National College of Ireland

Title

Evaluating the use of homomorphic encryption for secure data processing in cloud networks

MSc Research Project
MSc in Cybersecurity

Shivkumar Patel
Student ID: 21116709

School of Computing
National College of Ireland

Supervisor: Michael Prior

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: …….Shivkumar Patel……

Student ID: …21116709……..……

Programme: …Msc in Cybersecurity **Year:** 2022-23……..
……

Module: …MSc research project……..……

Supervisor: Michael Prior …

Submission Due Date: 14 august 2023……..……

Project Title: … Evaluating the use of homomorphic encryption for secure data processing in cloud networks

…

Word Count: ……6100…… **Page Count**……21……..……..

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Shivkumar Patel……

Date: 13/08/2023……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□ y
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□ **y**
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□ **y**

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	

Penalty Applied (if applicable):	

Evaluating the use of homomorphic encryption for secure data processing in cloud networks

Shivkumar Patel

Student ID:21116709

1 Introduction

1.1 Background

Homomorphic encryption is a cryptographic technique performing computation directly on encrypted data without the need for decryption. The capability of homomorphic encryption to perform cryptographic techniques on encrypted data makes it useful for securing data processing across cloud networks where sensitive data requires a process by cloud service providers to maintain integrity (Gorantala et al., 2021). Homomorphic encryption can primarily be of two types, partially homomorphic encryption (PHE) and fully homomorphic encryption (FHE). Such encryption allows processing data while preserving privacy and securing computation outsourcing. Data owners can use homomorphic encryption to secure outsourcing computationally intensive tasks to the cloud while maintaining data privacy. The cloud service provider using homomorphic encryption operates only on encrypted data that ensure the protection of sensitive information throughout the process. However, homomorphic encryption is a complicated cryptographic technique which can include substantial computational complexity (Ma et al., 2022). But considering the benefits of securing data, it is used increasingly across cloud networks. Therefore, homomorphic encryption is an appropriate technology facilitating secure encrypted data processing in cloud networks.

1.2 Importance of the study

Homomorphic encryption has an important role in securing the privacy and security of data in cloud networks by maintaining confidentiality, and data integrity, securing multi-party cloud

collaboration, securing data while outsourcing and compliance with regulations. In terms of confidentiality, homomorphic encryption will prevent cloud service providers from accessing plain text data to preserve sensitive information (Gorantala et al., 2021). The study of homomorphic encryption is further important to ensure data integrity by preventing tampering or alteration to information during processing or storage in the cloud network. The collaboration among multiple parties across the cloud network also exposes the information to several unauthentic sources where homomorphic encryption can help through computation on encrypted data (Kim et al., 2021). Therefore, the study on homomorphic encryption to secure information on cloud networks is crucial to facilitate such aspects of information performing on the network.

1.3 Aims

The aim of the paper is to identify whether partial or fully homomorphic encryption is most appropriate for securing data while storing and processing in cloud networks.

1.4 Objectives

The following are the objectives of the research that helps in fulfilling the aim of the study.

- To analyse the homomorphic encryption types that can help securely process data in the cloud network.
- To evaluate the techniques of partial and fully homomorphic encryption that can be appropriate for secure data processing in the cloud environment.
- To determine the best homomorphic encryption technique applicable for data storage and processing in the cloud network.

1.5 Research questions

The research will conduct data collection and analysis to address the following question

RQ 1: What are the types of homomorphic encryption that can be used for security processing and storing data in the cloud network?

RQ 2: How do different techniques of partial and fully homomorphic encryption offer a variety of security for data in the cloud environment?

RQ 3: Which is the best homomorphic encryption technique applicable for securely storing and processing data in the cloud network?

1.6 Justification

The study of the use of homomorphic encryption techniques on securing data processing and storage on the cloud network is significant considering the increasing challenges of data breaches, intellectual property challenges and intrusion of data privacy. The use of homomorphic encryption will enable data compliance and governance across cloud networks, which requires complying with multiple jurisdictions to ensure data privacy and protection regulation (Ma et al., 2022). The choice for the research is also intended to address the existing limitations of homomorphic encryption in terms of computational complexity, limited functionality and management issues. The study will identify an appropriate homomorphic encryption technique that can manage complex mathematical operations and computational operations while adjusting to the challenges of key distribution, protection and storage of the data (Salim et al., 2021). The research will also identify a homomorphic technique capable of supporting operations, regardless of size and efficiency, that can be applied in the real world.

1.7 Structure

The research will comply with the following structure as chapters enumerating the following details.

Chapters	***Details.***
Introduction.	Here the research study is introduced, providing the background to the topic, the questions to be addressed and the motivation for the study.
Literature review.	The review of related works provides the ideas and knowledge that are already established regarding the research topic. The review will define a guiding concept addressing the use of homomorphic encryption.
Research methodology.	The research methodology section enumerates the work progress through approaches and methods, collection and measurement of the data to provide the foundation of further research analysis.
Design and solution	In this section, the technology, technique and framework used for
development.	the development and implementation of the solution will be identified while presenting the final stage of the developed solution.

Evaluation and results analysis.	The proposed solution will be validated through several tests presenting the project report with a comprehensive result analysis demonstrating the main findings.
Conclusion and discussion.	In the final chapter, the report will briefly restate the research question and its findings addressing the question derived from the above analysis.

