Hindawi
Security and Communication Networks
Volume 2021, Article ID 8759922, 12 pages
https://doi.org/10.1155/2021/8759922

Research Article

Secure KNN Classification Scheme Based on Homomorphic Encryption for Cyberspace

Jiasen Liu,[1,2] **Chao Wang**,[2] **Zheng Tu**,[2] **Xu An Wang**,[1,2] **Chuan Lin**,[1] **and Zhihu Li**[3]

[1]*Guizhou Provincial Key Laboratory of Public Big Data, GuiZhou University, Guiyang 550025, China*
[2]*Key Laboratory for Network and Information Security of the PAP, Engineering University of the PAP, Xi'an 710086, China*
[3]*China Electric Power Research Institute, Beijing 100001, China*

Correspondence should be addressed to Xu An Wang; 1261510059@qq.com and Chuan Lin; clin@gzu.edu.cn

Received 3 August 2021; Accepted 19 October 2021; Published 3 November 2021

Academic Editor: Gu Zhaoquan

With the advent of the intelligent era, more and more artificial intelligence algorithms are widely used and a large number of user data are collected in the cloud server for sharing and analysis, but the security risks of private data breaches are also increasing in the meantime. CKKS homomorphic encryption has become a research focal point in the cryptography field because of its ability of homomorphic encryption for floating-point numbers and comparable computational efficiency. Based on the CKKS homomorphic encryption, this paper implements a secure KNN classification scheme in cloud servers for Cyberspace (CKKSKNNC) and supports batch calculation. This paper uses the CKKS homomorphic encryption scheme to encrypt user data samples and then uses Euclidean distance, Pearson similarity, and cosine similarity to compute the similarity between ciphertext data samples. Finally, the security classification of the samples is realized by voting rules. This paper selects IRIS data set for experimental, which is the classification data set commonly used in machine learning. The experimental results show that the accuracy of the other three similarity algorithms of the IRIS data is around 97% except for the Pearson correlation coefficient, which is almost the same as that in plaintext, which proves the effectiveness of this scheme. Through comparative experiments, the efficiency of this scheme is proved.

1. Introduction

With the gradual maturity of various AI algorithms, data has gradually become the basis of social operation, playing an essential role in important areas such as economic investment, social management, scientific and technological development, and national security. Large amounts of data are uploaded to cloud servers from personal front-ends, social networks, sensor networks, and the Internet for sharing and analysis. With the development of cloud computing, many artificial intelligence algorithms derived from large data have been developed and widely used in various APPs. Major manufacturers generate recommendation algorithms or make price decisions by analyzing large data of user behavior, such as TikTok, Taobao, small videos recommended by Meituan for users, commodities, and stores. Drip travel also analyzes user travel segments to increase prices for older users. As one of the classic algorithms for big data classification, the KNN algorithm realizes data classification by calculating the similarity between the test data set and the training data set. Because of the simple structure, high efficiency, and accuracy of the KNN algorithm, it is adopted in various scenarios; for example, it is used in image recognition, traffic classification, sensor detection, and especially text classification. Commercial clouds such as Google, Microsoft, and Amazon also provide related services. With the popularity and application of cloud computing, large numbers of users upload local data to cloud servers to enjoy storage and computing services provided by cloud platforms. However, the security of commercial clouds is often not fully trusted [1–3]. As shown in Figure 1, various users and cloud servers transmit various data and classification results in network space, and the security of data needs to be protected.

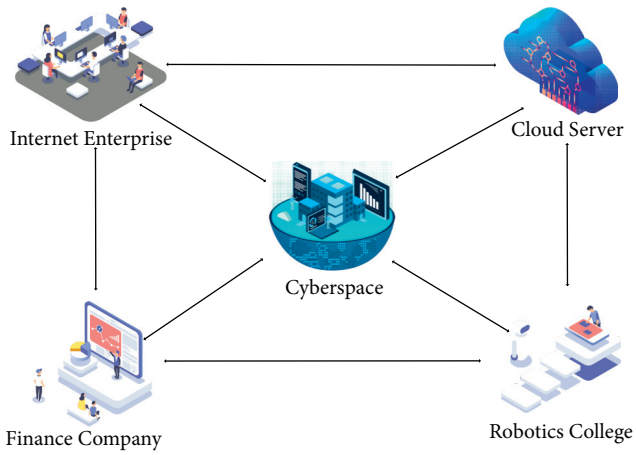

Figure 1: Data transmission in network space.

In recent years, privacy leakage incidents such as malicious collection and theft of user-sensitive privacy data by APP have emerged one after another. In 2007, the National Security Agency (NSA) and the Federal Bureau of Investigation (FBI) launched the infamous "Prism" secret surveillance project, which directly entered the central server of the US Internet company to mine data and collect intelligence; nine international network giants including Microsoft, Yahoo, Google, and Apple have participated in it. In 2018, the Z. power risk monitoring platform detected NASDAQ: HTHT data leak, involving sensitive and private data of about 500 million citizens. In 2021, China Internet Network Information Center issued a notice on the illegal collection and use of personal information by 105 apps including TikTok. Under this trend, people's personal data has gradually become another commodity, and many manufacturers privately collect or even sell users' personal data. In the traditional KNN classification schemes [4–7], the user uploads the local raw data to the cloud platforms for storage and calculation, and then, the cloud platforms compute the similarity between samples and return the result to the user. However, for some highly confidential data, such as personal privacy data, commercial confidential data, medical privacy data, and national security data, once these data are leaked or stolen, the consequences are unimaginable. Therefore, the core issue of this study is to implement a secure privacy-preserving KNN classification scheme efficiently and accurately in the cloud environment. An effective solution for the secure KNN classification scheme is to encrypt local data through a cryptographic algorithm and compute the similarity on the ciphertext data. However, due to the limitation of the encryption algorithm, the computational overhead and storage overhead brought by this solution are extremely large compared to plaintext [8, 9]. First of all, the basic logic operations supported by general encryption schemes are limited, and iterative algorithms are needed instead. Second, some encryption schemes are restricted by modulus reduction and can only support a limited number of multiplications. Finally, traditional encryption schemes can only encrypt digits or vectors one by one, requiring a lot of resources to store the ciphertext. In 2017, Cheon et al. [10] proposed a scheme of homomorphic encryption, CKKS, which supports real number/complex number approximations. This scheme has the ability of homomorphic encryption for floating-point numbers and comparable computational efficiency, which has become a research focal point in the cryptography field, and is widely used in machine learning and big data analysis. To keep users' privacy safely, ensure the security of data and implement KNN classification safely and efficiently in the cloud environment, and enable it to maintain computational efficiency and classification accuracy in plain text fields, as proposed by this paper, a secure KNN classification scheme ground on the CKKS homomorphic encryption scheme in cloud sever for Cyberspace (CKKSKNNC) will be presented. We realize the computation of the similarity through the Euclidean distance, Pearson correlation coefficient, and cosine similarity, at last, to make the secure KNN classification can operate in the domain of ciphertext, which can avoid any user privacy data leakage.

