SEMARAK ILMU PUBLISHING

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Privacy Preserving Image Retrieval Using Multi-Key Random Projection Encryption and Machine Learning Decryption

Alaa Mahmoud Ibrahim[1,*], Mohamed Farouk[2], Mohamed Waleed Fakhr[3]

[1] College of Computing and Information Technology, Arab Academy for Science, Technology and Maritime Transport (AAST), Heliopolis, Cairo, Egypt
[2] College of Computing and Information Technology, Arab Academy for Science, Technology and Maritime Transport (AAST), Abu Qir, Alexandria, Egypt
[3] Department of Computer Engineering, Arab Academy for Science, Technology and Maritime Transport (AAST), Heliopolis, Cairo, Egypt

ARTICLE INFO

Article history:
Received 18 August 2023
Received in revised form 21 November 2023
Accepted 12 February 2024
Available online 3 April 2024

Keywords:
Privacy Preservation; Random Projection; Machine Learning

ABSTRACT

Homomorphic Encryption (HE), Multiparty Computation (MPC), Differential Privacy (DP) and Random Projection (RP) have been used in privacy preserving computing. The main benefit of the random projection approach is the lighter time and space complexity compared to the other available techniques. However, RP is typically used in a symmetric encryption mode, with one random projection matrix single key, making it vulnerable to attacks. An enhanced multi-key RP approach is proposed in this paper where a set of N random matrices are used as projection keys. Moreover, a randomly chosen one is used for each new query. Machine learning models are trained to perform specific vector operations on the randomly projected vectors and produce another randomly projected results vector. Another machine learning model is trained to decrypt the final result at the user's side. The proposed system is shown to offer privacy against known plaintext and cipher-only attacks while preserving Euclidean distance calculations accuracy in the randomly projected domain which are demonstrated on the COREL 1K image retrieval task. Results show that the cyphertext space took sixteen times less than the ciphertext done with homomorphic encryption, and the computation of distance using random projection was 8 times faster than homomorphic encryption distance calculation.

1. Introduction

Privacy preserving computing has become an important feature in various domains such as smart homes [1], autonomous vehicles [2], smart grids [3], portable medical information [4], and decision support systems [5]. Data is processed on cloud as it is capable of performing the required amount of computations with minimum latency, however, on the cost of potential privacy leakage [6].

To address this concern, privacy-preserving techniques mostly employ one of 4 approaches: Homomorphic Encryption (HE) [7-10], Multiparty Computation (MPC) [11], Differential Privacy (DP)

* *Corresponding author.*
E-mail address: alaa.mahmoud@aast.edu

https://doi.org/10.37934/araset.42.2.155174

[12,13], and Random Projection (RP) [10-12]. These techniques are sometimes combined for the purpose of achieving the needed privacy result.

Homomorphic Encryption (HE) [14,15] is a secure computing technique that allows computations to be performed on encrypted data, without first decrypting the data. The main idea behind homomorphic encryption is to use cryptography to create encrypted representations of data that can be manipulated in a meaningful way, without having to reveal the data [16]. The HE approaches guarantees privacy on all levels (input data, model parameters and output); however, it has significant drawbacks in terms of the high storage and large encryption delay [16]. HE can support addition and multiplication operations; a Homomorphic Encryption that supports both operations with unlimited balance called Fully Homomorphic Encryption scheme (FHE) [14,15]. Multiplication operation causes more noise to the ciphertext than addition. Using HE to encrypt training data requires expressing the data as a low degree polynomial [17].

Multiparty Computation (MPC) is a secure computing technique in which multiple parties can collaborate to perform a joint computation, without revealing their individual inputs to each other. The main idea is to encrypt the data to ensure that the computation is performed in such a way that no single party can see the inputs of the others, and the final result is revealed only after the computation is complete. Multi-party computation is useful in situations where multiple parties have sensitive data that they want to use in a joint computation, but do not trust each other to reveal their data. This technique is commercially ready and highly accurate, but there may exist corrupted parties who try to discover other parties' inputs. It also has high communication complexity.

Differential Privacy (DP) is a data perturbation method that provides privacy by "adding noise" [18]. DP introduces randomness to preserve the data and thus suffers from potential accuracy problems as well as attacks vulnerability [12]. It provides mathematical guarantees [19] about the privacy of individual data items in a dataset. The main idea is to add random noise to the data in such a way that individual data items cannot be easily distinguished from the noise, while still allowing meaningful statistical analysis to be performed on the overall dataset. Differential privacy is widely used in areas such as medical research [20], where it is important to protect the privacy of individual patients while still allowing meaningful insights to be gained from the data. However, the DP approach still suffers from potential accuracy problems as well as attacks vulnerability [12].

Random Projection (RP) and Compressed Sensing (CS) have been used as privacy preserving techniques [21-26] due to being lightweight compared to other techniques. In this paper, CS is considered as a special case of the more general RP approach and the term RP will be used in the paper for both of them. Random projection is a secure computing technique that involves transforming data into a lower-dimensional, and in this research a higher-dimensional, space in a random way. The main idea is to use random projections to obscure the structure of the original data, making it more difficult for an attacker to obtain meaningful information about the data. Random projection can be used in combination with other privacy-preserving techniques, such as differential privacy, to provide additional protection for sensitive data [27]. RP works by projecting any given feature vector by a random matrix. If the feature vector is sparse and its dimension is reduced, RP is performing compressed sensing (CS). Otherwise, it is a general-sense random projection.

There are a few weaknesses in the RP approach in privacy preserving computing; Firstly, it is a single-key, symmetric encryption paradigm; thus, both the owner and the user have to share the same key (random projection matrix). This would compromise the secrecy and privacy of the data. Secondly, a semi-honest cloud or a malicious adversary could ultimately decrypt the data and make sense of the results if it accumulates a sufficient number of encrypted data [28,29].

A multi-key RP system is proposed in this paper, maintaining the light-weight features of the RP while enhancing its security by using a potentially very large number of random keys. This is done by

employing randomly selected keys for the users and cloud, taken from a finite set of random matrices. Also, by using two trained machine learning model banks; one to do the encrypted computations at the cloud and the other to decrypt the computation results at the legitimate user side.

The paper is organized as follows: Section 2 background and related work. Section 3 explains the details of the proposed model. Followed by section 4 that covers the security analysis of the proposed model. Section 5 show the experimental results, and finally discussion is on section 6, followed by the conclusion.

2. Background and Related Work

2.1 Comparison of Private Computing Methods

Homomorphic encryption strength lays in its ability to perform computations on encrypted data, which provides the needed privacy for sensitive data, and it could also be used on a variety of use cases. However, it is computationally expensive, and it provides a limited functionality compared to the ones that could be done on unencrypted data. HE provides a tighter security guarantee, as it allows computations to be performed on encrypted data without first revealing the data itself. This makes it more difficult for an attacker to obtain sensitive information about the data. However, the tightness of security in homomorphic encryption depends on the strength of the encryption scheme and the ability to correctly implement the homomorphic encryption. If the encryption scheme is weak or the implementation is incorrect, the security guarantee can be compromised.

MPC provides privacy for individual inputs, it could be applied on multiple use cases as well, and it could handle computations with multiple parties. On the other hand, it could cause communication overhead, could be difficult to coordinate between multiple parties, and could be vulnerable to rogue parties who do not follow the protocol. MPC provides a tight security guarantee by allowing computations to be performed on private data without revealing it to any of the parties involved. In a multiparty computation, each party inputs their private data and the computation is performed in such a way that no party can learn any information about the private data of another party. This makes it difficult for an attacker to obtain sensitive information about any individual's data. However, the tightness of security in multiparty computation depends on the ability to coordinate between the parties involved and the trustworthiness of the parties, as a rogue party could potentially manipulate the computation to obtain sensitive information.

DP represents the mathematical guarantees of privacy, it could handle large datasets, and could be applied to multiple use cases. But it could result in significant loss of information [30], it could also be computationally expensive in some cases [31], could be difficult to tune the amount of privacy protection, Vulnerable to attacks that try to infer individual data items from the noise added by differential privacy [30]. DP provides a strong mathematical guarantee of privacy, but the tightness of this guarantee can depend on the amount of noise added to the data and the choice of privacy parameters. The privacy guarantee in DP is usually expressed as a bound on the likelihood of an attacker being able to determine the presence or absence of an individual data item in a dataset. The tighter this bounds, the stronger the privacy guarantee, but this can come at the cost of adding more noise to the data and reducing the accuracy of the results.

RP can obscure the structure of data [32] and can be computationally efficient. And similar to DP it can result in significant loss of information, and it can be vulnerable to attacks that try to infer information about the original data. RP uses a random projection matrix to map the data to a lower or a higher dimensional space. The projection matrix is not known to the attacker and provides a level of security for the data, so the computations could be performed securely.

Each of the above techniques has its own strengths and weaknesses, and the best method for a particular use case will depend on the specific requirements and constraints of the situation. For example, if privacy is a primary concern, differential privacy or fully homomorphic encryption might be the best approach, while if the goal is to perform a joint computation with multiple parties, multi-party computation might be the best choice, RP and CS shine in their fast performance and being lightweight.

