Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.083887

ARTICLE

A Cross-Modal Searchable Encryption Scheme with Result Verification

Peixuan Wang[1], **Lingyun Yuan**[1,2,*], **Yi Xiang**[1], **Tianyu Xie**[1,2], **Haochen Bao**[1] **and Kexin Wang**[1,2]

[1]The School of Information Science and Technology, Yunnan Normal University, Kunming, China
[2]The Key Laboratory of Educational Information for Nationalities, Ministry of Education, Kunming, China

*Corresponding Author: Lingyun Yuan. Email: yuanlingyun@ynnu.edu.cn

Received: 12 April 2026; Accepted: 09 June 2026; Published: 13 August 2026

ABSTRACT: With the development of the Internet of Things (IoT), there is a rising demand for ciphertext retrieval. However, existing searchable encryption schemes mainly support single-modal retrieval, while current cross-modal searchable encryption methods often suffer from high computational overhead and lack reliable result verification. To address these problems, we propose a cross-modal searchable encryption scheme with result verification (VCMSE). First, we design a cross-modal hash extraction method that combines contrastive learning with a residual similarity matrix to generate encryption-friendly binary features with enhanced semantic consistency. Second, we designed a lightweight garbled circuit-based matching mechanism that enables efficient similarity computation in the ciphertext domain. Third, we propose a triple verification mechanism to ensure the search results from the cloud server are correct, complete, and comprehensive. Experimental results demonstrate that, compared with other cross-modal searchable encryption schemes, our method improves mean average precision (MAP) by 3.02%–16.9% on the NUS-WIDE dataset, while also reducing trapdoor generation time by 94.8%.

KEYWORDS: Searchable encryption; cross-modal retrieval; cross-modal hashing; result verification; lightweight garbled circuit

1 Introduction

Recently, the exponential growth of IoT devices has generated a massive volume of multimodal data, positioning cross-modal retrieval as a key technology in the IoT domain. Faced with the management of such large-scale multimodal data, outsourcing data to cloud servers has become a mainstream solution, but such services may leak sensitive information [1]. To mitigate such risks, data encryption and authentication mechanisms have become important methods for ensuring cloud data security, such as multilevel image encryption with independent keying strategies [2] and image-embedded password authentication with variable key lengths [3], which can prevent the leakage of sensitive information. However, while encryption protects data confidentiality, it also renders traditional plaintext retrieval methods directly ineffective, as users cannot efficiently obtain the required information without decrypting the entire dataset.

To address the above problems, searchable encryption techniques [4–6] have emerged, which allow for direct searches over encrypted data without decryption. However, existing searchable encryption schemes [7–13] primarily focus on a single data modality, which exhibits significant limitations, addressing the prevalent need for cross-modal ciphertext retrieval in IoT environments. To address this, cross-modal searchable encryption (CMSE) was proposed. Its primary objective is to allow users to submit a query in one modality to securely retrieve semantically relevant data from another modality, all within an encrypted

multimodal dataset. The core idea of existing CMSE schemes [14–20] is to map data from different modalities into unified encrypted feature vectors and perform similarity matching directly in the ciphertext domain. Although CMSE has achieved initial progress, its development is still in an exploratory stage and faces the following core challenges.

Firstly, the challenge is achieving high-precision retrieval on encrypted multimodal data. In the medical Internet of Things, inaccurate results can lead to clinical misdiagnosis and even endanger patient safety. The root of this challenge lies in the inherent semantic gap among different data modalities [21], whose heterogeneous distributions prevent the direct measurement of similarity. Although mainstream solutions employ contrastive learning to map multimodal data into a unified feature space, their process pulls positive pairs closer while indiscriminately separating all negative pairs. Consequently, nuanced semantic correlations among those negative pairs are unavoidably neglected, leading to a learned feature space structure that deviates from the true semantic distribution, which in turn limits the upper bound of retrieval accuracy.

Secondly, the challenge is satisfying the requirements of real-time retrieval for multimodal data. In the medical Internet of Things, an encryption scheme must be sufficiently lightweight to support instant retrieval in critical medical scenarios. From a security perspective, both Paillier-KNN [22] and homomorphic encryption (HE) [23] can be used to construct CMSE schemes. The former applies multiple sets of invertible linear transformations to each component of both the query and data vectors, combining with Paillier's public-key encryption mechanism, and maps the original plaintext features into the ciphertext space. The latter performs similarity computations directly in the ciphertext domain. However, these approaches often incur expensive modular arithmetic or homomorphic computations, which lead to considerable time overhead in practical multimodal retrieval settings.

Thirdly, the challenge is verification of results returned by the cloud server. In the medical Internet of Things, verifying the integrity, correctness, and comprehensiveness of data is crucial for ensuring that clinical decisions are based on information that is authentic, relevant, and without any omissions. Existing CMSE schemes [14–20] generally lack mechanisms the results returned by the server. Even among the few studies on searchable encryption that do tackle result validation [24–28], there are almost no solutions that simultaneously address correctness, integrity, and comprehensiveness. Therefore, a triple verification mechanism for CMSE is critically needed to ensure all three properties in returned search results.

In response to the above challenges, we propose a cross-modal searchable encryption scheme (VCMSE) that supports verifiable retrieval results. We first design a cross-modal hash (CMH) extraction method to generate high-quality hashes for accurate encrypted multimodal retrieval. After obtaining these CMHs, the core challenge shifts to performing their precise comparison efficiently and in a privacy-preserving manner. To address this, we design a lightweight garbled circuit (LGC) optimized for Hamming distance calculation. Finally, we introduce a triple verification mechanism to verify the comprehensiveness, integrity, and correctness of the returned results. The main contributions of this paper are as follows:

- We propose a cross-modal hash extraction method that combines contrastive learning with a residual similarity matrix. The method generates encryption-friendly and semantically consistent binary hash codes for multimodal data, thereby improving the retrieval accuracy of cross-modal searchable encryption.
- We design a lightweight garbled circuit-based matching mechanism for encrypted cross-modal hash codes. By specializing the circuit's functionality and forgoing the generality of traditional garbled circuits, our LGC reduces computational overhead. thereby requirements of real-time retrieval of multimodal encrypted data in the IoT.

- We propose a triple-verification mechanism that jointly considers comprehensiveness, integrity, and correctness. Only when the retrieval results satisfy all three properties simultaneously can the data received by the user be considered untampered, accurate, and complete, thereby compensating for the lack of a verification mechanism in existing CMSE schemes.

2 Related Work

Searchable encryption was first proposed by Song et al. [4], enabling data user retrieval over ciphertext. However, traditional searchable encryption schemes are limited to text-based retrieval and do not support cross-modal retrieval. With the increasing demand for cross-modal ciphertext retrieval in the IoT environment, CMSE schemes [14–20] have emerged. Cao et al. [14] extract text and image features by using pre-trained image encoders and text encoders in the CLIP model, and protect the features by transforming and downscaling them using a three-layer MLP. However, their approach suffers from degraded retrieval accuracy due to the obfuscation of feature representations. Chen et al. [15] proposed a secure cross-modal retrieval model based on inner-product functional encryption. The model enables similarity computation over encrypted multimodal data while preserving data privacy, but suffers from significant time overhead. Guo et al. [16] designed a privacy-preserving search scheme by integrating collective matrix factorization with homomorphic encryption. Wang et al. [17] combined federated learning with homomorphic encryption. However, both schemes suffer from high computational cost due to the overhead of homomorphic operations. Hu et al. [18] perform plaintext retrieval in dedicated hardware; however, the scheme remains susceptible to plaintext data leakage under hardware-based fault injection attacks. Li et al. [19] use knowledge distillation to train a lightweight model for extracting feature vectors in a shared semantic space, and then encrypt the vectors using secure kNN. Yang et al. [20] proposed the FECMR scheme, which employs an SFB-IPFE based encryption algorithm to construct the index structure. However, the SFB-IPFE encryption algorithm and secure KNN both involve large-scale matrix operations, and the time overhead of the scheme is high.

It is worth noting that, in public cloud storage environments, the above schemes have not yet fully considered the risk that servers may return incorrect search results, either to save computational resources or for other motives. To address this issue, Miao et al. [24] combine document hash values with a bilinear pairing technique to verify result integrity. Shi et al. [25] design a novel data structure called CBF to verify the correctness of search results. Zhang et al. [26] ensure both correctness and comprehensiveness by adopting a multilevel hash function. Liu et al. [27] introduce a convergent key mechanism to double-check the correctness and comprehensiveness of search results. Li et al. [28] utilize homomorphic MAC and random polling techniques to verify the correctness and comprehensiveness of the returned results, respectively. However, these schemes [24–28] fail to simultaneously satisfy the three verification requirements: correctness, comprehensiveness, and integrity. In addition, existing CMSE schemes generally lack verification capabilities. Therefore, it is urgent to develop a verification mechanism for CMSE that can ensure all three properties in the returned search results.

Comprehensive analysis indicates that existing CMSE schemes still struggle to simultaneously achieve low time overhead and high retrieval accuracy. On the one hand, many existing schemes rely on complex cryptographic operations or large-scale matrix computations, which introduce considerable computational overhead and limit their applicability to real-time retrieval scenarios. On the other hand, the feature vectors generated by some schemes fail to fully capture the fine-grained semantic relationships between different modalities, which restricts retrieval accuracy. Moreover, existing CMSE schemes generally focus on encrypted cross-modal matching but pay insufficient attention to result verification. By comparison,

searchable encryption schemes that incorporate verification mechanisms are mostly designed for single-modal data and generally lack support for multimodal retrieval. To address these limitations, we propose a CMSE scheme that supports result verification and lightweight encrypted retrieval, aiming to enable efficient and verification over encrypted multimodal data.

3 Scheme Design

3.1 System Model

We show the system framework in Fig. 1, which contains four entities: data owner (DO), data user (DU), cloud server (CS), and trusted third party (TTP).

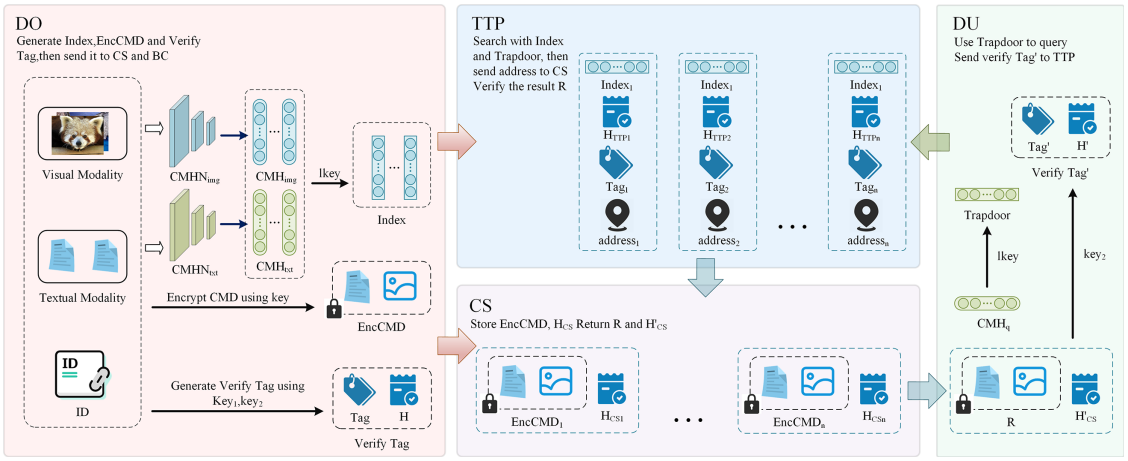

Figure 1: The system framework of our schemes.

