Hindawi
Complexity
Volume 2021, Article ID 7937922, 7 pages
https://doi.org/10.1155/2021/7937922

Research Article

Cooperative Cloud-Edge Feature Extraction Architecture for Mobile Image Retrieval

Chao He[1] and Gang Ma[2]

[1] *State Key Laboratory of Network and Switching Technology Institute, Beijing University of Posts and Telecommunications, Beijing, China*
[2] *School of Mathematics and Statistics, Shandong University of Technology, Zibo, Shandong, China*

Correspondence should be addressed to Chao He; chaohe@bupt.edu.cn

Received 8 September 2021; Revised 14 October 2021; Accepted 18 October 2021; Published 1 November 2021

Academic Editor: Xuyun Zhang

Mobile image retrieval greatly facilitates our lives and works by providing various retrieval services. The existing mobile image retrieval scheme is based on mobile cloud-edge computing architecture. That is, user equipment captures images and uploads the captured image data to the edge server. After preprocessing these captured image data and extracting features from these image data, the edge server uploads the extracted features to the cloud server. However, the feature extraction on the cloud server is noncooperative with the feature extraction on the edge server which cannot extract features effectively and has a lower image retrieval accuracy. For this, we propose a collaborative cloud-edge feature extraction architecture for mobile image retrieval. The cloud server generates the projection matrix from the image data set with a feature extraction algorithm, and the edge server extracts the feature from the uploaded image with the projection matrix. That is, the cloud server guides the edge server to perform feature extraction. This architecture can effectively extract the image data on the edge server, reduce network load, and save bandwidth. The experimental results indicate that this scheme can upload few features to get high retrieval accuracy and reduce the feature matching time by about 69.5% with similar retrieval accuracy.

1. Introduction

Mobile image retrieval plays an important role in processes such as identification of crop diseases and insect pests, the protection of pedestrians in autonomous vehicles, suspect identification, and medical services [1–6]. It has penetrated into all aspects of people's lives. Feature extraction and feature matching are two important factors in image retrieval tasks. Feature matching is the most time-consuming, and feature extraction affects the matching time and retrieval results. As a result of the limited computing and storage resources of user equipment, many mobile image retrieval tasks are based on mobile cloud computing architecture [7–11]. For instance, Shelly et al. [12] proposed a cloud computing-based iris retrieval solution based on the Hadoop framework and proved that the use of cloud servers can effectively accelerate retrieval tasks. Hassan et al. [13] proposed a face retrieval method based on mobile cloud computing architecture. In this framework, the mobile device performs a lightweight task and the cloud server runs the computationally intensive tasks. In these solutions, the user equipment performs a lightweight task and the cloud server runs the computationally intensive tasks with powerful computing and storage resources. The above methods all use the powerful computing and storage resources of cloud servers. That is, mobile users first use their user equipment to capture images and then upload the captured image data to the cloud server for further processing. After obtaining the uploaded image data, the cloud server processes it and gets the result and sends it to the mobile users. To reduce the network traffic, many studies [14, 15] have also preprocessed these captured image data, and even extracted features from these image data, and only uploaded the extracted features to the cloud server. Kan et al. [14]

presented an automatic classification method for medicinal plant leaves instead of manual classification, which can greatly improve the accuracy and speed of retrieval. This method preprocesses the image of the leaves of medicinal plants, extracts the shape and texture features, and then classifies the leaves of the medicinal plants through a support vector machine (SVM) classifier. Mannan et al. [15] proposed an enhanced cloud-based biometric identification method for identifying individuals on campus. By using the computing and storage resources of the cloud server, the system's computing and storage burdens are reduced. This method uses the local binary pattern (LBP) algorithm to extract features of encrypted images to protect personal privacy. At the same time, it uses the principal component analysis algorithm to reduce transmission delay and calculation time.

Mobile image retrieval greatly facilitates our lives by providing various retrieval services. However, as a result of the long distance between mobile users and the cloud server, it is difficult to provide a fast response to massive mobile image retrieval tasks, and with limited coverage and spectrum resources, mobile users are usually far away from cloud servers, leading to network delays and interruptions, which will bring a poor user experience. Since fast response is a great demand for mobile image retrieval tasks, a new computing paradigm multiaccess edge computing (MEC) [16–20] architecture has emerged as a promising solution to address the long response time of mobile cloud architecture by providing computing and storage resources at the edge of the network. In the MEC environment, feature extraction is mainly performed on edge servers and the time-consuming operation is performed on the cloud server. In recent years, many related studies have been proposed [21, 22]. For instance, Soyata et al. [21] presented a hybrid mobile-cloudlet-cloud computing architecture that uses the cloudlet as a lightweight server and applies an optimal task-partitioning method for distributing computing load among cloud servers. Hu et al. [22] proposed a face retrieval framework based on fog computing architecture. In the proposed framework, fog nodes perform face detection, image preprocessing, feature extraction, and face identifier from the raw image transmitted by the client with a local binary pattern algorithm. Then, the cloud server performs face matching and identity information acquisition after receiving the identifier from fog nodes. Results show that the proposed framework can reduce bandwidth consumption and response time.

However, in most of the existing methods, the feature extraction on the edge server is separated from the cloud servers. This leads to the lack of effectiveness of the extracted features, which affects the performance of image retrieval tasks. Sharma et al. [23] presented a coordinated architecture for edge and cloud computing that can analyse big data effectively in the Internet of Things networks. The key point is that the cloud server utilizes the network knowledge and historical information to guide the edge server to provide various customized services. To this end, we aim to study mobile image retrieval in the MEC environment. And, we propose a cloud-edge collaboration feature extraction solution for mobile image retrieval in MEC. In the proposed framework, the effective features are extracted through the collaboration of the cloud server and edge servers, rather than through the cloud server or the edge servers alone. Meanwhile, we store the extracted features and results on the edge server to respond to the same image retrieval service faster.

The rest of this paper is organized as follows: Section 2 presents the system framework. Section 3 presents the experimental results and analysis. Finally, we conclude this paper in Section 4.

2. Design of the Proposed Architecture

2.1. Problem Statement. In this paper, we study the problem of image retrieval in the MEC environment, which is described as follows: As illustrated in Figure 1, the system architecture of MEC consists of three layers of components: user equipment, edge servers, and cloud servers. User equipment communicates with edge servers through a network gateway, and the edge servers connect to cloud servers via the Internet backbone. A large amount of labelled image data is stored on cloud servers. Mobile users first use their user equipment to capture images and preprocess them and then upload the preprocessed image data to edge servers. After receiving the preprocessed images, the edge servers use a feature extraction algorithm to extract effective features, store the features, and then upload the extracted features to the cloud server for further processing. After receiving the extracted features, the cloud server processes them and gets the results and finally returns the results to the edge servers and user equipment. The symbols used in this paper are summarized in Table 1.

2.2. Detailed Design of the Proposed Architecture. The architecture of the proposed framework consists of three layers of components: user equipment, edge servers, and cloud servers. User equipment refers to some devices with limited computing and storage resources, such as smartphones, laptops, and Apple watches with a mobile broadband adapter. Edge servers are usually a group of servers that are deployed at the edge of the network, such as microservers. Cloud servers are usually Alicloud servers, Amazon Web Service (AWS) Cloud servers, and Microsoft Azure Cloud servers.

In order to verify the feasibility and effectiveness of the architecture, we implemented a prototype system that uses a Karhunen–Loève transform (KLT) [24, 25] algorithm for feature extraction. In practical scenarios, our framework can flexibly select algorithms according to application scenarios, including but not limited to KLT algorithms. The proposed framework can be divided into offline stages and online stages. In the offline stage, we get the projection matrix A with a KLT algorithm from the image data set on the cloud server. We assume that x_i, $i = 1, \ldots, m$, is the i-th image in the image data set. The number of features of the image is d, and the number of class labels is K. The calculation process of KLT is as follows:

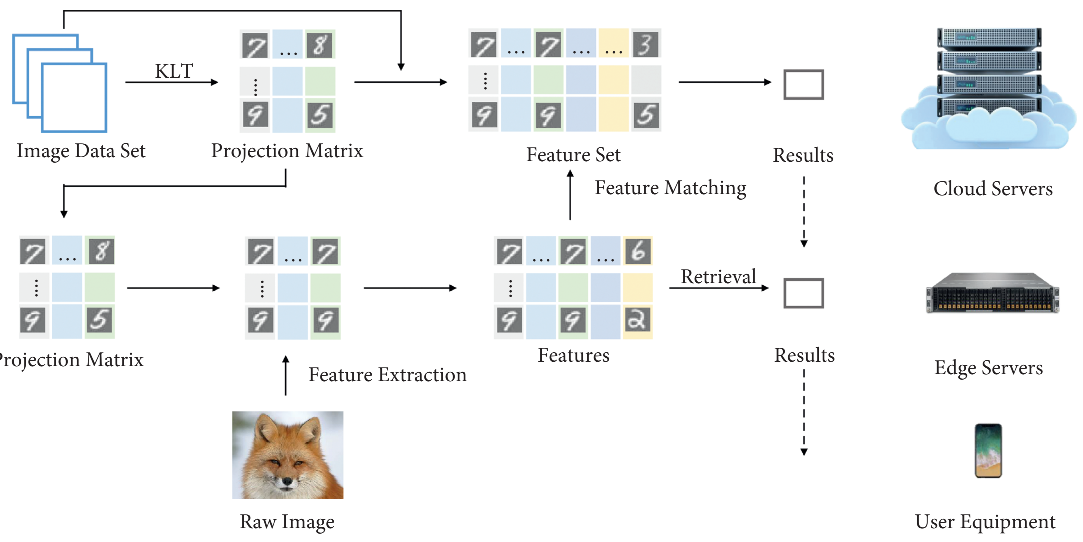


Figure 1: The detailed design of the proposed framework.

Table 1: Frequently used symbols.

Symbols	Descriptions
$\mathbf{X}$	The pixel matrix of an image data set
x_i	The i-th image in the image data set
μ	The mean of the image
$x_i(j)$	The j-th feature of the i-th image
σ_j	The standard deviation of the j-th feature
C	The covariance matrix for the features in the image data set
A	The projection matrix
$\mathrm{d}(x_i, v_i)$	The Euclidean distance between two vectors x_i and v_i
R	The accuracy rate of the feature matching
$\varphi(a_i, b_i)$	Step function, equals 1 if $a_i = b_i$ and equals 0 otherwise
K	K-nearest neighbor

(1) Standardize the data set on the cloud server. Most machine learning and optimization algorithms perform better when all the features are along the same scale. To do this, a standardization approach can be implemented. Sample x_i can become the standardized feature by using the following calculation:

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i, \tag{1}$$

where μ is the mean of the sample

$$x_i = x_i - \mu. \tag{2}$$

Then, the sample x_i has zero mean:

$$\sigma_j = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(x_i(j)\right)^2}, \tag{3}$$

where σ_j is the standard deviation of the corresponding feature, $x_i(j)$, $j = 1, \ldots, d$, is the j-th feature, and finally, the standardized feature $x_i(j)$ is as follows:

$$x_i(j) = \frac{x_i(j)}{\sigma_j}. \tag{4}$$

(2) Calculate the covariance matrix for the features in the data set. The covariance matrix C is as follows:

$$\begin{aligned} C &= \frac{1}{m}\sum_{i=1}^{m} x_i (x_i)^T, \\ &= \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,m} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m,1} & c_{m,2} & \cdots & c_{m,m} \end{pmatrix}, \\ &= \begin{pmatrix} \sigma_1^2 & c_{1,2} & \cdots & c_{1,m} \\ c_{2,1} & \sigma_2^2 & \cdots & c_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m,1} & c_{m,2} & \cdots & \sigma_m^2 \end{pmatrix}. \end{aligned} \tag{5}$$

(3) Calculate the eigenvalues and eigenvectors for the covariance matrix.

(4) Sort eigenvalues and their corresponding eigenvectors.

(5) Form the projection matrix A by selecting the corresponding eigenvectors of the t largest eigenvalues.

$$A = [p_1, \ldots, p_t]. \tag{6}$$

The feature set on the cloud server can also be calculated by $X \cdot A$. Then, the projection matrix A is transmitted to the edge server for feature extraction from the raw image uploaded by the user equipment. Matrix A gets the whole information of the image data set on the cloud server; therefore, the feature extraction from the raw image on the edge server is effective and has fewer features.

We use the K-nearest neighbor (KNN) [26, 27] algorithm to match features. KNN assumes the similarity between the new sample and available cases and puts the new case into the category that is most similar to the available categories. The processing flow of KNN is that we first choose the number of K, where K represents the number of neighbors. Then, we measure the distance of the K-nearest neighbors of the test data. Followed by that, we count the number of neighbors of each category. Finally, we assign the test data to the category with the most neighbors. Note that we use the most used Euclidean distance to calculate the distance between samples. Given two samples x_i and v_i, their Euclidean distance can be written as

$$\mathrm{d}\left(x_i, v_i\right) = \sqrt{\sum_{j=1}^{d}\left(x_{ij} - v_{ij}\right)^2}. \tag{7}$$

We also store the results of feature extraction and feature matching for saving computing resources when there are the same image retrieval requests. We assume that for the image x_i, a_i and b_i are the results of the feature matching. To quantify the performance of retrieval, the accuracy rate is defined as follows:

$$\mathrm{R} = \frac{1}{m}\sum_{\mathrm{i}}^{m}\varphi\left(a_i, b_i\right), \tag{8}$$

where $\varphi(a_i, b_i)$ equals 1 if $a_i = b_i$ and equals 0 otherwise. In the online stage, mobile users use their user equipment to capture images and upload the captured image data to the edge server. After receiving the image data, the edge server first performs object detection, using object segmentation algorithms to preprocess it. Then, the edge server uses the projection Matrix A to extract features from the preprocessed image data and uploads the extracted feature data to the cloud server for feature matching. Note that KNN is used to test the effectiveness of the extracted features. For convenience, we set K = 1.

The detailed cooperative cloud-edge process is given in Algorithm 1.

The novelty of our scheme is that the proposed framework uses the collaboration between the cloud servers and edge servers to extract efficient features to reduce feature matching time and get similar retrieval accuracy with fewer features, thus improving the user experience of mobile image retrieval applications.

3. Experiment

In this section, we use six data sets: ORL, YALE, UMIST, MNIST, COIL20, and LEAVES [28, 29] to verify our proposed architecture. Table 2 lists the details used in the experiment, and all the data sets are divided randomly into the image data set on the cloud server and multiple raw images on the edge server uploaded by user equipment mentioned in Section 2. Hence, the raw image is the image that has been preprocessed in our experiment. The MNIST data set is rescaled to 28×28 pixels, and the other image data sets are rescaled to 32×32 pixels.

We evaluate the proposed framework in terms of accuracy and feature matching time. Figure 2 shows the accuracy of the raw image and our method using different training samples. In Figure 2(a), in the beginning, the accuracy of using the original image method is higher than using our method. When the number of training images reaches 280, our method begins to outperform the original image method. At first, in Figure 2(b), the accuracy of our proposed method is not as good as the original image method, but when the number of training images increases to 83, the accuracy of the original image method starts to decline relative to our method. In Figure 2(c), the accuracy changes of the original image method and our method are similar to those of Figure 2(a) and Figure 2(c), but the accuracy of the two is more similar. Then, in Figure 2(d) and Figure 2(e), the accuracy of the original image method is better than of our method and the accuracy difference between the two is even higher than that of other data sets. In Figure 2(f), the accuracy of our method has always been better than that of the original image method, and after the number of training images is set to greater than 130, the accuracy of our method improves faster. In most cases, the accuracy of our method is similar to that of using raw images. Note that our method uses only a few features to participate in the retrieval tasks.

Our method achieves accuracy similar to that of using all features, which proves the effectiveness of the features extracted by our method. In addition, with the increase in training images, the accuracy of directly using the raw images and our method has increased, which proves the importance of collecting enough training images.

Our method has the least feature matching time. Figure 3 shows that the matching time of raw images and the matching time of the extracted features have similar accuracy. We observe that compared to using raw images, using the extracted features can save a lot of matching time. For instance, compared with using raw images, on the ORL data set, our method reduces the feature matching time by about 69.5% in the case of similar retrieval accuracy. This is because our method extracts a small number of features. Therefore, with the same number of images, fewer features result in less feature matching time. This also shows that using our method can save a lot of feature matching time. Moreover, on the COIL20 data set, the matching time between our method and the original image method has the largest change, and on the YALE data set, we have the smallest change in matching time between the original image method and our method. The size of matching time changes of other data sets is in the order of MNIST data set, UMIST data set, ORL data set, and LEAVES data set.

```
(i) Input: image data set, raw image
(ii) Output: result of image retrieval
(iii) Cloud gets the projection Matrix A with a KLT algorithm for the image data set storing on the cloud server
(iv) Cloud sends the projection Matrix A to the edge server
(v) Edge server gets the raw image uploading from the user equipment
(vi) Edge server preprocesses the raw image and gets the pixel Matrix $\mathbf{X}$, then extracts features by $\mathbf{X} \cdot \mathbf{A}$
(vii) Edge server sends the features of raw image to the cloud server for feature matching;
(viii) Cloud gets the result of image retrieval with a KNN algorithm
(ix) Cloud sends the result to the edge server, and the user equipment gets the image retrieval result sending by the edge server
```

ALGORITHM 1: Cooperative cloud-edge process.

TABLE 2: Description of benchmark data sets.

Data sets	Images	Dimensions	Classes
ORL	400	1024	40
YALE	165	1024	15
UMIST	564	1024	20
MNIST	70000	256	10
COIL20	1440	1024	20
LEAVES	186	1024	3

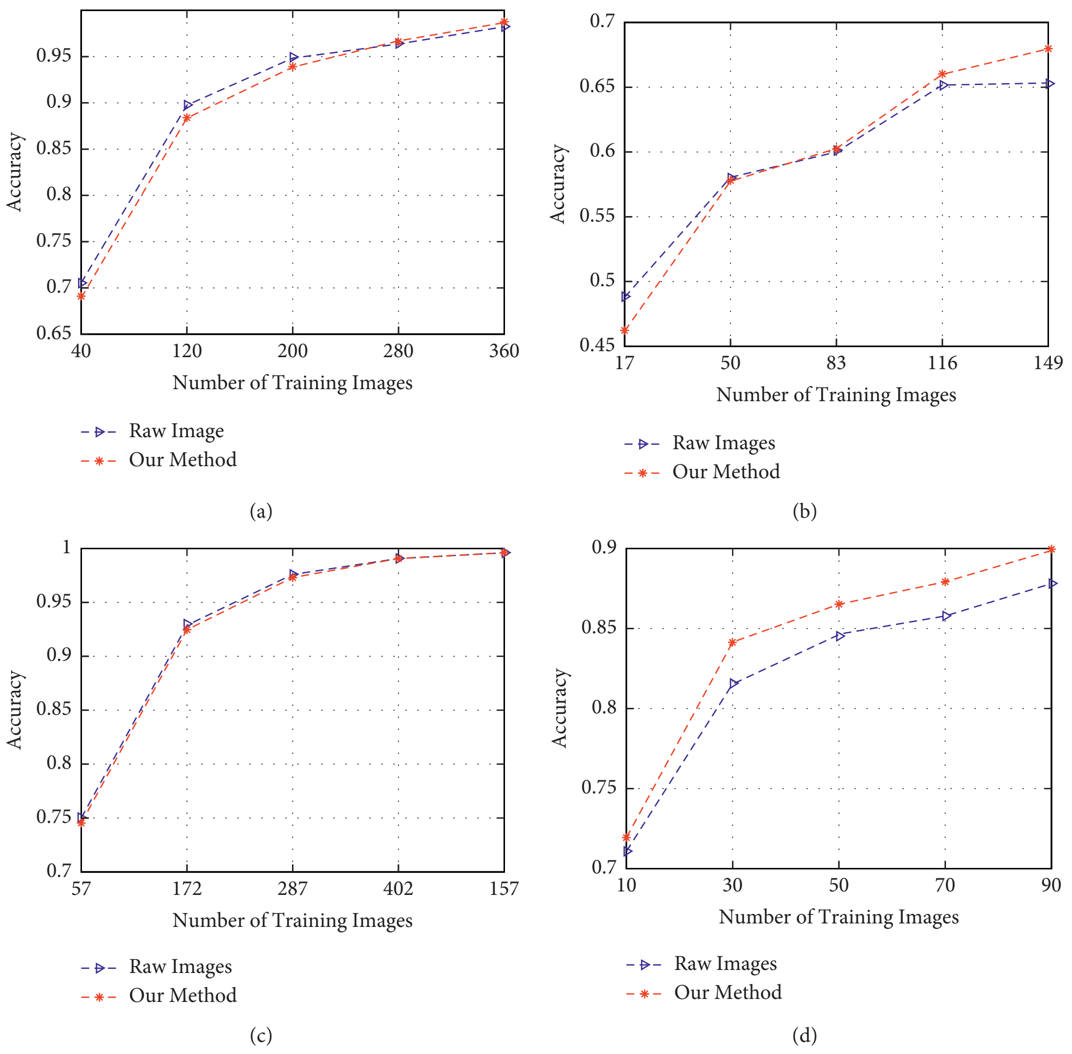
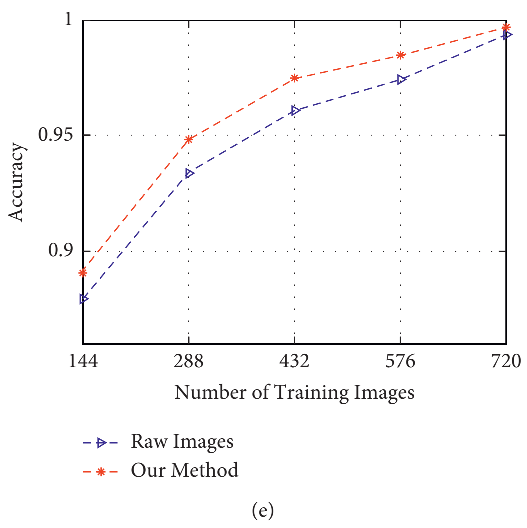


FIGURE 2: Continued.



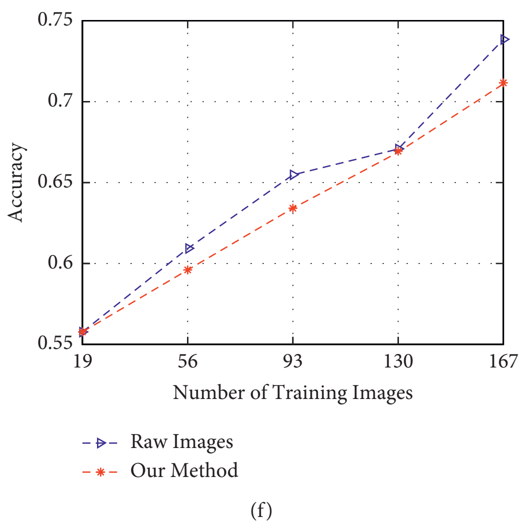


FIGURE 2: Accuracy vs number of training images on (a) ORL, (b) YALE, (c) UMIST, (d) MNIST, (e) COIL20, and (f) LEAVES data sets.

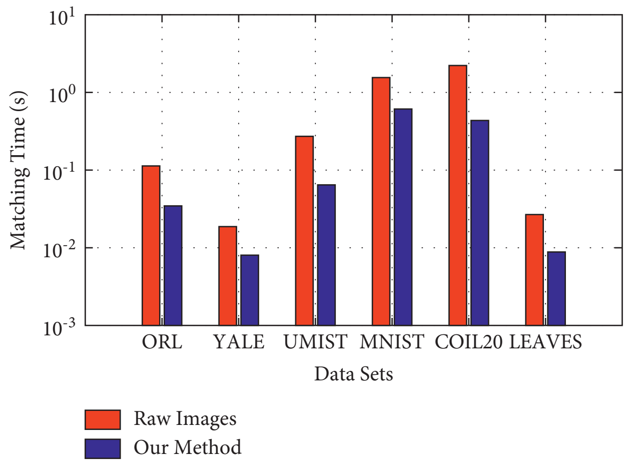


FIGURE 3: Matching time of the raw image vs matching time of our method.

4. Conclusions

In this article, the advantages of our architecture are the collaboration between cloud servers and edge servers. The cloud server performs the computing-intensive part of the image retrieval task, that is, projection matrix generation and feature matching. The edge server performs the lightweight part of the task, i.e., extracting features from the raw image with the projection matrix. Although the edge server can also perform feature extraction on the original image uploaded by user equipment, using our method can make the feature extraction of the original image more effective. The experiment also proved this, and the experiment shows that feature extraction is more effective as the number of images in the data set increases. Therefore, the projection matrix generated from the entire data set on the cloud server can better guide the original image feature extraction work on the edge server and make feature extraction more effective. Also, the accuracy of image retrieval and the matching time verify the effectiveness of our proposed architecture. Moreover, our proposed framework uses the KLT algorithm to extract features. In future work, the efficiency of the adopted algorithm for different data sets in different scenarios might prove important. This is an issue for future research to explore.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank professorate senior engineer Hui Xu and Dr. Zhimi Cheng for their valuable comments on the paper. This research and publication were supported by Shandong Provincial Natural Science Foundation of China under Grant no. ZR2019MA012.

References

[1] X. Sun, S. Mu, Y. Xu, Z. Cao, and T. Su, "Image retrieval of tea leaf diseases based on convolutional neural network," in *Proceedings of the 2018 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, Jinan, China, December 2018.

[2] H. Wang, B. Wang, B. Liu, X. Meng, and G. Yang, "Pedestrian recognition and tracking using 3D LiDAR for autonomous vehicle," *Robotics and Autonomous Systems*, vol. 88, pp. 71–78, 2017.

[3] W.-J. Chang, L.-B. Chen, C.-H. Hsu, C.-P. Lin, and T.-C. Yang, "A deep learning-based intelligent medicine retrieval system for chronic patients," *IEEE Access*, vol. 7, pp. 44 441–444 458, 2019.

[4] V. D. A. Kumar, V. D. A. Kumar, S. Malathi, K. Vengatesan, and M. Ramakrishnan, "Facial recognition system for suspect identification using a surveillance camera," *Pattern Recognition and Image Analysis*, vol. 28, no. 3, pp. 410–420, 2018.

[5] K. K. Singh and A. Singh, "Diagnosis of COVID-19 from chest X-ray images using wavelets-based depthwise convolution network," *Big Data Mining and Analytics*, vol. 4, no. 2, pp. 84–93, 2021.

[6] J. Mabrouki, M. Azrour, D. Dhiba, Y. Farhaoui, and S. E. Hajjaji, "IoT-based data logger for weather monitoring using arduino-based wireless sensor networks with remote graphical application and alerts," *Big Data Mining and Analytics*, vol. 4, no. 1, pp. 25–32, 2021.

[7] J. Yang, N. Xiong, A. V. Vasilakos et al., "A fingerprint recognition scheme based on assembling invariant moments for cloud computing communications," *IEEE Systems Journal*, vol. 5, no. 4, pp. 574–583, 2011.

[8] P. Duan, W. Wang, W. Zhang, F. Gong, P. Zhang, and Y. Rao, "Food image retrieval using pervasive cloud computing," in *Proceedings of the 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing. IEEE*, pp. 1631–1637, Beijing, China, December 2013.

[9] W. Zhang, X. Chen, and J. Jiang, "A multi-objective optimization method of initial virtual machine fault-tolerant placement for star topological data centers of cloud systems," *Tsinghua Science and Technology*, vol. 26, no. 1, pp. 95–111, 2021.

[10] D. Kim, J. Son, D. Seo, Y. Kim, H. Kim, and J. T. Seo, "A novel transparent and auditable fog-assisted cloud storage with compensation mechanism," *Tsinghua Science and Technology*, vol. 25, no. 1, pp. 28–43, 2020.

[11] X. Tan, J. Zhang, Y. Zhang, Z. Qin, Y. Ding, and X. Wang, "A PUF-based and cloud-assisted lightweight Authentication for multi-hop body area network," *Tsinghua Science and Technology*, vol. 26, no. 1, pp. 36–47, 2021.

[12] Shelly and R. Nallanthighal, "Iris retrieval on hadoop: aA biometrics system implementation on cloud computing," in *Proceedings of the 2011 IEEE International Conference on Cloud Computing and Intelligence Systems. IEEE*, pp. 482–485, Beijing, China, September 2011.

[13] G. Hassan and K. Elgazzar, "The case of face recogni- tion on mobile devices," in *Proceedings of the 2016 IEEE wireless communications and networking conference. IEEE*, pp. 1–6, Doha, Qatar, April 2016.

[14] H. X. Kan, L. Jin, and F. L. Zhou, "Classification of medicinal plant leaf image based on multi-feature extraction," *Pattern Recognition and Image Analysis*, vol. 27, no. 3, pp. 581–587, 2017.

[15] J. Ma and Y. Yuan, "Dimension reduction of image deep feature using pca," *Journal of Visual Communication and Image Representation*, vol. 63, Article ID 102578, 2019.

[16] Q.-V. Pham, F. Fang, V. N. Ha et al., "A survey of multi-access edge computing in 5g and beyond: fundamentals, technology integration, and state-of-the-art," *IEEEAccess*, vol. 8, pp. 116974–117017, 2019.

[17] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: a survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[18] R. Bi, Q. Liu, J. Ren, and G. Tan, "Utility aware offloading for mobile-edge computing," *Tsinghua Science and Technology*, vol. 26, no. 2, pp. 239–250, 2021.

[19] T. Zhao, F. Ye, M. Yan, H. Liu, and S. Basodi, "A survey on algorithms for intelligent computing and smart city applications," *Big Data Mining and Analytics*, vol. 4, no. 3, pp. 155–172, 2021.

[20] Z. Xue and H. Wang, "Effective density-based clustering algorithms for incomplete data," *Big Data Mining and Analytics*, vol. 4, no. 3, pp. 183–194, 2021.

[21] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-vision: real-time face retrieval using a mobile-cloudlet-cloud acceleration architecture," in *Proceedings of the 2012 IEEE symposium on computers and communications (ISCC). IEEE*, Cappadocia, Turkey, July 2012.

[22] P. Hu, H. Ning, T. Qiu, Y. Zhang, and X. Luo, "Fog computing based face identification and resolution scheme in internet of things," *IEEE transactions on industrial informatics*, vol. 13, no. 4, pp. 1910–1920, 2016.

[23] S. K. Sharma and X. Wang, "Live data analytics with collaborative edge and cloud processing in wireless iot networks," *IEEE Access*, vol. 5, pp. 4621–4635, 2017.

[24] K. Koutroumbas, *Sergios Theodoridis, Pattern Retrieval*, pp. 327–334, Academic Pressvol, Amsterdam, Netherlands, 2008.

[25] F. Kherif and A. Latypova, "Principal component analysis," *Machine Learning*, Elsevier, Amsterdam, Netherlands, pp. 209–225, 2020.

[26] G. Amato and F. Falchi, "Knn based image classification relying on local feature similarity," in *Proceedings of the Third International Conference on Similarity Searchand Applications*, pp. 101–108, