our MHE method. This communication overhead makes a difference even in the LAN setting, thus making our MHE method faster for a moderate number of divisions. At scale, when considering the full supported batch of divisions in MHE, we can clearly see that the SMPC method requires 235 GB of data to be transferred between the parties, thus making it impractical even in a LAN setting. The full amortized runtime cost of running secure batch division using our MHE method is 4.60×10^{-4} s/division compared to the SMPC method's 7.52×10^{-2} s/division, which is a more than a 100× improvement in runtime.

In terms of relative error, while we can see that the MHE method appears slightly more accurate than the SMPC method, this is most likely due to the larger number of iterations that the Goldschmidt division has to be run for in our MHE method owning to the large range of possible values.

6.2. Secure Federated t-Test

We tested our secure federated t-test protocol against a polygenic risk score validation use case using the 1000Genomes dataset [26]. We conduct a t-test based on the polygenic risk score for cholesterol (PGS000192 [27] from the PGS Catalog) against three pairs of classes—American and East Asian (Data ID 1, 1075 samples), European and African (Data ID 2, 1526 samples), and Male and Female (Data ID 3, 3202 samples). We similarly compare the communication overhead (MB), runtime (s), and relative error (%) of computing the t^2 value and degree of freedom across the three pairs of classes in Tables 4–6, respectively.

It is clear that even when working with a small number of samples as in the American and East Asian setting, the calculation of the mean and variance of the two groups creates a huge bottleneck for the SMPC method, which has to exchange 50× more data amongst the two parties, thus resulting in a 2× slowdown in runtime when compared to our MHE method. This effect is further exacerbated in the other settings with the SMPC method requiring 12 GB of communication between the parties in the Male and Female setting. This

clearly shows that for even medium scale computations, the SIMD property of MHE vastly improves performance and can potentially overcome the efficiency losses of performing more complicated operations such as division.

Table 4. Communication overhead (MB) of running the secure federated *t*-test on each pair of classes as indicated by the ID.

ID	MHE	SMPC
1	86.0	4008
2	86.0	5661
3	86.0	11,810

Table 5. Runtime (s) of running the secure federated *t*-test on each pair of classes as indicated by the ID.

ID	MHE	SMPC
1	25.3	42.9
2	25.5	60.9
3	29.2	129

Table 6. Relative error (%) of calculating *t* statistic and degree of freedom (d.f.) in the secure federated *t*-test on each pair of classes as indicated by the ID.

ID	*t*		d.f.	
	MHE	SMPC	MHE	SMPC
1	4.04	0.0433	0.0897	0.0286
2	4.23	0.0161	0.0317	1.75
3	4.07	0.341	0.363	12.7

Overall, both methods have roughly the same relative error with our MHE method being more accurate in calculating the degrees of freedom whereas the SMPC method is more accurate in calculating the *t* statistic. On the whole, our MHE method has an average relative error of 2.14% compared to the SMPC method's 2.45%, which shows that both methods are relatively accurate.

6.3. Secure Federated χ^2-Test

We tested our secure federated χ^2-test protocol against 3 of the 12 real world datasets used by Wang et al. [14] listed below in Table 7. The three datasets (Mushroom, Credit, and Adult) were chosen to cover a range of sizes for the contingency table ($|\mathcal{D}|$) while keeping the number of divisions practical for the SMPC method. We compare the communication overhead (MB), runtime (s) and relative error (%) of computing the χ^2-statistic across the three datasets in Tables 8–10, respectively.

Table 7. Details of datasets used in benchmarking the secure federated χ^2-test on various datasets.

Dataset	Mushroom	Credit	Adult
Attr 1	Cap color	Feature 6	Occupation
# Attr 1	10	10	14
Attr 2	Odor	Feature 7	Native Country
# Attr 2	9	10	41
$\lvert\mathcal{D}\rvert$	90	100	574

Table 8. Communication overhead (MB) of running the secure federated χ^2-test on various datasets.

Dataset	$\lvert\mathcal{D}\rvert$	MHE	SMPC
Mushroom	90	196	7814
Credit	100	196	8680
ADULT	574	196	49,631

Table 9. Runtime (s) of running the secure federated χ^2-test on various datasets.