DO: The DO uses multimodal data to generate an n-bit binary vector, denoted as CMH. This CMH is then encrypted using the key $lkey$ to generate the index $Index$. Next, the DO encrypts each piece of multimodal plaintext data (CMD_i) using the encryption key key, resulting in $EncCMD_i$. Meanwhile, a multilevel hash operation is performed on the ID of each CMD_i to generate an authentication tag H_i. Then, using H_i, $Nonce$, and $EncCMD_i$, the DO computes another authentication tag, denoted as Tag_i. We collectively refer to the set H (composed of all H_i) and the set Tag (composed of all Tag_i) as the $VerifyTag$. The DO uploads the $Index$ and the $VerifyTag$ set to the TTP and uploads $EncCMD$ and the corresponding set H to the CS. Finally, the DO sends the key, $lkey$, key_1, key_2, and $nonce$ to the DU. For the purpose of differentiation, we will denote the multi-level hash set sent to the CS as H_{CS} and the multi-level hash set sent to the TTP as H_{TTP}.

DU: The DU receives $lkey$, key_1, key_2, key, and $Nonce$ from the DO. It then transforms the search keyword w into an n-bit binary vector, denoted as CMH_q, and further converts it into a trapdoor, referred to as $Trapdoor$, which is sent to the TTP. After obtaining the result set R that matches the query word w and the associated H'_{CS}, the DU utilizes key_1, key_2, R, H_{CS}, and $Nonce$ to compute the authentication tag Tag' and the multi-level hash value H'. We refer to Tag' and H' collectively as $VerifyTag'$. DU then sends the $VerifyTag'$ to the TTP for verification. Upon completion of this process, we receive the result, $IsTrue$.

CS: As a third-party storage service, the CS stores encrypted multimodal data, $EncCMD$, and its corresponding multi-level hash set, H_{CS}. Upon receiving the address from the TTP, it returns the relevant result set R and H'_{CS} to the DU.

TTP: The TTP stores the security key Z_1, the $Index$, the set H_{TTP}, and the set Tag. TTP is responsible for performing the matching between the $Trapdoor$ and the $Index$, and returns the matched address to the CS. After receiving Tag' and H' from the DU, TTP compares Tag' with the stored Tag to verify result integrity. If the integrity check passes, the TTP takes all elements in the multi-level hash set H_{TTP}, which corresponds to R, as input and computes the multi-level hash H_c. It then compares H_c with H'; if they match, it sets the verification flag $IsTrue = True$ and returns the result to the DU.

3.2 Formal Definition

This scheme consists of seven polynomial-time algorithms, and the system process is shown in Fig. 2. The design of each algorithm is described as follows:

(1) $\boldsymbol{KeyGen(\lambda) \rightarrow lkey, Z_1, key, key_1, key_2}$**:** Given the security parameter λ as input, the algorithm outputs the following keys: the key $lkey$ and Z_1 for generating the $Index$ and $Trapdoor$, the encryption key key for protecting CMD, the key key_1 for multilevel hash computation, and the key key_2 for generating the authentication tag Tag.

(2) $\boldsymbol{ModelGen(TD) \rightarrow CMHmodel}$**:** The ModelGen algorithm takes a multimodal training dataset TD as input and outputs a CMH extraction model $CMHModel$. The model is responsible for mapping data from different modalities into a common Hamming space.

(3) $\boldsymbol{IndexBuild(CMD, lkey) \rightarrow Index}$**:** The IndexBuild algorithm utilizes the $CMHmodel$ to extract features from the CMD. Subsequently, these features are encrypted using the key $lkey$, resulting in the encrypted index, $Index$, for both images and text.

(4) $\boldsymbol{CMDEnc(CMD, key) \rightarrow EncCMD}$**:** In the encryption phase, the algorithm takes the multimodal data CMD and the encryption key key as input, and outputs the encrypted data $EncCMD$
$\boldsymbol{CMDDec(EncCMD, key) \rightarrow CMD}$**:** In the decryption phase, the algorithm takes $EncCMD$ and key as input, and returns the decrypted multimodal data CMD.

(5) $\boldsymbol{TrapdoorBuild(w, lkey) \rightarrow Trapdoor}$**:** The TrapdoorBuild algorithm takes the query keyword w and the key $lkey$ as inputs, and outputs the encrypted trapdoor $Trapdoor$.

(6) $\boldsymbol{Search(Trapdoor, Index, Z_1) \rightarrow Address}$**:** The search algorithm takes the trapdoor $Trapdoor$, the encrypted index $Index$, and the key Z_1 as input. and outputs the addresses of the top-m best-matched entries selected from the complete candidate set, denoted as $Address = \{address_1, address_2, \ldots, address_m\}$.

(7) $\boldsymbol{VTagGen(EncCMD, ID, Nonce, key_1, key_2) \rightarrow H_{TTP}, Tag, H_{CS}}$**:** The DO utilizes the encrypted content $EncCMD$, the corresponding identifier ID, a random nonce $Nonce$, and keys key_1 and key_2 to produce an authentication tag set Tag along with two multilevel hash sets, H_{CS} and H_{TTP}. These outputs are subsequently sent to the CS and the TTP, respectively.

(8) $\boldsymbol{verify(R, H'_{CS}, H'_{TTP}, Tag, key_1, key_2, Nonce) \rightarrow IsTrue}$**:** The algorithm is implemented by two entities, DU and TTP. The DU uses the encrypted query result set R, together with the keys key_1 and key_2, the received H'_{CS}, and the nonce $Nonce$, to compute a set of authentication tags Tag' and a new multilevel hash value H'. These values are sent to the TTP. TTP takes H'_{TTP} as input to compute the multilevel hash H_C and compares it with the received H'. At the same time, it compares Tag' with the stored Tag. If both $Tag' = Tag$ and $H' = H_C$ hold, the verification result $IsTrue$ is set to True; otherwise, it is set to False.

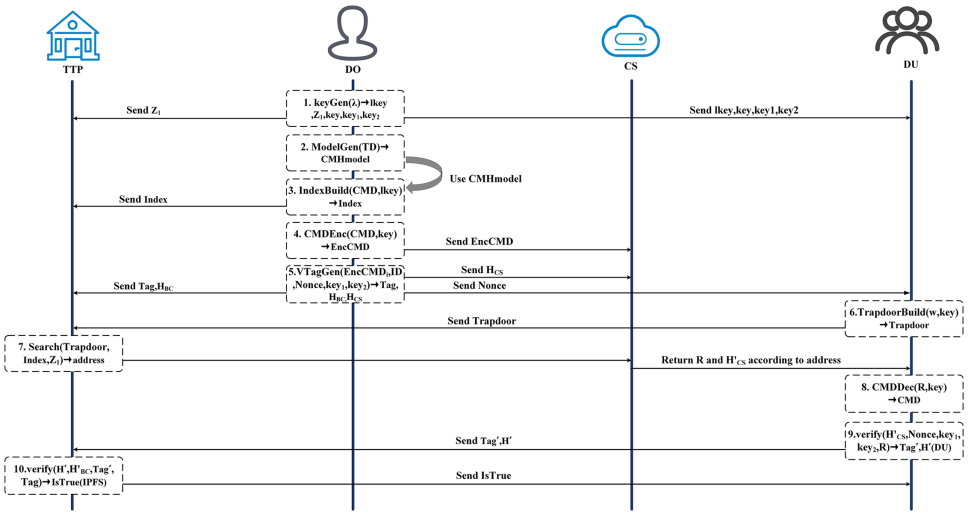

Figure 2: The system process of our schemes.

3.3 Threat Model

This scheme considers the DO and TTP as trusted entities, the DU as an untrusted entity, and the CS as a semi-honest but curious entity. Unlike the traditional honest-but-curious model in searchable encryption, the CS in this scheme, while generally following the protocol and not actively tampering with data, may still omit certain matching results to reduce computation costs or act maliciously. It may also return results that are inconsistent with those obtained from the TTP. As a result, verification mechanisms are required to ensure the correctness, integrity, and comprehensiveness of the returned results. Herein, we define our triple verification mechanism, which ensures that search results simultaneously satisfy correctness, comprehensiveness, and integrity. The definitions for the triple verification mechanism and its three properties are as follows:

Correctness: For a given query q, a result set R is considered correct if every document $R_i \in R$ satisfies the query q.

Comprehensiveness: For a given query q, let $DB(q)$ be the complete candidate set of all documents that satisfy the query. In top-m retrieval, R is considered comprehensive if it contains the m highest-scoring documents from $DB(q)$ without omitting any higher-scoring candidate.

Integrity: For a given query q, a result set R is considered to have integrity if every document $R_i \in R$ has not been tampered with or forged during the transmission process.

Triple verification mechanism: For a given query q, a result set R is considered verified by the triple verification mechanism if it satisfies the condition $R \subseteq DB(q)$, $|R| = m$, and there does not exist $D_i \in R$ and $D_j \in DB(q) \backslash R$ such that $Fraction_j > Fraction_i$, while every $R_i \in R$ has not been tampered with or forged during transmission.

Any tampering with the results, whether compromising their integrity, correctness, or comprehensiveness, will cause verification to fail, ensuring that the DU can identify any anomalies in the results in a timely and reliable manner.

Furthermore, to prevent dishonest DU from falsely claiming that the results are incorrect or incomplete as a pretext to avoid paying search fees, we delegate the final result verification to the TTP, thereby ensuring fairness between the CS and DU.

To prevent potential leakage of sensitive information, this scheme stores the encrypted multimodal data exclusively on the CS and places the encrypted index on the TTP. The TTP is modeled as a fully trusted entity and is therefore not considered part of the adversary. Its trustworthiness lies in its faithful execution of the protocol, rather than in unrestricted access to all underlying information. For this reason, both the trapdoor and the index are maintained in protected forms to avoid unnecessary semantic exposure. Under this threat model, we adopt a leakage-aware IND-CKA security model for the proposed searchable encryption scheme. Due to the inherent nature of similarity search, certain search-related leakage is unavoidable during retrieval. The defined leakage includes the public parameters, the difference information restored during $Search$, the resulting $Fraction$ scores, and the returned addresses and ranking order. In this model, the adversary A attempts to distinguish two search results generated from challenge instances with the same defined leakage. The security goal is that, except for the defined leakage, $\mathcal{A}$ cannot obtain additional information about the underlying CMH values or plaintext multimodal data. The formalization is as follows:

Setup: Challenger $\mathcal{C}$ runs $KeyGen$ once to generate the secret keys, including $lkey$, Z_1, key, key_1, and key_2. The generated $lkey$ is then used in both $IndexBuild$ and $TrapdoorBuild$.

Query Phase: Adversary $\mathcal{A}$ is allowed to make polynomially many adaptive queries. For each query, Challenger $\mathcal{C}$ runs the corresponding algorithm using the same keys as in the real scheme and returns the generated index, trapdoor, or search result.