2.2 Related Work

This section aims to cover the work done using private computing methods with machine learning, private computing methods with multikey encryption, and covering some of research that used random projection and compressed sensing for privacy preserving private computing.

Hesamifard *et al.,* [33] presented a framework to train deep neural networks on encrypted data using HE within its provided limitations. The limitations provided by the Fully Homomorphic Encryption (FHE) result in the direction of having a hybrid system that uses at least two of the commonly used privacy preservation techniques. Following this track some researchers lean into not using DP as the sole privacy preserving technique, as it would lack the robustness of the needed privacy protection. Owusu *et al.,* [34] has developed a privacy preserving deep neural network using secure MPC and DP fusion due to the impracticability of the FHE, their approach is secure with a good performance.

Multikey encryption also poses as a further security enhancement. Li *et al.,* [35] has proposed a collaborative multi-key scheme to train a neural network for multiple data owners on an untrusted cloud platform. They proposed the system on a basic model using only HE, then used both HE and MPC as an advanced model. Their system shows it provides the required security for the encrypted data. Similarly, López *et al.,* [36] has introduced an MPC scheme that encrypts the system with multi-key FHE, the ciphertext is processed then deciphered using a secret key of the user who issued the operation. Likewise, Mukherjee *et al.,* [37] has proposed a multiparty computation protocol using multi-key HE. The scheme developed by Li *et al.,* [31] reached the goal of private computing without any privacy leakage by combining MPC and DP with multiple public key encryption. The presented model provides simpler computation complexity, less spatial expansion, and highly accurate results.

On the other hand, the almost negligible processing needed for encryption using RP and CS makes it a promising approach for privacy preservation. Outsourcing images to the cloud for private data mining and image retrieval was used such as presented by Wang *et al.,* [22] and Boyle *et al.,* [32].

Random Projection was used by Peng *et al.* [38] to assist in protecting a biometric cryptosystem. The concept is to derive for each user's feature vector a projection of random space from a projection matrix, they also used backpropagation neural network to link the projected vector with a randomly generated key. This approach secured the vulnerability of cross matching attack. Jiang *et al.,* [39] has introduced a lightweight privacy preserving approach to using gaussian projections on IoT objects, they use deep learning on projected data to train the classifier. Ratra *et al.,* [25] has used dimensionality reduction as a privacy preserving technique along with feature selection. Both random projection and principal component analysis hybrid approach results in a more improved overall performance and classification.

Privacy preserving encryption using compressed sensing was used [23]. The researchers use compressive sensing to encrypt sensitive data, their targeted data were videos, they return the privacy preserved result based on the user level, if the user is not fully authorized the encryption will de-identify the image (face section) in the video, depending on the user's clearance level. Compressed sensing encryption helped their scheme to be reversible i.e., their scheme could

navigate the users from decrypted image to de-identified one depending on their level of clearance back and forth, and it has the benefits of CS as a lightweight encryption compared to HE.

Similarly, Kuldeep *et al.,* [24] has introduced multi-class privacy preserving cloud computing scheme, using CS they have two user levels a super-user who could access sensor data and another level who could only retrieve reports and some statistical information done on the encrypted data, using CS signal recover, with an advantage of lower computational complexity.

Fakhr and Mohamed [21] has developed a multiple-key random projection model which uses machine learning for results decryption, depicted in Figure 1. y_u denotes user's encrypted vector while y_o denotes data owner's randomly encrypted vectors using Φ_u & Φ_o respectively. The Support Vector Machine for Regression 1 (SVR1) module at the cloud produces an encrypted output using their own cloud key Φ_c while the (SVR2) at the user's side reveals the plaintext computation result. The paper introduced random projection with Machine Learning to solve the high computation problem of the HE and MPC since it requires matrix multiplication at encryption and support vector regression inference at decryption. The paper tested the proposed algorithm on two applications, Squared Euclidean Distance computation on COREL 1K image dataset and data prediction Time Series using Google Stock data. The results show a negligible difference between predicted data and plaintext results. The main drawback is that using fixed keys to train the SVR modules is vulnerable to attacks, and if the keys were to be changed or refreshed would require frequent retraining to the SVR modules which poses a scalability problem.

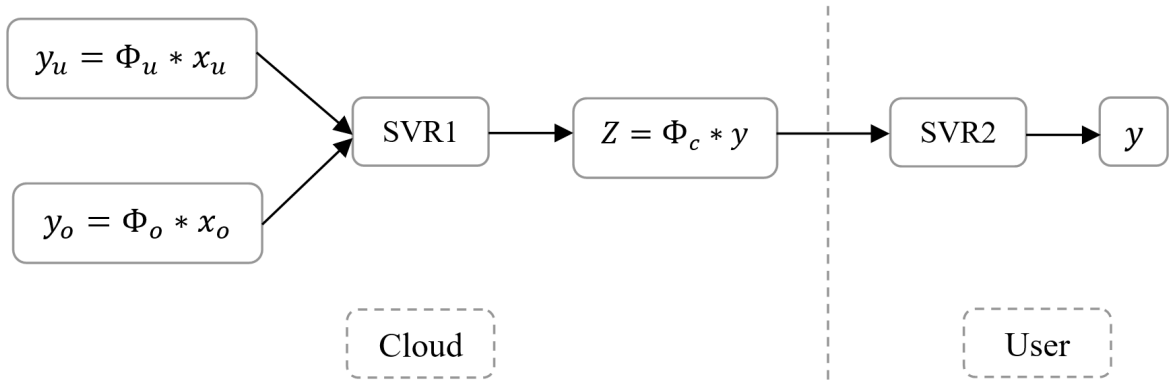

Fig. 1. Private computing using multiple keys and compressed sensing developed by [21] u denotes data user, o denotes data owner, c denotes the cloud

3. Proposed Private Computing Model

3.1 Basic Idea

Building on the work by Fakhr and Mohamed [21], this paper proposes a multi-key system, where both the user and the cloud encrypt their data vectors using randomly selected matrix from a potentially very large set of random matrices.

Figure 2 highlights the proposed model. User 1 is the one who initiates the computation and requires the final outcome, and User 2 is the database owner which is encrypted and stored on the cloud. Each matrix $\Phi_{U1}\ and\ \Phi_{U2}$ is of dimension $m\,x\,n$, and are sampled randomly from two different sets with size $J\ and\ K\ respectively$. They are used to encrypt x_1 and x_2 which are image feature vectors with size n each for Users 1 & 2 respectively as shown in Figure 2.

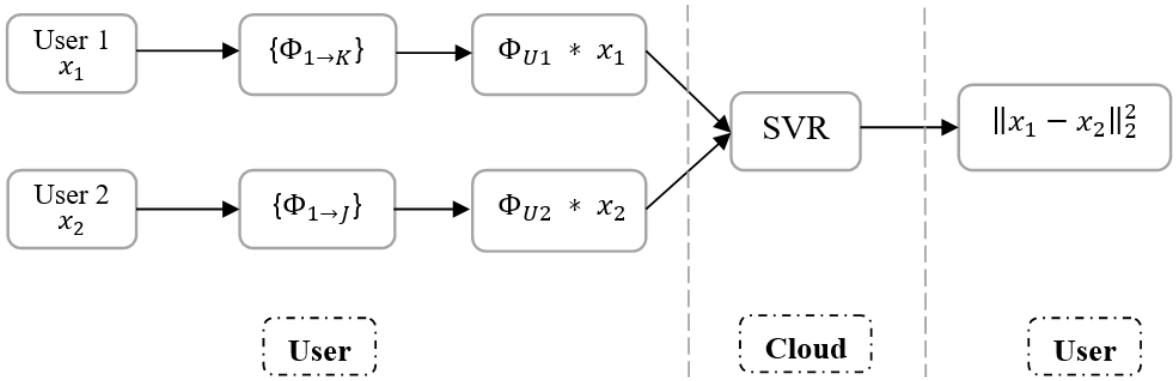

Fig. 2. The data flow of the basic idea of the proposed model

In this paper, the model is trained to calculate the Euclidean distance between two feature vectors; however, this is merely an example; it could also be trained to do other computations such as addition, subtraction, and dot product.

Algorithm 1 describes the process of the basic idea, each user gets a random Φ from their set, encrypts the corresponding feature vector, then the encrypted vectors are sent to the cloud. The machine learning at the cloud to do the operation then returns a plaintext distance without compromising the used features. The experimental results should show the model's ability to maintain the distance between features while two different keys were used. This model is outstanding compared to the previous one in terms of having multiple keys available for each user. However, the model has one SVR available at the could with one set assigned to it, which could expose the system to vulnerabilities easily as it could be exhausted, and the set could be figured out after multiple unique trials that matches the set length.

