Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.038911
Article

An Efficient Encrypted Speech Retrieval Based on Unsupervised Hashing and B+ Tree Dynamic Index

Qiuyu Zhang*, Yugui Jia, Fangpeng Li and Letian Fan

School of Computer and Communication, Lanzhou University of Technology, Lanzhou, 730050, China
*Corresponding Author: Qiu-yu Zhang. Email: zhangqy@lut.edu.cn
Received: 03 January 2023; Accepted: 10 April 2023; Published: 09 June 2023

Abstract: Existing speech retrieval systems are frequently confronted with expanding volumes of speech data. The dynamic updating strategy applied to construct the index can timely process to add or remove unnecessary speech data to meet users' real-time retrieval requirements. This study proposes an efficient method for retrieving encryption speech, using unsupervised deep hashing and B+ tree dynamic index, which avoid privacy leakage of speech data and enhance the accuracy and efficiency of retrieval. The cloud's encryption speech library is constructed by using the multi-threaded Dijk-Gentry-Halevi-Vaikuntanathan (DGHV) Fully Homomorphic Encryption (FHE) technique, which encrypts the original speech. In addition, this research employs Residual Neural Network18-Gated Recurrent Unit (ResNet18-GRU), which is used to learn the compact binary hash codes, store binary hash codes in the designed B+ tree index table, and create a mapping relation of one to one between the binary hash codes and the corresponding encrypted speech. External B+ tree index technology is applied to achieve dynamic index updating of the B+ tree index table, thereby satisfying users' needs for real-time retrieval. The experimental results on THCHS-30 and TIMIT showed that the retrieval accuracy of the proposed method is more than 95.84% compared to the existing unsupervised hashing methods. The retrieval efficiency is greatly improved. Compared to the method of using hash index tables, and the speech data's security is effectively guaranteed.

Keywords: Encrypted speech retrieval; unsupervised deep hashing; learning to hash; B+ tree dynamic index; DGHV fully homomorphic encryption

1 Introduction

The quantity of multimedia data stored in the cloud has increased as a result of the fast development of cloud storage and information technology, including images, videos, and audio, increases daily. The bottleneck problem in speech retrieval is that retrieve the required speech data accurately and quickly from the massive speech data in the cloud [1]. Data stored in the cloud, which is an untrustworthy third party, causes security and privacy issues [2,3].

The volume of speech data has rapidly increased in recent years, rendering it increasingly difficult for users to meet real-time retrieval demands with the current content-based speech retrieval. Therefore, constructing the index's structure and dynamic update strategy are the determinants of speech retrieval efficiency. Based on the different data structures and needs, there are already a variety of indices for people to choose from. For example, hash indices are highly effective when searching for key values, whereas B-Trees are more suitable for requests with range requirements, and Bloom-filters indices are used primarily to check the existence of data. Most index structures in the existing multimedia retrieval field use a tree structure [4]. The binary tree index structure is widely used, which can speed up data retrieval and reduce the retrieval time. However, binary tree index only supports range retrieval [5], data growth results in an unsuitable complex tree index structure for large-scale data retrieval. In addition, existing content-based encrypted speech retrieval systems [6–11] primarily utilize the hash index. The query efficiency is excellent for unique and equal-length hash codes, but the query efficiency is low when faced with a different index with the same hash value. In response to the above questions, scholars have proposed an index structure based on a B+ tree that supports range query and retrieval, can use indices to complete sorting. It supports the multi-column common joint index's leftmost matching concept, there is no hashing collision problem in the event of a high number of duplicate key-values [12,13]. However, the existing B+ tree index is designed for plaintext [14] and rarely uses an index for ciphertext.

Aiming at the existing retrieval performance is bad, the index table cannot be dynamically updated, and range retrieval for encrypted data are not supported. By using unsupervised deep hash and B+ tree dynamic indexing, this research proposes an efficient encrypted speech retrieval approach. This approach creates an index structure based on the external B+ tree index (the B+ tree index table), improving the retrieval efficiency of encrypted unsupervised binary hash codes, and implements the dynamic update of the B+ tree index table. The speech multi-threaded DGHV FHE scheme [15] developed in the previous research work is employed to encrypt the speech. The main contributions of this work are as follows:

1) The ResNet18-GRU network structure is used to create an unsupervised learning hash coding model, which can simultaneously extract deep feature and perform hash function learning on speech data, getting high-quality unsupervised compact binary hash codes, reducing computation and improving speech feature representation and system retrieval efficiency.
2) An external index structure based on the B+ tree is designed to enable the index table's dynamic update, reduce the index structure's complexity, save storage space, and improve the retrieval system.
3) To construct the encrypted speech library and encrypt the nodes of the external index structure based on the B+ tree, a multi-threaded DGHV FHE scheme [15] is used, reducing the risk of leakage of cloud-based speech data and unsupervised hash codes and improving the security of the speech retrieval system.

The rest of this paper is organized as follows: Section 2 reviews related research works. Section 3 elaborates on the proposed scheme, including building the external index structure of the B+ tree and unsupervised deep hashing speech retrieval method. Section 4 presents experimental simulations of the proposed speech retrieval method and performance comparisons with various existing audio/speech retrieval approaches. Section 5 concludes the study's work.

2 Related Works

2.1 Content-Based Encrypted Speech Retrieval

Scholars have proposed content-based encrypted speech retrieval technology to safeguard the privacy of speech data and indexes stored. Zhao et al. [6] introduced a novel retrieval algorithm over encrypted speech to increase the robustness, discrimination, and retrieval speed in large-scale datasets. Zhao et al. [7] suggested a syllable-level perceptual hashing retrieval method. Zhang et al. [8] proposed a retrieval method based on perceptual hashing second feature extraction. Zhang et al. [9] presented a deep hashing content-based encrypted speech retrieval method. The limitations of manual features and poor feature semantics in the feature extraction process of existing content-based encrypted speech retrieval methods, Zhang et al. [10] suggested a retrieval method for encrypted speech based on improved power normalized cepstrum coefficients (PNCC) and perceptual hashing. Zhang et al. [11] proposed an encrypted speech retrieval scheme relying on multiuser searchable encryption.

2.2 Unsupervised Deep Hashing Retrieval

Most existing deep learning-based speech retrieval adopts the supervised learning method. Although this technology can exploit the deep features of speech, it requires manual annotation of massive data, consuming a considerable amount of manual annotation resources. However, the unsupervised learning method does not need to label the data, and the unsupervised deep hashing technology has been successfully implemented in image retrieval [16,17]. Some scholars have also conducted unsupervised learning hash technology in speech retrieval [18–22]. To address the issue that the binary hash code cannot maintain similarity, resulting in low retrieval performance. Panyapanuwat et al. [18] proposed a audio retrieval method based on unsupervised learning hash. Panyapanuwat et al. [19] proposed a unsupervised deep hashing approach for content-based audio retrieval to address the issue that the binary hash code cannot adequately represent semantic similarity. Jia et al. [20] proposed a speaker recognition method based on adaptive clustering. Rao et al. [21] proposed a signal recognition method based on unsupervised. They developed an image processing approach for extracting the necessary keywords from a pair of utterances, employing a convolutional neural network as a binary classifier to distinguish true and false keyword match possibilities. Panyapanuwat et al. [22] proposed an unsupervised similarity-preserving hashing method for content-based audio retrieval. Li et al. [23] proposed cross-modal retrieval method based on semantic-guided autoencoder adversarial hashing. Li et al. [24] introduced a cross-modal hashing method based on the unsupervised knowledge distillation, which can use similarity information distilled from an unsupervised method to guide the supervised method. Mikriukov et al. [25] introduced a novel unsupervised cross-modal contrastive hashing method for text-image retrieval in remote sensing. Zou et al. [26] presented a cross-modal deep hashing framework based on the multi-label modality enhanced attention and self-supervised.

2.3 Index Technology

The index structure of the hash index table is used for content-based multimedia retrieval. The hash index tables have the same hash code value and length, the retrieval performance is excellent, but the range retrieval is not supported. Li [12] proposed a B+ tree index structure-based blockchain information fuzzy retrieval algorithm. Ho et al. [14] proposed a encrypted data retrieval method based on external B+ tree index. Du et al. [27] proposed a secure image retrieval scheme using an deep pairwise-supervised hashing algorithm. Gupta et al. [28] proposed a novel indexing method based on wavelet trees for retrieving data, requiring less memory to store the index and less time to fetch data

from the index. To support high-quality content-based retrieval over such a large volume of music data, Shen et al. [29] introduced a novel indexing technique called the effective music indexing framework to facilitate scalable and accurate music retrieval. Niu et al. [30] implemented a vector space model to reduce the complexity of the text and an efficient system for text retrieval and used the B+ tree to construct an index tree, minimizing the space complexity of the index structure and also increasing the retrieval speed of ciphertext transactions on the blockchain. Llaveshi et al. [31] proposed an algorithm that integrates a linear regression model into B+ tree search, which can predict the correct leaf nodes with 80% to 96% accuracy and reduce search time by 33% to 57%.

2.4 Index Dynamic Update

Most of the existing index tables are static. Due to the continuous data expansion, users must operate the entire index table to add, delete, change, and search the index structure, which is time-consuming, labor-consuming, and impractical. Therefore, the research on the dynamic update strategy of index structure is fundamental. Ferragina et al. [32] proposed a fully dynamic Piecewise Geometric Model (PGM) index to increase search efficiency and reduce the storage capacity of indices. Wang et al. [33] introduced the operations of the improved quadtrees spatial index: insertion, deletion, and query. Cao et al. [34] presented a searchable encryption cloud storage (DUSECS) scheme based on dynamically updatable. The DUSECS scheme adopts a linked list structure and achieves dynamic data update and integrity detection. To improve the efficiency and effectiveness of dynamic index updating, Cherniak et al. [35] proposed the update methods for the maintainability of a multidimensional index on non-ordered discrete vector data. Wu et al. [36] proposed a learned index with precise positions learning index structure to solve the dynamic update of the learning index structure. Zhou et al. [37] suggested an index update method using the Monte Carlo tree search, which maintains a strategy tree to represent the existing index and updates the index by looking up the strategy tree.

In summary, an effective encrypted speech retrieval method was proposed in this study, which can achieve safe and efficient retrieval of massive unlabeled speech data in cloud storage, in order to address the issues of the complex index structure, the inability to dynamically update the index table, the bad performance of retrieval system, and the privacy leakage of speech and indexes.

3 The Proposed Method

3.1 System Model

Fig. 1 shows the system model of the proposed method. The model consists of three entities: data owner (DO), cloud server (CS), and data user (DU).

Fig. 1 indicates that the system model is predominantly separated into three handling processes: the construction of speech encryption library, the B+ tree index sequence generation and the B+ tree index table construction, and speech retrieval. The multi-threaded speech DGHV FHE scheme [15] ensures the privacy and security of the cloud server's data, when building the encrypted speech library. During the process of B+ tree index sequence generation and B+ tree index table generation, this study designs the ResNet18-GRU network model that is fit for unsupervised hash coding of speech data, using extracted speech data time-series-features as network input. Parallel to completing deep speech feature extraction and hashing function learning, high-quality unsupervised compact binary hash codes with rich semantics are generated. A pre-training is added to the model training process to reduce model overfitting, speed up model convergence, and enhance network model performance. The B+ tree index table is constructed for the obtained unsupervised compact binary hashing code. Keys and pointers are made up the index table, and the leaf nodes point to the primary key field

value that is related to the key value. In the system model shown in Fig. 1, the unsupervised hashing learning method and the construction of the B+ tree index table can enhance the performance of speech retrieval system and the index table is dynamically updated to meet the needs of DU real-time retrieval.

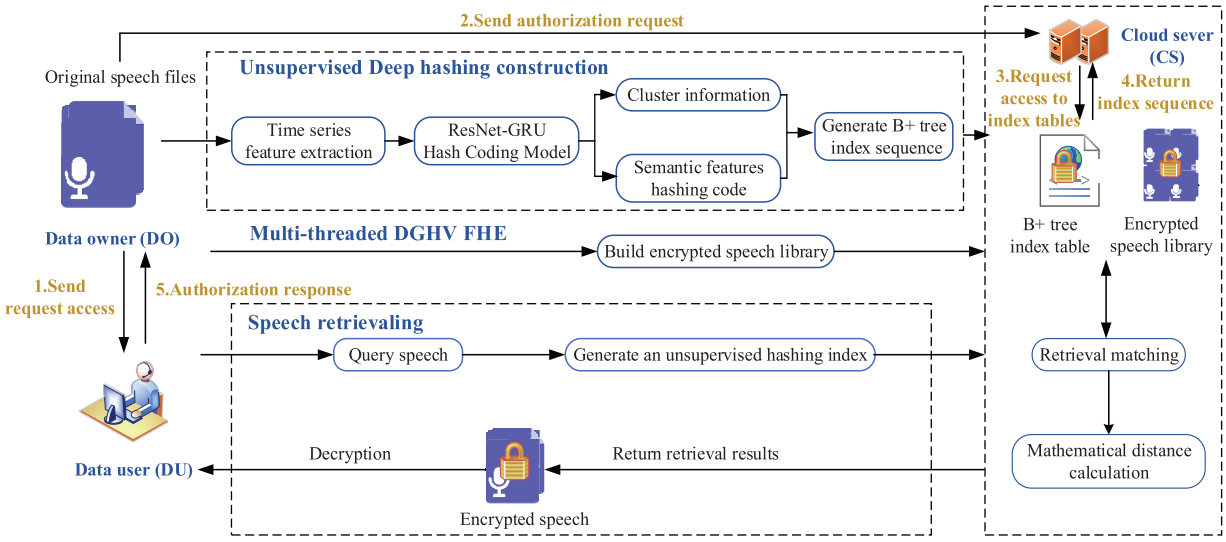


Figure 1: The system model of the proposed method

When the DU has retrieval requirements, it must obtain the authorization of the DO. The authorization process is shown in steps 1, 2, 3, 4, and 5 in Fig. 1. The specific processing flow is as follows:

Step 1: When the DU has retrieval requirements, it sends a request to the DO to access the B+ tree index table and waits for the DO authorization;

Step 2: After receiving the request from the DU, the DO sends an authorization request to the CS to request access to the B+ tree index table stored in the CS. After the CS receives the request, the CS returns the index sequence in the B+ tree index table, and the DO obtains the index sequence returned by the CS;

Step 3: The DO sends an authorization grant response to the DU and returns the index sequence value to the DU.

3.2 Encrypted Speech Library Construction

Fig. 2 shows the speech encryption/decryption using the multi-threaded DGHV FHE scheme [15]. It is employed to safeguard the security and privacy of speech stored in the cloud, and complete the construction of the encrypted speech library.

The specific processing process of the speech multi-threaded DGHV FHE scheme is as follows:

Step 1: Sampling, quantization, and analog-digital conversion (A/D). Initially, the original speech signal is sampled and quantized, and then the sampled and quantized analog speech signal is converted into a digital speech signal.

Step 2: Speech data pre-processing. The digital speech signal is processed by pre-weighting, framing, and windowing.

Step 3: Positive integer processing. To enable homomorphic encryption of speech data and to raise the efficiency of encryption, it is necessary to convert floating-point speech data into positive integer speech data. Encode floating-point speech data by multiplying it by a large number with fixed precision, rounding the result, and decoding it by dividing by this large number. By multiplying the original speech data of float type by 10^{φ} while maintaining φ bits valid, where φ is 6.

Step 4: The processed positive integer speech data are disassembled into the binary string by bit, and the disassembled speech data are formed into a matrix for cyclic encryption.

Step 5: Speech multi-threaded DGHV FHE. The encryption scheme contains three algorithms: The key generation algorithm (KeyGen (λ)), the encryption algorithm (Enc $(SK, \boldsymbol{pk}, m)$), and the ciphertext calculation algorithm (Eval $(\boldsymbol{pk}, C, c_1, c_2, \ldots, c_t)$), First, the private key SK and the public key vector $\boldsymbol{pk}$ are generated based on the key generation algorithm, After inputting the t ciphertexts c_i and the public key $\boldsymbol{pk}$ to the arithmetic circuit C with γ input ports, addition and multiplication are performed to obtain the ciphertext result c^*.

Step 6: Re-encryption. After the re-encryption technology, the re-encryption technique can reduce the noise generated by multiplication operations, and the ciphertext $c^{*'}$ can be obtained.

Step 7: Batch processing. Since the DGHV algorithm can only encrypt single-bit plaintext, and the Chinese remainder theorem (CRT) can encrypt information containing multiple bits as a ciphertext, this study uses CRT to convert the above encryption scheme into batch DGHV FHE scheme.

Step 8: Multi-threaded parallel processing. Using multi-threaded technology, multi-core processors can simultaneously execute the encryption and decryption process to achieve parallelism and reduce the overall execution time.

Step 9: Encrypted speech library construction. The ciphertext speech E_n is uploaded to the encryption speech library of the cloud server through the above encryption Steps 1–8.

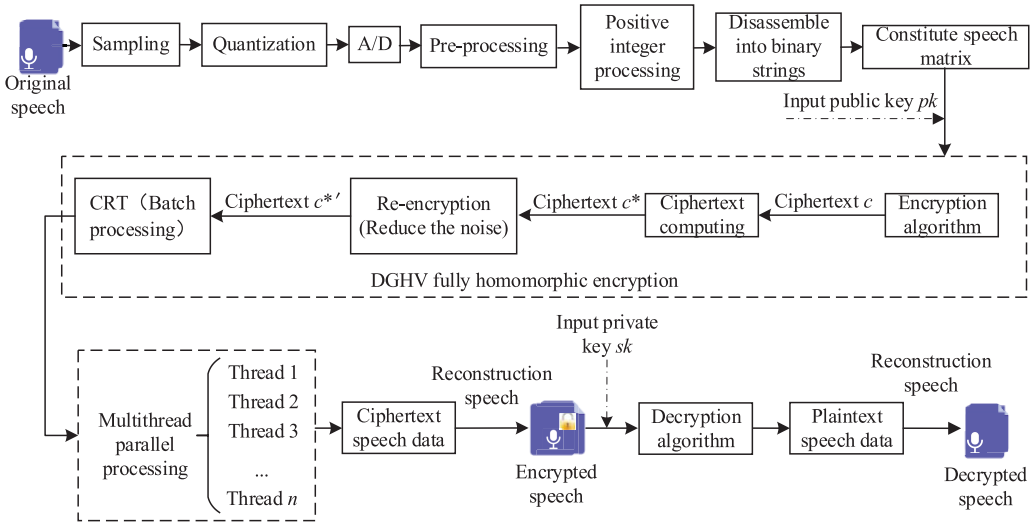


Figure 2: The processing flow of speech encryption/decryption using the multi-threaded DGHV FHE

Fig. 1 demonstrates when the data user (DU) obtains the retrieval result. It must decrypt the retrieved result (encrypted speech) using the decryption processing flow shown in Fig. 2. The decryption step is: input the private key SK and the ciphertext calculation result c^*, use $m' =$

[c^* mod p] mod 2 to obtain the plaintext speech data m', and reconstruct the plaintext speech data to obtain the decrypted speech.

3.3 Unsupervised Deep Hashing Learning Framework

In this study, the advantages of strong learning ability of deep learning are considered, the advantage that unsupervised learning does not require label information. The ResNet can solve the problem of gradient explosion and gradient loss brought, whereas GRU can solve the gradient problems in long-term memory and backpropagation. The ResNet18-GRU model can obtain more semantic deep features from unlabeled speech data by combining the above two network models. Fig. 3 depicts the unsupervised deep hashing network model based on ResNet18-GRU. Table 1 lists the specific parameter design of the ResNet18-GRU encoding model.

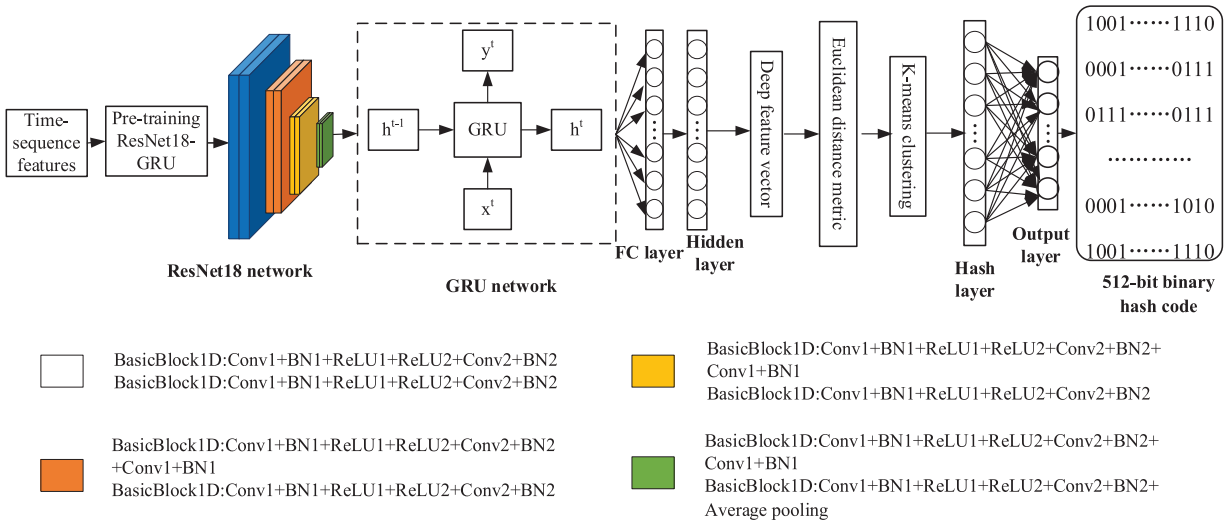


Figure 3: Unsupervised deep hashing network model based on ResNet18-GRU

Table 1: ResNet18-GRU encoding model parameters

Layer	Parameter settings
Conv1	Conv1d (kernel_size = (3, 3), stride = 2, padding = 1)
Conv2	Conv1d (kernel_size = (3, 3), stride = 1, padding = 1)
BN1	BatchNorm1d (out_channels = 64, eps = 1×10^{-5}, momentum = 0.1)
BN2	BatchNorm1d (out_channels = 256, eps = 1×10^{-5}, momentum = 0.1)
ReLU1	ReLU1 (activation = 'tanh')
ReLU2	ReLU1(activation = 'softmax")
Average pooling layer	Avgpool (kernel_size = (2, 2), stride = 1, padding = 1)
Hash layer	Dense (512, activation = 'sigmoid')
Output layer	Dense (512, 10/11)

The specific process steps of the unsupervised deep hashing construction of the ResNet18-GRU model are as follows:

Step 1: Extract time-sequence features. Initially, the speech samples $X=\{x_1, x_2, \ldots, x_n\}$ in the training set are pre-weighted, framed, windowed, and along with other pre-processing operations. Next, the time series feature package "tsfresh" in the Python tool is utilized to extract the time-sequence features, each speech time-sequence features $\boldsymbol{S}=\{\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_n\}$ in the training set samples is obtained.

Step 2: Extraction of deep speech features. The time-sequence features of speech $\boldsymbol{S}=\{\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_n\}$ are fed into the pre-trained network model ResNet18-GRU, the network parameters are initialized, the optimal network model is saved, and the end-to-end network model ResNet18-GRU applicable to the training of time-sequence features of speech is derived, and deep speech features are extracted.

Step 3: Extraction of deep speech feature vector. The FC layer contains rich semantic information, and the FC layer is added after the network model to output the extraction of deep speech feature vectors.

Step 4: K-means clustering. The obtained deep feature vector is measured by Euclidean distance, and K-means clustering is performed. The clustered deep speech feature vector is passed through the hash layer, and the deep feature vector is converted into the feature value in the [0, 1] interval.

Step 5: Unsupervised binary hash code generation. To better express the semantic information of speech, the number of nodes in the hash layer is 512, and the sign function (*sgn*) is utilized for the hashing construction to obtain a 512-bit binary hash code.

3.3.1 Hash Function Learning

Linear hashing can map high-dimensional input into compact binary hash codes to improve the performance of the retrieval system. The definition of the linear hashing function is illustrated in Eq. (1):

$$y = h(x) = \mathrm{sgn}(f(\boldsymbol{w}^H x + b)) \tag{1}$$

where $y\in\{0, 1\}$ or $\{-1, 1\}$, $\boldsymbol{w}$ is the weight vector, and b is the bias parameter. If $f(\boldsymbol{w}^H x+\mathbf{b}) \geq 0$, then $y=1$, otherwise $y=0$ or -1. The purpose of learning the hashing function $f(\cdot)$ is to learn the L-bit hashing function $h=\{h_1, h_2, \ldots, h_L\}$, Map the d-dimensional data $\boldsymbol{x}\in\mathrm{R}^d$ in the original space R to an L-bit binary hash code $\boldsymbol{y}=\{y_1, y_2, \ldots, y_L\}=\{h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_L(\boldsymbol{x})\}$.

To be able to learn compact hash codes, Using UH-BDNN [38] to optimize the objective function "$min_{w,b}Loss$"in the hidden layer of ResNet18-GRU, all of the hash codes that are returned from the hash layer belong to {0, 1}. Specific optimization requires learning five constraints on Eq. (2), as indicated by the five terms of Eq. (2). The optimized result is depicted in Eq. (3).

$$\begin{aligned} min_{w,b}Loss = \{\mathrm{sgn}(x)\} &+ \left\{\frac{\lambda_1}{2}\sum_{i=1}^{M}||\boldsymbol{w}_i||^2\right\} + \left\{\frac{\lambda_2}{2}||\boldsymbol{H}_M - \boldsymbol{Y}||^2\right\} \\ &+ \left\{\frac{\lambda_3}{2}\left|\left|\frac{1}{n}\boldsymbol{H}_M\boldsymbol{H}_M^T - \boldsymbol{I}\right|\right|^2\right\} + \left\{\frac{\lambda_4}{2}||\boldsymbol{H}_M[1]_{n\times 1}||^2\right\} \end{aligned} \tag{2}$$

$$s.t. y \in \{0, 1\}^{K\times n} \tag{3}$$

where λ_1, λ_2, λ_3, and λ_4 are the parameters of the constraints in the objective optimization function. The first term of Eq. (2) can map $\boldsymbol{X}$ into a binary hash code; the second term is to reduce the weight; the third term is to control binary constraints; the fourth and fifth terms support the independence

and balance of binary hash codes respectively, Eq. (3) makes every optimized hash code to belong to {0, 1}.