Dataset	$\lvert\mathcal{D}\rvert$	MHE	SMPC
Mushroom	90	19.0	41.8
Credit	100	19.3	46.9
ADULT	574	19.3	267

Table 10. Relative error (%) of calculating χ^2-statistic in the secure federated χ^2-test on various datasets.

Dataset	$\lvert\mathcal{D}\rvert$	MHE	SMPC
Mushroom	90	1.06×10^{-6}	1.60×10^{-5}
Credit	100	2.03×10^{-6}	4.50×10^{-5}
ADULT	574	3.19×10^{-3}	1.51×10^{-4}

For the secure federated χ^2-test, we can see that our MHE method is clearly much more efficient in terms of the communication overhead and runtime as this is the perfect setting to leverage the amortized cost of our secure batch division protocol. The minimum number of divisions required is 90 for the Mushroom dataset where our MHE method is already 2× faster than the SMPC method. At the high end, the ADULT dataset required 574 divisions to be performed simultaneously, resulting in a 10× improvement in runtime and 300× improvement in communication for our MHE method over the SMPC method.

The secure batch division protocol remained highly accurate in this setting, with an average relative error of 1.06×10^{-3}% for calculating the χ^2 statistic in our MHE method and 2.06×10^{-4}% in the SMPC method.

6.4. p-Value Computation

We tested our hybrid protocol for p-value computation that starts from encrypted t^2 and degree of freedom values and performs the lookup using our MHE method against an equivalent SMPC protocol, which starts from the secret shared t^2 and degree of freedom values and performs the lookup using SMPC. The size of the lookup table used was 16,320. For the SMPC table lookup, we use the `semi-bin-party` from the MP-SPDZ library that implements a semi-honest SMPC protocol over the binary domain instead of the SPDZ2K protocol previously used for other benchmarks so as to present a fair comparison as lookup operations are far more efficiently implemented in the binary domain as opposed to an arithmetic domain. The runtime (s) and absolute error of computing the p-value across the three test cases from the 1000Genome dataset previously used in Section 6.3 can be found in Tables 11 and 12, respectively. The absolute error has been reported here instead of the relative error as some of the p-values are very close to 0.

Table 11. Runtime (s) of calculating the p-value of the t statistic and degree of freedom previously calculated in Table 6.

ID	Hybrid	SMPC
1	6.24	9.08
2	7.72	8.64
3	6.79	8.99

Table 12. Absolute error of calculating the p-value of the t statistic and degree of freedom previously calculated in Table 6.

ID	Hybrid	SMPC
1	5.45×10^{-3}	5.47×10^{-3}
2	1.68×10^{-9}	1.53×10^{-12}
3	2.52×10^{-2}	2.52×10^{-2}

We notice that our hybrid method is much faster compared to the pure SMPC method. This is mainly due to the number of rounds of communication taken up by the SMPC lookup, which was 33,245. Compared to the SMPC lookup, our hybrid method only took 1150 rounds, which is 30× less. In fact, the number of rounds of communication made such an impact that even though our hybrid method resulted in 236 MB of communication compared to the SMPC lookup, which only took 126 MB, our hybrid method was up to 1.5× faster than the SMPC method in the LAN environment. The increase in communication for the MHE lookup is mainly due to the extra MHE to SMPC protocol that had to be run before the lookup could be performed, with the actual lookup protocol only taking about 36 MB of communication. This clearly shows that our hybrid method is far more efficient than the pure SMPC method for p-value calculation. We see that the table lookup method is fairly accurate with very small errors across the three experiments.

7. Conclusions

This work considered the problem of computing statistical tests securely in a federated setting using MHE. To that end, we presented a new method for secure division based on MHE that is communication efficient when performing large batches of divisions simultaneously. We applied this method to two common statistical tests—Welch's t-test and χ^2-test—in the vertical and horizontal federated settings, respectively. We evaluated our methods against equivalent methods over secret shared data using the MP-SPDZ framework and found that our methods are a lot more communication-efficient. Even when the parties are communicating over a high speed LAN network, this communication efficiency is leveraged to achieve a maximum of 5× and 10× improvement in runtime when computing the Welch's t-test and χ^2-test, respectively. Moreover, we designed and implemented novel protocols to switch between MHE and SMPC for the CKKS scheme and to perform a table lookup using a secret shared index. We used the protocols to compute the p-value for the Welch's t-test and found that our method was 1.5× faster than the pure SMPC-based method. We believe that our method opens up the possibility to perform large scale complicated analyses on encrypted data efficiently by switching between MHE and SMPC for approximate arithmetic, which was not previously possible with integer-based arithmetic. As for future work, we plan to expand our work and apply our methods to more applications with larger scale settings.