Challenge: Adversary $\mathcal{A}$ submits two challenge instances, each consisting of a dataset and a query history. The two challenge instances must have identical leakage under the defined leakage function. Challenger $\mathcal{C}$ selects a random bit $b \leftarrow \{0,1\}$, runs $IndexBuild$, $TrapdoorBuild$, and $Search$ on the selected challenge instance, and returns the corresponding indexes, trapdoors, and search results to adversary $\mathcal{A}$.

Guess: Adversary $\mathcal{A}$ outputs a bit b' according to the received searchable result.

Winning Condition: If $b = b'$, adversary $\mathcal{A}$ is considered to have won the game. The advantage of $\mathcal{A}$ is defined as $\mathrm{Adv}_{\mathcal{A}}^{\text{IND-CKA}}(\kappa) = \left|Pr[b' = b] - \frac{1}{2}\right|$.

Security Definition: If for any polynomial-time adversary $\mathcal{A}$, the advantage $\mathrm{Adv}_{\mathcal{A}}^{\text{IND-CKA}}(\kappa)$ is negligible for any two challenge instances with identical defined leakage, then the proposed scheme is considered secure under the leakage-aware IND-CKA model.

4 Scheme Realization

4.1 KeyGen

Input the security parameter λ, and generate the CMD encryption key key, the key key_1 used for multi-level hash function calculations, and the key key_2 for generating the verification tag Tag. Then use the pseudo-random number generator (PRNG) to generate λ-bit Δ and n λ-bit k_0, where $LSB(\Delta) = 1$. Then generate n k_1 according to the formula $k_{i,1} = k_{i,0} \oplus \Delta$. Let $lkey = (lkey_1, lkey_2, \ldots, lkey_n)$, and $lkey_i = \{k_{i,0}, k_{i,1}\}$. Let $Z_1 = (Z_{1,1}, Z_{2,1}, \ldots, Z_{n,1})$, where $Z_{i,1} = k_{i,0} \oplus k_{i,1}$.

4.2 ModelGen

In order to accurately extract CMH that lie in the same Hamming space from multimodal data, we propose the Residual Alignment Cross Modal Hashing Network (RACMH-Net). As shown in Fig. 3, RACMH-Net is composed of the following six main components: the image feature extraction module $Feature_{img}$, the text feature extraction module $Feature_{txt}$, the image cross modal hash (CMH_{img}) mapping network $CMHN_{img}$, the text cross modal hash (CMH_{txt}) mapping network $CMHN_{txt}$, the multimodal alignment module $Alignment_{CM}$ and the residual similarity calibration module RSC.

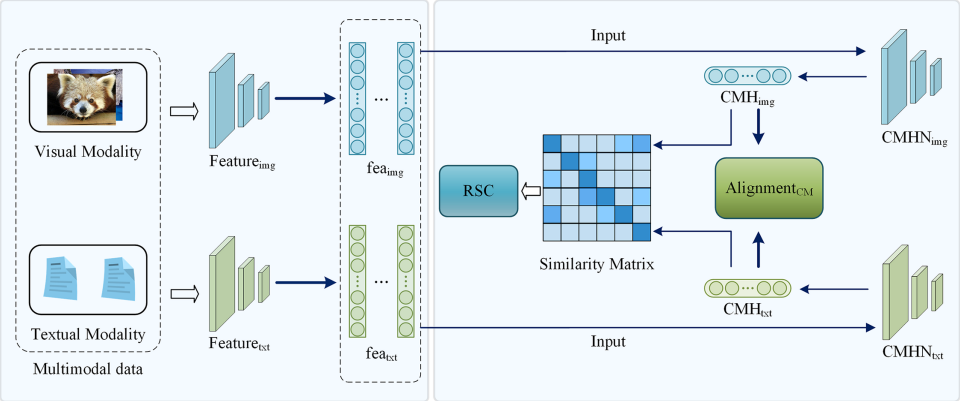

Figure 3: Framework of the proposed ModelGen.

RACMH-Net utilizes Vgg11 and BoW to extract features from the image and text modalities, respectively. In each training round, the extracted image features Fea_{img} and text features Fea_{txt} are fed into $CMHN_{img}$ and $CMHN_{txt}$ to train these two modules. $CMHN_{img}$ and $CMHN_{txt}$ share the same structure, each consisting of a neural network Net followed by a $Sign()$ function. The input features are compressed into an n-bit vector fea after passing through Net, denoted as $fea = (x_1, x_2, \ldots, x_n)$. The fea is then transformed into CMH_{img} and CMH_{txt} via the $Sign()$ function, which binarizes each x_i in fea: if $x_i > 0$, it is set to 1; otherwise, it is set to −1, as shown in Eq. (1).

$$Sign = \begin{cases} 1, & x_i > 0 \\ -1, & x_i < 0 \end{cases} \tag{1}$$

To ensure CMH_{img} and CMH_{txt} remain in the same Hamming space, the scheme adopts a contrastive learning approach for feature alignment. $CMHN_{img}$ and $CMHN_{txt}$ are trained to minimize the distance between similar pairs (CMH_{img}, CMH_{txt}) and maximize the distance between dissimilar ones. The loss function for this module is shown in Eq. (2).

$$\mathcal{L}_{align} = \frac{1}{2}(\mathcal{L}_{i2t} + \mathcal{L}_{t2i}) \tag{2}$$

Here, $\mathcal{L}_{i2t}$ uses CMH_{img} as the anchor to match the corresponding CMH_{txt}, and $\mathcal{L}_{t2i}$ does the reverse. Due to the similarity in principle, only Eq. (3) is presented.

$$\mathcal{L}_{i2t} = -\sum_{i=1}^{B} \log \frac{\exp\left(\left\langle \hat{b}_i^p, \hat{b}_i^t \right\rangle / \tau\right)}{\sum_{j=1}^{B} \exp\left(\left\langle \hat{b}_i^p, \hat{b}_j^t \right\rangle / \tau\right)}, \quad \hat{b} = \frac{b}{\|b\|_2} \tag{3}$$

Here τ is the temperature coefficient, $\hat{b}$ is the normalized binary vector, $\hat{b}_i^p$ represents the i-th normalized CMH_{img}, and $\hat{b}_j^t$ represents the j-th normalized $CMHt_{xt}$.

While contrastive learning establishes a basic discriminative capability for cross-modal retrieval by maximizing positive pair similarity and minimizing negative pair similarity, it suffers from an inherent limitation: it tends to uniformly push all negative samples apart in the Hamming space. Consequently, the potential fine-grained semantic correlations among these negative samples are ignored. To ensure the structure of the CMH aligns closely with the true semantics, our proposed scheme incorporates an RSC module. RSC computes a similarity matrix between the outputs of $CMHN_{img}$ and $CMHN_{txt}$, then models

and minimizes the residuals between this matrix and the true similarity matrix, thereby improving cross-modal alignment quality and semantic consistency. The loss function of this module is shown in Eq. (4).

$$\mathcal{L}_{\text{residual}} = \left\| \frac{1}{D} B_v B_t^{\mathrm{T}} - M_r \right\|_F^2 + \left\| \frac{1}{D} B_t B_v^{\mathrm{T}} - M_r \right\|_F^2 \tag{4}$$

here, B_v denotes the normalized CMH_{img}, and B_t^T is the transpose of the normalized CMH_{txt}. Their product M_t is equal to $D - 2Hamming(B_i^v, B_j^t)$, which is linearly related to the Hamming distance and reflects the similarity between CMH_{img} and CMH_{txt} in Hamming space. D is the normalization factor of M_t, and M_r is the similarity matrix constructed from dataset labels. The first term of the loss uses CMH_{img} as the anchor to match the corresponding CMH_{txt}, and the second term does the reverse.

To achieve the optimal CMH mapping between $CMHN_{img}$ and $CMHN_{txt}$, the scheme minimizes both $\mathcal{L}_{residual}$ and $\mathcal{L}_{algin}$ simultaneously. Therefore, a joint loss function is defined as Eq. (5).

$$\mathcal{L} = \alpha \mathcal{L}_{\text{residual}} + \beta \mathcal{L}_{\text{align}} \tag{5}$$

The ModelGen algorithm is shown in Algorithm 1.

Algorithm 1: MoodelGen

Input: The training image-text pairs $TD = \{img, txt\}$, the length of the hash codes l, batch size Num and learning rate lr.

Output: $CMHModel$

$fea_{img} = Vgg11(img)$
$fea_{txt} = BoW(txt)$
for each epoch
  train(fea_{img}, fea_{txt}):
    $fea_{img} = Net_{img}(fea_img)$
    $fea_{txt} = Net_{txt}(fea_txt)$
    $CMH_{img} = Sign(fea_{img})$
    $CMH_{txt} = Sign(fea_{txt})$
    Calculate $\mathcal{L}_{align}$
    Calculate $\mathcal{L}_{residual}$
    $\mathcal{L} = \alpha \mathcal{L}_{\text{residual}} + \beta \mathcal{L}_{\text{align}}$
    Update the gradients
  end train
end for
Storage Model

4.3 IndexBuild

In the index generation stage, to achieve privacy protection, the extracted CMH needs to be encrypted. The Hamming distance between CMH can be used to measure their similarity. For this purpose, we have optimized the garbled circuit for the specific scenario of Hamming distance calculation. By abandoning the garbled tables and oblivious transfer protocols typically found in garbled circuits, and instead extracting only the core idea of substituting real values with secret labels, we designed a minimalist, noninteractive algebraic structure. This Lightweight Garbled Circuit (LGC) trades generality for high efficiency in Hamming distance computation. As shown in Fig. 4, the method for generating the index is primarily composed of two main modules: cross modal hash generation (CMHGen) and cross modal hash encryption (CMHEnc).

1. CMHGen: DO maps fea_{img} and fea_{txt} to CMH_{img} and CMH_{txt}, which lie in the same Hamming space, by utilizing the $CMHN_{img}$ and $CMHN_{txt}$ from the model $CMHModel$. Given a cross modal hash set CMH, such that $\forall\, CMH_{\text{img}}, CMH_{\text{txt}} \subseteq CMH$ holds, and for each CMH_i in $CMH = (x_1, x_2, \ldots, x_n)$, there is $x \in \{-1, 1\}$.
2. CMHEnc: For each CMH_i in the CMH, CMHEnc applies Eq. (6) sequentially to each component x_i in CMH_i, using the key $lkey$, thereby generating the index $Indx_i = (L_1, L_2, \ldots, L_n)$.

$$L_i = \begin{cases} k_{i,0} & \text{if } x_i = -1 \\ k_{i,1} & \text{if } x_i = 1 \end{cases} \tag{6}$$

Here, component x_i denotes the value of the vector CMH_i in dimension i, and component L_i denotes the value of the vector $Index_i$ in dimension i. The key $lkey$ is defined in Eq. (7).

$$\begin{aligned} &lkey = (lkey_1, \ldots, lkey_n), \quad lkey_i = \{k_{i,0}, k_{i,1}\} \\ &k_{i,0} \leftarrow \{0,1\}^{\kappa}, \quad k_{i,1} = k_{i,0} \oplus \Delta \\ &\Delta \leftarrow \{0,1\}^{\kappa} \text{ and } LSB(\Delta) = 1 \end{aligned} \tag{7}$$

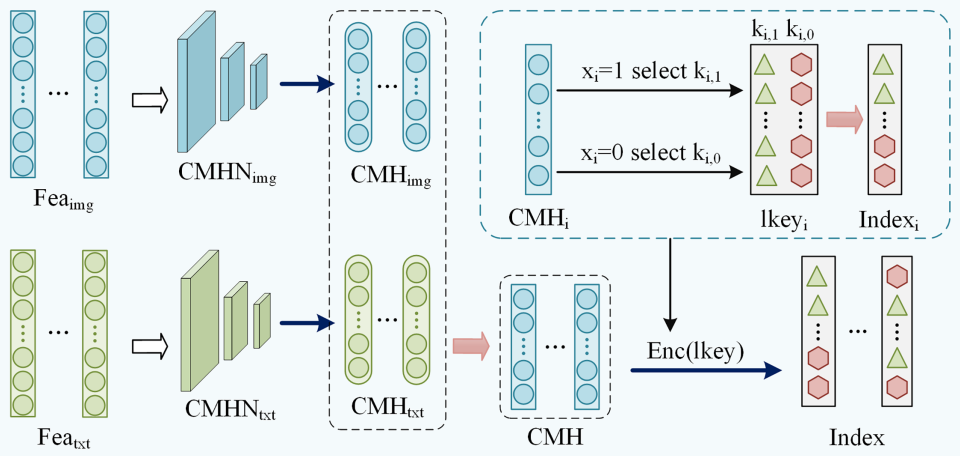

Figure 4: IndexBuild.