2. Related Work

The traditional KNN classification scheme is divided into two types: one is to assign the same optimal K value for every sample in the test [11–13], and the other is for different test samples; the experts assign individual K values [14–17]. In recent years, a lot of categories is designed based on the KNN algorithm. In 2016, Deng et al. [18] first divided the data set into several categories by K-means algorithm and then classified them by KNN, which realized the efficient KNN algorithm for big data. In 2017, Zhang et al. [19] came up with a kTree method to effectively implement KNN classification with different neighbor numbers. In 2020, Zhang et al. [20] designed two effective cost-sensitive KNN classifiers to classify unbalanced data. In 2021, Zhu et al. [21] proposed an ML-KNN integration scheme which can realize classification algorithm recommendations, and the scheme can take advantage of the diversity of different data features. Levchenko et al. [22] implemented a KNN query on a large time-series database based on iSAX and sketch. In the meantime, for protecting the users' private data, researchers also carry out a lot of research on how to perform KNN classification on the ciphertext. As early as 2009, Wong et al. [23] designed an asymmetric vector product preservation encryption scheme (ASPE) to support KNN calculations on encrypted data, which supports KNN computation of encrypted data by retaining a special type of scalar product, but the scheme assumes that the querying user is fully trusted, which is not suitable for practical application in complex network environments. In 2013, Zhu et al. [24] proposed a secure KNN calculation scheme for encrypted cloud data, and it does not need to share the key with the querying user, but they increase the communication overhead compared to the ASPE scheme. In 2014, Elmehdwi et al. [25] proposed a secure KNN scheme in an outsourcing environment based on the Paillier homomorphic encryption scheme, which can query the data without leaking any information to the cloud server by using the feature of homomorphic encryption and hides the query and data access

mode of users, but the computing cost is large. In 2015, Xia et al. [26] proposed a secure dynamic multikeyword ranking search scheme based on encrypted cloud data, which achieves sublinear search time and handles document deletion and insertion flexibly with a special tree-based index structure. Samanthula et al. [27] proposed a KNN classification scheme, which can be used for encrypted data stored in the cloud based on Paillier and multiparty security protocol. In 2016 and 2017, based on the Paillier homomorphic encryption scheme, similarly, Kim et al. [28, 29] designed a privacy protection KNN classification algorithm using the tree index structure and Yao's garbled code, respectively. However, the KNN classification scheme based on Paillier homologous encryption scheme is inefficient to compute, has some limitations in calculation method, and has a high computation cost. Li et al. [30] presented two secure and effective dynamic searchable symmetric encryption (SEDSSE) schemes for medical cloud data, they combined the secure KNN scheme and ABE technology to design a dynamic searchable symmetric encryption scheme and a key sharing scheme, and they implement both forward and backward privacy security and propose an enhanced scheme to effectively solve the key sharing problem caused by search encryption using KNN. In 2018, Wu et al. [31] used Paillier and ElGamal encryption schemes to implement a secure KNN classification scheme on a semantically secure hybrid encrypted cloud database. Later, Liu et al. [32] proposed a privacy protection KNN classification scheme in the dual cloud model based on secret sharing and additive homomorphic cryptography. In 2020, Parvin et al. [33] developed an electronic medical record analysis system on the blockchain based on KNN and LDA algorithms to automatically and safely share medical data sets among medical experts. In the same year, in order to realize the classification of large-scale ciphertext data in distributed servers, Yang et al. [34] proposed a vector homomorphic encryption (VHE) scheme through constructing key switching matrix and noise matrix and constructed a secure distributed KNN classification algorithm (seed KNN) based on it. Recently, Kim et al. [35] proposed an index-based KNN query processing algorithm and improved query processing efficiency through Yao's garbled code and data packaging technology. Liu et al. [36] achieved secure KNN classification by a secure and efficient query processing (SecEQP) scheme, which encodes location information through a projection function and implements location-based query processing based on the encrypted geospatial data stored in the cloud.

3. Preliminaries

3.1. CKKS Homomorphic Encryption Algorithm. In 2017, Cheon et al. [10] proposed a scheme of homomorphic encryption, CKKS, which supports real number/complex number approximations. This article mainly analyzes the CKKS homomorphic encryption algorithm. As shown in Figure 2 which is drawn referring to Cheon's Report [10], the following describes the main algorithm flow of CKKS.

Set safety parameters λ, and choose the power of two integers N. Set distributions $\chi_{\text{key}}, \chi_{\text{err}}, \chi_{\text{enc}}$ for key, learning with errors, and encryption on $R = \mathbb{Z}[X]/(X^N + 1)$ individually. To get a basic integer p and the number of levels L, set the modulus of the ciphertext $q_l = p^l (1 \le l \le L)$, where l is the level of ciphertext, then create an integer P at random, and output $pp = (N, \chi_{\text{key}}, \chi_{\text{err}}, \chi_{\text{enc}}, L, q_l)$:

(1) $\textbf{Key Gen}(\textbf{params}) \longrightarrow (\textbf{pk}, \textbf{sk}, \textbf{ks}, \textbf{rk}_r, \textbf{ck})$. Randomly generate $s \longleftarrow \chi_{\text{key}}$, and set private key$sk \longleftarrow (1, s)$. Randomly generate $a \longleftarrow U(R_{q_L})$ and $e \longleftarrow \chi_{\text{err}}$, set $pk \longleftarrow (-as + e, a) \in R^2_{q_L}$. Randomly generate $a' \longleftarrow U(R_{q2/L})$ and $e' \longleftarrow \chi_{\text{err}}$, and set evk $\longleftarrow (-a's + e' + q_L s^2, a') \in R^2_{q2/L}$.

(2) $\textbf{Encrypt}(m, pk) \longrightarrow \textbf{ct}$. Randomly generate $r \longleftarrow \chi_{enc}$ and $e_0, e_1 \longleftarrow \chi_{\text{err}}$; output the ciphertext $\textbf{ct} = r \cdot pk + (m + e_0, e_1) (\text{mod}\, q_L)$.

(3) $\textbf{Decrypt}(\textbf{ct}, sk) \longrightarrow m$. For ciphertext of level l, compute and output the plaintext $m' = \langle \textbf{ct}, \textbf{sk} \rangle (\text{mod}\, q_l)'$.

(4) $\textbf{Add}(\textbf{ct}, \textbf{ct}') \longrightarrow \textbf{ct}_{\textbf{add}}$. For the ciphertext $\textbf{ct}, \textbf{ct}'$ of the same level l, compute and output the addition result $\textbf{ct}_{\text{add}} = \textbf{ct} + \textbf{ct}' (\text{mod}\, q_l)$.

(5) $\textbf{Mult}_{\text{ks}}(\textbf{ct}, \textbf{ct}') \longrightarrow \textbf{ct}_{\textbf{mult}}$ $\textbf{ct} = (c_0, c_1), \textbf{ct}' = (c_0', c_1') \in R^2_{q_l}$. Compute $(d_0, d_1, d_2) = (c_0 c_0', c_0 c_1' + c_0' c_1, c_1 c_1') (\text{mod} q_l)$; output the result of ciphertext multiplication $\textbf{ct}_{\textbf{mult}} = (d_0, d_1) + \lfloor P^{-1} \cdot d_2 \cdot \textbf{sk} \rfloor \ (\text{mod}\, q_l)$. Given that the CKKS encryption scheme has a nature of being homomorphic, the cloud server computes ciphertext equivalent to plaintext, which would ensure both privacy of the user and the efficiency of the encryption.

3.2. K-Nearest Neighbor. Proposed by Cover and Hart in 1968, KNN came into the public view quite some time ago [37], and it ranks among the simplest algorithms for machine learning. Due to its simple structure and remarkable classification performance, it became one of the most popular algorithms in the data mining and statistics fields, granting it a seat among the top ten data mining algorithms [6], and is used very commonly in classification, regression, and missing value interpolation and other fields [38–40]. At present, many algorithms for machine learning have been developed to better determine the value of k in the KNN algorithm and the distance measurement algorithm. Being one of the most classic data mining classification technology algorithms, the main idea of the KNN nearest neighbor classification algorithm is to establish the category objects to be classified, based on the category of the majority of samples in a certain range adjacent to the object to be classified. The working principle of the KNN nearest neighbor classification algorithm is to compare the sample waiting for classification with the others which are of established categories in the database, and to compute the similarity between these two sets of different samples, and select the k samples of known categories with the closest similarity to the sample to be classified. According to the voting rule (minority obeys

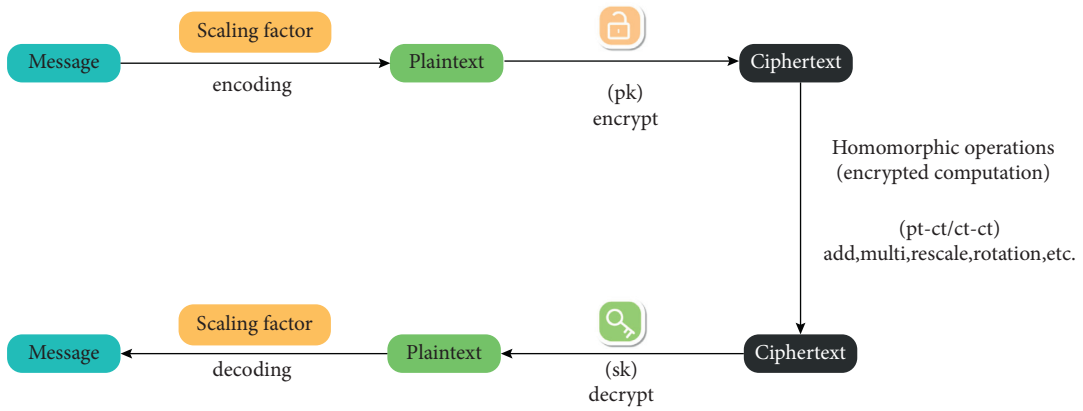

Figure 2: Ciphertext matrix transpose operation.