Author Contributions: Conceptualization, M.S.M.S.A., C.J. and K.M.M.A.; methodology, M.S.M.S.A. and C.J.; Writing—original draft, M.S.M.S.A., C.J. and K.M.M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by A*STAR, Singapore under its RIE2020 Advanced Manufacturing and Engineering (AME) Programmatic Programme (Award A19E3b0099).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Patient consent was waived due to the fact that only publicly available data was used.

Data Availability Statement: 1000 Genome Dataset: https://www.internationalgenome.org/data; Mushroom Dataset: https://archive.ics.uci.edu/ml/datasets/mushroom; Credit Dataset: https://www.kaggle.com/datasets/praveengovi/credit-risk-classification-dataset; Adult Dataset: https://archive.ics.uci.edu/ml/datasets/adult.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sheller, M.J.; Edwards, B.; Reina, G.A.; Martin, J.; Pati, S.; Kotrotsou, A.; Milchenko, M.; Xu, W.; Marcus, D.; Colen, R.R.; et al. Federated learning in medicine: Facilitating multi-institutional collaborations without sharing patient data. *Sci. Rep.* **2020**, *10*, 12598. [CrossRef] [PubMed]
2. Strasak, A.M.; Zaman, Q.; Marinell, G.; Pfeiffer, K.P.; Ulmer, H. The use of statistics in medical research: A comparison of The New England Journal of Medicine and Nature Medicine. *Am. Stat.* **2007**, *61*, 47–55. [CrossRef]
3. Bogdanov, D.; Laur, S.; Willemson, J. Sharemind: A framework for fast privacy-preserving computations. In Proceedings of the European Symposium on Research in Computer Security, Málaga, Spain, 6–8 October 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 192–206.
4. Keller, M. MP-SPDZ: A versatile framework for multi-party computation. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual, 9–13 September 2020; pp. 1575–1590.
5. Ishai, Y.; Kilian, J.; Nissim, K.; Petrank, E. Extending oblivious transfers efficiently. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 145–161.
6. Sim, J.J.; Chan, F.M.; Chen, S.; Meng Tan, B.H.; Mi Aung, K.M. Achieving GWAS with homomorphic encryption. *BMC Med. Genom.* **2020**, *13*, 90. [CrossRef] [PubMed]
7. Al Badawi, A.; Jin, C.; Lin, J.; Mun, C.F.; Jie, S.J.; Tan, B.H.M.; Nan, X.; Aung, K.M.M.; Chandrasekhar, V.R. Towards the alexnet moment for homomorphic encryption: Hcnn, the first homomorphic cnn on encrypted data with gpus. *IEEE Trans. Emerg. Top. Comput.* **2020**, *9*, 1330–1343. [CrossRef]
8. Jin, C.; Ragab, M.; Aung, K.M.M. Secure transfer learning for machine fault diagnosis under different operating conditions. In Proceedings of the International Conference on Provable Security, Singapore, 29 November–1 December 2020; Springer: Cham, Switzerland, 2020; pp. 278–297.
9. Jin, C.; Al Badawi, A.; Unnikrishnan, J.B.; Mun, C.F.; Brown, J.M.; Campbell, J.P.; Chiang, M.; Kalpathy-Cramer, J.; Chandrasekhar, V.R.; Krishnaswamy, P.; et al. CareNets: Efficient homomorphic CNN for high resolution images. In Proceedings of the NeurIPS Workshop on Privacy in Machine Learning (PriML), Vancouver, BC, Canada, 13–14 December 2019.
10. Wang, J.; Jin, C.; Tang, Q.; Liu, Z.; Aung, K.M.M. CryptoRec: Novel Collaborative Filtering Recommender Made Privacy-Preserving Easy. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 2622–2634. [CrossRef]
11. Froelicher, D.; Troncoso-Pastoriza, J.R.; Raisaro, J.L.; Cuendet, M.A.; Sousa, J.S.; Cho, H.; Berger, B.; Fellay, J.; Hubaux, J.P. Truly privacy-preserving federated analytics for precision medicine with multiparty homomorphic encryption. *Nat. Commun.* **2021**, *12*, 5910. [CrossRef]