In addition, during the execution of CMHEnc, IndexBuild packs all the CMH_i in CMH to be encrypted at once into a tensor of shape [N, L], where N is the number of CMH_i, and L is the dimensionality of each CMH_i vector. Afterward, the random labels corresponding to each wire are extracted in parallel via one-time advanced indexing, generating a tensor of shape [N, L, kappa(32)], where kappa represents the byte length of each CMH_i component. This tensor forms the index set $Index$.

4.4 CMSE-Process

DO divides the plaintext data into a number of chunks $(M_1, M_2, \ldots, M_n)$ and generates a random number $Nonce$ for the entire set. After that, an $C_i = M_i \oplus E_{key}(Nonce \parallel i)$ operation is performed on each plaintext chunk M_i, and all the resulting ciphertext chunks C_i are concatenated to construct the encrypted multimodal data $EncCMD$.

DU, upon receiving $EncCMD$, splits it into the original chunks $(C_1, C_2, \ldots, C_n)$ based on the chunk length used during encryption. It then performs the $M_i = C_i \oplus E_{key}(Nonce \parallel i)$ operation on each

encrypted chunk C_i, and concatenates the resulting plaintext blocks M_i in order to reconstruct the original multimodal data CMD.

4.5 TrapdoorBuild

Assuming that img and txt represent the image modality and text modality, respectively, if DU wants to query img, the type of w is txt; conversely, if txt is queried, the type of w is img. The system loads either $CMHN_{img}$ or $CMHN_{txt}$ based on the modality type of the w and transforms the input into the corresponding CMH_{img} or CMH_{txt}. For uniform representation, this is denoted as CMH_q.

Given CMH_q, it can be expressed as $CMH_q = (y_1, y_2, \ldots, y_n)$, and satisfies $y_i \in \{-1, 1\}$. The trapdoor generation algorithm TrapdoorBuild receives CMH_q and $lkey$ as inputs, and applies Eq. (8) to each component y_i of CMH_q to generate the trapdoor $Trapdoor = (L_1, L_2, \ldots, L_n)$.

$$L_i = \begin{cases} k_{i,0} & \text{if } y_i = -1 \\ k_{i,1} & \text{if } y_i = 1 \end{cases} \tag{8}$$

here, component y_i denotes the value of the vector CMH_q in dimension i, component L_i denotes the value of the vector $Trapdoor_i$ in dimension i, and $lkey$ is defined as shown in Eq. (7).

4.6 Search

As shown in Fig. 5, the search method consists of two major modules: the similarity calculation module (SimilarCalculate) and the fraction sorting module (FSorting).

1. SimilarCalculate: Given $Trapdoor$ and the set $Index$, denote $Trapdoor$ as $Trapdoor = (y_1, y_2, \ldots, y_n)$, and each $Index_i$ in the set $Index$ as $Index_i = (x_1, x_2, \ldots, x_n)$. SimilarCalculate takes $Index_i$, $Trapdoor$, and Z_1 as inputs and computes the fraction of correlation ($Fraction$) between each $Index_i$ and $Trapdoor$ by performing Eq. (9).

$$Fraction = \sum_{i=1}^{n} x_i \oplus y_i \oplus Z_{1,i} \tag{9}$$

here, component x_i denotes the value of vector $Index_i$ in dimension i, component y_i denotes the value of vector $Trapdoor_i$ in dimension i, and Z_1 is defined in Eq. (10).

$$\begin{aligned} Z_1 &= (Z_{1,1}, \ldots, Z_{1,n}) \\ Z_{i,1} &= k_{i,0} \oplus k_{i,1} \end{aligned} \tag{10}$$

The value of $Fraction$ is a bit-wise matching score between the $Index_i$ and the $Trapdoor$. For two n-bit $CMHs$, a larger $Fraction$ indicates more matching bit positions and therefore corresponds to a smaller Hamming distance between the associated CMH and CMH_q. Thus, a larger $Fraction$ indicates a better match, while a smaller $Fraction$ indicates a weaker match.

2. FSorting: The address-score pairs ($address_i$, $Fraction_i$) corresponding to each $Index_i$ are sorted in ascending order according to their $Fraction$ values. The top m addresses with the highest $Fraction$ scores are selected to form the address set $Address$, which is Address = $\{address_1, \ldots, address_m\}, Fraction_1 \geq \ldots \geq Fraction_n$

During the execution of similarCalculate, the search algorithm extracts the least significant bits of both $Index$ and $Trapdoor$ in a single operation and precomputes the decoding mask. Then, $Trapdoor$ and $Index$ are batch-wise broadcasted and XORed to generate intermediate results, which are further XORed with Z_1

to recover the true difference bits. Finally, a summation is performed along the bit dimension to compute the *Fraction* in parallel.

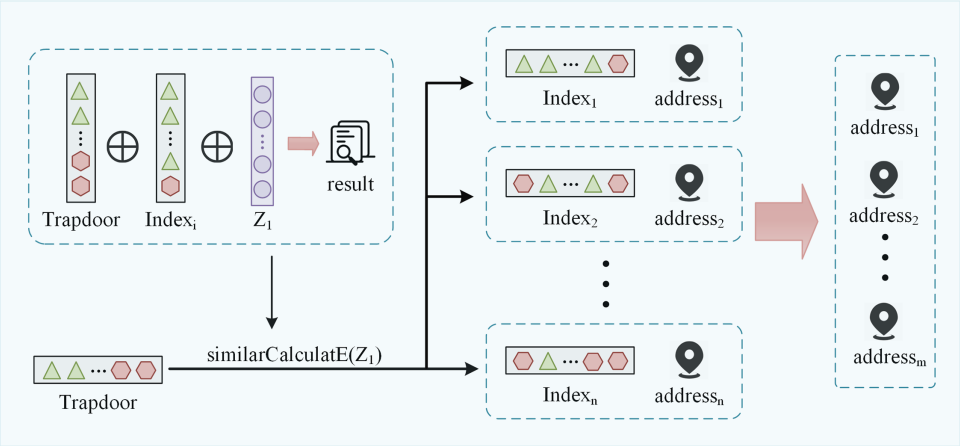

Figure 5: Search.

4.7 Verify

The verify method has an execution process that includes two stages: verification tag generation and result verification. The verification tag generation stage occurs before the DU submits a query q. In this stage, the VTagGen module precomputes and generates the tags H_i and Tag_i for subsequent validation. The result verification stage is handled by the verify module. After the DU receives the returned result set R from the CS, the Verify module is responsible for verifying its integrity, comprehensiveness, and correctness.

1. VTagGen: The module computes the multilevel hash H_i based on the ID corresponding to CMD_i, according to Eq. (11), where $H_i = \{h, c, r\}$.

$$\begin{aligned} h &= H_{key,}(0 \parallel r) \oplus H_{key,}(1 \parallel ID) \\ c &= 1 \bmod 2^m, \quad r \leftarrow \{0,1\}^m \end{aligned} \tag{11}$$

Then, VTagGen traverses $EncCMD$, and for each $EncCMD_i$ and its corresponding H_i, the tag Tag_i is computed according to Eq. (12).

$$Tag_i = G_k(\text{key}_2, G_k(H_i, \text{EncCMD}_i)) \oplus \text{pad}(\text{Nonce}) \tag{12}$$

here, $G_k(X, Y)$ denotes a truncated compression function applied to Y using key X, and is defined in Eq. (13).

$$\text{DM}[E, m](X, Y) = E_X(Y) \oplus Y[1..m] \tag{13}$$

The VTagGen algorithm is shown in Algorithm 2.

2. Verify: DU receives the set R and the set H'_{CS} sent by CS, where $R = \{R_1, \ldots, R_n\}$ and $H'_{CS} = \{H'_{CS_1}, \ldots, H'_{CS_n}\}$, and each R_i corresponds one-to-one with an element in H'_{CS}. DU traverses R and H'_{CS}, computes the corresponding Tag'_i for each R_i and H'_{CS_i} based on Eq. (14), and aggregates them into the set Tag', such that $Tag' = \{Tag'_1, \ldots Tag'_n\}$.

$$Tag'_i = G_k(key_2, G_k(H'_{CS_i}, R_i)) \oplus \text{pad}(Nonce) \tag{14}$$

After that, DU computes the multilevel hash H' of the set H'_{CS} according to Eq. (15), and sends Tag' and H' to TTP.

$$\begin{aligned} c &= (c_1 + c_2 + \ldots + c_n) \bmod 2^m, \quad r \leftarrow \{0,1\}^m \\ h &= \left(H_{key_1}(0 \parallel r) \oplus h_1\right) \oplus \ldots \oplus \left(H_{key_1}(0 \parallel r) \oplus h_n\right) \end{aligned} \tag{15}$$

Assume that $H'_{TTP} = \{H'_{TTP_1}, \ldots, H'_{TTP_n}\}$ and $Tag = \{Tag_1, \ldots, Tag_n\}$, where elements in H'_{TTP} correspond one-to-one with elements in R. Upon receiving Tag' and H' from DU, TTP checks whether the sets Tag and Tag' are equal. At the same time, TTP computes the multilevel hash of H'_{TTP} according to Eq. (15) and compares the computed hash H_C with the received H'. If both conditions $Tag = Tag'$ and $H_C = H'$ are satisfied, TTP sets the Boolean variable $IsTrue$ to True. Otherwise, $IsTrue$ is set to False. Finally, $IsTrue$ is returned to DU. The DU executes the Verify algorithm shown in Algorithm 3, while the TTP executes the Verify algorithm shown in Algorithm 4.

Algorithm 2: VTagGen

Input: $EncCMD$, ID, $Nonce$, key_1, key_2
Output: Tag, H_{TTP}, H_{CS}

$r \leftarrow \{0,1\}^m$, $c = 1 \bmod 2^m$
$h = H_{\text{key}_1}(0 \parallel r) \oplus H_{\text{key}_1}(1 \parallel ID)$
$H_i = \{h, c, r\}$, $H'_i = h \parallel c \parallel r$
$rr = E_{H'_i}(\text{EncCMD}_i) \oplus \text{EncCMD}_i[1..m]$
$v = rr \oplus pad(Nonce)$
$Tag_i = E_{\text{key}_2}(V) \oplus V[1..m]$
$H_{TTP} = \{H_1, \ldots, H_n\}$
$H_{CS} = \{H_1, \ldots, H_n\}$
$Tag = \{Tag_1, \ldots, Tag_n\}$
return Tag, H_{TTP}, H_{CS}

Algorithm 3: Verify (DU)

Input: R, H'_{CS}, Tag, $Nonce$, key_1, key_2
Output: $H' = \{h, c, r\}$

$Tag' = []$
$c = H'_{CS}.c$, $r \leftarrow \{0,1\}^m$
$h = H_{\text{CS}'_1} \cdot h \oplus H_{\text{key}_1}(0 \parallel r)$
for R_i, H_i in R and H'_{CS}:
    $h'_{CS_i} = H'_{CS_i} \cdot h \parallel H'_{CS_i} \cdot c \parallel H'_{CS_i} \cdot r$
    $rr = E_{H'_{CS_i}}(R_i) \oplus R_i[1..m]$
    $v = rr \oplus pad(Nonce)$
    $Tag_{temp} = E_{\text{key}_2}(v) \oplus v[1..m]$
    $Tag'.append(Tag_{temp})$
    if $H'_{CS_i} = H'_{CS_1}$: continue
    $c = (H_i.c + c) \bmod 2^m$
    $h = h \oplus (H_i \cdot h \oplus H_{\text{key}_1}(0 \parallel r))$
$H' = \{h, c, r\}$

Algorithm 4: Verify (TTP)

Input: Tag', R, H'_{TTP}, Tag, key_1, key_2
Output: $IsTrue$
for Tag'_i and Tag_i in Tag' and Tag
    if $Tag'_i = Tag_i$: continue
    else
        $IsTrue$=False
        return $IsTrue$
$c = H'_{TTP_1}.c$
$h = H'_{TTP_1} \cdot h \oplus H_{key_1}(0 \parallel H'_{TTP_1} \cdot r)$
$r \leftarrow \{0,1\}^m$
H'_{TTP}=H'_{TTP} delete H'_{TTP_1}
for H'_{TTP_i} in H'_{TTP}
$c = (H'_{TTP_i}.c + c) \bmod 2^m$
    $h = h \oplus (H'_{TTP_i}.h \oplus H_{key_1}(0 \parallel H'_{TTP_i}.r))$
$H_c = \{h, c, r\}$
if $H_c = H'$: $IsTrue$=True
else
    $IsTrue$=False
return $IsTrue$

5 Security Analysis

5.1 Index-Trapdoor Confidentiality

In this subsection, we analyze the confidentiality of $Index$ and $Trapdoor$ under the leakage-aware IND-CKA model defined in Section 3.3. Due to the inherent nature of similarity search, retrieval necessarily reveals limited search-related information, such as the restored difference information, $Fraction$ scores, and ranking results, which are treated as defined leakage in our security model. Beyond such leakage, the labels in $Index$ and $Trapdoor$ should not reveal additional information about the underlying CMH values. Under the HXI assumption, these labels are computationally indistinguishable from random labels. We present the label indistinguishability proof below.

Definition 1: HXI computational hardness problem. The task of the adversary is to distinguish between two label distributions as described in $L \in (\{0,1\}^\kappa)^n$. D_{pair}: A global difference Δ is randomly chosen, where $\Delta \leftarrow \{0,1\}^\kappa$ and $LSB(\Delta) = 1$. For each $i \in \{1, \ldots, n\}$, there are $k_{i,0} \leftarrow \{0,1\}^\kappa$, and one element is randomly selected from $\{k_{i,0}, k_{i,0\oplus\Delta}\}$ as L_i. D_{rand}: For each $i \in \{1, \ldots, n\}$, one element is selected independently and uniformly at random from $\{0,1\}^\kappa$ as L_i. This scheme assumes that the HXI assumption holds if no probabilistic polynomial-time (PPT) adversary can distinguish D_{pair} from D_{rand} with non-negligible advantage.

The following games are constructed in this paper following a standard hybrid argument for indistinguishability-based security analysis. Game 0 corresponds to the real label-generation process of the proposed scheme. Game 1 replaces the encoded bit by an independently sampled random bit, and Game 2 further replaces the paired-label distribution with a fully random distribution. These games are adapted to the label structure defined in $KeyGen$, $IndexBuild$, and $TrapdoorBuild$.

Game 0

Challenger $\mathcal{C}$ generates a global difference Δ according to $\Delta \leftarrow \{0,1\}^{\kappa}$, and ensures that $LSB(\Delta) = 1$.

Adversary $\mathcal{A}$ performs polynomially many queries to the encryption oracle OEnc(.).

$\mathcal{A}$ sends x_i and x_i' to $\mathcal{C}$.

$\mathcal{C}$ outputs $x^{(b)}$ according to $b \leftarrow \{0,1\}$

$\mathcal{C}$ generates $k_{i,0}$ and $k_{i,1}$, satisfying $k_{i,0} \leftarrow \{0,1\}^{\kappa}$ and $k_{i,1} = k_{i,0} \oplus \Delta$. If $x_i^{(b)} = 0$ holds, then $L_i = k_{i,0}$ applies; if $x_i^{(b)} = 1$ holds, then $L_i = k_{i,1}$ applies. Finally, $L = \{L_1 \ldots, L_n\}$ is derived, and the label set L is sent to $\mathcal{A}$.

$\mathcal{A}$ outputs a prediction b'.

Game 1

Challenger $\mathcal{C}$ generates a global difference Δ according to $\Delta \leftarrow \{0,1\}^{\kappa}$, and ensures that $LSB(\Delta) = 1$.

Adversary $\mathcal{A}$ performs polynomially many queries to the encryption oracle OEnc(.).

$\mathcal{A}$ sends x_i and x_i' to $\mathcal{C}$.

$\mathcal{C}$ outputs $x^{(b)}$ according to $b \leftarrow \{0,1\}$

$\mathcal{C}$ generates $k_{i,0}$, $k_{i,1}$ and r_i, satisfying $k_{i,0} \leftarrow \{0,1\}^{\kappa}$, $k_{i,1} = k_{i,0} \oplus \Delta$ and $r_i \leftarrow \{0,1\}$. If $r_i = 0$ holds, then $L_i = k_{i,0}$ applies; if $r_i = 1$ holds, then $L_i = k_{i,1}$ applies. Finally, $L = \{L_1 \ldots, L_n\}$ is derived, and the label set L is sent to $\mathcal{A}$.

$\mathcal{A}$ outputs a prediction b'.

Game 2

Challenger $\mathcal{C}$ generates a global difference Δ according to $\Delta \leftarrow \{0,1\}^{\kappa}$, and ensures that $LSB(\Delta) = 1$.

Adversary $\mathcal{A}$ performs polynomially many queries to the encryption oracle OEnc(.).

$\mathcal{A}$ sends x_i and x_i' to $\mathcal{C}$.

$\mathcal{C}$ outputs $x^{(b)}$ according to $b \leftarrow \{0,1\}$

$\mathcal{C}$ generates $L = \{R_1, \ldots, R_n\}$, where $R_i \leftarrow \{0,1\}^{\kappa}$, and sends L to $\mathcal{A}$

$\mathcal{A}$ outputs a prediction b'.

Lemma 1: *For any PPT adversary $\mathcal{A}$, Game 0 and Game 1 are perfectly indistinguishable.*

Proof: The following probability derivation are derived from the uniform sampling of $k_{i,0}$, the relation $k_{i,1} = k_{i,0} \oplus \Delta$, and the independence of labels across different dimensions. For Game 0, we define the distribution D_0. For Game 1, we define the distribution D_1. Due to $k_{i,0} \leftarrow \{0,1\}^{\kappa}$, $\Pr[k_{i,0} = \alpha] = 2^{-\kappa}$ follows. Furthermore, because of $k_{i,1} = k_{i,0} \oplus \Delta$, we get $\Pr[k_{i,1} = \alpha] = 2^{-\kappa}$, where $\alpha \leftarrow \{0,1\}^{\kappa}$.

For distribution D_0, the labels $L^{(0)} = (\kappa_{1,x_1}, \ldots, \kappa_{n,x_n})$, satisfy:

$$\Pr\left[L_i^{(0)} = \alpha\right] = \Pr\left[k_{i,x_i} = \alpha\right] = 2^{-\kappa}$$

For distribution D_1, the labels $L^{(1)} = (\kappa_{1,r_1}, \ldots, \kappa_{n,r_n})$ hold. Since $r_i \leftarrow \{0,1\}$ holds, it follows that $\Pr\left[L_i^{(1)} = k_{i,0}\right] = \frac{1}{2}$ and $\Pr[L_i^{(1)} = k_{i,1}] = \frac{1}{2}$ hold. Therefore, we obtain

$$\begin{aligned}
&\Pr\left[L_i^{(1)} = \alpha\right] = \Pr\left[L_i^{(1)} = k_{i,0}\right] \cdot \Pr\left[k_{i,0} = \alpha\right] \\
&+ \Pr\left[L_i^{(1)} = k_{i,1}\right] \cdot \Pr\left[k_{i,1} = \alpha\right] \\
&= \frac{1}{2} \cdot 2^{-\kappa} + \frac{1}{2} \cdot 2^{-\kappa} \\
&= 2^{-\kappa}
\end{aligned}$$

Since $\{k_{i,0}\}$ for different i are independently sampled, the $\{L_i\}$ are also mutually independent in both distributions. Hence, for any $\ell = (\ell_1, \ldots, \ell_n) \in \{0,1\}^\kappa$, there is:

$$\Pr\left[L^{(0)} = \ell\right] = \prod_{i=1}^{n} \Pr\left[L_i^{(0)} = \ell_i\right] = \left(2^{-\kappa}\right)^n = 2^{-n\kappa}$$

Similarly, we get:

$$\Pr\left[L^{(1)} = \ell\right] = \prod_{i=1}^{n} \Pr\left[L_i^{(1)} = \ell_i\right] = \left(2^{-\kappa}\right)^n = 2^{-n\kappa}$$

leading to $\Pr[L^{(0)} = \ell] = \Pr[L^{(1)} = \ell]$, therefore $\Pr[\mathcal{A}(L^{(0)}) = 1] = \Pr[\mathcal{A}(L^{(1)}) = 1]$, which satisfies $Pr[L^{(0)} = \ell] - Pr[L^{(1)} = \ell] = 0$. This completes the proof. □

Lemma 2: *For any PPT adversary $\mathcal{A}$, Game 1 and Game 2 are computationally indistinguishable.*

Now we construct a reduction proof. The reduction is constructed according to the equivalence between the distributions in Game 1 and Game 2 and the distributions defined in the HXI assumption. Suppose there exists a PPT adversary $\mathcal{A}$ that distinguishes between Game 1 and Game 2 with non-negligible advantage $\varepsilon(\kappa)$. We construct a new PPT algorithm $\mathcal{B}$ using $\mathcal{A}$ as a subroutine, and $\mathcal{B}$ will break the HXI assumption with the same advantage. In this setting, the label set $L^{(1)}$ in Game 1 is distributionally equivalent to D_{pair}, and the label set L'' in Game 2 is equivalent to D_{rand}.

Construction of algorithm $\mathcal{B}$: Algorithm $\mathcal{B}$ receives a label set L' from an external HXI challenger. Here, L' is either sourced from D_{pair} or from D_{rand}. $\mathcal{B}$'s goal is to determine the source of L'. $\mathcal{B}$ sends the received L' to $\mathcal{A}$. If $L' \leftarrow D_{pair}$ holds, it is equivalent to $\mathcal{B}$ sending the label set $L^{(1)}$ of Game 1; if $L' \leftarrow D_{rand}$ holds, it is equivalent to $\mathcal{B}$ sending the tag set L'' of Game 2. After completing its subsequent computations, $\mathcal{A}$ will output a decision to distinguish between Game 1 and Game 2. To simplify the analysis, we assume that $\mathcal{A}$ outputs 1 when it believes it is in Game 1, at which point $\mathcal{B}$ determines that L' comes from D_{pair}; when in Game 2, it outputs 0, at which point $\mathcal{B}$ determines that L' comes from D_{rand}.

Advantage analysis for $\mathcal{B}$: If $L' \leftarrow D_{pair}$ holds, then the environment for $\mathcal{A}$ is identical to Game 1. In this case, the probability that $\mathcal{A}$ outputs 1 is $\Pr[\mathcal{A}(L^{(1)}) = 1]$, $\Pr[\mathcal{B}(D_{pair}) = 1] = Pr[\mathcal{A}(L^{(1)}) = 1]$; If $L' \leftarrow D_{rand}$ holds, then the environment for $\mathcal{A}$ is identical to Game 2. In this case, the probability that A outputs 1 is $\Pr[\mathcal{A}(L'') = 1]$, $\Pr[\mathcal{B}(D_{rand}) = 1] = \Pr[\mathcal{A}(L'') = 1]$, so the distinguishing advantage of $\mathcal{B}$ is:

$$\begin{aligned}
&\mathrm{Adv}_B^{HXI}(\kappa) \\
&= \left|\Pr\left[\mathcal{B}\left(D_{\mathrm{pair}}\right) = 1\right] - \Pr\left[\mathcal{B}\left(D_{\mathrm{rand}}\right) = 1\right]\right| \\
&= \left|\Pr\left[\mathcal{A}\left(L^{(1)}\right) = 1\right] - \Pr\left[\mathcal{A}\left(L''\right) = 1\right]\right| \\
&= \varepsilon(\kappa)
\end{aligned}$$

Since $\mathrm{Adv}_B = \varepsilon(\kappa) \notin negl(\kappa)$, exactly violates *Definition 1* that the advantage of any PPT algorithm in distinguishing D_{pair} from D_{rand} must be negligible. From this contradiction, we know that the assumption that there exists a PPT adversary $\mathcal{A}$ that can distinguish between Game 1 and Game 2 with a non-negligible advantage $\varepsilon(\kappa)$ is invalid. Therefore, for any PPT adversary $\mathcal{A}$, the advantage it gains in distinguishing between Game 1 and Game 2 is negligible. This completes the proof.

According to the triangle inequality, for an adversary $\mathcal{A}$:

$$
\begin{aligned}
&\left|\Pr[\mathcal{A}(L^{(0)}) = 1] - \Pr[\mathcal{A}(L'') = 1]\right| \\
&\leq \left|\Pr[\mathcal{A}(L^{(0)}) = 1] - \Pr[\mathcal{A}(L^{(1)}) = 1]\right| \\
&+ \left|\Pr[\mathcal{A}(L^{(1)}) = 1] - \Pr[\mathcal{A}(L'') = 1]\right|| \\
&\leq 0 + \mathrm{negl}(\kappa) \\
&\leq \mathrm{negl}(\kappa)
\end{aligned}
$$

Because L'' is completely random, we have $\Pr[\mathcal{A}(L'') = 1] = 2^{-1}$.

Therefore, $\mathrm{Adv}_{\mathcal{A}}^{\text{IND-CPA}}(\kappa) = \left|\Pr[\mathcal{A}(L^{(0)}) = 1] - \frac{1}{2}\right| \leq \mathrm{negl}(\kappa)$. This completes the proof. □

5.2 Verification Analysis

The VCMSE scheme supports the verification of comprehensiveness, integrity, and correctness of the results returned by the CS. To prevent the DU from maliciously claiming that the received data is incomplete or incorrect in order to refuse payment, the scheme introduces a trusted third party, TTP. The DU cannot perform verification independently and must cooperate with the TTP for joint verification.

The trusted third party verifies the integrity of the returned results by comparing the set Tag' with the set Tag. For a given query q, if any record R_i in the result set R or its corresponding H_i is tampered with during transmission, the new tag Tag_i' generated from the altered data will not match the original, authentic tag Tag_i. The security of this integrity verification relies on the collision resistance of the truncated compression function DM. We argue that for an adversary's tampering to go undetected, they would need to find a Tag_{fake} that is identical to the original Tag. However, the collision resistance of DM makes such an action computationally infeasible. The collision resistance of DM has been proven in [29].

The TTP verifies the comprehensiveness and correctness of the returned results by determining whether H_c and H' are equal. If $\exists D_i \in DB(q)$ and $D_i \notin R$, it will cause $H_c \neq H'$, leading to a comprehensiveness verification failure. Similarly, if $\exists R_i \in R$ and $D_i \notin DB(q)$, it will also cause a correctness verification failure due to $H_c \neq H'$. Due to the collision resistance of the multi-level hash function [30], it is considered computationally infeasible for any PPT adversary $\mathcal{A}$ to forge a multi-level hash value H_{fake} that is identical to H_C, as described in Eq. (16):

$$
\Pr\left[x_{fake} \neq x_c \wedge H_{fake} = H_c\right] \leq \mathrm{negl}(\kappa) \tag{16}
$$

where x_c denotes the true input to the multi-level hash function, and x_{fake} denotes the forged input crafted by the adversary.

6 Experimental Analysis

6.1 Functional Analysis

In this section, the proposed scheme is compared with other CMSE schemes [16,17,20], and the results are presented in Table 1. For clarity, we abbreviate the schemes in [16,17,20] as SCMR, SCMS-FL, and

FECMR/FECMR+, respectively. In terms of the index structure, our scheme introduces a lightweight garbled circuit (LGC) designed to achieve multimodal retrieval in the ciphertext domain. Moreover, it verifies the integrity, comprehensiveness, and correctness of the retrieval results received by the DU, whereas the other schemes do not consider any result verification mechanism. Therefore, the proposed scheme not only achieves multimodal retrieval in the ciphertext domain but also verifies the retrieval results from multiple dimensions, offering more comprehensive functional coverage.

Table 1: Comparison of schemes.

Scheme	Index Structure	Cross-Modal	SE	Integrity	Comprehensiveness	Correctness
SCMR	Paillier HE	✓	✓	×	×	×
FECMR	ESFB-IPFE	✓	✓	×	×	×
FECMR+	ESFB-IPFE	✓	✓	×	×	×
SCMS-FL	LSH + Secure KNN	✓	✓	×	×	×
Ours	LGC	✓	✓	✓	✓	✓

6.2 Theoretical Analysis

To rigorously evaluate the efficiency of our proposed scheme, this section provides a theoretical comparison of its time complexity against current schemes (including SCMR, FECMR, and FECMR+). The analysis focuses on three core operational phases: index generation, trapdoor generation, and search. The complexity comparison is summarized in Table 2, followed by a detailed analysis. The notations used are defined as follows: n represents the number of documents; k is the dimension of the unified feature vector; d_t is the dimension of the original feature vector in SCMR; λ is the bit length of the security key in the Paillier cryptosystem used by SCMR; L and a are the height and the number of child nodes for each non-leaf node of the KM-tree in FECMR+, respectively; and $T_{Cluster}$ represents the computational cost of the K-modes clustering algorithm in FECMR+.

Table 2: Time complexity comparison.

Scheme	IndexBuild	TrapdoorBuild	Query
SCMR	$O(n \cdot (k + \lambda^3))$	$O(dt^3)$	$O(k \cdot dt \cdot \lambda^3)$
FECMR	$O(n^{2k})$	$O((k+n)k)$	$O(n(k+n))$
FECMR+	$2(a-1)(a^{L-1}-1)O(nk) + 2(L-1)T_{Cluster}$	$O((k+n)k)$	$O(a(L-1)(k+n))$
Ours	$O(nk)$	$O(k)$	$O(nk)$

IndexBuild: The computational bottleneck of SCMR lies in its use of Paillier homomorphic encryption, resulting in a complexity of $O(n \cdot (k + \lambda^3))$. The overhead of FECMR primarily stems from the matrix-vector multiplication in its ESFB-IPFE encryption, with a complexity of $O(n^{2k})$. FECMR+ introduces additional overhead by constructing a KM-tree via K-modes clustering and encrypting all node vectors. In contrast, our proposed scheme primarily relies on selection operations, achieving a lower complexity of $O(nk)$.

TrapdoorBuild: SCMR requires a computationally expensive matrix inversion, leading to a complexity of $O(d_t^3)$. Both FECMR and FECMR+ rely on matrix-vector multiplication, with a complexity of $O((k+n)k)$. Our scheme exhibits a significant advantage, requiring only selection operations with a linear complexity of $O(k)$.

Query: The search in SCMR involves dense homomorphic operations, primarily dominated by secure feature computations, resulting in a complexity of $O(k \cdot d_t \cdot \lambda^3)$. FECMR performs a linear scan involving n decryption operations, leading to a complexity of $O(n(k+n))$. FECMR+ improves upon this by searching a path within the KM-tree, reducing the cost to $O(a(L-1)(k+n))$. The search phase in our scheme is mainly composed of efficient XOR operations, with a complexity of $O(nk)$.

This analysis indicates that our scheme demonstrates theoretically superior efficiency by avoiding high-cost operations such as homomorphic encryption and matrix inversion. This advantage is particularly pronounced in the trapdoor generation phase.

6.3 Experimental Simulation

6.3.1 Environmental Settings

The simulation experiments in this paper were conducted on Ubuntu 22.04 and implemented using Python 3.8.20. The hardware environment consisted of an 11th Gen Intel(R) Core(TM) i7-11700F CPU @ 2.50 GHz, 32 GB of RAM, and a GeForce RTX 3090 GPU. The software environment included PyCharm 2024.3.1 and PyTorch.

6.3.2 Data Preparation

This experiment uses the NUS-WIDE dataset [31], which contains 269,648 real-world web images from Flickr, each associated with one or more text tags from 81 semantic concept categories. We selected 100,000 image-text pairs as the database entries and randomly chose 2100 image-text pairs to construct the query set, ensuring that the database and query sets are disjoint. The keys k_0 and k_1 in the index generation key $lkey$ are uint8 tensors of length kappa (32) bytes, and each byte has a value in the range of [0, 255]. The lengths of the multimodal data encryption key key, the multilevel hash function computation key key_1, and the authentication tag Tag generation key key_2 are all 16 bytes. The length of the $Nonce$ is 12 bytes. The experimental results are averaged over 100 local executions.