3.4 External B+ Tree Index Table Construction

The B+ tree index has the benefits of supporting range query, storing data only in leaf nodes, and having stable query time in response to the demand for rapid retrieval of massive multimedia data [39]. The fundamental principle of external B+ tree index structure is as follows: First, an external index is created for the speech file before encryption. The keys in the external B+ tree index are arranged sequentially, and the B+ tree structure is constructed on this arrangement. The B+ tree index structure is consisted of keys and pointers, with the leaf nodes pointing to the primary key field value corresponding to key values. Based on the order of the key values in the external B+ index, the leaf nodes will be retrieved quickly in the index as the retrieval progresses. Compared to the B-tree data structure, the B+ tree can maintain the sequential structure of the data, which can support range retrieval, and has a faster retrieval speed.

Fig. 4 displays the construction of the B+ tree index. In order to retrieve the target object, the first I/O operation is to access the root node, the second I/O operation is to access the intermediate node, and the third I/O operation is to access the leaf node, traverse the inside node in order, and retrieve the target object.

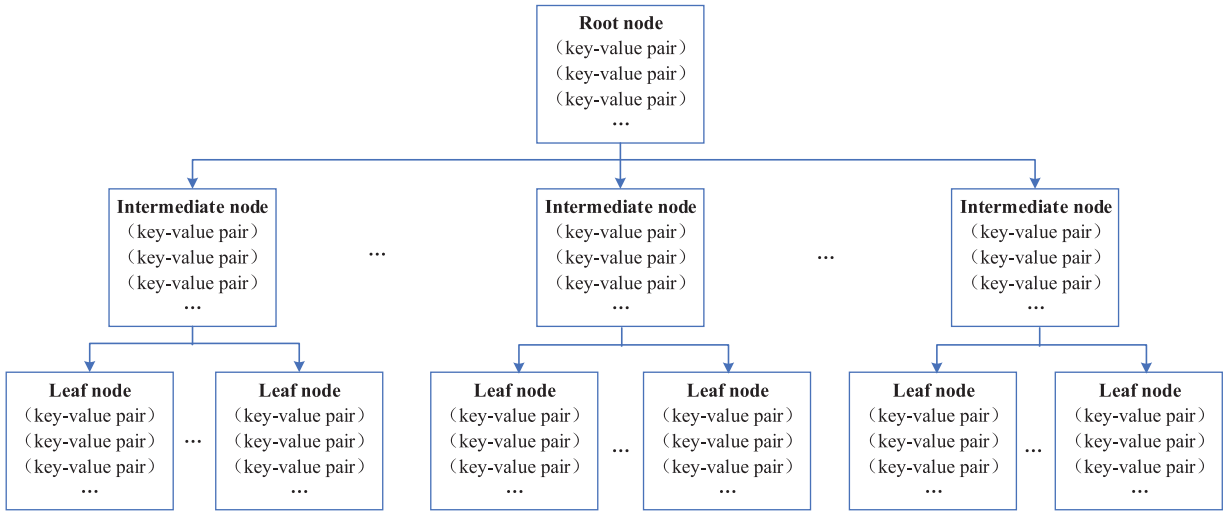


Figure 4: The construction principle of the B+ tree index structure

Indexing is the optimal method for efficiently retrieving massive encryption speech stored on the cloud servers, but an efficient indexing structure is required. Fig. 5 demonstrates the design and construction process of the external B+ tree index table.

The particular procedure can be summed up as follows:

Step 1: The original speech files are processed using time sequence feature extraction, the unsupervised deep hash coding of the ResNet18-GRU model, the unsupervised deep hashing construction described in Section 3.3, and the clustering information and semantic feature hash codes of each speech file are obtained. Perform classification to obtain unsupervised semantic feature hash codes that determine the category.

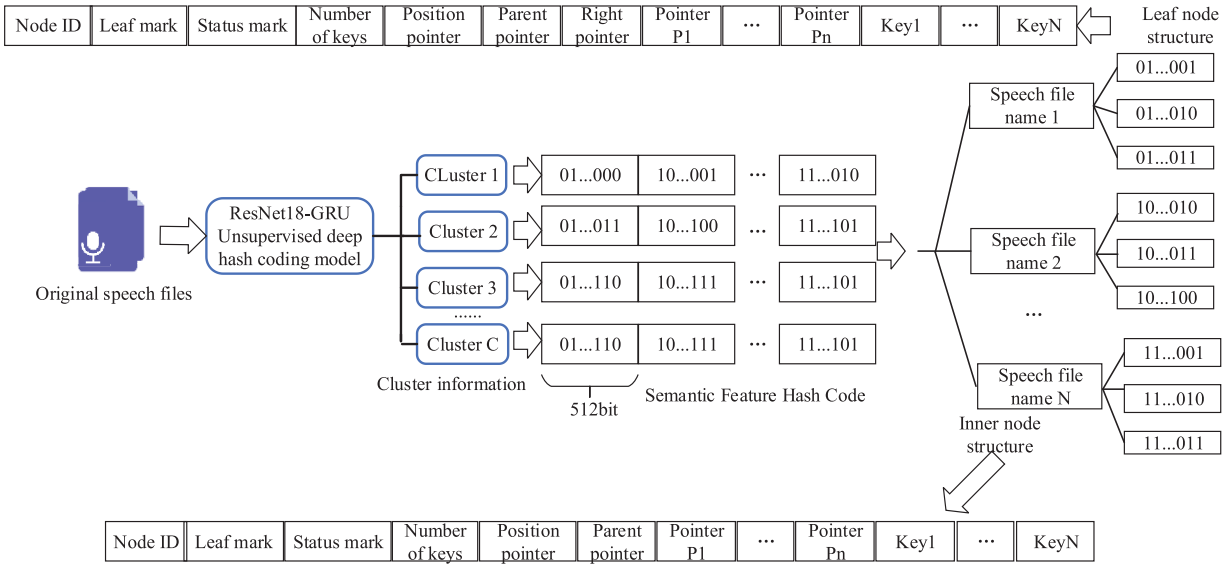


Figure 5: Process for building external B+ tree index tables

Step 2: The unsupervised semantic feature hash code obtained in Step 1 is employed to construct the external B+ tree index table. Based on the classified unsupervised semantic feature hash codes, *m* B+ tree index tables are established, which can be expressed: B+ tree index Table 1, B+ tree index Table 2, B+ tree index Table 3, ..., B+ tree index table *m*.

Step 3: The height of the B+ tree in this paper is considered to be 3, relying on different datasets, the file name of each speech file, and the hash value of various speech. The structure of pointer, pointer, pointer, ..., pointer, data, data, data, ..., data is adopted to facilitate the halving of the keywords in the node. Under certain conditions, inserting a right pointer to a leaf node can increase the retrieval speed. Adding a parent pointer and a position pointer to an inner node reduces the search time.

Table 2: Comparison of the storage space occupied by different index structures

CPOs	Hash index table (kB)		B+ tree index table (kB)	
	THCHS-30	TIMIT	THCHS-30	TIMIT
Amplitude reduction/−3 dB	1501	2001	500	670
Amplitude increase/+3 dB	1501	2001	500	670
Re-quantization/16-8-16 kbps	1501	2001	500	670
Volume reduction/75%	1501	2001	500	670
Volume increase/150%	1501	2001	500	670
MP3 compression	1501	2001	500	670
Resampling/rq8_16	1501	2001	500	670
Noise addition/GN	1501	2001	500	670

Table 3: Efficiency analysis of speech retrieval

Total of speech	Hash index table (s)		B+ tree index table (s)	
	THCHS-30	TIMIT	THCHS-30	TIMIT
3,000	0.022	0.022	0.018	0.017
5,000	0.034	0.034	0.025	0.023
7,000	0.043	0.044	0.035	0.031
10,000	0.065	0.063	0.051	0.049
15,000	0.079	0.078	0.068	0.062
30,000	0.099	0.097	0.082	0.079

Step 4: Map the semantic feature binary hash codes in all speech files to the B+ tree index table of the cluster to which they belong, establish a one-to-one mapping relation, and finally upload it to the cloud to complete the B+ tree index table.

3.5 Index Node Encryption and Decryption

The cloud server is an untrusted third party, and the B+ tree index table stored in the cloud server has certain security risks. Hence, this study encrypts the B+ tree index node. The specific processing steps are as follows:

Step 1: The leaf nodes of the B+ tree are determined and then encrypted using the speech multi-threaded DGHV FHE scheme [15].