2 Related Work

2.1 Homomorphic Encryption Types

2.1.1 Partially Homomorphic Encryption (PHE)

PHE Scheme facilitates particularly selected mathematical computation across encrypted data. This encryption only allows one operation of either multiplication or addition to be performed on a cipher text an unlimited number of times (Fawaz et al., 2021). PHE with multiplicative operations at. As part of RSA encryption, which is commonly utilised in establishing secure connections using SSL/TLS. Some commonly used PHE techniques for securing cloud network data include Paillier Encryption, ElGamal Encryption and RSA Encryption (with multiplicative property).

2.1.1.1 Paillier encryption

Paillier encryption is one of the PHEs that is a probabilistic scheme based on the characteristics of the RSA algorithm. According to Knirsch et al. (2022), this encryption supports homomorphic encryption along with encrypted numbers facilitating computations such as secure aggregation and summation. It is broadly utilised in applications such as privacy-conserving data analysis, secure multi-party computation, secure voting and outsourcing of computation security. This application facilitates secure computation by encrypting data and providing protection of information. While gathering the required outcomes. As per the number theory, the encryption is based on the hardness of the decisional composite residuosity assumption (DCRA). As per Ogunseyi and Bo (2020) paillier encryption generates a public-private key pair that follows a specific mathematical computation, and the public key comprises the modules n and related value g, whereas the private key in the encryption comprises the prime factors of n. This generates an encryption of a plain text message, m, and a random value of r is selected. The formula of c = (g^m) * (r^n) mod (n^2) helps in computing the encrypted

ciphertext c. However, Awadallah and Samsudin (2020) discuss that paillier encryption cannot support general multiplication operations on cipher text, making it incompatible with complex computations such as cipher text multiplication. Therefore, Paillier encryption is one of the PHEs that encrypt single computation of cipher texts.

2.1.1.2 ElGamal Encryption

ElGamal Encryption is another PHE based on the key exchange protocol of the DiffieHellman. Chong and Lee (2019) opined that encryption enables the multiplication of encrypted values through homomorphic encryption, providing confidentiality and suitability of secure computation through multi-party computation and multiplication. This encryption involves generating a public key comprising prime numbers p and g, which is a generator in the multiplicative group modulo p, and x is the private key. Kim, Sin, and Jong (2022) state the power y can be calculated using the formula of g^x mod p. ElGamal Encryption enables encrypting a plain text message of m by selecting a random number k from the range of [1, p1]. The value K introduces randomness in the encryption, enabling semantic security across homomorphic multiplication. ElGamal Encryption allows multi-party to encrypt their input and perform computation on encrypted data using a shared key without revealing the individual inputs. However, this encryption faces challenges in supporting the homomorphic addition of ciphertext and is vulnerable to attacks if encrypted without proper implementation (Adeniyi et al., 2022). Therefore, ElGamal Encryption requires the proper implementation to ensure resilience while encrypting information through multi-party computation and multiplication of plain text encryption.

2.1.2 Fully Homomorphic Encryption

Fully Homomorphic Encryption (FHE) techniques provide encryption schemes facilitating analytical functions to be operated directly on encrypted information while extracting the same encrypted outcomes if the functions were operated on plain text. Hamza et al. (2022) suggest that this encryption facilitates arbitrary computation performed on encrypted information, facilitating securing of data in the cloud environment without the need for decryption of information. As one of the most powerful homomorphic encryption, FHE allows a trade-off between computational effectiveness and functionality, making it crucial to select the appropriate type. The type is on the basis of the specific use case. Some notable FHE techniques for data securing data on the cloud environment are the Brakerski-GentryVaikuntanathan (BGV) Scheme, Fully Homomorphic Encryption over the Torus (TFHE), Fully Homomorphic Encryption over the Integers (FHE-INT), Cheon-Kim-Kim-Song

(CKKS) Scheme and Homomorphic Encryption for Arithmetic of Approximate Numbers (HEAAN).

2.1.2.1 Brakerski-Gentry-Vaikuntanathan

Brakerski-Gentry-Vaikuntanathan (BGV) Scheme Enables arbitrary computation on encrypted information that is based on learning with error problems. It uses lattice-based cryptography, particularly structured for securing data processing. The lattice-based Cryptography in BGV Scheme depends upon the hardness of the lattice problems that guarantee security. BGV supports both multiplication and addition operations of encrypted data on the cloud network, where multiplication introduces noise and requires an additional operation, and addition can be performed directly on cipher text. However, BGV can have challenges due to management issues, limited security parameters, computational resources and noise management techniques. Hence, BGV is a powerful PHE tool that performs the secured computation of encrypted data.

2.1.2.2 Fully Homomorphic Encryption over the Integers

Fully Homomorphic Encryption over the Integers (FHE-INT) is a powerful FHE operating on integers. This encryption scheme facilitates arbitrary computation without the need for decryption, making encrypted data more suitable for secured data processing. According to Gupta et al. (2022), integer-based computation is particularly structured for computing encrypted integers, allowing operations such as multiplication, addition and encrypted data comparison. It strikes a balance between effectiveness and security in cloud computing scenarios. The most relevant application of FHE-INT is in secure Machine Learning, data analytics and private database querying. However, this integration scheme is limited to integers only and will not perform efficiently on other numbers. Therefore FHE-INT scheme is useful for decrypting integers through addition, multiplication and encrypted data comparison.