6.3.3 Time-Consuming

Index generation time: We evaluated the influence of CMH dimensionality on index construction time under a fixed database size of 100,000. As shown in Fig. 6a, the index generation time of all schemes increases approximately linearly as the CMH dimensionality grows. Among the compared schemes, the proposed scheme consistently achieves the lowest time overhead, reducing the index construction time by 99.92%–99.95% compared with SCMR and by 99.4%–99.93% compared with FECMR. Fig. 6b further compares the index construction cost when the database size increases from 20,000 to 100,000, with the CMH dimensionality fixed at 256. The results show that the index generation time of the proposed scheme remains below 1 s in all tested cases, even as the database size increases significantly. These observations are consistent with the theoretical complexity analysis and demonstrate that the proposed IndexBuild algorithm scales well with both CMH dimensionality and database size.

Trapdoor generation time: Fig. 7 illustrates the variation in individual trapdoor generation time as the CMH dimensionality increases. Across all tested dimensions, the proposed scheme consistently achieves the lowest overhead, reducing trapdoor generation time by 94.79%–95.34% compared with the existing schemes [16,20]. This observation is consistent with the theoretical complexity analysis and confirms the low computational overhead of the proposed TrapdoorBuild algorithm.

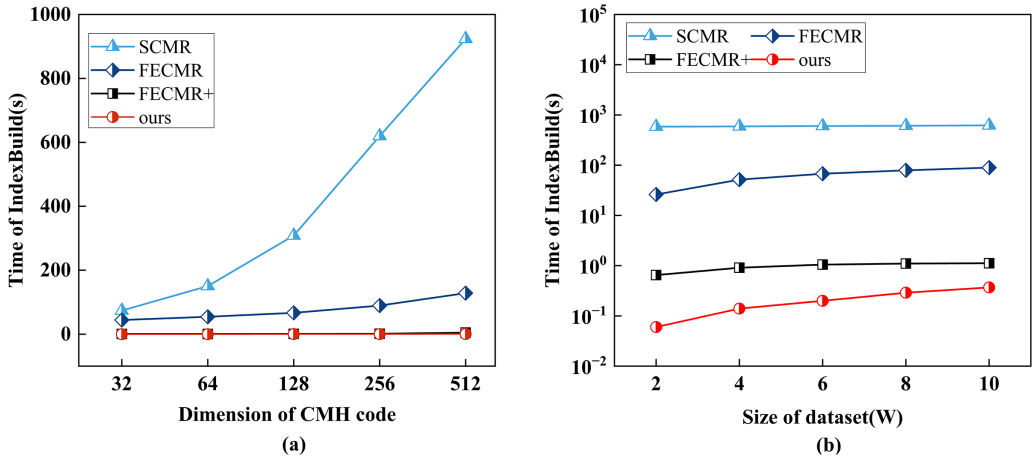

Figure 6: Index construction overhead. (**a**) Effect of dimensions on TrapdoorBuild, (**b**) effect of size on IndexBuild.

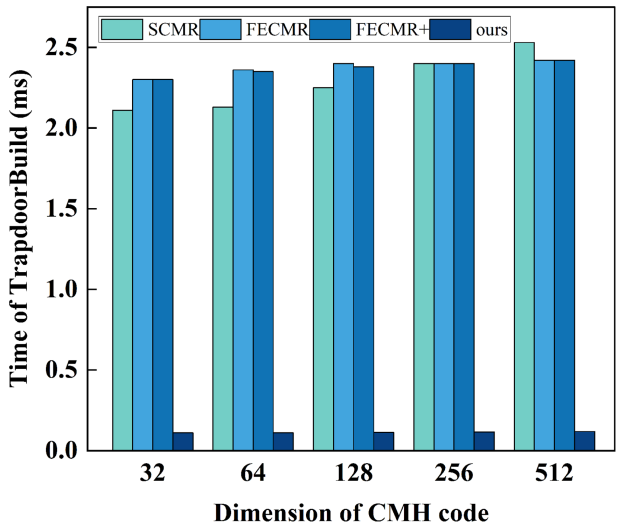

Figure 7: Effect of dimensions on TrapdoorBuild.

Search time: As shown in Fig. 8a, under a fixed index size of 100,000, the single query search latency of the proposed scheme remains below 1 ms when the CMH dimensionality does not exceed 512. Fig. 8b further reports the impact of index size on search time with the CMH dimensionality fixed at 256. The search time of most schemes increases approximately linearly with the number of indexes, whereas FECMR+ exhibits a sub-linear trend due to its optimized index structure. These results are consistent with the theoretical complexity analysis and indicate that the proposed search algorithm maintains practical efficiency in large-scale encrypted retrieval.

Verification time: The verification process consists of two stages: the generation of Tag_i and H_i, and the verification phase between the DU and the TTP. In the generation phase, the proposed scheme employs the Truncated Davies-Meyer construction instead of a conventional hash function, where the core computation is mainly based on lightweight XOR operations. Moreover, since the length of ID is fixed in the multi-level hash computation, the corresponding computational overhead remains limited. During the verification phase, the DU recomputes the corresponding tag Tag_i' for each returned result R_i and performs multi-level

hash computations over all associated H_i'. The TTP aggregates the multi-level hash values to obtain H_C, and then compares H' and the aggregated tag Tag' with H_C and the locally stored tag set Tag, respectively.

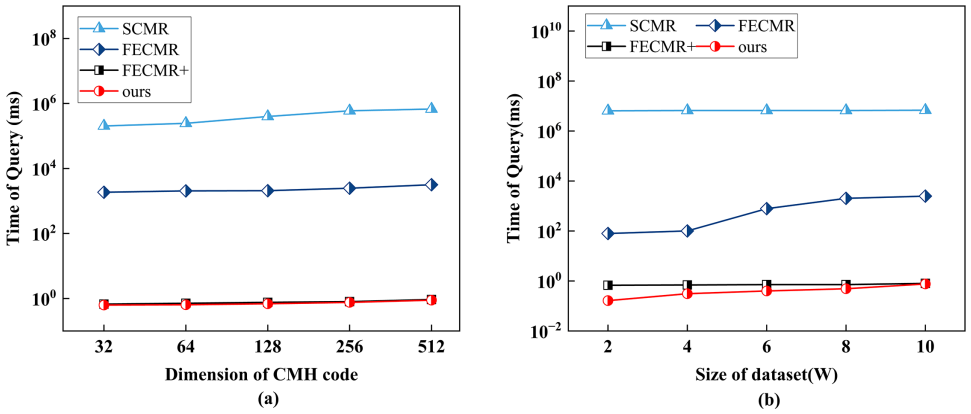

Figure 8: Search overhead. (**a**) Effect of dimensions on query, (**b**) effect of size on query.

Fig. 9 illustrates the relationship among the number of CMD, the size of CMD, and the tag generation time, where the CMD sizes are set to 100, 200, 300, 400, and 500 bytes, respectively. Since tag generation involves encrypted content processing, its overhead increases with both the size and the number of CMD. Fig. 10 further presents the relationship among the number of CMD, the size of CMD, and the verification time. The results indicate that the verification time is mainly affected by the number of CMD, while the influence of CMD size is primarily reflected in the recomputation of Tag_i'. In contrast, the remaining verification operations mainly process fixed-length hash values and tags, making them less sensitive to the size of the original multimodal data. These results indicate that the verification overhead is primarily determined by the number of returned results, whereas the effect of data size is mainly confined to content-related tag computation.

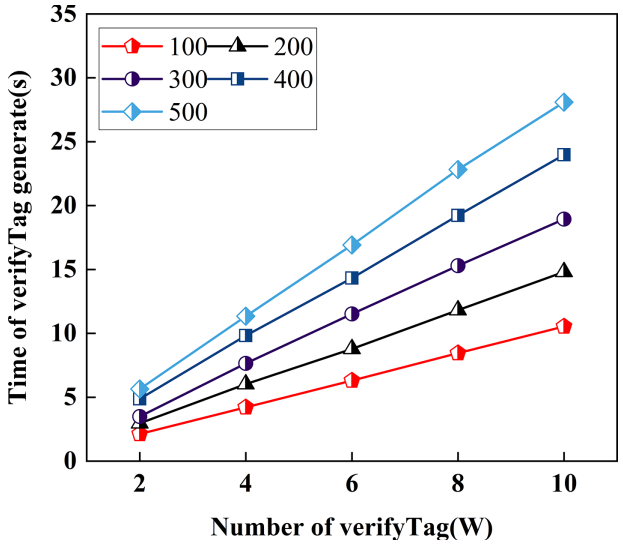

Figure 9: VerifyTag generate time.

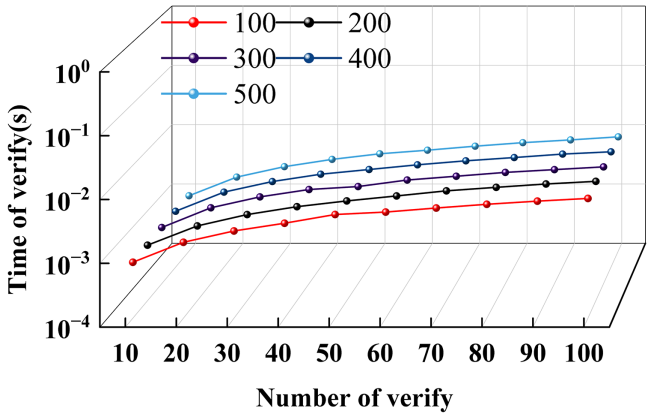

Figure 10: Verify time.

Search Effectiveness Evaluation: To evaluate the accuracy of our proposed scheme, we conducted two types of cross-modal retrieval tasks: using an image to retrieve relevant texts (Image → Text) and using a text to retrieve relevant images (Text → Image). A retrieval result is considered correct if there is at least one shared semantic category between the query and the retrieved item.

The evaluation metric used is the widely adopted Mean Average Precision (MAP), which is defined as the mean of the Average Precision (AP) across all queries. The AP is calculated according to Eq. (17):

$$AP = \frac{1}{R}\sum_{i=1}^{k} P(i)\cdot rel(i) \tag{17}$$

where R is the number of relevant samples, $P(i)$ is the precision at the i-th retrieved result, and $rel(i)$ equals 1 if the result is relevant to the query, and 0 otherwise.

As shown in Table 3, we compare the retrieval accuracy of the proposed method with FECMR, FECMR+, SCMR, and SCMS-FL under different CMH dimensionalities. Overall, the MAP values tend to increase as the CMH dimensionality grows, indicating that higher-dimensional hash codes can preserve richer semantic information within a certain range. The improvement is relatively significant when the dimensionality increases from 32 to 64 and from 64 to 128, whereas the gain becomes less pronounced from 128 to 256. This trend suggests that increasing the CMH dimensionality can improve retrieval accuracy, but the marginal benefit gradually decreases as the dimensionality becomes larger. Considering that higher-dimensional hash codes also introduce additional storage and computation costs, the 128-dimensional setting provides a favorable balance between retrieval accuracy and efficiency.