Step 2: Based on the characteristics of the B+ tree index table, locate its leaf nodes, and encrypt the index nodes in the order of the root node and the leaf node until the last node is encrypted; then the entire B+ tree index table the nodes are encrypted finish.

Step 3: When DU has a retrieval request and sends a request to DO to access the B+ tree index table, it must decrypt the B+ tree index table's nodes. Decryption follows the same sequence as encryption. Then DO obtains the decrypted B+ tree index tables and returns the value to DU for future use.

3.6 Dynamic Update Strategy for an Index Structure

The index structure of the B+ tree can support the dynamic update of indices, which is suitable for practical applications. This study designs the dynamic update strategy of the B+ tree index table. The specific processing steps are as described below.

Step 1: When DO needs to add, delete, or change index tables, it initiates the dynamic update process with the dynamic-update-request function.

Step 2: Upon receiving the request, CS searches for the updated speech file and modifies the root node's hash value.

Step 3: When adding/inserting a request, if the previous node has no hash value, add/insert directly to the current node; if adding to a complete node, it must look up the adjacent nodes left and right to add/insert.

Step 4: When deleting a request, alternately check the left and the right N same leaf nodes of the deleted node and determine whether all the hash values of the node and each of the left and right N children can be placed in $2N$ nodes, if so, perform the delete operation; otherwise, reject the request and perform the merge operation.

Step 5: Only the children of non-leaf nodes are updated when each update request is processed. CS updates all requested nodes without changing the entire B+ tree index table if a batch update is required.

3.7 Speech Retrieval

When a data user (DU) needs to retrieve a speech file on the client side, he/she can do so by completing the steps outlined below:

Step 1: Create unsupervised hashing indexes. Generate the unsupervised binary hash codes y_{ps} of the speech to be queried using the unsupervised deep hashing construction scheme described in Section 3.3.

Step 2: DU sends request access to DO, waiting for authorization from DO, then DO sends the authorization request to CS and requests the B+ tree index table stored in CS. After receiving the request, CS returns the B+ tree index table depth hash sequence y_n, DO gets the y_n returned from CS, sends authorization consent response to DU, and returns the y_n to DU.

Step 3: Retrieval matching. First, the normalized Hamming distance $D(y_n, y_{ps})$ (bit error rate (BER)) is calculated between the unsupervised binary hash code y_{ps} and the unsupervised hashing sequence y_n in the cloud of B+ tree index table, set this distance threshold τ. If $D(y_n, y_{ps}) > \tau$, which indicates that no speech file matching the speech to be queried is retrieved; if $D(y_n, y_{ps}) \leq \tau$, which demonstrates that there is a retrieval result matching the speech to be queried. Then the unsupervised hashing sequences are sorted by the size of the normalized Hamming distance for conditions that satisfy $D(y_n, y_{ps}) \leq \tau$. The minimum normalized Hamming distance is the B+ tree index sequence corresponding the speech to be retrieved. The normalized Hamming distance is determined by Eq. (4):

$$D(yn, yps) = \frac{1}{N}\sum_{i=1}^{N}(|yn(i) - yps(i)|) = \frac{1}{N}\sum_{i=1}^{N} yn(i) \oplus yps(i), i = 1, 2, \ldots, N \tag{4}$$

where N is the length of the binary hash code ($N = 512$), and $\oplus$ is the XOR operation.

Step 4: Decryption processing. Based on the retrieval matching result, the corresponding encrypted speech file in the encrypted speech library is returned to the data user (DU) by the decryption operation in Section 3.2.

4 Experimental Results and Performance Analysis

4.1 The Dataset and Experimental Environment

This study uses the speeches in the open Chinese speech database THCHS-30 [40] database and TIMIT [41] (Texas Instruments and Massachusetts Institute of Technology) to evaluate the proposed method. THCHS-30 and TIMIT datasets are sampled at 16 kHz with 16-bit sampling accuracy in single-channel wave format. The speech content in THCHS-30 is 1,000 news clips with different contents and this dataset contains 13,388 speech clips with a total duration of approximately 30 h. The speech content in TIMIT includes 6,300 sentences, each of which is approximately 4 s long, and 630 individuals from United States.

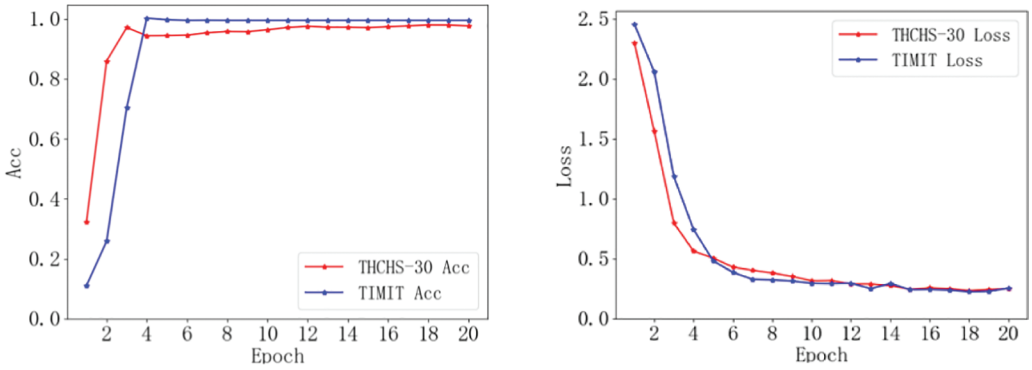
In this paper, 10 different speech contents spoken by 17 people and 11 various speech contents spoken by 23 people are selected from the THCHS-30 and TIMIT, and many varying content preserving operations (CPOs), such as volume adjustment, noise addition, re-quantization, resampling, and Moving Picture Experts Group Audio Layer III (MP3) are also performed. The two speech database training sets contain 3,060 and 2,277 samples. In the analysis of the mean average precision (mAP), recall rate, precision rate, and F1-score, 1,000 random speech from two speech libraries are evaluated. In retrieval efficiency evaluation, 3,000, 5,000, 7,000, 10,000, 15,000, and 30,000 speech from two speech libraries are randomly selected.

The encryption performance and security analysis of the speech multi-threaded DGHV FHE scheme have been thoroughly analyzed in the encryption scheme [15] and will not be analyzed.

The experimental hardware platform has an Intel(R) Core(TM) i7-11800H Central Processing Unit (CPU). The software environment is Windows 10, JetBrains PyCharm Community Edition 2020, and MATLABR2017b. The deep learning framework is Pytorch.

4.2 Performance Analysis of Unsupervised Network Model

A speech retrieval system's performance can only be determined through the steps of extracting speech features and creating binary hash codes. In addition, the length of the speech data hash code, which represents the network model's capacity to represent the features of the input speech data. The accuracy and loss rate of the network model are evaluated through repeated experiments to obtain the optimal network model. In the paper, there are 512 nodes in the hash layer. Fig. 6 shows the test accuracy and loss rate curves of the ResNet18-GRU.



(a) Accuracy curve (b) Loss rate curve

Figure 6: The test accuracy and loss rate curves of the ResNet18-GRU network model

Fig. 6a indicates that in THCHS-30, after 18 training batches, the test accuracy curve is close to 1. In the TIMIT dataset, after 18 training batches, the test accuracy curve remains unchanged and is 1. Fig. 6b indicates that in the THCHS-30 and TIMIT, after 18 training batches, the test loss curve is close to 0, showing that the network on different speech datasets have excellent performance. The accuracy of Fig. 6a is infinitely close to 1. Because the higher the number of nodes, the stronger the nonlinear capability of the model, which has a higher fit to the input speech data, and the accuracy of the model is higher. Fig. 6b shows that the model has a large drop in the initial stage, indicating that the learning rate is appropriate and gradient descent is being performed. After a training period, the loss gradually stabilizes infinitely close to 0. Hence, the ResNet18-GRU network model has good

performance, high accuracy, and low loss rate on different datasets. The performance of the speech retrieval is significantly improved.

4.3 B+ Tree Index Table Analysis of Space Occupancy

This study performs eight kinds of different CPOs operations on speech data from the THCHS-30 and TIMIT to evaluate the size of storage space occupied by B+ tree index tables. Each content preserving operation (CPO) selects 1,000 speech data, for a total of 8,000 speech data in each dataset, and evaluates the storage space occupied by hash index tables and B+ tree index tables for these speech data sizes. Table 2 compares the storage space occupied by the two index structures on different datasets.

As can be seen from Table 2, the storage space occupied is the same size under different CPOs with the same index structure and dataset. When the THCHS-30 and TIMIT are used, the storage space occupied is different because the two datasets are various languages, and extracted depth feature vectors will be different, leading to the different space occupied by the feature vectors converted into hash codes. The memory space occupied using the B+ tree index table is about one-third of the memory space occupied by the hash index table, saving memory space. Hence, the designed B+ tree index table can effectively reduce the storage space size and is suitable for retrieving massive encrypted speech.

4.4 Retrieval Efficiency Analysis

To evaluate the proposed method's retrieval time, 3,000, 5,000, 7,000, 10,000, 15,000, and 30,000 speech data with a speech length of 10 s were randomly selected from the THCHS-30 dataset, and the same number of data with a speech length of 4 s was randomly selected from the TIMIT dataset for retrieval to test the average retrieval time. Table 3 displays the efficiency analysis of the proposed method for speech retrieval.

As can be seen from Table 3, the average retrieval time grows as the number of speech entries increases, whereas the retrieval time for the TIMIT dataset is shorter. The TIMIT dataset contains only 4 s speech files, while the THCHS-30 dataset includes 10 s, so the required retrieval time will be longer. When using different index structures with the same dataset and number of speech entries, it is evident that a B+ tree index table requires less retrieval time than a hash index table because a B+ tree is a sequential search, and a hash index table is an iterative search and will require more retrieval time. This method has high retrieval efficiency under two different datasets and measured the time required for 3,000, 5,000, 10,000, 15,000, and 30,000 speech entries. In the THCHS-30 and TIMIT, the retrieval time used for the B+ tree index structure is about an average of 0.81 and 0.76 times that of the hash index table. Therefore, the method significantly improves the performance of retrieval system, making it ideally suited to the demand for efficient retrieval of massive ciphertext speech.

4.5 Retrieval Performance Analysis

4.5.1 Retrieval Accuracy

This paper chooses the mean Average Precision (mAP), to evaluate the retrieval accuracy of the ResNet18-GRU under various hash code lengths, obtaining the ideal node setting for the network. In the experiment, various nodes are placed for the speech after various CPOs, and the AP and mAP

values are determined. The calculation formulas are given in Eqs. (5) and (6):

$$AP(q) = \frac{\sum_{k=1}^{n} (P(k) \times rel(k))}{L}, rel(k) \in \{0, 1\} \tag{5}$$

$$mAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{6}$$

where n is the total number of speeches in the speech library, L is the total number of speeches related to the query speech, Q is the number of query speeches, and $rel(k)$ indicates whether the k-th speech of the database is related to the query speech, where related is one, and unrelated is zero.

Table 4 lists the mAP values of the ResNet18-GRU with various datasets and hashing code lengths. Fig. 7 depicts mAP with curves for different datasets and hashing code lengths.

Table 4: The mAP for different datasets and different hashing code lengths (%)

Hashing code length (bits)	THCHS-30	TIMIT
32	86.49	86.52
64	88.64	89.56
128	91.45	91.52
256	93.54	93.65
384	95.24	95.32
512	95.84	99.95
544	95.84	95.95
576	95.84	95.95

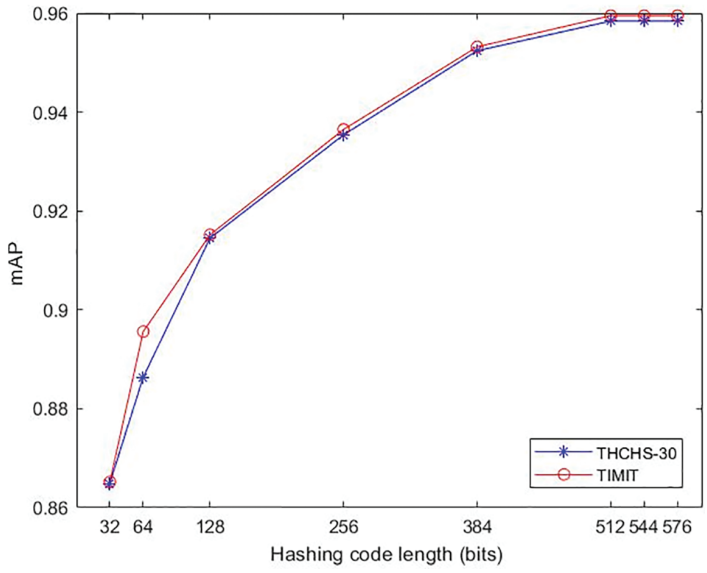


Figure 7: The mAP with curves for different datasets and hashing code lengths

As can be seen from Table 4, the mAP values of the ResNet18-GRU increase with the length of the hash code in the two datasets, as longer hashing codes can accurately represent the semantic features of the input speech. When the length of hash code is 512, the mAP value of the ResNet18-GRU exceeds 95%, and the difference between the mAP values of 512 and 576 is insignificant, indicating that the ResNet18-GRU has achieved optimal performance with 512 nodes. Therefore, the hash code length of this proposed method hash code is 512, yielding a 512 bits binary hash code for subsequent retrieval analysis.

Fig. 7 illustrates that as the hash code length increases, the mAP is increasing in this paper. The mAP value does not increase once the length of the hash code reaches 512 bits, indicating that when the number of bits in the hash code is 512, the performance of retrieval is optimal.

4.5.2 Recall Rate, Precision Rate and F1 Score Analysis, and Precision Rate-Recall Rate(P-R) Curves Analysis

The Recall rate (R) and precision rate (P) are utilized to measure the retrieval system's quality, and their respective formulas are shown in Eqs. (7) and (8):

$$R = \frac{TP}{TP + FN} \times 100\% \tag{7}$$

$$P = \frac{TP}{TP + FP} \times 100\% \tag{8}$$

where TP is the number of related speech segments in the retrieval result, FN is the number of related speech segments not retrieved, and FP is the number of unrelated speech segments in the retrieval result.

As the links between recall and accuracy rates are mutually restricted, the F1 score of Eq. (9) is chosen as the evaluation index to evaluate the retrieval performance of the scheme further. The larger the value of F1 score, the better the performance of the speech retrieval system.

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{9}$$

Table 5 indicates the recall rate (R), precision rate (P), and F1 score of the speech with different datasets and CPOs. Fig. 8 displays P-R curves with various datasets.

As can be seen from Table 5, the recall and precision of the first seven CPOs in the THCHS-30 dataset are higher than 98%. Because the operation of amplitude reduction and amplitude increase only change the amplitude of the speech file. Re-quantization operation first compresses the original speech to 8 kbps and then restores it to 16 kbps; Volume increase and reduction only change the volume; MP3 compression lowers the high-frequency band signal while preserving the low-frequency band signal without distortion. The purpose of resampling is to improve the speech waveform accuracy, demonstrating that speech quality is unaffected by the seven types of CPOs. Therefore, the values of recall rate, precision rate, and F1 score of the three evaluation criteria all exceed 98%. For the noise-addition operation, speech quality and semantic information change, resulting in poor epistasis of the extracted speech time-sequence features. Hence, the R, P, and F1 after the noise-addition operation are lower than the first seven CPOs. In this study, the R, P, and F1 of TIMIT are the same as the test results in THCHS-30 under four CPOs: amplitude decreasing, amplitude increasing, volume decreasing, and volume increasing. For the noise addition operation, the method in this paper performs better under the TIMIT dataset because THCHS-30 is a Chinese speech set, while TIMIT belongs to the English speech set. Since various languages are affected differently by

the same operation, the values of the measurement indices will be different. Testing the recall rate, precision rate, and F1 score under two distinct datasets indicates the universality of this method.

Table 5: Recall rate, precision rate, and F1 score under different datasets and CPOs (%)

CPOs	THCHS-30			TIMIT		
	R	*P*	F1	*R*	*P*	F1
Amplitude reduction/−3 dB	99.99	99.99	99.99	100.0	100.0	100.0
Amplitude increase/+3 dB	99.99	99.99	99.99	99.99	99.99	99.98
Re-quantization/16-8-16 kbps	98.58	98.55	98.54	97.80	97.81	97.80
Volume reduction/75%	99.98	99.98	99.98	99.99	100.0	99.99
Volume increase/150%	99.48	99.50	99.51	99.98	99.99	99.99
MP3 compression	99.95	99.97	99.96	97.88	97.81	97.89
Resampling/rq8_16	98.56	98.52	98.54	97.88	97.81	97.89
Noise addition/GN	89.94	89.94	89.94	99.98	99.98	99.98

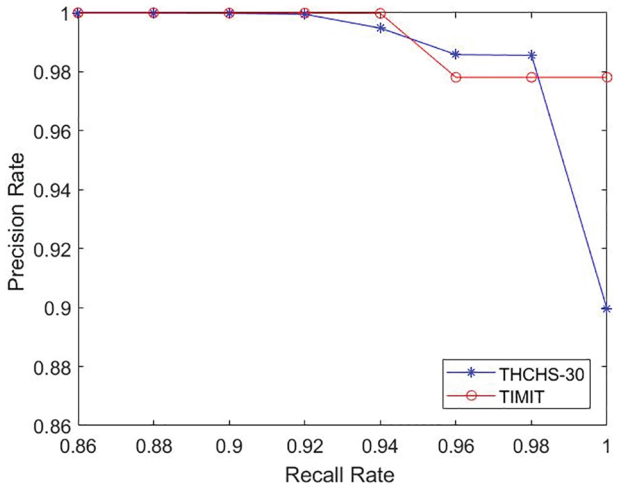


Figure 8: P-R curves with different datasets

Fig. 8 exhibits that the larger area enclosed by the P-R curve and the x-y axis, the better the retrieval performance of the retrieval algorithm. The area enclosed by the P-R curve and the x-y axis of the proposed scheme is larger in the TIMIT dataset than in the THCHS-30 dataset, showing that the TIMIT dataset has superior retrieval performance. Because the THCHS-30 dataset is Chinese, there are cases of multiple words with one sound and multiple meanings in Chinese, while there are only 26 letters in English, and various letter combinations have different meanings. The retrieval performance of the scheme presented in this paper is superior to the TIMIT dataset.