the majority), the category of the sample to be classified falls in rank with the category which has the highest proportion of the k-nearest samples. Suppose that we have samples of known categories $[(\mathbf{X}_1, Y_1), (\mathbf{X}_2, Y_2), \ldots, (\mathbf{X}_n, Y_n)]$, where X represents the characteristic index of the sample, and Y represents the category label of the sample. For a given sample $\mathbf{X}'$ to be classified, we select the k samples with the highest similarity in the vicinity of $\mathbf{X}'$, and these samples vote for the category of $\mathbf{X}'$ according to their own category. The category label with the most votes is called category Y' of $\mathbf{X}'$, as shown in Figure 3. The green dots represent samples to be classified, and the blue squares and red triangles represent the other two samples of known categories. When $k = 3$, the proportion of red triangles in the nearest neighboring range is 2/3, and the green dots are judged as red triangle samples. When $k = 5$, the proportion of the blue square in the nearest neighbor is 3/5, and the green dot is judged as a blue square sample.

The KNN method is more suitable than other methods in the sample to be classified with more intersections or overlaps in the class domain. There are many methods for calculating similarity in the KNN algorithm, such as the Euclidean distance, cosine similarity, Pearson correlation, Manhattan distance, and Chebyshev distance. The most commonly used method is the Euclidean distance.

3.3. Ciphertext Matrix Transpose Operation. Since this scheme is implemented in the TenSEAL homomorphic encryption library, although TenSEAL provides the ciphertext matrix multiplication function mul and the inner product function dot, it does not provide a ciphertext transpose function. Therefore, this part will introduce the process of transposing ciphertext matrix in the TenSEAL homomorphic encryption library. TenSEAL provides a very useful function reshape; its function can be expressed as $\mathbf{A}^{m\times n} = \mathbf{A}^{1\times m*n}.\text{reshape}([m, n])$. Suppose that there is a ciphertext matrix $\mathbf{A}^{m*n}$. First, the transition matrix $\mathbf{D}^{m*n\times m*n}$ is generated, and the transposition process can be shown in Figure 4.

It can be seen that $\mathbf{A}^{m\times n}$ is converted to $\mathbf{A}^{1\times m+n}$ through $\text{reshape}([1, m+n])$. Afterward, the internal elements are rearranged through $\text{dot}(\mathbf{D}^{m+n\times m+n})$ and finally transposed through $\text{reshape}([n, m])$.

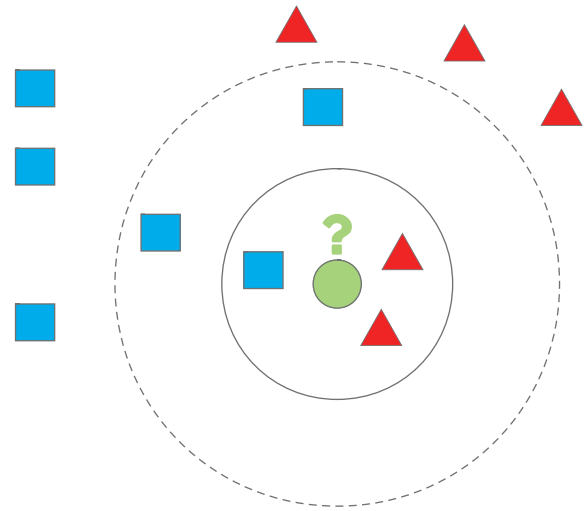

Figure 3: KNN algorithm example.

3.4. Symbols and Parameters. In order to show the algorithm in this article more intuitively, we briefly introduce the related symbols that are often used in this article, as shown in Table 1. The vectors are illustrated in lowercase bold letters and the matrices are shown in uppercase bold letters. Add enc_ in front to indicate the ciphertext form of the data.

4. System Models

4.1. Proposed Model. According to Figure 5, the CKKSKNNC protocol model designed in this paper is composed of two parts, namely, the user (USER) and the cloud service provider (CSP). Among them, CSP can provide remote storage and computing services for users, which is "honest and curious". Users have a large amount of local data and enjoy the services provided by CSP. The division of labor of each part is as follows:

(1) USER: generate public and private keys locally, encrypt data and upload them to CSP, and decrypt ciphertext computation results

(2) CSP: provide remote storage and services for computing for USER, with capable storage and

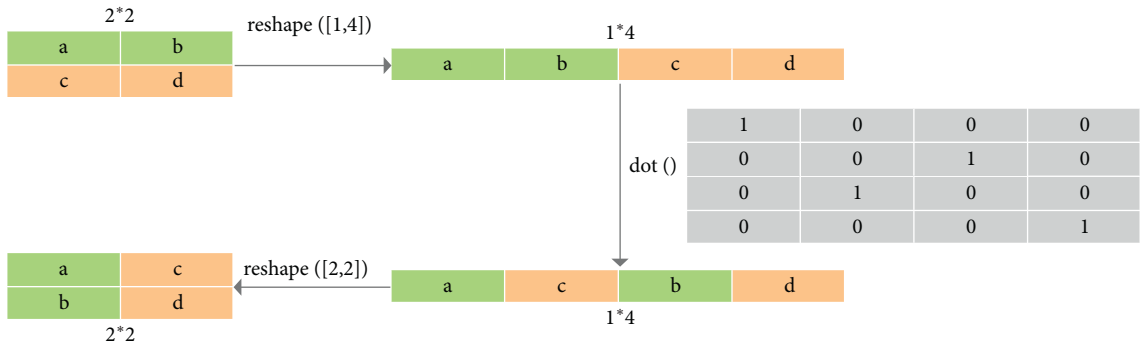

Figure 4: Ciphertext matrix transpose operation.

Table 1: Notations.

Symbols	Description
$\mathbf{X}$	Known category sample
$\tilde{\mathbf{X}}$	Standardized data sample
Y	Category label of known category sample
$\mathbf{X}\prime$	Sample to be classified
Y'	Category label of sample to be classified
$x_{ij}, \quad i \in [1,n],\ j \in [1,m]$	The j-th feature index of the i-th sample
(pk, sk)	Public and private key
result	Similarity
d	Euclidean distance
p	Pearson correlation coefficient
c	Cosine similarity

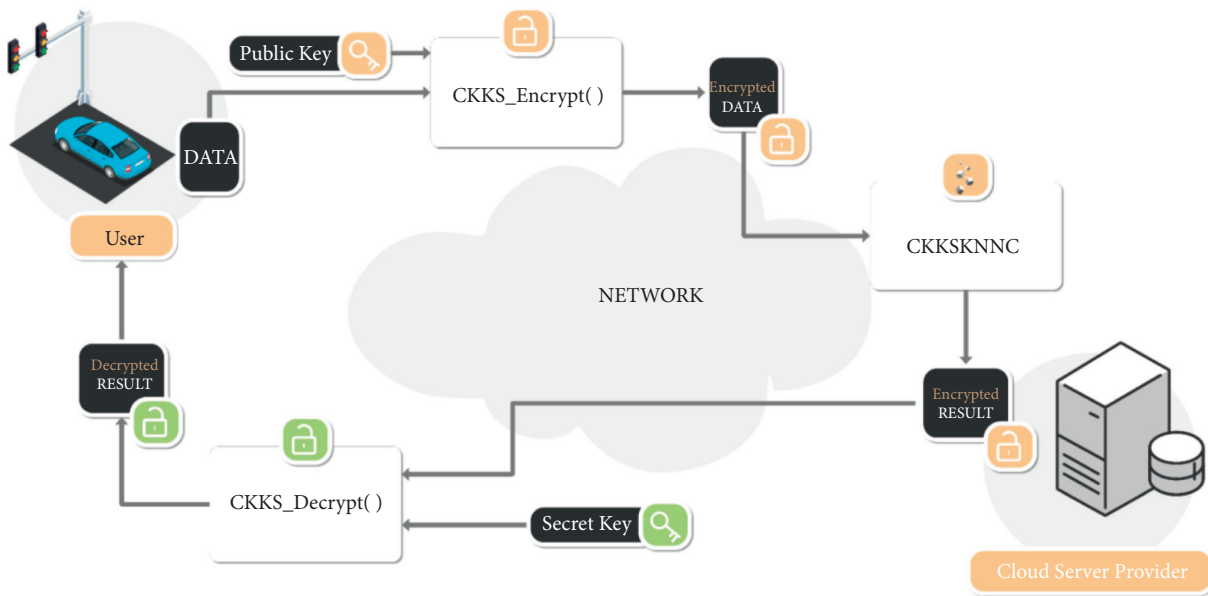

Figure 5: CKKSKNNC protocol model.