For the Text-to-Image task, although the proposed method is slightly lower than SCMR and SCMS-FL at 32 dimensions, it achieves clear advantages as the dimensionality increases. Specifically, the MAP of the proposed method reaches 81.77% at 64 dimensions, exceeding SCMR by 3.47 percentage points. At 128 dimensions, the proposed method achieves 84.10% MAP, outperforming SCMS-FL by 5.90 percentage points. At 256 dimensions, it reaches 85.25% MAP, exceeding FECMR+ by 5.60 percentage points. For the Image-to-Text task, the proposed method consistently achieves the best performance across all tested dimensionalities. Compared with the best-performing comparison schemes under the same dimensionalities, the MAP improvements are 7.58, 11.40, 12.80, and 18.21 percentage points at 32, 64, 128, and 256 dimensions, respectively. These results indicate that the proposed residual-alignment CMH generation method provides stable retrieval accuracy across both retrieval directions.

Table 3: MAP comparison of different schemes under varying dimensions.

MAP	Text-to-Image (T-I)				Image-to-Text (I-T)			
Dim	**32**	**64**	**128**	**256**	**32**	**64**	**128**	**256**
FECMR	66.9	68.9	69.9	79.61	65.6	67.9	68.0	70.01
FECMR+	N/A	N/A	N/A	79.65	N/A	N/A	N/A	70.31
SCMR	76.4	78.3	N/A	N/A	72.4	73.3	N/A	N/A
SCMS-FL	77.6	78.0	78.2	N/A	73.1	73.9	74.8	N/A
Ours	76.07	81.77	84.1	85.25	80.68	85.3	87.6	88.52

Note: N/A indicates that the corresponding data were not available or not reported in the compared references.

7 Conclusion

In this paper, we propose a cross-modal searchable encryption scheme with verification functionality. Experimental results demonstrate that the proposed method improves MAP by up to 5.90 percentage points compared with existing schemes, while reducing trapdoor-generation time by 94.79%–95.34%. To ensure the security and trustworthiness of the results returned from the cloud, we design a triple-verification mechanism covering integrity, correctness, and comprehensiveness. This mechanism guarantees that the retrieved data received by data users is unaltered and includes all relevant entries, thereby enhancing the overall reliability of the system. In future work, we plan to extend the scheme to support more complex multimedia forms, such as audio and video, enhance the protection of access patterns and search patterns, and enable dynamic cross-modal searchable encryption while ensuring both forward and backward security.

Acknowledgement: Not applicable.

Funding Statement: This work was supported in part by the National Natural Science Foundation of China under Grant 62262073; Yunnan Provincial Applied Basic Research Program under Grant 202101AT070098; in part by the Yunnan Provincial Ten Thousand People Program for Young Top Talents under Grant YNWR-QNBJ-2019-237; and in part by the Yunnan Provincial Major Science and Technology Special Program under Grant 202402AD080002.

Author Contributions: Peixuan Wang conceptualized the research, designed the methodology, and wrote the main manuscript text. Lingyun Yuan supervised the project, acquired funding, and contributed to the writing, review, editing, and validation. Tianyu Xie contributed to the conceptualization, methodology, and formal analysis. Yi Xiang contributed to the visualization, validation, and formal analysis. Haochen Bao contributed to the validation and formal analysis. Kexin Wang contributed to the data curation and formal analysis. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The datasets used in this study are available at the following website: https://huggingface.co/datasets/Lxyhaha/NUS-WIDE. Experimental data will be shared upon reasonable request.

Ethics Approval: Not applicable.

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

References

1. Li F, Ma J, Miao Y, Liu X, Ning J, Deng RH. A survey on searchable symmetric encryption. ACM Comput Surv. 2024;56(5):1–42. doi:10.1145/3617991.
2. Jumaa SS, Challoob MH, Humaidi AJ. Multilevel military image encryption based on tri-independent keying approach. Comput Mater Contin. 2026;87(1):1–10. doi:10.32604/cmc.2025.074752.

3. Jirjees SW, Alkhalid FF, Hasan AM, Humaidi AJ. A secure password-based authentication with variable key lengths based on the image-embedded method. Mesopotamian J Cybersecur. 2025;5(2):491–500.
4. Song DX, Wagner D, Perrig A. Practical techniques for searches on encrypted data. In: Proceedings of the 2000 IEEE Symposium on Security and Privacy (S&P 2000); 2000 May 14–17; Berkeley, CA, USA. p. 44–55.
5. Ji L, Li J, Zhang Y, Lu Y. Verifiable searchable symmetric encryption over additive homomorphism. IEEE Trans Inform Forensic Secur. 2025;20:1320–32. doi:10.1109/tifs.2025.3526062.
6. Zhang K, Hu B, Ning J, Gong J, Qian H. Pattern hiding and authorized searchable encryption for data sharing in cloud storage. IEEE Trans Knowl Data Eng. 2025;37(5):2802–15. doi:10.1109/tkde.2025.3537613.
7. Jiang J, Wang D. QPASE: quantum-resistant password-authenticated searchable encryption for cloud storage. IEEE Trans Inform Forensic Secur. 2024;19:4231–46. doi:10.1109/tifs.2024.3372804.
8. Yang Y, Hu Y, Li R, Dong X, Cao Z, Shen J, et al. LSE: efficient symmetric searchable encryption based on labeled PSI. IEEE Trans Serv Comput. 2024;17(2):563–74. doi:10.1109/tsc.2024.3356728.
9. Yang N, Tang C, Zhou Q, He D. Dynamic consensus committee-based for secure data sharing with authorized multi-receiver searchable encryption. IEEE Trans Inform Forensic Secur. 2023;18:5186–99. doi:10.1109/tifs.2023.3305183.
10. Cheng L, Meng F. Server-aided public key authenticated searchable encryption with constant ciphertext and constant trapdoor. IEEE Trans Inform Forensic Secur. 2024;19:1388–400. doi:10.1109/tifs.2023.3336160.
11. Xie T, Yuan L, Zhang Q, Wu J, Ren F. Ciphertext fuzzy retrieval mechanism with bidirectional verification and privacy protection. IEEE Internet Things J. 2024;11(24):41061–83. doi:10.1109/jiot.2024.3458457.
12. Bao H, Yuan L, Xie T, Chen H, Dai H. A blockchain-based efficient verification scheme for context semantic-aware ciphertext retrieval. Comput Mater Contin. 2026;86(1):1–30. doi:10.32604/cmc.2025.069240.
13. Liu P, He Q, Zhao B, Guo B, Zhai Z. Efficient multi-authority attribute-based searchable encryption scheme with blockchain assistance for cloud-edge coordination. Comput Mater Contin. 2023;76(3):3325–43. doi:10.32604/cmc.2023.041167.
14. Cao Y, Zhang H, Shang X. A privacy-preserving cross-modal retrieval scheme based on CLIP and deep hashing. In: Proceedings of the ICASSP 2025—2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2025 Apr 6–11; Hyderabad, India. p. 1–5. doi:10.1109/icassp49660.2025.10890094.
15. Chen J, Yan W, Qin W, Ni Z. Based on inner product function encryption secure cross-modal retrieval. In: Proceedings of the 2024 5th International Conference on Machine Learning and Computer Application (ICMLCA); 2024 Oct 18–20; Hangzhou, China. p. 580–3. doi:10.1109/icmlca63499.2024.10754561.
16. Guo C, Jia J, Jie Y, Liu CZ, Choo KR. Enabling secure cross-modal retrieval over encrypted heterogeneous IoT databases with collective matrix factorization. IEEE Internet Things J. 2020;7(4):3104–13. doi:10.1109/jiot.2020.2964412.
17. Wang X, Li J, Liu Z, Tang Q, Wang X. Enabling secure cross-modal search over encrypted data via federated learning. IEEE Internet Things J. 2025;12(2):1933–45. doi:10.1109/jiot.2024.3464760.
18. Hu S, Zhang LY, Wang Q, Qin Z, Wang C. Towards private and scalable cross-media retrieval. IEEE Trans Dependable Secure Comput. 2021;18(3):1354–68. doi:10.1109/tdsc.2019.2926968.
19. Li M, Zhu Y, Du R, Jia C. LP^{2}CR-IoT: lightweight and privacy-preserving cross-modal retrieval in IoT. IEEE Internet Things J. 2025;12(10):14812–27. doi:10.1109/jiot.2025.3526939.
20. Yang L, Zhang W, Miao Y, Liang Y, Li X, Choo KR, et al. Secure and efficient cross-modal retrieval over encrypted multimodal data. IEEE Trans Comput. 2025;74(4):1405–17. doi:10.1109/tc.2025.3525614.
21. Wang T, Li F, Zhu L, Li J, Zhang Z, Shen HT. Cross-modal retrieval: a systematic review of methods and future directions. Proc IEEE. 2024;112(11):1716–54. doi:10.1109/jproc.2024.3525147.
22. Zheng Y, Lu R, Zhang S, Shao J, Zhu H. Achieving practical and privacy-preserving kNN query over encrypted data. IEEE Trans Dependable Secure Comput. 2024;21(6):5479–92. doi:10.1109/tdsc.2024.3376084.
23. Marcolla C, Sucasas V, Manzano M, Bassoli R, Fitzek FHP, Aaraj N. Survey on fully homomorphic encryption, theory, and applications. Proc IEEE. 2022;110(10):1572–609. doi:10.1109/jproc.2022.3205665.

24. Miao Y, Tong Q, Deng RH, Choo KR, Liu X, Li H. Verifiable searchable encryption framework against insider keyword-guessing attack in cloud storage. IEEE Trans Cloud Comput. 2022;10(2):835–48. doi:10.1109/tcc.2020.2989296.
25. Shi Z, Fu X, Li X, Zhu K. ESVSSE: enabling efficient, secure, verifiable searchable symmetric encryption. IEEE Trans Knowl Data Eng. 2022;34(7):3241–54. doi:10.1109/tkde.2020.3025348.
26. Zhang Z, Wang J, Wang Y, Su Y, Chen X. Towards efficient verifiable forward secure searchable symmetric encryption. In: Computer Security—ESORICS 2019. Cham, Switzerland: Springer; 2019. p. 304–21. doi:10.1007/978-3-030-29962-0_15.
27. Liu X, Yang X, Luo Y, Zhang Q. Verifiable multikeyword search encryption scheme with anonymous key generation for medical Internet of Things. IEEE Internet Things J. 2022;9(22):22315–26. doi:10.1109/jiot.2021.3056116.
28. Li X, Tong Q, Zhao J, Miao Y, Ma S, Weng J, et al. VRFMS: verifiable ranked fuzzy multi-keyword search over encrypted data. IEEE Trans Serv Comput. 2023;16(1):698–710. doi:10.1109/tsc.2021.3140092.
29. Bellare M, Hoang VT. Efficient schemes for committing authenticated encryption. In: Advances in Cryptology—EUROCRYPT 2022. Cham, Switzerland: Springer; 2022. p. 845–75. doi:10.1007/978-3-031-07085-3_29.
30. Clarke D, Devadas S, van Dijk M, Gassend B, Suh GE. Incremental multiset hash functions and their application to memory integrity checking. In: Advances in Cryptology—ASIACRYPT 2003. Berlin/Heidelberg, Germany: Springer; 2003. p. 188–207. doi:10.1007/978-3-540-40061-5_12.
31. Rasiwasia N, Costa Pereira J, Coviello E, Doyle G, Lanckriet GRG, Levy R, et al. A new approach to cross-modal multimedia retrieval. In: Proceedings of the 18th ACM International Conference on Multimedia; 2010 Oct 25–29; Firenze, Italy. p. 251–60. doi:10.1145/1873951.1873987.