4.6 Performance Comparison with Existing Speech Retrieval Methods

The experimental results for speech retrieval methods based on perceptual hashing [9], supervised learning [8], and unsupervised learning [18,19,22] are compared to the existing methods in order to assess the performance of the proposed method. The comprehensive performance comparison results are illustrated in Table 6.

Table 6: Comparison between the proposed and different speech retrieval methods

Evaluation indicators	Ref. [8]	Ref. [9]	Ref. [18]	Ref. [19]	Ref. [22]	Proposed	
Dataset	THCHS-30	THCHS-30	Ballroom	Ballroom	Ballroom	THCHS-30	TIMIT
Network model	CNN	–	CNN	CNN	CNN	ResNet18-GRU	
Indexing method	Hash index	Hash index	Hash index	Hash index	Hash index	B+ tree dynamic index	
Index dynamic update	No	No	No	No	No	Yes	
mAP (%)	96.96	–	91.36	90.01	88.92	95.84	95.95
Recall rate (%)	100.00	100.00	91.00	91.00	88.60	99.99	100.00
Precision rate (%)	100.00	100.00	98.38	98.91	97.90	99.99	99.99
F1 score (%)	–	–	94.55	94.79	93.02	98.31	99.19
Retrieval time (s)	0.4854	0.2897	–	–	–	0.082	0.079
Total of audio	10000	1000	200	220	1200	30,000	
Security	Chaotic encryption	Chaotic encryption	No	No	No	DGHV FHE	

As can be seen from Table 6, in terms of the datasets used, the method in this paper performs many experimental analyses on the two datasets of different languages. In contrast, the scheme [8,9,18,19,22] only performs the corresponding analyses a single dataset, indicating that the proposed method is more generalizable. The scheme [8,18,19,22] all use Convolution Neural Network (CNN) network models. However, this paper employs an end-to-end network model with ResNet18 at the front end and GRU at the back end, which can better solve the gradient explosion and gradient loss problems in network model training, and utilizes a pre-training mechanism to accelerate model convergence and improve the network training accuracy. Regarding the index method, the scheme [8,9,18,19,22] all use a hash index table. The index method requires traversal retrieval during retrieval, which can result in a lengthy retrieval time. This study uses B+ tree index table that retrieves from root node to leaf node in order, requiring less retrieval time compared to the scheme [8,9], and the scheme [18,19,22] does not mention retrieval efficiency. Compared to the scheme [8,9,18,19,22], the method described in this study supports dynamic updating of the index structure (index table) and reduces the complexity of indexing. Under the assumption that the index structure does not change, new and useless speech data are processed on time to meet users' real-time retrieval needs. The scheme [18,19,22] does not mention the retrieval efficiency. Compared to the scheme [8,9,18,19,22], the method presented in this paper supports the dynamic update of the index structure (index table), which reduces the complexity of indexing, processes the new addition, and removes useless speech data on time to satisfy the user's real-time retrieval needs without changing the index structure. Regarding retrieval accuracy, the mAP values of the proposed method on two distinct datasets are slightly lower than those of the scheme [8] due to the use of an unsupervised training method that does not require label information, and there is a partial loss of category features. However, this partial feature loss does not cause a significant

decrease in retrieval accuracy and is more suitable for massive unlabeled data retrieval. Compared to the schemes [18,19,22], the mAP value of the proposed method is higher because this paper adopts the end-to-end network model of ResNet18-GRU, which can extract deep features with stronger semantic meaning, and the number of nodes in its hash layer is 512. The obtained deep hash code has higher quality and stronger semantic representation ability, resulting in a higher retrieval accuracy. The R and P of this paper are lower than the schemes [8,9] because the deep hash code in the scheme [8] consists of category hash code and semantic feature hash code, and the semantic feature information is richer, and scheme [8] adopts supervised training method, and the obtained deep feature vector contains label information. In contrast, the proposed scheme employs an unsupervised training method without label information. The depth feature extraction process has a small amount of feature loss. The scheme [8] extracts improved PNCC features, changes the Fourier transform to Dynamic Time Warping (DTW) with higher resolution, and adds to the original PNCC coefficients first-order difference coefficients representing the dynamic speech features. The time-sequence features extracted in this paper are time domain features, and some frequency domain features are inevitably lost in the extraction process, resulting in a low detection and accuracy rates. However, this paper's P, R, and F1 are higher than the schemes [18,19,22]. Because this study uses the network model of ResNet at the front end and GRU at the back end to extract speech depth features. As a result, ResNet can solve gradient explosion and gradient disappearance issues caused by the increasing number of network layers. In contrast, GRU can solve the problems of long-term memory and backpropagation, and this network model can extract deep features with more epistemic significance. The length of hash code are 512 bits, which can generate the 512 bits binary hash code, and the hash code is more epistemic as its length increases, so the subsequent retrieval has a higher R, P, and F1. The total number of retrieved speech datasets selected is 30,000, which is significantly greater than the number of speeches in the scheme [8,9,18,19,22]. The retrieval time is also shorter, indicating that the scheme is more suitable for the efficient retrieval demand of massive encrypted speech. Regarding security means, the scheme [8,9] used chaotic encryption to build the encrypted speech library, whereas the scheme [18,19,22] did not use any security measures, resulting in privacy leakage. In contrast, this paper uses a self-designed speech multi-threaded DGHV FHE scheme to build an encrypted speech library, which protects speech data from the privacy leakage problem more effectively.

5 Conclusion

This study proposes an efficient encrypted speech retrieval method using unsupervised deep hashing and B+ tree dynamic index. The method uses an end-to-end ResNet18-GRU network model to exact deep feature and unsupervised hash function learning simultaneously. It use the learnt unsupervised hash function compression mapping to produce high-quality unsupervised compact binary hash codes without using the category characteristics of data labels, thereby reducing the computational effort and improving the semantics of speech features and the efficiency of system retrieval. In order to reduce the complexity of the index structure and improve the performance of the speech retrieval system, an index structure based on the external B+ tree is designed, which can achieve the dynamic update of the index structure. A multi-threaded DGHV FHE scheme decreases the risk of cloud-based speech data and unsupervised deep hashing code leakage. The experimental verification analysis on the THCHS-30 and TIMIT shows that the proposed approach has higher retrieval accuracy and retrieval efficiency than existing supervised and unsupervised speech retrieval methods, can achieve dynamic index update of index structure, and can resist the privacy leakage problem of cloud-based speech data and unsupervised binary hash codes, and is more suitable for the

retrieval demand of massive encrypted speech. Currently, the proposed method relies primarily on two insufficient datasets.

Future work will transfer existing single-modal retrieval to cross-modal retrieval research and utilize learnable encryption techniques to enhance the privacy protection of speech.

Acknowledgement: The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

Funding Statement: This work is supported by the National Natural Science Foundation of China (No. 61862041).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. I. Shyla and S. S. Sujatha, "Efficient secure data retrieval on cloud using multi-stage authentication and optimized blowfish algorithm," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 1, pp. 151–163, 2022.

[2] L. Song, Y. Miao, J. Weng, K. R. Choo, X. Liu *et al.,* "Privacy-preserving threshold-based image retrieval in cloud-assisted internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13598–13611, 2022.

[3] X. Li, S. Liu, R. Lu, M. K. Khan, K. Gu *et al.,* "An efficient privacy-preserving public auditing protocol for cloud-based medical storage system," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 5, pp. 2020–2031, 2022.

[4] N. Deepa and P. Pandiaraja, "E health care data privacy preserving efficient file retrieval from the cloud service provider using attribute based file encryption," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 5, pp. 44877–4887, 2021.

[5] A. Law and A. Ghosh, "Multi-label classification using binary tree of classifiers," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 3, pp. 677–689, 2021.