12. Bogdanov, D.; Kamm, L.; Laur, S.; Pruulmann-Vengerfeldt, P.; Talviste, R.; Willemson, J. Privacy-preserving statistical data analysis on federated databases. In Proceedings of the Annual Privacy Forum, Athens, Greece, 20–21 May 2014; Springer: Cham, Switzerland, 2014; pp. 30–55.
13. Servan-Schreiber, S.; Ohrimenko, O.; Kraska, T.; Zgraggen, E. STAR: Statistical Tests with Auditable Results. *arXiv* **2019**, arXiv:1901.10875.
14. Wang, L.; Pang, Q.; Wang, S.; Song, D. FED-χ^2: Privacy Preserving Federated Correlation Test. *arXiv* **2021**, arXiv:2105.14618.
15. Chor, B.; Goldreich, O.; Kushilevitz, E.; Sudan, M. Private information retrieval. In Proceedings of the IEEE 36th Annual Foundations of Computer Science, Milwaukee, WI, USA, 23–25 October 1995; pp. 41–50.
16. Cheon, J.H.; Kim, A.; Kim, M.; Song, Y. Homomorphic encryption for arithmetic of approximate numbers. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Hong Kong, China, 3–7 December 2017; Springer: Cham, Switzerland, 2017; pp. 409–437.
17. Mouchet, C.; Troncoso-Pastoriza, J.; Bossuat, J.P.; Hubaux, J.P. Multiparty Homomorphic Encryption from Ring-Learning-with-Errors. *Proc. Priv. Enhancing Technol.* **2021**, *2021*, 291–311. doi: 10.2478/popets-2021-0071. [CrossRef]
18. Catrina, O.; Saxena, A. Secure computation with fixed-point numbers. In Proceedings of the International Conference on Financial Cryptography and Data Security, Tenerife, Spain, 25–28 January 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 35–50.
19. Damgård, I.; Fitzi, M.; Kiltz, E.; Nielsen, J.B.; Toft, T. Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation. In Proceedings of the Theory of Cryptography Conference, New York, NY, USA, 4–7 March 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 285–304.
20. Schulte, M.J.; Omar, J.; Swartzlander, E. Optimal initial approximations for the Newton-Raphson division algorithm. *Computing* **1994**, *53*, 233–242. [CrossRef]
21. Marden, J.R.; Walter, S.; Tchetgen Tchetgen, E.J.; Kawachi, I.; Glymour, M.M. Validation of a polygenic risk score for dementia in black and white individuals. *Brain Behav.* **2014**, *4*, 687–697. [CrossRef]
22. Wünnemann, F.; Sin Lo, K.; Langford-Avelar, A.; Busseuil, D.; Dubé, M.P.; Tardif, J.C.; Lettre, G. Validation of genome-wide polygenic risk scores for coronary artery disease in French Canadians. *Circ. Genom. Precis. Med.* **2019**, *12*, e002481. [CrossRef] [PubMed]
23. Xu, J.; Zhang, Z.; Xiao, X.; Yang, Y.; Yu, G.; Winslett, M. Differentially private histogram publication. *VLDB J.* **2013**, *22*, 797–822. [CrossRef]

24. Asharov, G.; Jain, A.; López-Alt, A.; Tromer, E.; Vaikuntanathan, V.; Wichs, D. Multiparty computation with low communication, computation and interaction via threshold FHE. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, 15–19 April 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 483–501.
25. Lattigo v3. EPFL-LDS, Tune Insight SA. 2022. Available online: https://github.com/tuneinsight/lattigo (accessed on 1 May 2022).
26. Consortium, .G.P.; et al. A global reference for human genetic variation. *Nature* **2015**, *526*, 68. [CrossRef]
27. Kathiresan, S.; Melander, O.; Anevski, D.; Guiducci, C.; Burtt, N.P.; Roos, C.; Hirschhorn, J.N.; Berglund, G.; Hedblad, B.; Groop, L.; et al. Polymorphisms associated with cholesterol and risk of cardiovascular events. *N. Engl. J. Med.* **2008**, *358*, 1240–1249. [CrossRef] [PubMed]