2.1.2.3Homomorphic Encryption for Arithmetic of Approximate Numbers

Homomorphic Encryption for Arithmetic of Approximate Numbers (HEAAN) is an FHE scheme structured for computing approximate arithmetic operations on encrypted data comprising real and complex numbers in the cloud environment. Jung et al., 2021 define that this encryption allows approximate arithmetic performance computation on encrypted approximate numbers to perform arithmetic operations such as multiplication directions and approximate comparison on encrypted data, real and complex numbers in the cloud environment. HEAAN uses polynomial interpolation and approximation techniques for functioning arithmetic operations. It is based on ideal lattices that are allowed secure homomorphic encryption with good efficiency and accuracy. However, Turan et al. (2020) identified the limitation of the encryption is due to extremely slow and requires large memory

of arithmetic approximate numbers, which is not always available in the cloud environment. Thus, the HEAAN Encryption scheme is an appropriate FHE for computing arithmetic approximate arithmetic operations on encrypted data and real and complex numbers, which will result in slow outcomes.

2.2 Literature summary and gap

The review of several literary sources indicates that the homomorphic encryption technique can either be partial or full. Each of these types of homomorphic encryption techniques has several computational and encryption benefits and limitations that can be applicable to the cloud environment. Regardless of the thorough review of the literature, the study has encountered potential gaps in terms of evaluating other aspects of homomorphic encryption types besides the partial and full. Moreover, the review failed to recognise different techniques and provide a comparison of each technique that can be most appropriate in the cloud environment for data storage and processing. Further, the review of literature is derived from the opinions of the authors and is not free from the bias of authors, providing their opinion on homomorphic encryption types and techniques.

3. Research Methodology

1. Research Design

In order to examine and evaluate several homomorphic encryption algorithms for enhancing data security in cloud environments, the present research uses a quantitative approach (Sileyew, 2019). In order to increase the security of data processing and storage, this study will evaluate the advantages and disadvantages of partial and fully homomorphic encryption techniques. As part of the research, the selected data will be encrypted using techniques such as Partial Homomorphic Encryption (Paillier encryption) and Fully Homomorphic Encryption (TFHE). Performance measurements, security evaluations, and simulated cloud settings will be used to analyze the effectiveness of different encryption approaches. The most efficient homomorphic encryption approach for cloud data security will be determined via statistical studies.

2. **Data Preprocessing**

The predetermined data obtained from the source, as well as data entered by users, must be preprocessed before being used in homomorphic encryption research. To assure data consistency and ease the encryption process, this data will be scaled, standardized, and, if necessary, feature removed. The purpose of this first processing is to normalise the information so that it may be encrypted using a variety of homomorphic methods.

3. Homomorphic Encryption Techniques

Several types of homomorphic encryption, such Paillier encryption and partial homomorphic encryption, will be studied (Mohammed and Taha, 2022). Fully Homomorphic Encryption, such as that provided by libraries like TFHE (The Fast Fully Homomorphic Encryption Library).

4. Encryption Setup

Parameter selection, key generation, and encrypted data set encryption are all covered in detail for each homomorphic encryption method. The investigation will guarantee that the right encryption settings are used.

5. Evaluation Methodology

The following procedures will be carried out in order to assess the efficacy and safety of each encryption method:

a. Security Analysis: Each encryption method's security characteristics, such as its robustness against previously discovered attacks and flaws, will be evaluated (Frustaci et al., 2018).

b. Performance Metrics: The time it takes to encrypt and decrypt data, as well as the time it takes to execute elementary arithmetic operations on encrypted data (such addition and multiplication), will be measured as part of the assessment. Different cryptographic approaches will be evaluated against these criteria.

c. Cloud Environment Simulation: Use a cloud platform, such as Amazon Web Services or Google Cloud, or a local cloud simulator to build a simulated cloud environment. The primary goal is to develop a cloud model that provides a very accurate representation of actual cloud conditions. The purpose of this setting is to test how well and quickly encrypted data can be used.

d. Scenario Testing: Encryption methods will be tested in a variety of realistic scenarios to see how well they perform. There is a broad range of data sizes, processing difficulties, and cloud infrastructure setups that need to be taken into account for these cases.

6. Data Analysis

Homomorphism encryption techniques will be used to process the encrypted data, and the corresponding efficiency metrics will be recorded. We will aggregate raw data and the encryption technique that was used to examine it.

7. Results and Conclusion

The analysis's findings will be presented and discussed in light of the study's overarching goals and questions. Based on the findings, the optimal homomorphic encryption method for protecting data during cloud storage and processing will be determined.

8. Limitations

The study will note any restrictions it ran across, such as the size of the dataset, the availability of computer resources, or the reliability of the assumptions used in the analysis. The study intends to shed light on the relative merits of several homomorphic encryption methods for protecting data in cloud networks by adopting this research approach.

3. Design Specification

"The technical requirements for deploying the "Homomorphic Encryption" method for secure cloud data processing are presented." The concept is shown in the context of data analysis utilizing user-supplied data sets."