ALGORITHM 1: RANDOM PROJECTION ENCRYPTION WITH SVR DECRYPTION BASIC IDEA

Inputs: *User 1 & 2 feature vectors* x_1 & x_2.
Output: *Calculated Euclidean distance* $\|x_1 - x_2\|_2^2$.
1 *SVR* $Rand(SVR_i)$.
2 $Rand(\Phi_{i,j})$ *// foreach user from available designated sets.*
3 *Encryption:* $Z = \Phi * x$ *// Multiply each user's vector with a key* Φ.
4 *Send encrypted features to the cloud.*
5 *Estimate* $\lambda = \|x_1 - x_2\|_2^2$.
6 *Return* λ.

3.2 Enhanced Proposed Model

The model is trained at the Trusted Third Party (TTP). TTP trains two different SVRs, one is at the cloud, which is a set of parallel SVRs, and the second SVR is a singular available at the user's side. The need to increase the number of available SVRs at the cloud is to increase the number of available sets of keys, and the system expansion would result in having a more complex system. The available set of SVRs at the cloud are identical in performance but each is trained using a different set of keys. The input to the cloud's SVR is two encrypted feature vectors that users wish to measure their distance against the database owner (BD_o) vectors. So, the TTP generates for each SVR available at the cloud three random matrix sets; $\{\Phi_{DB_o}, \Phi_U \; and \; \Psi_{SVR}\}$, each Φ is a potentially large set of keys, where the SVR has only one large key Ψ that encrypts the output distance. When a user inputs their vector, a

randomly chosen key from the randomly chosen SVR is multiplied with their vector, same is done with the database owner's vectors, as both vectors get randomly projected.

The cloud's SVR input is a concatenation of both randomly projected encrypted vectors as shown in Eq. (1) through (3).

$$\mathrm{w_U} = \Phi_{\mathrm{u}}{}^{*}x_{\mathrm{u}} \tag{1}$$

$$\mathrm{w_{DBO}} = \Phi_{\mathrm{DBO}}{}^{*}x_{\mathrm{DBO}} \tag{2}$$

$$\mathrm{w} = \left[\mathrm{w_u}\ \mathrm{w_{DB_O}}\right] \tag{3}$$

The user's SVR is trained to receive the encrypted vector in the form of Eq. 7 and retrieves the originally measured distance. Which will be explained in the user part. Algorithm 2 states the enhanced model procedure.

ALGORITHM 2: RANDOM PROJECTION ENCRYPTION WITH SVR DECRYPTION ENHANCED MODEL

Inputs: *User 1 & 2 feature vectors* x_1 & x_2.
Output: *Calculated Euclidean distance* $\lambda = \|x_1 - x_2\|_2^2$.
1 $Rand(SVR_i)$ *// User is assigned to a randomly chosen SVR.*
2 For each feature: *// for both vectors available designated sets.*
3 $Rand_{SVR_i}(\Phi)$
4 *Encryption:* $W = U(x_1) * Rand_{SVR_i}(\Phi_U) + DB_O(x_2) * Rand_{SVR_i}(\Phi_{DB_O})$.
5 *Send* Y *to the cloud's chosen* SVR.
6 *Foreach* SVR_j*:*
7 *Estimate:* $y_j = Z_j * SVR_j(\Psi)$ *//SVR produces an encrypted distance with random added noise.*
8 Calculate $Y = \sum_{a=0}^{L} \Psi_a * Z_a$
9 *Send* Y *to the user.*
10 *Decryption:* $\lambda = \Psi_i * \mathrm{Y}$

3.2.1 User to Cloud

Figure 3 displays the discussed further enhancements to the model in Figure 2. The cloud will offer multiple parallel SVRs instead of having just one. A user that wishes to do computations on their data against the (DB_o) database owner's data, will be assigned randomly one of the available SVRs at the cloud. Each SVR can calculate an encrypted distance and then sends an encrypted result to the user. The system at the cloud has L number of available SVRs. Each SVR has two sets of unique normalized and randomized Gaussian Φs, all sets of same size but different key combination. Each one of the SVRs is trained to calculate the distance and produce a sparse vector Z which is encrypted with a corresponding Ψ as an output of the cloud's SVR, which is an orthogonal key matrix.

The input to the cloud SVR is the user's and the DB_o's concealed vectors. All SVRs receive similar looking inputs to misguide the cloud from knowing the real working SVR. Each SVR at the cloud calculates the distance using both encrypted feature vectors.

A further processing on the real distance is done within the cloud's SVR, before multiplying it with the SVR's Ψ. The calculated distance is encoded in Z which is a sparse vector. In order to stabilize the vector's energy a random noise is added to it that is inversely proportional to the actual signal so their energy sum to a constant. Consequently, the sparse vector's encoded output would not be

directly proportional with the real distance. Then Z is multiplied with Ψ. And this represents each SVR's output at the cloud.

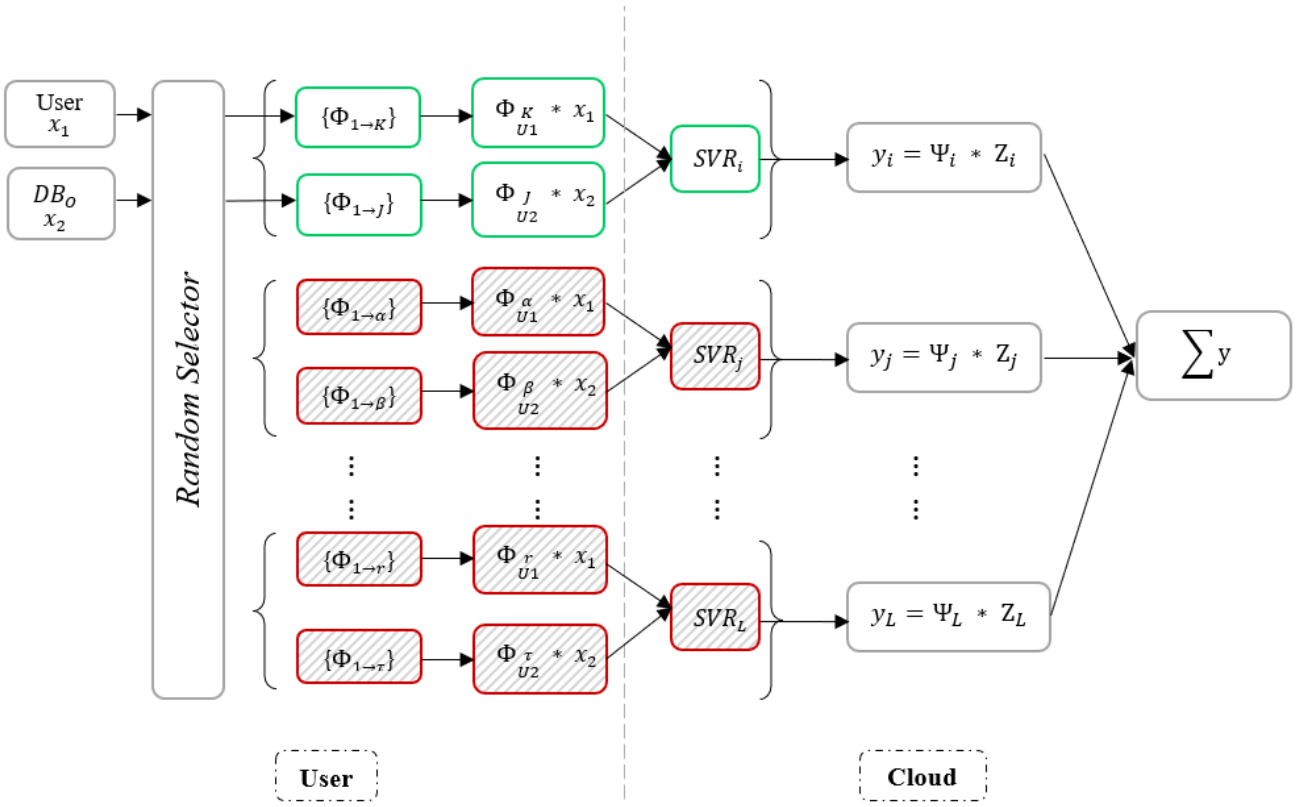

Fig. 3. User to cloud phase, where the user is assigned a random SVR and a random key from the available set of the randomly chosen SVR

The output of all SVRs is then summed together and sent to the user.

$$Y = \sum_{a=0}^{L} y_a = \sum_{a=0}^{L} \Psi_a * Z_a \tag{4}$$

The rest of the SVRs that were not chosen produces noise. Undoubtedly, the cloud should not learn which SVR was chosen, therefore, all SVRs' outputs are summed and then sent to the user.

The purpose of generating two different sets of Φs for each SVR is to impair the cloud from knowing which SVR was chosen, and which one is actually running, this produces a more complex system, and have a divergent set of SVR. Nonetheless, the Database Owner at cloud could have an agnostic set of Φs, which will make a less complex system, and a – somewhat – more similar trained SVR.