computing capabilities, taking charge for storing the ciphertext data uploaded by USER, calculating the similarity between the encrypted sample to be classified and other ciphertext samples, and returning the ciphertext result to USER

First of all, USER generates public and private keys locally, encrypts locally known samples of classes, and sends them to CSP. CSP accepts the ciphertext samples sent by USER and stores them. When USER receives a new sample to be classified, USER encrypts the sample to be classified locally and delivers it to the CSP. The CSP accepts and

computes the similarity between the encrypted sample to be classified and other ciphertext samples in the server and sends the ciphertext computation result to USER. The USER accepts the computation result from the CSP and decrypts them. Then, the USER selects the nearest k samples and obtains the category label of the sample to be classified according to the voting rule.

4.2. Security Model. Since the CSP is "honest and curious", the transmission network may also be subject to malicious attacks. Therefore, we list the following security issues that may occur when users upload data to the cloud server for KNN classification:

(1) CSP may strictly abide by the designed protocol, but it can infer other additional information through the information legally received in the process of the protocol

(2) CSP attempts to steal USER's public and private keys and relies on stored ciphertext data samples to try and decipher the USER's plaintext data samples and private keys

(3) During the transmission process between the user-uploaded ciphertext data and the ciphertext result returned by the cloud server, data samples may be maliciously intercepted by hackers and be used to crack the user's sensitive data

5. System Algorithm

5.1. CKKSKNNC Framework. Assuming that the user has sample data $[(\mathbf{X}_1, Y_1), (\mathbf{X}_2, Y_2), \ldots, (\mathbf{X}_n, Y_n)]$, which is known categories to be uploaded locally, where $\mathbf{X}_j = (x_1, x_2, \ldots, x_m)^T$, the system protocol framework is shown in Figure 6.

According to the protocol framework, the protocol algorithm is made up of two phases, namely, the data initialization phase and the classification phase. The specific operation procedures are listed as follows:

(1) Data initialization:

(a) First, USER standardizes the characteristic index of local data samples; compute $\widetilde{x}_{ij} = x_{ij} - \overline{x_j}/\sqrt{\mathrm{var}(x_j)}, i \in [1, n], j \in [1, m]$, where $\overline{x_j} = 1/n \sum_{i=1}^{n} x_{ij}$represents the average value of the j-th characteristic index, $\mathrm{var}(x_j) = 1/n - 1 \sum_{i=1}^{n} (x_{ij} - \overline{x_j})^2$ represents the standard deviation of the j-th characteristic index, then the standardized data is a $[(\widetilde{\mathbf{X}}_1, Y_1), (\widetilde{\mathbf{X}}_2, Y_2), \ldots, (\widetilde{\mathbf{X}}_n, Y_n)]$, the average value of its characteristic index is 0, the variance is 1, and it is dimensionless. *b*. USER generates public and private keys locally (pk, sk) and encrypts the characteristic index and category labels in the original data and standardized data, respectively, get $(\mathbf{enc_X}, \mathrm{enc}_Y)$and $(\mathbf{enc\_}\widetilde{\mathbf{X}}, \mathrm{enc}\_Y)$, and upload both to CSP for storage.

(2) Classification

(a) After receiving the new sample $\mathbf{X}'$ to be classified, USER first standardizes its characteristic index to obtain $\widetilde{\mathbf{X}}'$, then uses the public key pk to encrypt it to obtain $\mathbf{enc\_}\widetilde{\mathbf{X}}'$, and sends the encrypted result to the CSP as a query matrix.

(b) After receiving the query matrix, the CSP computes the similarity enc_result in the ciphertext between the sample waiting for classification and others that are of other known categories and returns it to USER.

(c) USER decrypts enc_result, selects the top k samples with the highest similarity, and obtains the category label Y' of the sample to be classified according to the voting rule.

5.2. Security Similarity Calculation. In the process of data mining and data analysis, there are many methods to measure the differences between samples. In the CKKSKNNC protocol, this paper uses the Euclidean distance, Pearson correlation coefficient, and cosine similarity to measure the similarity between samples.

5.2.1. Euclidean Distance. The Euclidean distance [41] is the most popular similarity measurement method. It has been widely used in various scenes such as face recognition. The traditional Euclidean distance computation method is to directly calculate the absolute distance between each point in the multidimensional space and the Euclidean distance between samples through the matrix inner product [42]. Two methods for calculating the Euclidean distance are introduced below. Method 1: since the ciphertext encrypted by the CKKS homomorphic encryption algorithm cannot be directly squared, the distance is not squared, and the ciphertext distance between the sample to be classified and the sample of the category is

$$\mathrm{enc}\_d_1 = \left(\mathbf{enc\_}\widetilde{\mathbf{X}}' - \mathbf{enc\_}\widetilde{\mathbf{X}}\right)^2 = \sum_{i=1}^{n} \left(\mathrm{enc}\_\widetilde{X}_i' - \mathrm{enc}\_\widetilde{X}_i\right)^2. \tag{1}$$

As the distance grows smaller, the similarity of the samples becomes higher. Method 2: before uploading data, USER computes $\widetilde{\mathbf{X}}_*' = (\mathbf{1}, \widetilde{\mathbf{X}}'^{\mathbf{T}}\widetilde{\mathbf{X}}', \widetilde{\mathbf{X}}^{\mathbf{T}})^{\mathbf{T}}$ and $\widetilde{\mathbf{X}}_* = (\widetilde{\mathbf{X}}^{\mathbf{T}}, \widetilde{\mathbf{X}}, 1 - 2\widetilde{\mathbf{X}}^T)^{\mathbf{T}}$, respectively, and uploads data to CSP after being encrypted. CSP can directly compute the ciphertext distance between two samples through the inner product:

$$\begin{aligned} \mathrm{enc}d_2 &= \mathbf{enc}\widetilde{\mathbf{X}}_*^T \mathbf{enc}\widetilde{\mathbf{X}}_*' \\ &= \mathbf{enc}X'^T \mathbf{enc}_{\widetilde{\mathbf{X}}}' + \mathbf{enc}\widetilde{\mathbf{X}}^T \mathbf{enc}_{\widetilde{\mathbf{X}}} - 2\mathbf{enc}\widetilde{\mathbf{X}}'^T \mathbf{enc}_{\widetilde{\mathbf{X}}} \\ &= \left(\mathbf{enc}_{\widetilde{\mathbf{X}}}' - \mathbf{enc}_{\widetilde{\mathbf{X}}}\right)^2. \end{aligned} \tag{2}$$

The result of this method is the same as that of the first method. Although it increases the computational complexity, it can be batch-processed computation.