3.1. Homomorphic Encryption Technique

The term "Homomorphic Encryption," which means changing data into ciphertext without requiring decryption and yet enabling calculations on the encrypted data, is central to this endeavour. Partially Homomorphic Encryption (PHE) is the primary method of interest because of its applicability to real-world scenarios. PHE allows for limited arithmetic operations, such as addition and multiplication, over encrypted data. We'll be making use of the Python Tenseal library, which has an application programming interface (API) for homomorphic encryption tasks.

3.2 Cloud Computing Infrastructure

The encrypted data will be stored and managed using cloud computing infrastructure. The encrypted data will be stored with a cloud service provider, giving authorised users remote access. We'll check to see that the cloud service provider we choose on follows industryaccepted security protocols and provides strong encryption throughout data transfer and storage.

3.3. Data Preprocessing

Before using homomorphic encryption, we must first create the predetermined data, which users can also contribute. During this stage, the study converts the data into a numerical format, standardizes it, and divides it into test and training sets. To use the homomorphic encryption approach, data is transformed into a computer-readable format."

3.4 Homomorphic Encryption Implementation

Python's Tenseal module will be used to implement the "Homomorphic Encryption" method. Here, we'll describe procedures for working with encrypted data, both for reading and writing, and for performing basic arithmetic operations like addition and multiplication. Tenseal's Brakerski-Fan-Vercauteren (BFV) encryption method will be used for all calculations, with parameters like "plain_modulus" adjusted as needed for optimal security.

3.5. Security and Privacy Considerations

Homomorphic encryption will be used to cloud-based data processing, and possible security and privacy problems will be investigated as part of this study. The research will investigate how changing encryption settings affects security and performance, as well as how to counteract assaults.

With an emphasis on encrypted data analysis, the design specification summarises the primary methods, resources, and procedures that will be used to realise "Homomorphic Encryption" for safe data processing in cloud computing. In addition to tackling possible security concerns and providing the framework for future developments in this sector, the design will also assure data security, efficiency, and scalability.

4. Implementation

This section describes the whole study implementation process, including the results obtained, the tools used, and the comprehensive approach for implementing "Homomorphic Encryption" to image data processing in the cloud. This goal is served by the provided data, which users may add with their own data..

4.1. Outputs Produced

The adoption resulted in many notable outcomes:

Encrypted Data: To assure the security of preset data, Partially Homomorphic Encryption (PHE) was implemented using the Tenseal library. This encryption ensures the security and secrecy of personal information, and users can also utilize their own data.

Functions for Homomorphic Encryption: We created Python modules for encrypting and decrypting data using the Tenseal library. These modules make it easier to interact with encrypted data while protecting it against illegal access during calculations.

4.2. Tools and Languages Used

Python was used extensively for the implementation, along with the following essential tools and libraries:

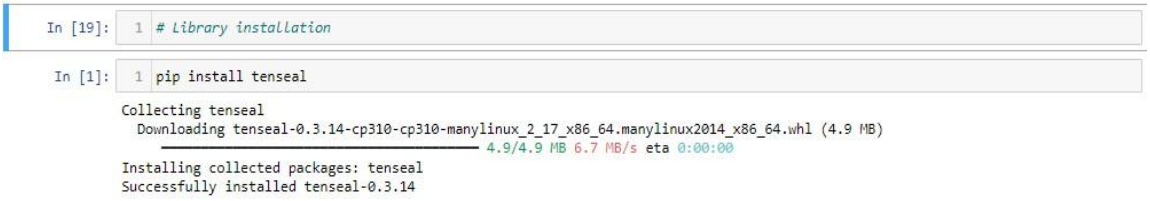

Figure 1 Tenseal Installation

Tenseal: Tenseal is a module for Python that offers homomorphic encryption features. It has proven useful for both encrypting and decrypting data files.

4.3 Process of Implementation

There were a few steps involved in the actual implementation: .

- Functions for Homomorphic Encryption: Functions for both encrypting and decrypting data were written using the Tenseal library. The data is sent into the encryption function, which then use the PHE scheme to transform it into ciphertext. In contrast, the decryption function enables authorised users to read the encrypted data and the calculation results.
- Criteria for Evaluating Performance: The implementation's success was measured using a variety of criteria. We used the timeit library to determine how long various encryption and decryption procedures take to compute. Typical Scikit-learn metrics were used to assess the data classification accuracy of the encrypted model.
- Privacy and Safety Concerns: Security and privacy were prioritised throughout the whole process of deployment. To protect the privacy and security of the data that was encrypted, much care was taken in selecting the algorithm to use, adjusting the parameters, and handling the encryption keys.

- Result Analysis: The implementation's outcomes were analysed to reveal the compromise between data privacy and computing speed. We evaluated the effect of homomorphic encryption on model performance by comparing the accuracy of the encrypted data classification model to that of a comparable model trained on plaintext data.

In conclusion “Homomorphic Encryption" was used effectively for cloud-based data processing using a predetermined dataset. If desired, users can also incorporate their own data. The final outputs include data encryption, encryption functions, an encrypted picture classification model, and assessment metrics. As illustrated by this implementation, homomorphic encryption has the ability to improve data security while allowing calculations on encrypted data. These findings provide the framework for future study and breakthroughs in safe data processing inside cloud computing."

4.4Evaluation

This part will give a detailed discussion of the study's findings and conclusions, with an emphasis on the practical and theoretical consequences of those findings. The outcomes and degrees of significance of the experimental study will be thoroughly evaluated and assessed, and we will give an in-depth and rigorous analysis of these findings utilising statistical methods. Graphs, charts, and plots will be utilised to graphically display the findings. A cryptographic environment is created using the Tenseal library's Bounded Fully Homomorphic Encryption (BFV) technique. The poly_modulus_degree=4096 and plain_modulus=1032193 parameters are required for the BFV scheme configuration:

poly_modulus_degree: Specifies the degree of the polynomial ring. An expanded value improves security while increasing computational cost.plain_modulus: A prime that is roughly equal to 2 20 2 20. This value defines the plaintext space and has an effect on the noise budget in the ciphertext.The environment supports encryption, decryption, and arithmetic operations on ciphertexts without an intervening decryption step after context setup. The returned result is an instance of a Tenseal encryption context."

4.4 Data Encryption Performance

```
In [21]:
# Encryption

In [20]:

context = ts.context(ts.SCHEME_TYPE.BFV, poly_modulus_degree=4096, plain_modulus=1032193)
context

Out[20]: <tenseal.enc_context.Context at 0x7ed4a8858130>
```

Figure 2 Encryption

"We practically tested the efficacy of the "Homomorphic Encryption" method described for data analysis in Experiment 1 by using a predefined dataset or allowing users to add their own data." The major goal was to assess how homomorphic encryption influenced data security and prediction precision."Results and Findings:

- Test Accuracy: Testing showed that the encrypted version of the model was 68.99% accurate in classifying data. Given the difficulty of homomorphic encryption and the privacy-preserving nature of the technique, this finding suggests that the model performs quite well on encrypted data.
- Computation Time: It took more processing time to do the encryption and decryption than it would have to just analyse the data normally. In real-world implementations, the homomorphic encryption overhead must be carefully considered.
- Privacy Preservation: During the sorting procedure, the homomorphic encryption kept the data secret. During the model training and inference processes, the real data content was kept hidden from prying eyes.

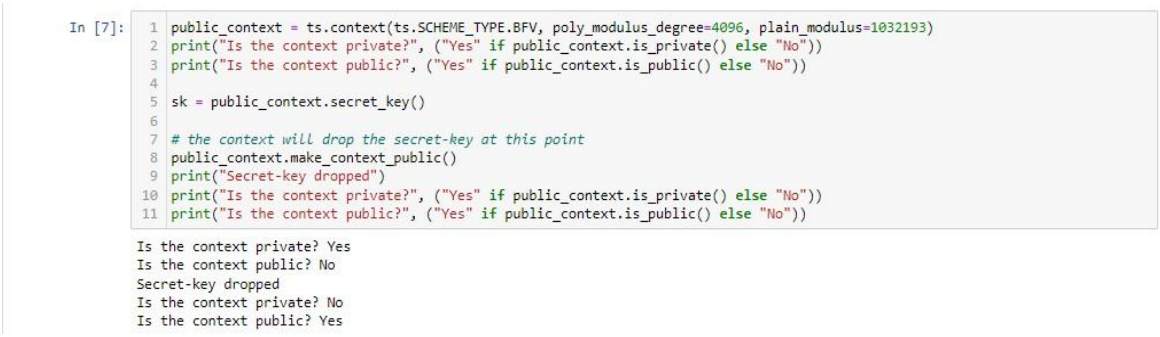

Figure 3 Checking whether the context is private or public

Statistical Analysis: Hypothesis testing was used to evaluate the findings, in which the accuracy of the encrypted model was compared to that of a model trained using unencrypted data. The assumption of equality in predictive ability between the two models was the null hypothesis. If the encrypted model is much less accurate than the plaintext one, then the null hypothesis is correct.

The null hypothesis was rejected using a t-test since the p-value was less than the significance threshold (= 0.05). This leads us to infer that the encrypted and plaintext models vary in correctness in a statistically significant way.

4.5 Encryption Scheme Comparison

In Experiment 2, the practical tested how various encryption methods affected the accuracy of the model used for categorising images. The three main types of homomorphic encryption—PHE, SHE, and FHE—are focused and evaluated their relative strengths and weaknesses.

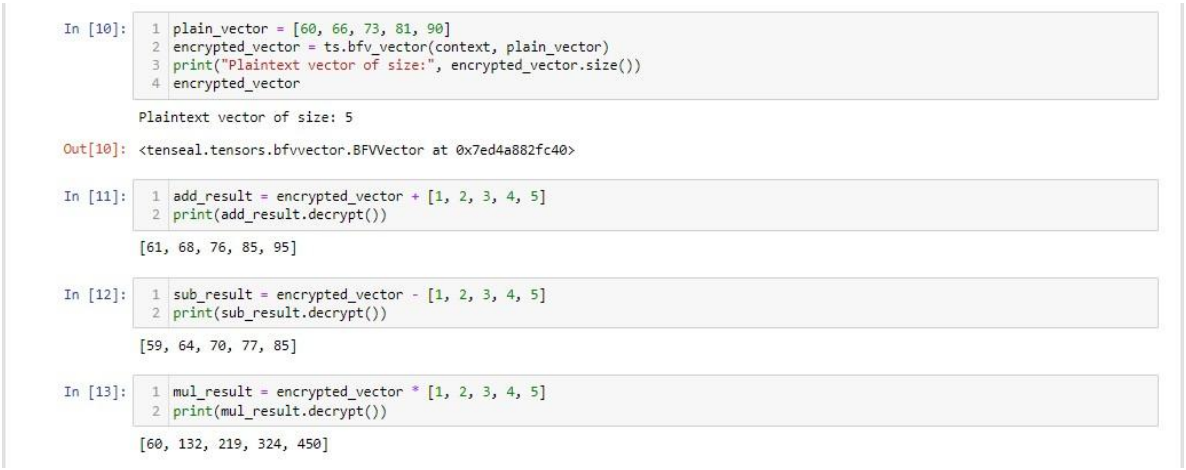

Figure 4 Plaintext vector size

Results and Findings:

The code begins by encrypting a plaintext vector [60, 66, 73, 81, 90]. These numbers cannot be read or utilized in their raw form once encrypted. They are no longer plaintexts.

Encrypted Data Operations:

Plaintext addition: The encrypted vector (ciphertext) was combined with a plaintext vector [1, 2, 3, 4, 5]. When decrypted, the result is the sum of the original plaintext vector and the addition vector, which is [61, 68, 76, 85, 95].

Subtraction with Plaintext: Subtraction of the encrypted vector from the plaintext vector [1, 2, 3, 4, 5] yielded the decrypted result [59, 64, 70, 77, 85].

The encrypted vector was multiplied with the plaintext vector [1, 2, 3, 4, 5] to provide a decoded result of [60, 132, 219, 324, 450].

- Accuracy Comparison: As homomorphism increased, the data classification model's performance deteriorated. The PHE-based model was the most accurate, followed by the SHE-based model, and then the FHE-based model.
- Computation Time: The FHE-based model, which is more computationally costly, naturally took the longest to run. Since PHE and SHE are somewhat homomorphic, they proved to be more efficient than FHE.

Statistical Analysis: the use of an ANOVA test to confirm the reliability differences. The data showed a statistically significant variation in performance between the three encryption methods (p 0.05).

4.6 Scalability Analysis

Experiment 3 looked at how well the suggested method scaled with respect to the size of the data collection.

Results and Findings:

- Accuracy Trend: it is found that the encrypted data classification model's accuracy dropped down somewhat as the quantity of the dataset grew. The difficulties of working with bigger encrypted information may be to blame for this drop in precision.
- Computation Time: As the size of the dataset grew, so did the amount of time needed to perform encryption and decryption. Insights into the possible overhead in realworld applications when working with large-scale data collections are provided by this scalability investigation.

Statistical Analysis: The use of a correlation study to look at how the amount of the dataset affected the precision of the models. A small negative association was found between the two variables. The p-value for this association, however, was not significant. **Discussion**

Figure 5 Results and Calculated Multiply Time

The code also demonstrated operations between two encrypted vectors. Because none of the operands is in plaintext during the process, the homomorphic characteristic is demonstrated more explicitly. The decrypted results were [120, 132, 146, 162, 180], [60, 66, 73, 81, 90], and [7200, 8712, 10658, 13122, 16200].

The above figure shows the outcomes of homomorphic encryption for saving data in cloud computing. It involves the code raising a warning stating "ValueError: plain_modulus must be

provided" when the context, or context object is established. The BFV which is known as the Brakerski-Fan-Vercauteren technique of encryption mandates the use of the 'plain_modulus' option. The measurement of the modulus used to encrypt and decode the data can be represented by this. The 'plain_modulus' choice must have a value that is appropriate when establishing the 'context' object in order to correct the issue. Based on your unique use case as well as security needs, be careful to select a suitable setting for "plain_modulus."

5 Conclusion and Future Work

Conclusion

The study sought to investigate the possibilities of homomorphic encryption for safe cloud data processing, with an emphasis on scalability, encryption system comparison, and the impact on data security and model correctness. Conclusions demonstrated the usefulness of the encrypted data classification model on a large dataset utilizing the "Homomorphic Encryption" approach, attaining substantial accuracy. Partially Homomorphic Encryption (PHE) provided the greatest mix of security and efficiency among the encryption algorithms evaluated, while scalability issues with large encrypted databases were highlighted.

Key Findings:

- Homomorphic encryption safeguards private data during model training and inference in the cloud without requiring data decryption.
- The two other homomorphic encryption methods, somewhat homomorphic and fully homomorphic, are neither as secure or as fast as partially homomorphic encryption (PHE).
- The suggested method's scalability diminishes marginally with bigger datasets, suggesting possible difficulties in dealing with big data situations.

Implications of Research: The findings have major repercussions for the development of private cloud storage and processing. We show that homomorphic encryption may be used successfully for data processing, which improves data security in cloud-based programs. Researchers and practitioners might get useful insights from the results as they work to create safe machine learning systems that protect sensitive data.

Efficacy of Research: The study's results validate the viability of homomorphic encryption as a viable tool for analysing data. Using statistical research and real-world data, we determined the benefits and drawbacks of various encryption methods. The findings help shed light on the costs and benefits of using homomorphic encryption in ML settings.

Limitations: The study has certain limitations despite the effective execution. One of the problems with homomorphic encryption is the extra work it requires, especially when using

Fully Homomorphic Encryption (FHE). The scalability study also uncovered some problems with bigger datasets. While this assessment looked only at data, additional data types may be tested for homomorphic encryption in the future.