3.2.2 Cloud to User

The proposed model uses machine learning for decryption to recover the mathematical computation done between the two vectors. In Figure 4 the user receives the output provided from the cloud Eq. (4); the user knows which is the correct SVR, so the summed output sent by the cloud to the user. The output is multiplied with Ψ Eq. (5). The result will be fed into the user's SVR and finally produces the distance as plaintext Eq. (6). Encrypting the cloud's output would protect the

system from eavesdroppers or curious clouds. The Ψ acts as another key dedicated for each SVR. Also summing the output of all SVRs adds another layer of complexity.

$$\text{User's SVR Input} = \Psi_i * \sum y \tag{5}$$

$$\text{User's SVR Output } \lambda = \|x_1 - x_2\|_2^2 \tag{6}$$

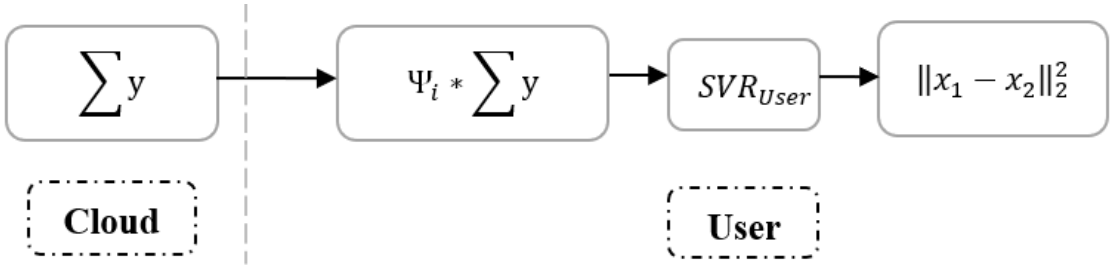

Fig. 4. User receives the cloud output and multiply it with the correct Ψ to get the energy vector and use it on their own SVR to compute the distance

4. Proposed Model Cryptanalysis

This section discusses the security concerns of the proposed model, from COA Ciphertext Only Attack [24] and KPA Known Plaintext Attack [40] perspectives: three attack scenarios are presented.

4.1 Scenario 1

The attacker is based at the cloud (Cipher-Only Attack). In the proposed approach, the SVR's input is encrypted with a random key from a subset and the output signal's energy is hidden by teaching the cloud's SVRs to produce an encrypted signal, which is multiplied with a key that belongs to each SVR. The attacker will not be able to reverse engineer the input nor the output correctly. And it will be hard to recover the correct vector with the wrong key. Figure 5 explains scenario 1.

Because all SVRs produce a randomly encrypted and energy normalized outputs, a curious cloud or an attacker based on the cloud will be misled and will not figure which SVR was the randomly chosen one.

And should the attacker identify the correct SVR for a specific distance request the attacker will have to try and obtain the keys used to encrypt each vector, along with the SVR's key. This would fall into an NP class problem, the attacker would know the key size, but will have to try all possible combinations to reach the correct key matrix used. And would reach a vector with minimal confidence, as a wrong key would produce the wrong vector. Rachlin *et al.,* [41] has demonstrated that compressed sensing is not able to reach perfect secrecy provided by Shannon's definition [42]. However, they tested trying to recover the plaintext given the ciphertext only while using a wrong key, and it resulted in a different plaintext which means that an adversary cannot recover the original signal without the correct key, given the key is sufficiently large, this happens with one-time scenarios. They also showed that using CS based encryption technique provides computational security and highlighted that the signal's energy is a weakness an adversary might use.

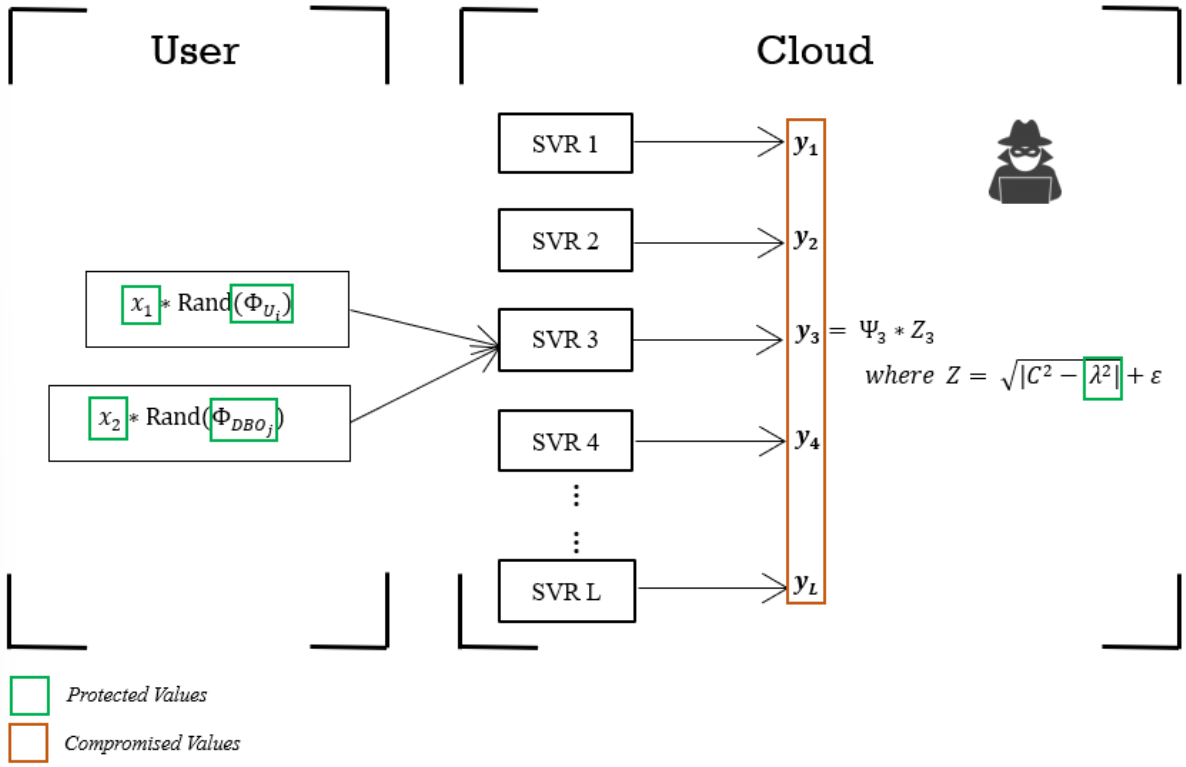

Fig. 5. Scenario 1: The attacker is based on the cloud, could see the output of the SVRs but cannot be sure which one is currently in use

This attack has a solution presented by Fay *et al.,* [28] by using multiple keys that changes with every transaction. The adversary could guess parts of the key matrix Φ when they collect enough ciphertexts which would weaken the encryption abilities. Therefore, having a carefully generated Gaussian distributed keys which are chosen using a random seed with each transaction will result in generating different encrypted output to the same plaintext.

Giving each user a limited number of system usage will also be beneficial in this case. As with each trial the output to the same plaintext will be different therefore the attacker will not be able to estimate anything useful. In this case the attacker will not only have to guess the number of keys and solve a clustering problem but will have to estimate the key that was used and the correlation in the data.

4.2 Scenario 2

In this case the attacker is both at the cloud and acts as a user as well to perform a KPA (Known Plaintext Attack). The attacker would have access to a black-box system where they get a random matrix to multiply their vector with, they get another key that represents the Ψ. The attacker could recover their specific Φ. Furthermore, they could exhaust the system in order to figure out all keys assigned in the users' sets to be able to learn other users' inputs. But the database owner is protected as their keys are independent of the users'. In this scenario the attack will require multiple queries using the same inputs to target the same key matrix and produce the same output, therefore, in order to replicate the framework an input should be queried twice using the same key. On the other hand, to replicate the system that is residing at the cloud only using brute force; each single key would require $O(mnr)$ to be figured, ignoring time complexity involved on the user's side. Figure 6 showcases the second scenario, where the attacker is both in the cloud and acting as a user. By

recovering both keys Φ & Ψ the attacker will be able to see the value of Z which contains the hidden value of the distance between the two vectors. The problem will be even more complex as the attacker will need to figure how many Φs are used in the system and will have to cluster the output data with those different Φs, so the attacker could apply inverse modeling to crack all keys. Cambareri *et al.,* [43] reviewed CS encryption against Known Plaintext Attack in a theoretical approach, their results show that there are many keys that could result in a similar pair of plaintext and ciphertext. Which will harden a brute force attack on the key matrix as the results would be inconclusive.