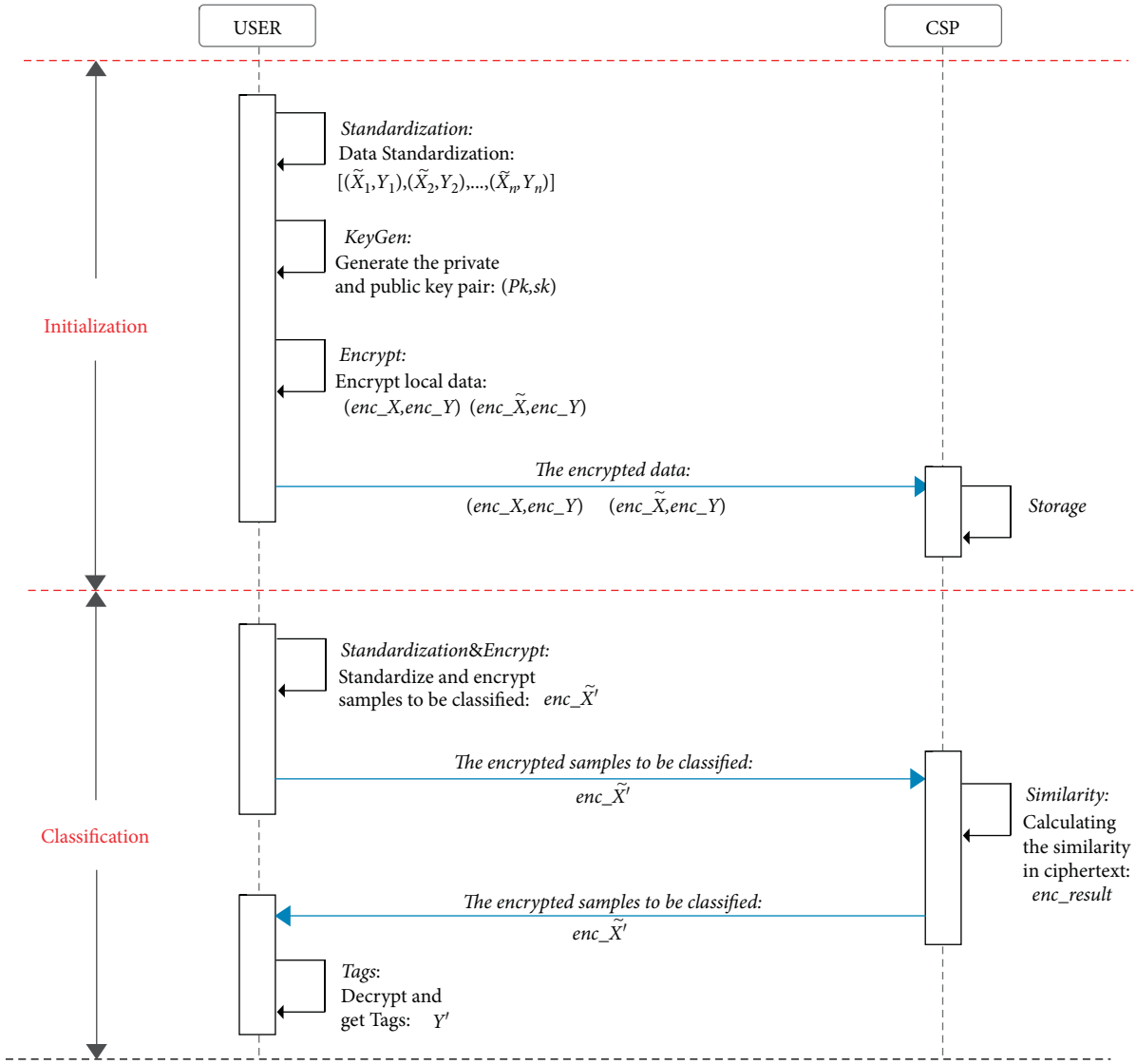

Figure 6: CKKSKNNC protocol framework.

5.2.2. Pearson's Correlation Coefficient. Since the magnitude of the different characteristic index of the sample has a greater impact on the Euclidean distance, in some applications, people often choose the Pearson correlation coefficient [43] that is not sensitive to the magnitude to measure the similarity between samples. The data sample has been standardized before uploading to the CSP; the computation process is as follows:

$$\begin{aligned} \text{enc\_}p &= \frac{(\mathbf{enc\_X} - \mathbf{enc\_\overline{X}})^T\left(\mathbf{enc\_X}' - \mathbf{enc\_\overline{X}\prime}\right)}{||\mathbf{enc\_X} - \mathbf{enc\_\overline{X}}||\left\|\mathbf{enc\_X}' - \mathbf{enc\_\overline{X}\prime}\right\|}, \\ &= \frac{\mathbf{enc\_X}^T\mathbf{enc\_}\tilde{\mathbf{X}}'}{n-1} \end{aligned} \tag{3}$$

5.2.3. Cosine Similarity. The angle cosine similarity is like the Pearson correlation coefficient and insensitive to the magnitude of the characteristic index. It is often used in the computation of text similarity, but it needs to be computed on the original data. It measures the similarity by calculating the cosine of the angle between both samples in the vector space. And the method pays more attention to what is different from the direction of one vector to another, rather than the distance measurement. Similarly, because the CKKS ciphertext cannot be directly used for square rooting, the cosine similarity computation process in the protocol is as follows:

$$\begin{aligned} \text{enc\_}c &= \left(\frac{\mathbf{enc\_X}'^T\mathbf{enc\_X}}{\left\|\mathbf{enc\_X}'\right\|\|\mathbf{enc\_X}||}\right)^2, \\ &= \frac{\left(\mathbf{enc\_X}'^T\mathbf{enc\_X}\right)^2}{\left(\mathbf{enc\_X}'^T\mathbf{enc\_X}''\right)\left(\mathbf{enc\_X}^T\mathbf{enc\_X}\right)}. \end{aligned} \tag{4}$$

In terms of actual implementation, the CKKS ciphertext cannot be directly performed division operations, so the CSP will actually return the two values of $(\mathbf{enc\_X}'^T\mathbf{enc\_X})^2$ and

$(\mathbf{enc\_X}'^{T}\mathbf{enc\_}X')(\mathbf{enc\_X}^{T}\mathbf{enc\_X})$ to the USER, and the USER will decrypt it and perform the division on the plaintext.

5.3. Batch. Assume that CSP stores ciphertext samples of known category $[(\mathbf{enc\_}\widetilde{\mathbf{X}}_1, \mathrm{enc\_}Y_1), (\mathbf{enc\_}\widetilde{\mathbf{X}}_2, \mathrm{enc\_}Y_2), \ldots, (\mathbf{enc\_}\widetilde{\mathbf{X}}_n, \mathrm{enc\_}Y_n)]$, when USER uploads multiple ciphertext samples to be classified $[\mathbf{enc\_X}'_1, \mathbf{enc\_X}'_2, \ldots, \mathbf{enc\_X}'_q]$, CSP needs to compute the similarity between each sample to be classified and each sample of a known category, and the encryption method is determined by the method of calculating the similarity.

5.3.1. Euclidean Distance. When CSP uses the Euclidean distance method 1 to compute similarity, batch processing cannot be performed. USER needs to encrypt each sample separately, and CSP needs to separately compute the ciphertext similarity between each sample to be classified and all of the other samples of known categories and returns the similarity matrix $\mathbf{enc\_D}_1 = [\mathrm{enc\_}d_{ij}], i \in [1, n], j \in [1, q]$, where $\mathrm{enc\_}d_{ij}$ is the similarity between the i-th known category sample and the j-th sample to be classified. When CSP uses the Euclidean distance method 2 to compute similarity, USER can directly encrypt the plaintext matrix of the sample to be classified and obtain the ciphertext matrix $\mathbf{enc\_}\widetilde{\mathbf{X}}' = [\mathbf{enc\_X}'_1, \mathbf{enc\_X}'_2, \ldots, \mathbf{enc\_X}'_q]$, CSP computes $[(\mathbf{enc\_}\widetilde{\mathbf{X}}_*)_1, (\mathbf{enc\_}\widetilde{\mathbf{X}}_*)_2, \ldots, (\mathbf{enc\_}\widetilde{\mathbf{X}}_*)_n]$, $[(\mathbf{enc\_}\widetilde{\mathbf{X}}_*')_1, (\mathbf{enc\_}\widetilde{\mathbf{X}}_*')_2, \ldots, (\mathbf{enc\_}\widetilde{\mathbf{X}}_*')_q]$, and similarity matrix is $\mathbf{enc\_D}_2 = [\mathrm{enc\_}d_{ij}] = (\mathbf{enc\_}\widetilde{\mathbf{X}}_*)^T(\mathbf{enc\_}\widetilde{\mathbf{X}}_*'), i \in [1, n], j \in [1, q]$, where $\mathrm{enc\_}d_{ij}$ is the similarity between the sample of i-th from a known category and the sample of j-th, which is yet to be classified.