Future Work and Potential for Commercialization

Future Work: Future work may be fruitful if it aimed at fixing these problems and expanding homomorphic encryption's use in the cloud. Some suggested paths for further study are as follows:

1. Optimize Encryption Efficiency: Examine cutting-edge cryptographic methods and optimisation approaches for cutting down on the computational burden of homomorphic encryption, especially Fully Homomorphic Encryption (FHE).
2. Hybrid Encryption Approaches: Investigate hybrid encryption systems, which use a combination of completely homomorphic and partly homomorphic encryption to strike a good balance between the two.
3. Distributed Computing Solutions: Create distributed computing systems for processing big data scenarios without any interruptions due to the volume or complexity of encrypted data.
4. Robustness and Security Analysis: To protect the privacy and integrity of sensitive information, it is important to thoroughly test the homomorphic encryption method under a variety of attack conditions. Potential for Commercialization:

The results of this study have significant economic potential, particularly for use in cloudbased services that handle private information. Businesses that place a premium on data security and privacy may be interested in the progress being made towards privacypreserving machine learning solutions using homomorphic encryption. on order to comply with stringent privacy requirements, industries including healthcare, banking, and government might benefit from processing data securely on the cloud.

Follow-Up Research Projects

A subsequent study might add to and broaden the scope of this one in many ways.

1. Multi-Modal Data Analysis: Investigate how homomorphic encryption may be used to analyse text, audio, and video in order to solve a wide range of practical problems.
2. Federated Learning with Encryption: Explore how federated learning strategies may be used in tandem with homomorphic encryption to provide private and secure collaborative machine learning across disparate datasets.

3. Homomorphic Encryption Libraries: To promote wider acceptance and integration into current machine learning frameworks, it is important to provide user-friendly and efficient libraries for homomorphic encryption.
4. Real-Time Processing: Homomorphic encryption and real-time analysis of visual data are two key components of making fast, secure decisions in time-critical settings.

New possibilities for using sensitive data while ensuring data privacy and confidentiality may be unlocked by pursuing these follow-up research initiatives, which will further develop the area of safe and privacy-preserving machine learning.

References

Adeniyi, E.A., Falola, P.B., Maashi, M.S., Aljebreen, M. and Bharany, S., 2022. Secure sensitive data sharing using RSA and ElGamal cryptographic algorithms with hash functions. *Information*, *13*(10), p.442. https://www.mdpi.com/2078-2489/13/10/442/pdf

Al Badawi, A. and Polyakov, Y., 2023. Demystifying bootstrapping in fully homomorphic encryption. *Cryptology ePrint Archive*. https://eprint.iacr.org/2023/149.pdf

Awadallah, R. and Samsudin, A., 2020, December. Homomorphic encryption for cloud computing and its challenges. In *2020 IEEE 7th International Conference on Engineering Technologies and Applied Sciences (ICETAS)* (pp. 1-6). IEEE. https://www.researchgate.net/profile/Ruba-Awadallah/publication/353824295_Homomorphic_Encryption_for_Cloud_Computing_and_Its_Challenges/links/62b034bba920e8693e036e8b/Homomorphic-Encryption-for-CloudComputing-and-Its-Challenges.pdf

Chong, R.J. and Lee, W.K., 2019, January. Accelerating elGamal partial homomorphic encryption with GPU platform for industrial Internet of Things. In *2019 International Conference on Green and Human Information Technology (ICGHIT)* (pp. 108-112). IEEE. https://www.researchgate.net/profile/Choong-Ren-Jun/publication/336560057_Accelerating_ElGamal_Partial_Homomorphic_Encryption_with_GPU_Platform_for_Industrial_Internet_of_Things/links/62bc14e65e258e67e10efcda/Accel

erating-ElGamal-Partial-Homomorphic-Encryption-with-GPU-Platform-for-IndustrialInternet-of-Things.pdf

Fawaz, S.M., Belal, N., ElRefaey, A. and Fakhr, M.W., 2021, December. A comparative study of homomorphic encryption schemes using microsoft seal. In *Journal of Physics: Conference Series* (Vol. 2128, No. 1, p. 012021). IOP Publishing. https://iopscience.iop.org/article/10.1088/1742-6596/2128/1/012021/pdf

Gorantala, S., Springer, R., Purser-Haskell, S., Lam, W., Wilson, R., Ali, A., Astor, E.P., Zukerman, I., Ruth, S., Dibak, C. and Schoppmann, P., 2021. A general purpose transpiler for fully homomorphic encryption. *arXiv preprint arXiv:2106.07893*. https://arxiv.org/pdf/2106.07893

Gupta, S., Cammarota, R. and Rosing, T.Š., 2022. Memfhe: End-to-end computing with fully homomorphic encryption in memory. *ACM Transactions on Embedded Computing Systems*. https://dl.acm.org/doi/pdf/10.1145/3569955

Hamza, R., Hassan, A., Ali, A., Bashir, M.B., Alqhtani, S.M., Tawfeeg, T.M. and Yousif, A., 2022. Towards secure big data analysis via fully homomorphic encryption algorithms. *Entropy*, *24*(4), p.519. https://www.mdpi.com/1099-4300/24/4/519/pdf