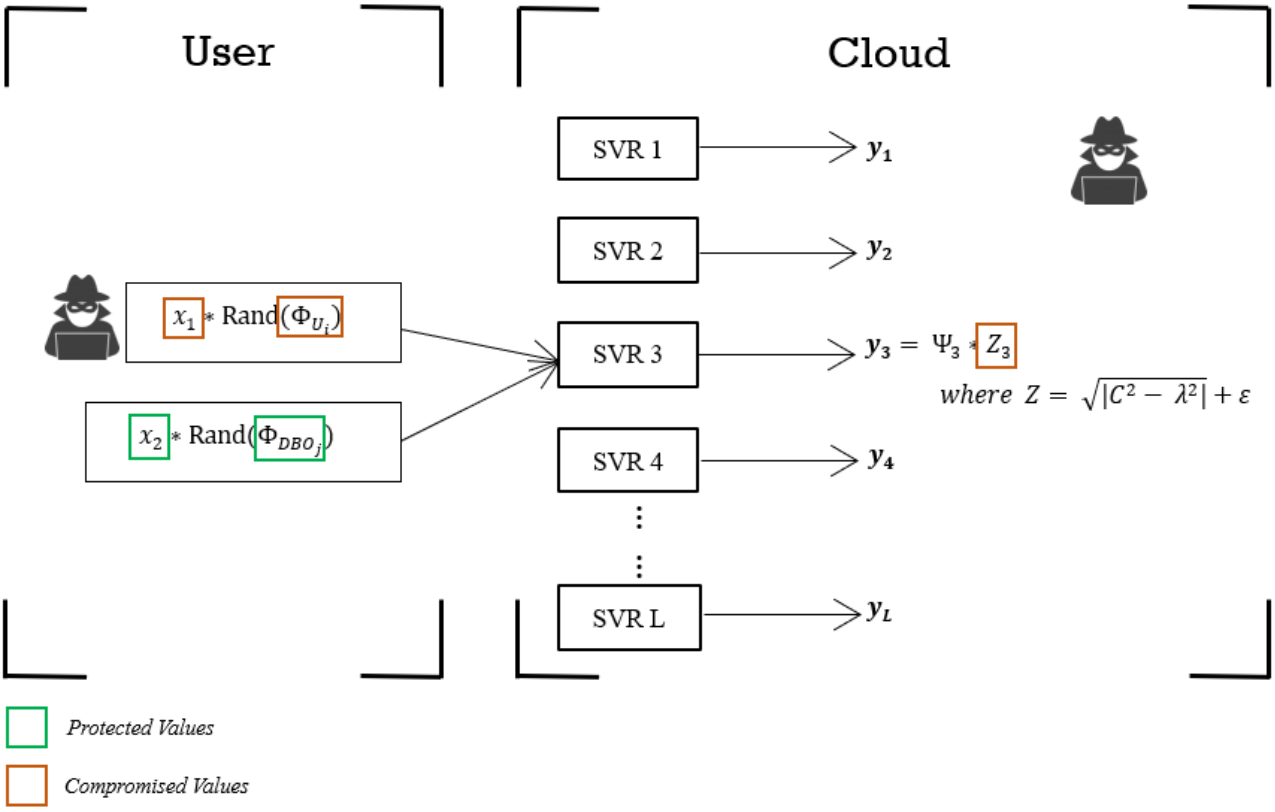

Fig. 6. Scenario 2: The attacker is based on the cloud and acting as a user as well, they could recover the $\boldsymbol{Z}$ which is a vector that contains the value of the encrypted distance

One of the algorithm an attacker could use is the Blind Source Separation (BSS) algorithm, which is the process of separating a mixture of K sources into its individual sources without the knowledge of the mixing coefficients [44]. There is a popular algorithm that is used in solving the BSS problem called Independent Component Analysis (ICA). The idea of ICA is observing random variables v which are modelled as the sum of hidden variables as in Eq. (7)

$$v_i = \sum_{j=1}^{m} a_{ij} s_j \tag{7}$$

Where a_{ij} is a constant referred to as mixing matrix, and s_j are the hidden independent components or source vectors. The target is to estimate both a_{ij} and s_j using a set of assumptions.

With the possibility of system expansion. The attacker could use such a technique after trying to exhaust the model to produce plaintext ciphertext pairs as much as possible, which enough data

points the attacker could apply a suitable BSS algorithm and try to separate the pairs, which could eventually let the attacker identify which pairs came from the same source.

4.3 Scenario 3

The third scenario would be the case of a TTP that acts as a user as well but cannot access the cloud. The attacker then would be able to generate enough feature vectors examples to train a machine learning algorithm that could resemble the entire system. Therefore, the more keys, key size, and system expansion the harder the process is for the attacker. Also changing the keys every while would also act as another protection layer against this attack. Figure 7 explains scenario 3 where the TTP is acting as a user and using the output to train a machine learning algorithm to mimic the framework.

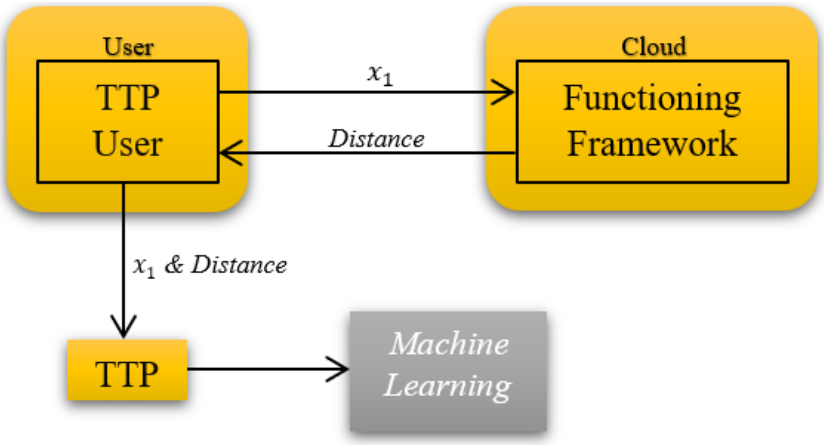

Fig. 7. Scenario 3: TTP acting as a user, then training a machine learning to mimic the framework

5. Experimental Results

5.1 Methodology

The research aims to prove that secure computations could be done on encrypted data, with low computational complexity, while preserving the data's integrity and security. Following the work conducted by Fakhr and Mohamed [21], the proposed model is used to perform secure image retrieval on the COREL 1K dataset. The privacy-preserved Euclidean distance is sought between query and database image feature vectors which are PCA-reduced to only 7 features. Training the cloud's SVRs and the user's SVR was done using synthetic data to control the number of training examples. Each cloud SVR was trained on a set of unique random keys, to produce an encrypted distance, the encrypted data is sent to the user to decrypt and get the distance.

The first experiment is designed to benchmark the retrieval accuracy of the proposed approach, in addition a designed nearest neighbour experiment was applied to measure the accuracy of the distance calculation before and after the encryption process. The second experiment is aimed to observe how changing the parameters of the system, by increasing the number of keys and SVRs, would affect the system. The third experiment represents the system's extrapolation beyond the parameters used in the second experiment. Experiment 4 is a comparison between the proposed approach and Homomorphic Encryption in terms of storage, delay, and performance.

5.2 Training the Support Vector Regression Models:

The training is done by using synthetic randomized Gaussian data that represent an image's feature vector, which mimic the COREL 1K dataset and the same pre-processing steps were done on both. The keys are generated as an orthogonal randomized Gaussian matrix. As each SVR has two sets of keys, each feature vector is encrypted with a randomly chosen matrix Φ from the available keys. Each SVR at the cloud is capable of producing the output as an encrypted result that is multiplied with another key Ψ specific for each SVR. Ψ is another random Gaussian matrix. The cloud's SVRs required more training examples as the number of the available set increased, the benchmark experiment was trained using 500 synthetically generated examples, where the user's SVR required in all experiments between 1000 and 2000 training examples. The user's SVR is trained similarly as the cloud's SVR. It also inherits the hyper parameters from the cloud SVR structure. The data for training and testing is generated and the same processing takes place.
The experiments aim is to reveal the following:

i. The amount of training data needed should be proportional to the complexity.
ii. The retrieval accuracy, under the mentioned different scenarios.
iii. Learning mean square error (MSE).
iv. How the system extrapolation would look like.

The experiments were done on the following specifications, the processor is a twelve cores AMD Ryzen 9 5900X, the used memory has a capacity of 32 GB.

5.3 Experiment 1: COREL 1K Retrieval Performance of Proposed Model

The benchmarking experiment was done to set the retrieval accuracy in the plaintext scenario. In this experiment to imitate the plaintext scenario, the user, and the database owner both have the same key Φ. While the number of SVRs at the cloud is set to only one.

In the experiment the cloud was trained on the same Φ (i.e., only one key matrix). Moreover, using one SVR at the cloud. Two versions of the benchmarking experiment were done, a version represents dimensional expansion RP that expanded a vector from 7 non-zero elements to 420 elements. The benchmark parameters setup is shown in Table 1.

Table 1
Benchmark Parameters

Parameters	Value
Number of available SVRs at the cloud	1
Number of available Φs for each user	1
Feature vector length of non-zero values	7
Expansion rate of vector	60
Cloud SVR output length	8
Expansion length of cloud output	2
Cloud's SVR encrypted input size	60*7 = 420

The experiment was done by generating 5000 synthetic data to feed the cloud's SVM for training, then generating 1000 examples to train the SVM at the User. The training MSE at the cloud reached a minimum of 10^{-6}, while training MSE of the user's SVR 10^{-5} with 60% retrieval accuracy when tested the classification on COREL1K dataset. The benchmarking experiment proved that the proposed model could preserve the distances even while encrypted.