5.3.2. Pearson's Correlation Coefficient. Similar to the above, CSP computes the similarity matrix $\mathbf{enc\_P} = [\mathrm{enc\_}p_{ij}] = (\mathbf{enc\_}\widetilde{\mathbf{X}})^T(\mathbf{enc\_}\widetilde{\mathbf{X}}')/\sqrt{(n-1)} * \sqrt{(q-1)}, i \in [1, n], j \in [1, q]$, where $\mathrm{enc\_}p_{ij}$ is between the sample of i-th from a known category and the sample of j-th, which is yet to be classified.

5.3.3. Cosine Similarity. When CSP uses cosine similarity to compute similarity, CSP first generates unit diagonal matrix $\Lambda_1^{n\times n}$和$\Lambda_2^{p\times p}$, vector $\mathbf{e}_1 = [1, 1, \ldots, 1]^{1\times n}$, and$\mathbf{e}_2 = [1, 1, \ldots, 1]^{1\times p}$; then, compute the distance matrix $\mathbf{L} = (\mathbf{enc\_X}^T\mathbf{enc\_X}).\mathrm{mul}(\Lambda_1^{n\times n}).\mathrm{dot}(e_1)$ and $\mathbf{L}\prime = (\mathbf{enc\_X}'^T\mathbf{\_X}')\mathbf{enc}.\mathrm{mul}(\Lambda_2^{n\times n}).\mathrm{dot}(\mathbf{e}_2)$. Finally, compute the similarity matrix $\mathbf{enc\_C} = [\mathrm{enc\_}c_{ij}] = (\mathbf{enc\_X}^T\mathbf{enc\_X}')^2/L^TL', i \in [1, n], j \in [1, q]$, where $\mathrm{enc\_}c_{ij}$ is the similarity between the sample of i-th from a known category and the sample of j-th, which is not yet classified; CSP returns $(\mathbf{enc_X}^T\mathbf{enc_X}')^2$and $\mathbf{L}^T\&\mathbf{L}\prime$ to the USER.

6. Security Analysis

According to the protocol model of CKKSKNNC, since the CSP is 'honest and curious', the USER's private key is only stored locally, and the CSP is only in charge of storing data and computing the user-uploaded ciphertext data; both the public and the private central information of the USER cannot be obtained. The security definition of the semitrusted model is listed as follows.

Definition (security of semitrusted model): assume function $f(x, y)$, where $f_1(x, y)$ and $f_2(x, y)$ are, respectively, the first and second elements of $f(x, y)$. Assume that Γ is a two-party protocol used to compute $f(x, y)$. $\mathrm{PARTY}_1(x, y)$ is a role that implements the Γ protocol, where $\mathrm{PARTY}_1(x, y) = (x, r, p_1, p_2, \ldots, p_t)$, x represents input, r represents randomness, and p_i represents the i-th data accepted. Also $\mathrm{PARTY}_1(x, y) = (y, r, p_1, p_2, \ldots, p_t)$ is available. If there exists probabilistic polynomial-time algorithms V_1 and V_2, such that

$$\begin{aligned} V_1(x, f_1(x, y)) &\longmapsto \mathrm{PARTY}_1(x, y), \\ V_2(y, f_2(x, y)) &\longmapsto \mathrm{PARTY}_2(x, y), \end{aligned} \tag{5}$$

where computational indistinguishability is represented by $\longmapsto$, it is said that computing $f(x, y)$ is secure when Γ protocol is against semitrusted adversary.

Theorem 1. *Under the semihonest model, CKKSKNNC is provably secure. CSP fails to obtain any helpful information from the stored data set or query matrix.*

Proof. In the protocol, the data set and query matrix that CSP can obtain are transmitted in the ciphertext. The view of CSP is $\mathrm{PARTY} = (e_1, e_2, \ldots, e_n)$, where e_i are all ciphertexts and n is the number of ciphertext data that CSP can access. We can design a simulator V to simulate the USER view and then set $V = (r_1, r_2, \ldots, r_n)$, where r_i are all random numbers. Since CKKS homomorphic encryption scheme is a semantically secure encryption scheme, V is computationally no different from PARTY. Thus, under the semihonest model, CKKSKNNC is provably secure, and CSP cannot gain any helpful information from the stored data set and query matrix. □

Theorem 2. *Assume that CSP and other attackers cannot perform key recovery attacks on stored or stolen ciphertext data and computation results, so they cannot recover the user's original data and keys.*

Proof. According to the protocol algorithm, in the transmission process, USER's sample data, category label, and query matrix are all transmitted in the form of CKKS ciphertext. Its security is protected by the CKKS homomorphic encryption scheme, which grants security, and security is resolved by its own algorithm. Therefore, the CSP cannot restore the original user data and keys through the stored user data and intermediate computation results. In the process of returning the result, the similarity computation result is transmitted to the user in ciphertext for decryption, so the attacker cannot recover the user's original sensitivity from the intercepted ciphertext data during the process of transmission and the data in the stolen cloud server data. In addition, no matter what method is used to compute the similarity, multiplications only need 3 times at most.

Therefore, the CKKSKNNC protocol algorithm does not have additional special requirements for the parameters of the CKKS encryption scheme. □

7. Experimental Test

With the aim of well testing the potency of the scheme in a proposition, we conduct our experiments on Windows10 operating system with Intel® Core™ i7-7700HQ CPU @ 2.80 GHz/16 GB RAM, using PyCharm 2020.1.1 × 64 to call TenSEAL-0.1.4 library to implement the CKKS encryption scheme, take poly_ modulus_ degree = 8192, coeff_ mod_ bit_ sizes = [50, 30, 30, 50], scale = 30 as the parameter of CKKS homomorphic encryption scheme, and test on IRIS data set.

7.1. Efficiency of Similarity Calculation. In this part, we test the computational efficiency of different similarity algorithms. We randomly selected 100 groups of IRIS data set as the known class samples and randomly selected the remaining 10, 20, 30, 40, and 50 groups of samples as the samples to be classified to form the test set, recorded the time to compute the similarity on the ciphertext, and recorded the results of 30 experiments, and the average value is regarded to be the final experimental data. The computational efficiency of the four similarity algorithms is shown in Figure 7.

In this part, we did not record the time to compute the transposition of the ciphertext matrix, because we make matrix transposition default as preprocessing work. It shows that as the number of samples to be classified in the test set increases, the computation costs of the four similarity algorithms increase linearly. Among them, the Euclidean distance 2 has the highest computation efficiency, and cosine similarity computation has the lowest efficiency because it performs more cipher multiplications. But it is worth mentioning that the ciphertext in the Euclidean distance 1 uses the CKKS_Vector data type, and other methods use the CKKS_Tensor data type. The storage overhead of different test sets is shown in Table 2 (unit: byte).

It shows that, with the rising amount of samples, the ciphertext data set of CKKS_Vector type occupies a linear increase in memory and occupies more memory than the same number of ciphertext data sets of CKKS_Tensor type, while the ciphertext data set of CKKS_Tensor type occupies a constant memory change. Therefore, when dealing with big data, try to avoid using the Euclidean distance 1 and cosine similarity to compute the similarity.

7.2. Accuracy of Similarity Calculation. In this part, we take a random number of samples to be classified between 35 and 45 as the data set and test the classification accuracy of the four similarity algorithms when $k = 3$, 5, 7, and 9, respectively. We record the results of 30 experiments and take the average value as the final experimental data. The computation accuracy of the four similarity algorithms is shown in Figure 8.

It shows that the accuracy of the Euclidean distance and cosine similarity is stable at about 97%, but the accuracy of the Pearson correlation coefficient is as low as about 65%. Therefore, when doing the KNN classification algorithm, try to avoid using the Pearson correlation coefficient to compute similarity degree.