[6] H. Zhao and S. He, "A retrieval algorithm for encrypted speech based on perceptual hashing," in *Proc. of ICNC-FSKD*, Zhangjiajie, China, IEEE, pp. 1840–1845, 2016.

[7] H. Zhao and S. He, "A retrieval algorithm of encrypted speech based on syllable-level perceptual hashing," *Computer Science and Information Systems*, vol. 14, no. 3, pp. 703–718, 2017.

[8] Q. Zhang, Z. Ge, Y. Hu, J. Bai and Y. Huang, "An encrypted speech retrieval algorithm based on Chirp-Z transform and perceptual hashing second feature extraction," *Multimedia Tools and Applications*, vol. 79, no. 9, pp. 6337–6361, 2020.

[9] Q. Zhang, X. Zhao, Q. Zhang and Li, Y., "Content-based encrypted speech retrieval scheme with deep hashing," *Multimedia Tools and Applications*, vol. 81, no. 7, pp. 10221–10242, 2022.

[10] Q. Zhang, J. Bai and F., Xu, "A retrieval method for encrypted speech based on improved power normalized cepstrum coefficients and perceptual hashing," *Multimedia Tools and Applications*, vol. 81, no. 11, pp. 15127–15151, 2022.

[11] Q. Zhang, M. Fu, Y. Huang and Z. Zhao, "Encrypted speech retrieval scheme based on multiuser searchable encryption in cloud storage," *Security and Communication Networks*, vol. 2022, pp. 9045259, 2022.

[12] J. Li, "Research on fuzzy retrieval method of blockchain information based on B+ tree index structure," in *Proc. of IoT and Big Data Technologies for Health Care: Second EAI Int. Conf.*, Leicester, Great Britain, pp. 308–325, 2022.

[13] W. Zhang, C. Zhao, L. Peng, Y. Lin, F. Zhang *et al.,* "High performance GPU concurrent B+ tree," in *Proc. of the 27th ACM SIGPLAN Symp. on Principles and Practice of Parallel Programming*, New York, United States, pp. 443–444, 2022.

[14] C. Ho., K. Pak, S. Pak, M. Pak and C. Hwang, "A study on improving the performance of encrypted database retrieval using external indexing system of B+ tree structure," *Procedia Computer Science*, vol. 154, pp. 706–714, 2019.
[15] Q. Zhang and Y. Jia, "A speech fully homomorphic encryption scheme for DGHV based on multithreading in cloud storage," *International Journal of Network Security*, vol. 24, no. 6, pp. 1042–1054, 2022.
[16] Y. Zhu, Y. Li and S. Wang, "Unsupervised deep hashing with adaptive feature learning for image retrieval," *IEEE Signal Processing Letters*, vol. 26, no. 3, pp. 395–399, 2019.
[17] X. Dong, L. Liu, L. Zhu, Z. Cheng and H. Zhang, "Unsupervised deep K-means hashing for efficient image retrieval and clustering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 8, pp. 3266–3277, 2020.
[18] P. Panyapanuwat, S. Kamonsantiroj and L. Pipanmaekaporn, "Unsupervised learning hash for content-based audio retrieval using deep neural networks," in *Proc. of 2019 11th Int. Conf. on Knowledge and Smart Technology (KST)*, Phuket, Thailand, pp. 99–104, 2019.
[19] P. Panyapanuwat and S. Kamonsantiroj, "Performance comparison of unsupervised deep hashing with data-independent hashing for content-based audio retrieval," in *Proc. of the 2019 2nd Int. Conf. on Electronics, Communications and Control Engineering*, Phuket, Thailand, pp. 16–20, 2019.
[20] Y. Jia, X. Chen, J. Yu, L. Wang, Y. Xu *et al.,* "Speaker recognition based on characteristic spectrograms and an improved self-organizing feature map neural network," *Complex & Intelligent Systems*, vol. 7, no. 4, pp. 1749–1757, 2021.
[21] K. K. Rao and K. S. Rao, "A novel approach to unsupervised pattern discovery in speech using convolutional neural network," *Computer Speech & Language*, vol. 71, pp. 101259, 2022.
[22] P. Panyapanuwat, S. Kamonsantiroj and L. Pipanmaekaporn, "Similarity-preserving hash for content-based audio retrieval using unsupervised deep neural networks," *International Journal of Electrical and Computer Engineering*, vol. 11, no. 1, pp. 879–890, 2021.
[23] M. Li, Q. Li, Y. Ma and D. Yang, "Semantic-guided autoencoder adversarial hashing for large-scale cross-modal retrieval," *Complex & Intelligent Systems*, vol. 8, no. 2, pp. 1603–1617, 2022.
[24] M. Li and H. Wang, "Unsupervised deep cross-modal hashing by knowledge distillation for large-scale cross-modal retrieval," in *Proc. of the 2021 Int. Conf. on Multimedia Retrieval*, Taipei, Taiwan, pp. 183–191, 2021.
[25] G. Mikriukov, M. Ravanbakhsh and B. Demir, "Unsupervised contrastive hashing for cross-modal retrieval in remote sensing," in *Proc. of ICASSP 2022*, Shenzhen, China, pp. 4463–4467, 2022.
[26] X. Zou, S. Wu, N. Zhang and E. Bakker, "Multi-label modality enhanced attention based self-supervised deep cross-modal hashing," *Knowledge-Based Systems*, vol. 239, pp. 107927, 2022.
[27] A. Du, L. Wang, S. Cheng and N. Ao, "A Privacy-protected image retrieval scheme for fast and secure image search," *Symmetry*, vol. 12, no. 2, pp. 282, 2020.
[28] A. Gupta and D. Yadav, "A novel approach to perform context-based automatic spoken document retrieval of political speeches based on wavelet tree indexing," *Multimedia Tools and Applications*, vol. 80, no. 14, pp. 22209–22229, 2021.
[29] J. Shen, M. Tao, Q. Qu, D. Tao and Y. Rui, "Toward efficient indexing structure for scalable content-based music retrieval," *Multimedia Systems*, vol. 25, no. 6, pp. 639–653, 2019.
[30] S. Niu, J. Wang, B. Wang, X. Jia and X. Du, "Ciphertext sorting search scheme based on B+ tree index structure on blockchain," *Journal of Electronics and Communications*, vol. 41, no. 10, pp. 2409–2415, 2019.
[31] A. Llaveshi, U. Sirin and A., Ailamaki, "Accelerating B+ tree search by using simple machine learning techniques," in *Proc. of the 1st Int. Workshop on Applied AI for Database Systems and Applications*, Los Angeles, California, pp. 1–10, 2019.
[32] P. Ferragina and G. Vinciguerra, "The PGM-index: A fully-dynamic compressed learned index with provable worst-case bounds," in *Proc. of the VLDB Endowment*, vol. 13, no. 8, Tokyo, Japan, pp. 1162–1175, 2020.
[33] H. Wang and J. Zhu, "A quadtree spatial index method with inclusion relations and its application in landcover database update," *Ingénierie des Systèmes D'Information*, vol. 24, no. 3, pp. 241–247, 2019.

[34] L. Cao, Y. Kang, Q. Wu, R. Wu, X. Guo *et al.,* "Searchable encryption cloud storage with dynamic data update to support efficient policy hiding," *China Communications*, vol. 17, no. 6, pp. 153–163, 2020.
[35] R. Cherniak, Q. Zhu and S. Pramanik, "Update methods for maintainability of a multidimensional index on non-ordered discrete vector data," in *Proc. of the Int. Symp. on Computer Architecture*, vol. 27, no. 3, Valencia, Spain, pp. 94–106, 2020.
[36] J. Wu, Y. Zhang, S. Chen, J. Wang, Y. Chen *et al.,* "Updatable learned index with precise positions," *arXiv preprint*, vol. 14, no. 8, pp. arXiv:2104.05520, 2021.
[37] X. Zhou, L. Liu, W. Li, L. Jin, S. Li *et al.,* "Autoindex: An incremental index management system for dynamic workloads," in *ICDE*, Kuala Lumpur, Malaysia, pp. 2196–2208, 2022.
[38] T. Do, A. Doan D and N. Cheung, "Learning to hash with binary deep neural network," in *Proc. of the European Conf. on Computer Vision*, Amsterdam, The Netherlands, pp. 219–234, 2016.
[39] D. Abel, "A B+ tree structure for large quadtrees," *Computer Vision, Graphics, and Image Processing*, vol. 27, no. 1, pp. 19–31, 1984.
[40] D. Wang and X. Zhang, "Thchs-30: A free Chinese speech corpus," *arXiv Preprint*, pp. arXiv: 1512.01882, 2015.
[41] V. Zue, S. Seneff and J. Glass, "Speech database development at MIT: TIMIT and beyond speech communication," *Speech Communication*, vol. 9, no. 4, pp. 351–356, 1990.