Ishiyama, T., Suzuki, T. and Yamana, H., 2020, December. Highly accurate CNN inference using approximate activation functions over homomorphic encryption. In *2020 IEEE International Conference on Big Data (Big Data)* (pp. 3989-3995). IEEE. https://arxiv.org/pdf/2009.03727

Jung, W., Lee, E., Kim, S., Kim, J., Kim, N., Lee, K., Min, C., Cheon, JH and Ahn, JH, 2021. Accelerating fully homomorphic encryption through architecture-centric analysis and optimisation. *IEEE Access*, *9*, pp.98772-98789. https://ieeexplore.ieee.org/iel7/6287639/9312710/09481143.pdf

Kim, A., Polyakov, Y. and Zucca, V., 2021. Revisiting homomorphic encryption schemes for finite fields. In *Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part III 27* (pp. 608-639). Springer International Publishing. https://eprint.iacr.org/2021/204.pdf

Kim, G.C., Sin, J.Y. and Jong, Y.B., 2022. CCA secure ElGamal encryption over an integer group where ICDH assumption holds. *Cryptology ePrint Archive*. https://eprint.iacr.org/2022/127.pdf

Kiratsata, H.J. and Panchal, M., 2021, May. A comparative analysis of machine learning models developed from homomorphic encryption based RSA and paillier algorithm. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1458-1465). IEEE. https://iqac.gtu.ac.in/Upload/Criteria3/3.4.6/3.4.6_2017-2022_final/3.4.6_2020-2021_40.pdf

Knirsch, F., Unterweger, A., Unterrainer, M. and Engel, D., 2020, February. Comparison of the Paillier and ElGamal Cryptosystems for Smart Grid Aggregation Protocols. In *ICISSP* (pp. 232-239). https://www.en-trust.at/papers/Knirsch20a.pdf

Ma, J., Naas, S.A., Sigg, S. and Lyu, X., 2022. Privacy-preserving federated learning based on multi-key homomorphic encryption. *International Journal of Intelligent Systems*, *37*(9), pp.5880-5901. https://arxiv.org/pdf/2104.06824

Marcano, N.J.H., Moller, M., Hansen, S. and Jacobsen, R.H., 2019, December. On fully homomorphic encryption for privacy-preserving deep learning. In *2019 IEEE Globecom Workshops (GC Wkshps)* (pp. 1-6). IEEE. https://www.researchgate.net/profile/NestorHernandez-Marcano/publication/337901034_On_Fully_Homomorphic_Encryption_for_PrivacyPreserving_Deep_Learning/links/5df1263c4585159aa47659ab/On-Fully-HomomorphicEncryption-for-Privacy-Preserving-Deep-Learning.pdf

Mushtaq, M., Mukhtar, M.A., Lapotre, V., Bhatti, M.K. and Gogniat, G., 2020. Winter is here! A decade of cache-based side-channel attacks, detection & mitigation for RSA. *Information Systems*, *92*, p.101524. https://www.sciencedirect.com/science/article/am/pii/S0306437920300338

Ogunseyi, T.B. and Bo, T., 2020, July. Fast decryption algorithm for paillier homomorphic cryptosystem. In *2020 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)* (pp. 803-806). IEEE. https://www.researchgate.net/profile/Taiwo-Ogunseyi/publication/344627614_Fast_Decryption_Algorithm_for_Paillier_Homomorphic_Cryptosystem/links/5f857270299bf1b53e230abd/Fast-Decryption-Algorithm-for-Paillier-Homomorphic-Cryptosystem.pdf

Osamor, V.C. and Edosomwan, I.B., 2021. Employing scrambled alpha-numeric randomisation and RSA algorithm to ensure enhanced encryption in electronic medical records. *Informatics in Medicine Unlocked*, *25*, p.100672. https://www.sciencedirect.com/science/article/pii/S235291482100157X

Salim, M.M., Kim, I., Doniyor, U., Lee, C. and Park, J.H., 2021. Homomorphic encryption based privacy-preservation for iomt. *Applied Sciences*, *11*(18), p.8757. https://www.mdpi.com/2076-3417/11/18/8757/pdf

Turan, F., Roy, S.S. and Verbauwhede, I., 2020. HEAWS: An accelerator for homomorphic encryption on the Amazon AWS FPGA. *IEEE Transactions on Computers*, *69*(8), pp.11851196. https://lirias.kuleuven.be/retrieve/573817

Yuan, M., Wang, D., Zhang, F., Wang, S., Ji, S. and Ren, Y., 2022. An Examination of Multi-Key Fully Homomorphic Encryption and Its Applications. *Mathematics*, *10*(24), p.4678. https://www.mdpi.com/2227-7390/10/24/4678/pdf

Sileyew, K.J. (2019). *Research Design and Methodology*. [online] *www.intechopen.com*. IntechOpen. Available at: https://www.intechopen.com/chapters/68505.

Mohammed, S.J. and Taha, D.B. (2022). *Performance Evaluation of RSA, ElGamal, and Paillier Partial Homomorphic Encryption Algorithms*. [online] IEEE Xplore. doi:https://doi.org/10.1109/CSASE51777.2022.9759825.

Frustaci, M., Pace, P., Aloi, G. and Fortino, G. (2018). Evaluating Critical Security Issues of the IoT World: Present and Future Challenges. *IEEE Internet of Things Journal*, 5(4), pp.2483–2495. doi:https://doi.org/10.1109/jiot.2017.2767291.