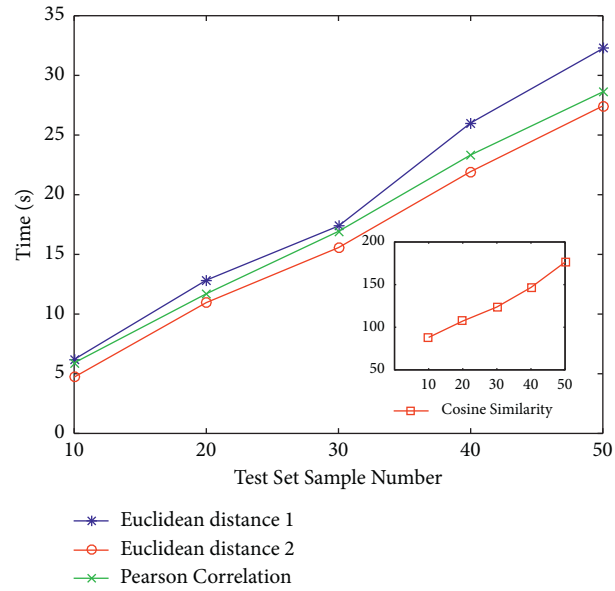

Figure 7: Computation efficiency of four similarity algorithms in different test sets.

Table 2: Storage costs for different sample sizes.

Type\samples	10	20	30	40	50
CKKS-vector	192	264	344	432	528
CKKS-tensor	56	56	56	56	56

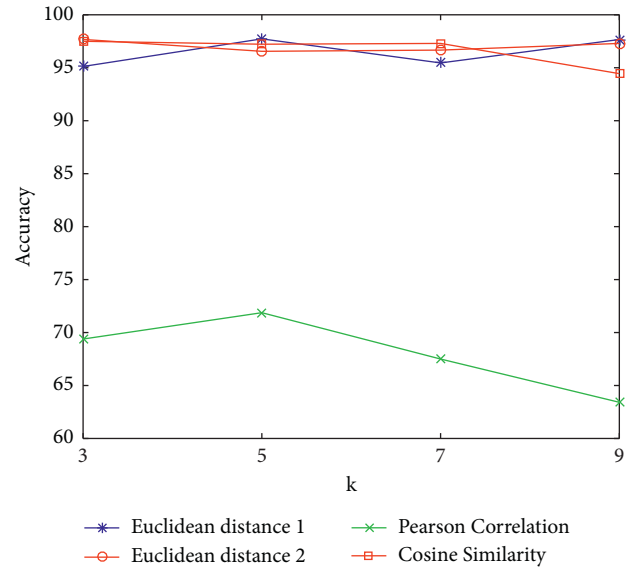

Figure 8: The accuracy of the four similarity algorithms in different k values.

7.3. Comprehensive Performance Comparison. Because the mature security KNN classification schemes applied in the market are ground on the Paillier homologous encryption

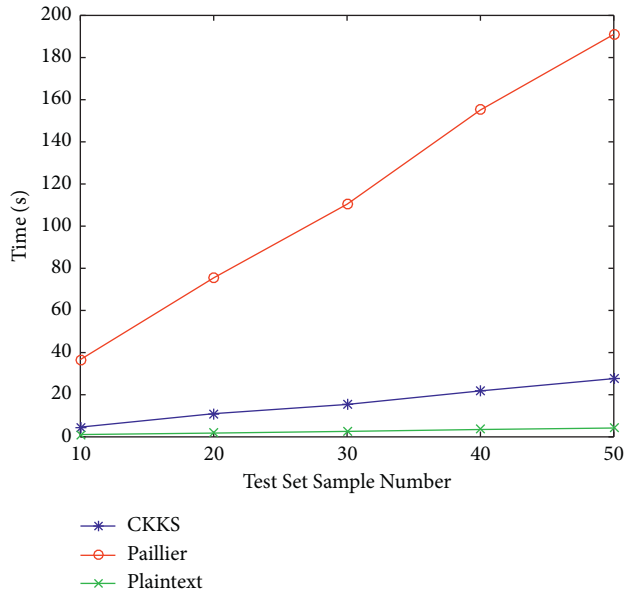

Figure 9: Computing efficiency of two encryption schemes in different test sets.

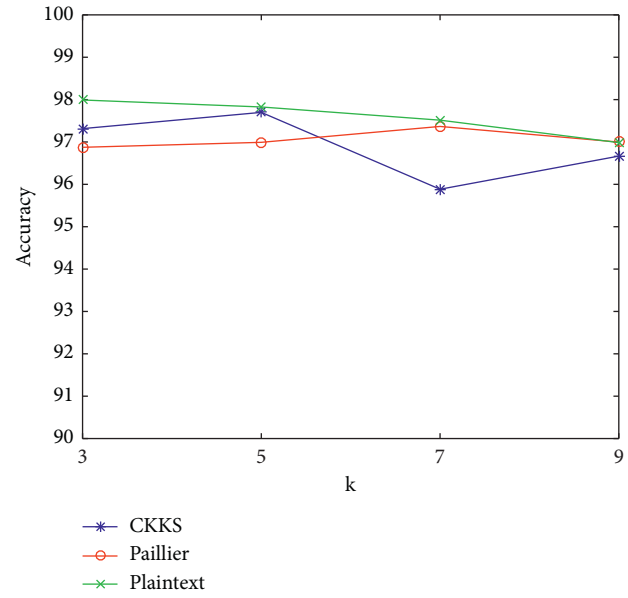

Figure 10: The accuracy of three schemes in different k values.

scheme, we test the efficiency and accuracy of computing the Euclidean distance in the Paillier and CKKS homologous encryption schemes with the same encryption parameters and in plain text. It should be emphasized that since the CKKS scheme is implemented in the TenSEAL homologous encryption library, which encapsulates the seal encryption library based on C++ into a dynamic library called python, the speed of the CKKS scheme depends on the efficiency of the sealed library in C++. Therefore, we did a comparative experiment in C++ with the Paillier homologous encryption scheme by calling NTL and GMP libraries with encryption parameter $|N| = 1024$. First, we compare the computational efficiency of the three schemes. We randomly selected the remaining 10, 20, 30, 40, and 50 groups of samples as the test set to be classified and calculated the similarity time as shown in Figure 9.

Clearly, the efficient CKKS schemes are more computational and closer to plain text than the Paillier schemes, and CKKS can support batch processing of data. In the encryption mode, the CKKS scheme can directly encrypt the data matrix, while the Paillier scheme can only encrypt numbers one by one and does not support floating-point operation. Then, we put together, in comparison, the accuracies of the three different schemes. We take a random number of samples to be classified between 35 and 45 as a dataset and test the classification accuracy of the four similarity algorithms at $k = 3$, 5, 7, and 9, respectively. The result can be found in Figure 10.

It is evident that whether the CKKS scheme or the Paillier scheme is used for security calculation, the calculation accuracy is not different from that calculated directly in plain text. We then tested the storage costs of the three schemes in datasets with different sample sizes, as shown in Table 3 (in bytes).

Table 3: Storage costs for different sample sizes.

Type\samples	10	20	30	40	50
CKKS-tensor	56	56	56	56	56
Paillier	144	224	304	384	464
Plaintext	144	224	304	384	464

As the number of samples goes up, the encrypted dataset of the CKKS scheme occupies the least memory and remains unchanged, but the Paillier scheme and the plain scheme occupy more memory and increase linearly. The secure KNN classification algorithm that chooses the CKKS scheme to process large data has absolute advantages.

8. Conclusions

To protect sensitive privacy data of cloud servers and users during transmission while meeting classification accuracy and computational efficiency requirements of classification algorithms, this paper implements a secure KNN classification scheme in ciphertext domain for Cyberspace (CKKSKNNC), based on the KNN classification scheme and CKKS algorithm. We use the TenSEAL homomorphic encryption library to implement the CKKS homomorphic encryption scheme and select two schemes of the Euclidean distance, Pearson correlation coefficient, and cosine similarity as the algorithm for calculating similarity in the KNN classification algorithm and test the computational efficiency, storage cost, and classification accuracy of the four similarity algorithms on IRIS data set. Through experimental tests, we found that the Euclidean distance Scheme 1 has the largest storage cost, the computation efficiency of cosine similarity is the lowest, and the classification accuracy of Pearson's correlation coefficient is the lowest. Nevertheless, the specific algorithm used as the similarity algorithm varies depending on the specific data.

Data Availability

Marked datasets, which support the conclusion of this study, can be obtained upon request to the corresponding authors.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the Foundation of State Key Laboratory of Public Big Data (No. 2019BDKFJJ008), Engineering University of PAP's Funding for Scientific Research Innovation Team (No. KYTD201805), and Engineering University of PAP's Funding for Key Researcher (No. KYGG202011).