Table 2 contains the evaluated measurements of the retrieval. Figure 10 shows the precision and recall of COREL dataset, the green dashes are the plaintext calculated Euclidean distance and the red

line is the distance calculated using the SVR at the cloud then decrypted by the SVR residing at the user.

Table 2
Benchmark Measurements

Measurement	Value
Precision	0.6272
Recall	0.6220
Accuracy	0.6220
Specificity	0.9580
F1 score	0.6200

To measure the accuracy of distance calculation before and after the encryption, the nearest neighbour experiment was done, by measuring the Euclidean distance between COREL features in training and testing sets, then retrieving the nearest neighbours. Once before encryption and once after encryption. The results of the K-Nearest Neighbour (KNN) before and after encryption were 99% matching each other as shown in Figure 8 and Figure 9 showcases the precision and recall results of the COREL data.

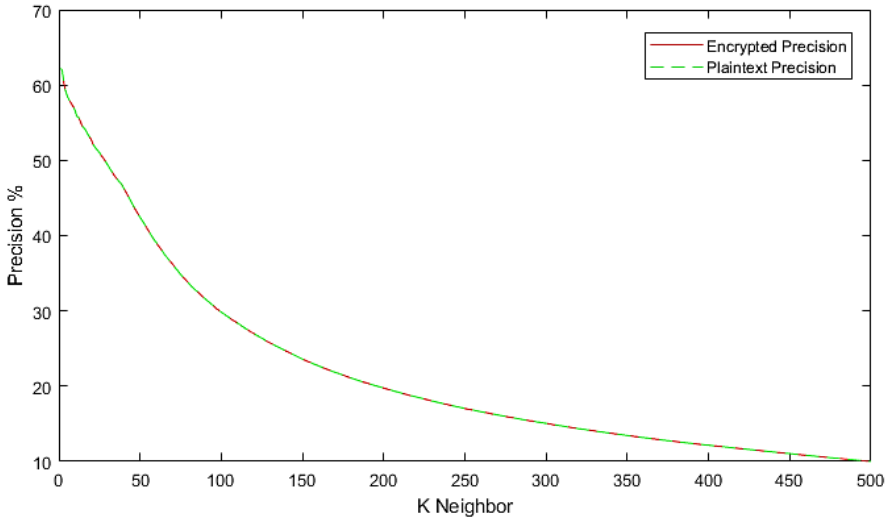

Fig. 8. Precision of COREL Data; the red line represents encrypted data precision, and the green dashes represents the plaintext data precision

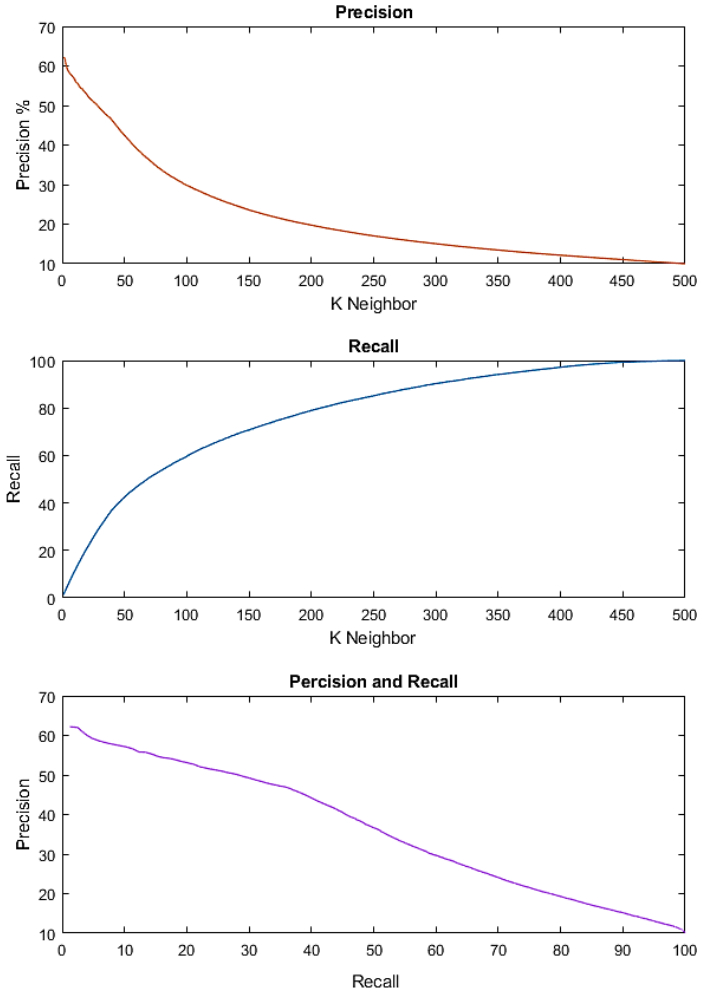

Fig. 9. Precision and recall at each K

5.4 Experiment 2: Advanced Settings

The 2nd experiment was designed to increase both the number of available SVMs and Φs at the cloud. And to examine the minimum training data needed to successfully train all SVRs. As well as to reach the set MSE for the cloud and user of at least 3 decimal points, besides reaching 60% when testing on COREL 1K dataset the classification of the SVR. The retrieval percentage could be improved by increasing the feature vector length. Figure 10 and Figure 11 showcase experiment 2 upper and lower ranges of minimum training length given the hyperparameters mentioned in Table 3, note that the figures also include the results of the extrapolated parameters.

Table 3
Experiment 2 hyper parameters

Parameters	Value
L: Number of available SVRs at the cloud	{1→8}
k: Number of available Φs for each user	{1→8}
Vector length	7
Expansion rate of the user's vector	60
Cloud SVR output length	8
Expansion length of cloud output	2
SVR input at cloud size	60*7 = 420

5.5 Experiment 3: System Expansion

In this experiment the system's extrapolation was modelled using polynomial regression to measure the minimum training length needed for the system when expanded up to 16 SVRs at the cloud and 16 Φs in each available set. The experiments were built based on the data acquired in experiment 2. This experiment is motivated by the need to measure the complexity of the system if an adversary attempts to attack. The deeper the system is the harder the task at hand. Figures 10 & 11 display the lower and upper ranges and mean of minimum training length of all data displayed in experiment 2 and extrapolated from 9 to 16.

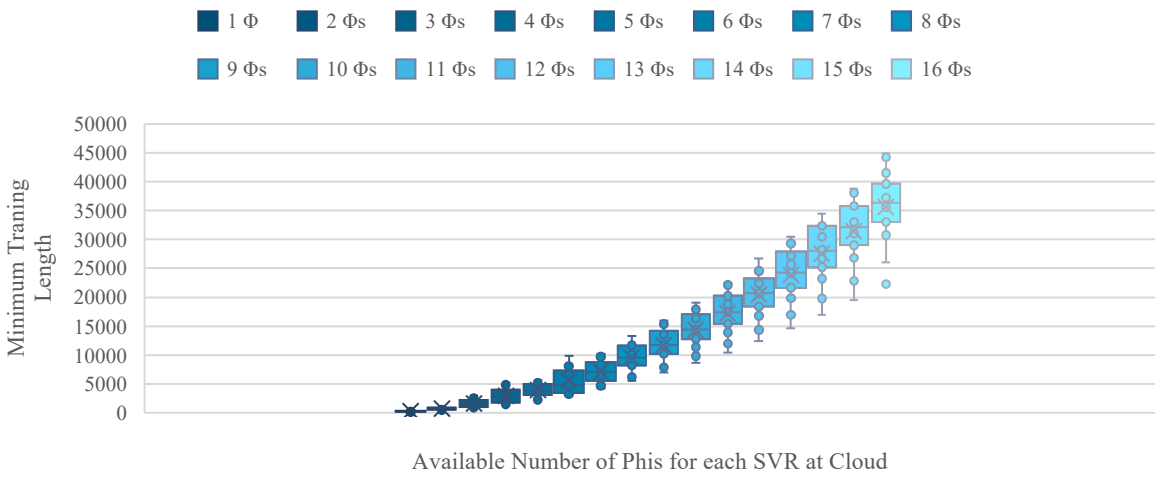

Fig. 10. Expected Minimum Training Length Ranges and Mean of Training Data for each Phi Available at Cloud (Extrapolation from 9 to 16 Phis)

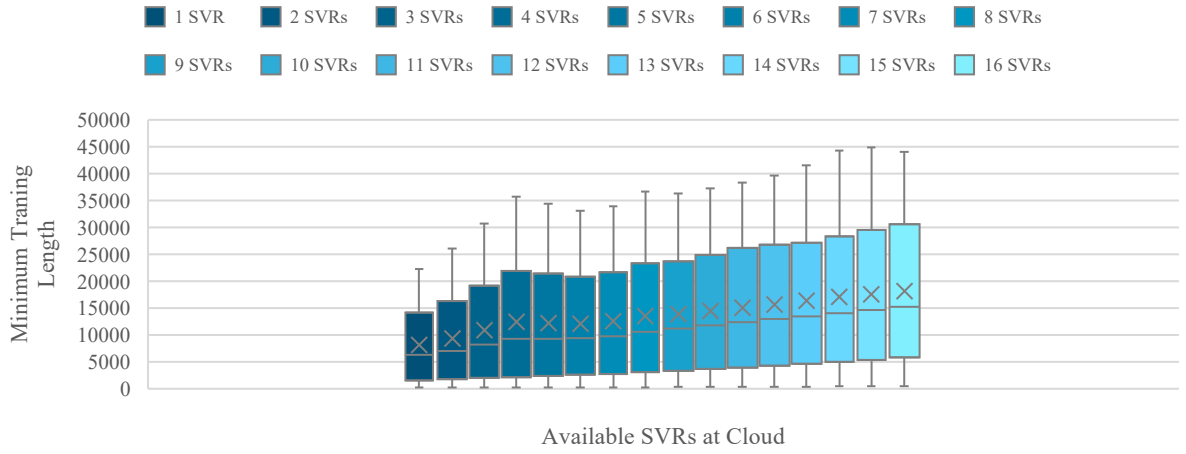

Fig. 11. Expected Minimum Training Length Range and Mean of each SVR Available at Cloud (Extrapolation from 9 to 16 SVRs)

5.6 Experiment 4: Time and Space Comparison with FHE

In this experiment a comparison was made between Fully Homomorphic Encryption and the proposed RP system. Microsoft Research has an open-source Simple Encrypted Arithmetic Library called Microsoft SEAL [45]. The HE algorithm that was used for the comparison is the BFV [44,46].

The time comparison was measured in milliseconds and an average was calculated between two vectors. And finally, ciphertext size expansion in HE and RP case. Note that all three methods have 7 features to start with.

The HE experiment is conducted by having two plaintext vectors, each vector has 7 features, then both vectors get encrypted and Euclidean distance is calculated. The process was as follows: encode plaintext vectors, then encrypt both, execute distance calculation, then decrypt and decode. The degree of polynomial modulus is set at 8192, the larger this number is the more calculations it allows with more noise budget, allowing multiple usages. But on the other hand, it also results in a larger ciphertext. When the degree was set to less than 4096 the encoding failed, while at 4096 there was 0 noise budget left for computations after the vectors squaring homomorphically. Table 4 encloses a comparison between HE, and RP.

Table 4
Comparison between Homomorphic Encryption, Random Projection

	Homomorphic Encryption	Random Projection
Encryption Time	0.6076 milliseconds	0.0610 milliseconds
Distance Calculation Time	0.5816 milliseconds	0.0654 milliseconds
Ciphertext Expansion Rate	1:987	1:60

6. Discussion

The proposed model shows in experiment 2 that increasing SVRs affects the minimum required training length less than adding more Φs, this means the anchor of the system is having multiple keys. The keys are generated randomly as orthogonal Gaussians. The model was tested by using all available SVRs with all available Φs to make sure they all reach the satisfying MSE, so there wouldn't be any ill performing SVR. This reflects the integrity of the system and the complexity of how hard it could take for an attacker to break the system with more SVRs and Φs.

The user is capable of concealing their data before sending it to the cloud using the key sent to them, the database owner has their own sets of keys as well, the cloud receives encrypted inputs and the SVR produces encrypted output. While all SVRs run a cloud's eavesdropper will not be able to identify the currently working SVR on a user's data. And the only symmetrical key is used for distance decryption and not used to recover the user's vector at all. Which is a reference to the mentioned scenario 1 in section V.

Experiments 2 & 3 displays the amount of data to be used to reach certain system learnability accuracy, when an adversary tries to duplicate the system as a black box, these data would give a solid image of how much data is needed to replicate the system, as the system's complexity is proportional to the minimum training length values. As mentioned in the second scenario, this is considered the minimum where the system has the ability to be verified, as for the replica it will need more than that to reach the same performance stability. Also, it will require a lot of trials and time especially where the system goes beyond the extrapolated values. Increasing the number of available Φs would definitely heighten the system's resilience and attack complexity. The data increases in an almost exponential behaviour when increasing the key count as shown in Figure 8. Therefore, increasing the key size and key count is recommended to be done together. The encryption time is $O(n^2)$ as it's a multiplication of a vector with a random matrix.

Users will benefit from this system by securely having computations done on their data against others'. The proposed system's strength lays in its ability to be expanded while giving a robust performance with its growth. Having larger key space with RP gives an advantage against attacks. Although the same concept could be applied on CS but in a limited conservative manner. RP exceeds

HE when compared in encryption and calculations speed, in addition to the less space it takes, as shown in experiment 4, taking into consideration that in RP the expansion rate is controlled and set as a parameter.

Moreover, the distance was measured between training and testing sets, once before projection and again after, then the index of the nearest neighbour was retrieved at each possible K, Figure 8 showcased the retrieval before and after projection which means that the projection doesn't affect the original data.

The trained SVR at the cloud was taught to calculate the distance between two vectors, however, it could also be trained to do any computation securely and on matrices as well. Furthermore, this presented system's expansion could be tuned to a given system, to find the perfect size, performance, privacy, and security combinations.

7. Conclusion

This paper introduced a lightweight secure expandable approach to enable parties to do computations privately, by giving an example of computing the Euclidean distance between images feature vectors securely, using random projection encryption and machine learning decryption. The distance is calculated in an encrypted manner by an SVR that is available at the cloud, and another SVR is used to decrypt the distance at the user's side. The proposed model eliminated the weakness of symmetric encryption methods. It also could be applied to various operations, not only distance calculation. The experiments done showed the ability of expansion which is an added layer of protection. It shows its supremacy over HE in speed and storage. In the future the system could be tested on larger different datasets and applications, also adding a multilevel encryption system.

Acknowledgement
This research was not funded by any grant.

References
[1] Chakraborty, Partha, and Sajeda Sultana. "IoT-based smart home security and automation system." In *International Conference on Micro-Electronics and Telecommunication Engineering*, pp. 497-505. Singapore: Springer Nature Singapore, 2021. https://doi.org/10.1007/978-981-16-8721-1_48

[2] Sugawara, Etsuko, and Hiroshi Nikaido. "Properties of AdeABC and AdeIJK efflux systems of Acinetobacter baumannii compared with those of the AcrAB-TolC system of Escherichia coli." *Antimicrobial agents and chemotherapy* 58, no. 12 (2014): 7250-7257. https://doi.org/10.1128/AAC.03728-14

[3] Goudarzi, Arman, Farzad Ghayoor, Muhammad Waseem, Shah Fahad, and Issa Traore. "A Survey on IoT-Enabled Smart Grids: Emerging, Applications, Challenges, and Outlook." *Energies* 15, no. 19 (2022): 6984. https://doi.org/10.3390/en15196984

[4] Kanchana, V., Surendra Nath, and Mahesh K. Singh. "A study of internet of things oriented smart medical systems." *Materials Today: Proceedings* 51 (2022): 961-964. https://doi.org/10.1016/j.matpr.2021.06.363

[5] Rathinaeaswari, S. P., and V. Santhi. "A New Efficient and Privacy-Preserving Hybrid Classification model for Patient-Centric Clinical Decision Support System." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 33, no. 1 (2023): 299-316. https://doi.org/10.37934/araset.33.1.299316

[6] Zuo, Chaoshun, Zhiqiang Lin, and Yinqian Zhang. "Why does your data leak? uncovering the data leakage in cloud from mobile apps." In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 1296-1310. IEEE, 2019. https://doi.org/10.1109/SP.2019.00009

[7] Phong, Le Trieu, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. "Privacy-preserving deep learning: Revisited and enhanced." In *Applications and Techniques in Information Security: 8th International Conference, ATIS 2017, Auckland, New Zealand, July 6–7, 2017, Proceedings*, pp. 100-110. Springer Singapore, 2017. https://doi.org/10.1007/978-981-10-5421-1_9

[8] Zhang, Yushu, Leo Yu Zhang, Jiantao Zhou, Licheng Liu, Fei Chen, and Xing He. "A review of compressive sensing in information security field." *IEEE access* 4 (2016): 2507-2519. https://doi.org/10.1109/ACCESS.2016.2569421

[9] Iezzi, Michela. "Practical privacy-preserving data science with homomorphic encryption: an overview." In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 3979-3988. IEEE, 2020. https://doi.org/10.1109/BigData50022.2020.9377989

[10] Lou, Qian, and Lei Jiang. "Hemet: A homomorphic-encryption-friendly privacy-preserving mobile neural network architecture." In *International conference on machine learning*, pp. 7102-7110. PMLR, 2021. https://doi.org/https://doi.org/10.48550/arXiv.2106.00038

[11] Cramer, Ronald, and Ivan Bjerre Damgård. *Secure multiparty computation*. Cambridge University Press, 2015. https://doi.org/10.1017/CBO9781107337756

[12] Truex, Stacey, Ling Liu, Mehmet Emre Gursoy, and Lei Yu. "Privacy-preserving inductive learning with decision trees." In *2017 IEEE International Congress on Big Data (BigData Congress)*, pp. 57-64. IEEE, 2017. https://doi.org/10.1109/BigDataCongress.2017.17

[13] Fletcher, Sam, and Md Zahidul Islam. "Differentially private random decision forests using smooth sensitivity." *Expert systems with applications* 78 (2017): 16-31. https://doi.org/10.1016/j.eswa.2017.01.034

[14] Gentry, Craig. *A fully homomorphic encryption scheme*. Stanford university, 2009.

[15] Gentry, Craig. "Fully homomorphic encryption using ideal lattices." In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169-178. 2009. https://doi.org/10.1145/1536414.1536440

[16] Morris, Liam. "Analysis of partially and fully homomorphic encryption." *Rochester Institute of Technology* 10 (2013): 1-5.

[17] Graepel, Thore, Kristin Lauter, and Michael Naehrig. "ML confidential: Machine learning on encrypted data." In *International conference on information security and cryptology*, pp. 1-21. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. https://doi.org/10.1007/978-3-642-37682-5_1

[18] Li, Chao, and Balaji Palanisamy. "Privacy in internet of things: From principles to technologies." *IEEE Internet of Things Journal* 6, no. 1 (2018): 488-505. https://doi.org/10.1109/JIOT.2018.2864168

[19] Zhao, Jingwen, Yunfang Chen, and Wei Zhang. "Differential privacy preservation in deep learning: Challenges, opportunities and solutions." *IEEE Access* 7 (2019): 48901-48911. https://doi.org/10.1109/ACCESS.2019.2909559

[20] Ziller, Alexander, Dmitrii Usynin, Rickmer Braren, Marcus Makowski, Daniel Rueckert, and Georgios Kaissis. "Medical imaging deep learning with differential privacy." *Scientific Reports* 11, no. 1 (2021): 13524. https://doi.org/10.1038/s41598-021-93030-0

[21] Fakhr, Mohamed Waleed. "A multi-key compressed sensing and machine learning privacy preserving computing scheme." In *2017 5th International Symposium on Computational and Business Intelligence (ISCBI)*, pp. 75-80. IEEE, 2017. https://doi.org/10.1109/ISCBI.2017.8053548

[22] Wang, Qia, Wenjun Zeng, and Jun Tian. "A compressive sensing based secure watermark detection and privacy preserving storage framework." *IEEE transactions on image processing* 23, no. 3 (2014): 1317-1328. https://doi.org/10.1109/TIP.2014.2298980

[23] Yamaç, Mehmet, Mete Ahishali, Nikolaos Passalis, Jenni Raitoharju, Bulent Sankur, and Moncef Gabbouj. "Reversible privacy preservation using multi-level encryption and compressive sensing." In *2019 27th European Signal Processing Conference (EUSIPCO)*, pp. 1-5. IEEE, 2019. https://doi.org/10.23919/EUSIPCO.2019.8903056

[24] Kuldeep, Gajraj, and Qi Zhang. "Multi-class privacy-preserving cloud computing based on compressive sensing for IoT." *Journal of Information Security and Applications* 66 (2022): 103139. https://doi.org/10.1016/j.jisa.2022.103139

[25] Ratra, Ritu, Preeti Gulia, Nasib Singh Gill, and Jyotir Moy Chatterjee. "Big Data Privacy Preservation Using Principal Component Analysis and Random Projection in Healthcare." *Mathematical Problems in Engineering* 2022 (2022). https://doi.org/10.1155/2022/6402274

[26] Majhi, Mukul, and Ajay Kumar Mallick. "Random projection and hashing based privacy preserving for image retrieval paradigm using invariant and clustered feature." *Journal of King Saud University-Computer and Information Sciences* 34, no. 9 (2022): 6829-6846. https://doi.org/10.1016/j.jksuci.2022.04.018

[27] Li, Ping, and Xiaoyun Li. "Differential Privacy with Random Projections and Sign Random Projections." *arXiv preprint arXiv:2306.01751* (2023). https://doi.org/10.48550/arXiv.2306.01751

[28] Fay, Robin, and Christoph Ruland. "Compressive sensing encryption modes and their security." In *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, pp. 119-126. IEEE, 2016. https://doi.org/10.1109/ICITST.2016.7856681

[29] Lu, Wenjun, Avinash L. Varna, and Min Wu. "Confidentiality-preserving image search: A comparative study between homomorphic encryption and distance-preserving randomization." *IEEE Access* 2 (2014): 125-141. https://doi.org/10.1109/ACCESS.2014.2307057

[30] Truex, Stacey, Ling Liu, Mehmet Emre Gursoy, Wenqi Wei, and Lei Yu. "Effects of differential privacy and data skewness on membership inference vulnerability." In *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pp. 82-91. IEEE, 2019. https://doi.org/10.1109/TPS-

ISA48467.2019.00019
[31] Li, Ping, Heng Ye, and Jin Li. "Multi-party Security Computation with Differential Privacy over Outsourced Data." In *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, pp. 487-500. Cham: Springer International Publishing, 2017. https://doi.org/10.1007/978-3-319-72389-1_39
[32] Boyle, Elette, Kai-Min Chung, and Rafael Pass. "Large-scale secure computation: Multi-party computation for (parallel) RAM programs." In *Annual Cryptology Conference*, pp. 742-762. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015. https://doi.org/10.1007/978-3-662-48000-7
[33] Hesamifard, Ehsan, Hassan Takabi, Mehdi Ghasemi, and Rebecca N. Wright. "Privacy-preserving machine learning as a service." *Proc. Priv. Enhancing Technol.* 2018, no. 3 (2018): 123-142. https://doi.org/10.1515/popets-2018-0024
[34] Owusu-Agyemeng, Kwabena, Zhen Qin, Hu Xiong, Yao Liu, Tianming Zhuang, and Zhiguang Qin. "MSDP: multi-scheme privacy-preserving deep learning via differential privacy." *Personal and Ubiquitous Computing* (2021): 1-13. https://doi.org/10.1007/s00779-021-01545-0
[35] Li, Ping, Jin Li, Zhengan Huang, Tong Li, Chong-Zhi Gao, Siu-Ming Yiu, and Kai Chen. "Multi-key privacy-preserving deep learning in cloud computing." *Future Generation Computer Systems* 74 (2017): 76-85. https://doi.org/10.1016/j.future.2017.02.006
[36] López-Alt, Adriana, Eran Tromer, and Vinod Vaikuntanathan. "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption." In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pp. 1219-1234. 2012. https://doi.org/10.1145/2213977.2214086
[37] Mukherjee, Pratyay, and Daniel Wichs. "Two round multiparty computation via multi-key FHE." In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pp. 735-763. Springer Berlin Heidelberg, 2016. https://doi.org/10.1007/978-3-662-49896-5_26
[38] Peng, Jialiang, Bian Yang, Brij B. Gupta, and Ahmed A. Abd El-Latif. "A biometric cryptosystem scheme based on random projection and neural network." *Soft Computing* 25 (2021): 7657-7670. https://doi.org/10.1007/s00500-021-05732-2
[39] Jiang, Linshan, Rui Tan, Xin Lou, and Guosheng Lin. "On lightweight privacy-preserving collaborative learning for internet-of-things objects." In *Proceedings of the international conference on internet of things design and implementation*, pp. 70-81. 2019. https://doi.org/10.1145/3302505.3310070
[40] Liang, Jia, Di Xiao, Hui Huang, and Min Li. "Multi-level Privacy Preservation Scheme Based on Compressed Sensing." *IEEE Transactions on Industrial Informatics* (2022). https://doi.org/10.1109/TII.2022.3209153
[41] Rachlin, Yaron, and Dror Baron. "The secrecy of compressed sensing measurements." In *2008 46th Annual Allerton conference on communication, control, and computing*, pp. 813-817. IEEE, 2008. https://doi.org/10.1109/ALLERTON.2008.4797641
[42] Shannon, Claude E. "Communication theory of secrecy systems." *The Bell system technical journal* 28, no. 4 (1949): 656-715. https://doi.org/10.1002/j.1538-7305.1949.tb00928.x
[43] Cambareri, Valerio, Mauro Mangia, Fabio Pareschi, Riccardo Rovatti, and Gianluca Setti. "On known-plaintext attacks to a compressed sensing-based encryption: A quantitative analysis." *IEEE Transactions on Information Forensics and Security* 10, no. 10 (2015): 2182-2195. https://doi.org/10.1109/TIFS.2015.2450676
[44] Brakerski, Zvika. "Fully homomorphic encryption without modulus switching from classical GapSVP." In *Annual Cryptology Conference*, pp. 868-886. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. https://doi.org/10.1007/978-3-642-32009-5_50
[45] Microsoft Research. (2021) Microsoft SEAL (Release 3.7) [Online]
[46] Brakerski, Zvika, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) fully homomorphic encryption without bootstrapping." *ACM Transactions on Computation Theory (TOCT)* 6, no. 3 (2014): 1-36. https://doi.org/10.1145/2633600