References

[1] X. Xie, X. Yang, X. Wang, H. Jin, D. Wang, and X. Ke, "BFSI-B: an improved K-hop graph reachability queries for cyber-physical systems," *Information Fusion*, vol. 38, pp. 35–42, 2017.

[2] L. Qi, X. Wang, X. Xu, W. Dou, and S. Li, "Privacy-aware cross-platform service recommendation based on enhanced locality-sensitive hashing," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1145–1153, 2021.

[3] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[4] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on Systems, Man, And Cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[5] S. Zhang, "Nearest neighbor selection for iteratively KNN imputation," *Journal of Systems and Software*, vol. 85, no. 11, pp. 2541–2552, 2012.

[6] X. Wu, V. Kumar, and J. R. Quinlan, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.

[7] T. Wang, Z. Qin, S. Zhang, and C. Zhang, "Cost-sensitive classification with inadequate labeled data," *Information Systems*, vol. 37, no. 5, pp. 508–516, 2012.

[8] H. Zhou, N. Li, X. Che, and X. Yang, "Multi-key fully homomorphic encryption scheme over prime power cyclotomic rings," *Netinfo Security*, vol. 20, no. 5, pp. 83–87, 2020.

[9] N. Li, H. Zhou, X. Che, and X. Yang, "Design of directional decryption protocol based on multi-key fully homomorphic encryption in cloud environment," *Netinfo Security*, vol. 20, no. 6, pp. 10–16, 2020.

[10] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 409–437, Hong Kong, China, December 2017.

[11] G. Góra and A. Wojna, "RIONA: a classifier combining rule induction and k-NN method with automated selection of optimal neighbourhood," in *Proceedings of the European Conference on Machine Learning*, pp. 111–123, Helsinki, Finland, August 2002.

[12] B. Li, Y. W. Chen, and Y. Q. Chen, "The nearest neighbor algorithm of local probability centers," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 1, pp. 141–154, Feb. 2008.

[13] X. Zhu, H.-I. Suk, and D. Shen, "Multi-modality canonical feature selection for alzheimer's disease diagnosis," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 162–169, Boston, MA, USA, September 2014.

[14] J. Wang, P. Neskovic, and L. N. Cooper, "Neighborhood size selection in the k-nearest-neighbor rule using statistical confidence," *Pattern Recognition*, vol. 39, no. 3, pp. 417–423, 2006.

[15] U. Lall and A. Sharma, "A nearest neighbor bootstrap for resampling hydrologic time series," *Water Resources Research*, vol. 32, no. 3, pp. 679–693, 1996.

[16] D. Cheng, S. Zhang, Z. Deng, Y. Zhu, and M. Zong, "kNN algorithm with data-driven k value," *Advanced Data Mining and Applications*, in *Proceedings of the International Conference on Advanced Data Mining & Applications*, pp. 499–512, Guilin, China, December 2014.

[17] X. Zhu, S. Zhang, Z. Jin, Z. Zhang, and Z. Xu, "Missing value estimation for mixed-attribute data sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 1, pp. 110–121, Jan. 2011.

[18] Z. Deng, X. Zhu, D. Cheng, M. Zong, and S. Zhang, "Efficient k NN classification algorithm for big data," *Neurocomputing*, vol. 195, pp. 143–148, 2016.

[19] S. Zhang, X. Li, M. Zong M, X. Zhu, R. Wang, and X. Zhu, "Efficient kNN classification with different numbers of nearest neighbors," *IEEE Transactions On Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 1774–1785, 2017.

[20] S. Zhang, "Cost-sensitive KNN classification," *Neurocomputing*, vol. 391, pp. 234–242, 2020.

[21] X. Zhu, C. Ying, J. Wang, J. Li, and X. Lai, "Ensemble of ML-KNN for classification algorithm recommendation," *Knowledge-Based Systems*, vol. 221, Article ID 106933, 2021.

[22] O. Levchenko, B. Kolev, D.-E. Yagoubi, R. Akbarinia, F. Masseglia, and T. Palpanas, "BestNeighbor: efficient evaluation of kNN queries on large time series databases," *Knowledge and Information Systems*, vol. 63, no. 2, pp. 349–378, 2021.

[23] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, pp. 139–152, Rhode Island, USA, July 2009.

[24] Y. Zhu, R. Xu, and T. Takagi, "Secure k-NN computation on encrypted cloud data without sharing key with query users," in *Proceedings of the 2013 International Workshop on Security in Cloud Computing*, pp. 55–60, Hangzhou, China, May 2013.

[25] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *Proceedings of the 2014 IEEE 30th International Conference on Data Engineering*, pp. 664–675, Chicago, IL, USA, April 2014.

[26] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 340–352, 2015.

[27] B. K. Samanthula, Y. Elmehdwi, and W. Jiang, "K-nearest neighbor classification over semantically secure encrypted relational data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 5, pp. 1261–1273, 2015.

[28] K. I. Kim, H. J. Kim, and J. W. Chang, "A kNN query processing algorithm using a tree index structure on the

encrypted database," in *Proceedings of the 2016 International Conference on Big Data and Smart Computing (BigComp)*, pp. 93–100, Hong Kong, China, January 2016.

[29] H. J. Kim, H. I. Kim, and J. W. Chang, "A privacy-preserving kNN classification algorithm using Yao's garbled circuit on cloud computing," in *Proceedings of the 2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, pp. 766–769, Austin, TX, USA, June 2017.

[30] H. Li, Y. Yang, Y. Dai, Y. Shui, and X. Yong, "Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data," *IEEE Transactions on Cloud Computing*, vol. 8, no. 2, pp. 484–494, 2017.

[31] W. Wu, J. Liu, H. Rong, H. Wang, and M. Xian, "Efficient k-nearest neighbor classification over semantically secure hybrid encrypted cloud database," *IEEE Access*, vol. 6, pp. 41771–41784, 2018.

[32] L. Liu, J. Su, X. Liu, R. Chen, and K. Huang, "Toward highly secure yet efficient KNN classification scheme on outsourced cloud data," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9841–9852, 2019.

[33] S. Parvin, S. F. Nimmy, S. Venkatraman, S. Abed, and A. Gawanmeh, "A KNN approach for blockchain based electronic health record analysis," in *Proceedings of the International Conference on Systems Engineering*, pp. 455–464, Las Vegas, NV, USA, August 2020.

[34] H. Yang, S. Liang, J. Ni, H. Li, and X. S. Shen, "Secure and efficient k NN classification for industrial internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 11, pp. 10945–10954, 2020.

[35] H. J. Kim, H. J. Lee, and J. W. Chang, "A secure and efficient query processing algorithm over encrypted database in cloud computing," in *Proceedings of the 2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 219–225, Jeju Island, South Korea, January 2021.

[36] A. X. Liu and R. Li, "K-nearest neighbor queries over encrypted data," in *Algorithms for Data and Computation Privacy*, pp. 79–108, Springer, New York, NY, USA, 2021.

[37] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[38] H. Liu, X. Li, and S. Zhang, "Learning instance correlation functions for multilabel classification," *IEEE Transactions on Cybernetics*, vol. 47, no. 2, pp. 499–510, 2017.

[39] X. Zhu, X. Li, and S. Zhang, "Block-row sparse multiview multilabellearning for image classification," *IEEE Transactions on Cybernetics*, vol. 46, no. 2, pp. 450–461, 2015.

[40] K. Kaibing Zhang, X. Xinbo Gao, D. Dacheng Tao, and X. Xuelong Li, "Single image super-resolution with multiscale similarity learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 10, pp. 1648–1659, 2013.

[41] P.-E. Danielsson, "Euclidean distance mapping," *Computer Graphics and Image Processing*, vol. 14, no. 3, pp. 227–248, 1980.

[42] H. Yang, W. He, J. Li, and H. Li, "Efficient and secure kNN classification over encrypted data using vector homomorphic encryption," in *Proceedings of the 2018 IEEE International Conference on Communications*, pp. 1–7, Beijing, China, August 2018.

[43] K. Pearson, "Notes on the history of correlation," *Biometrika*, vol. 13, no. 1, pp. 25–45, 1920.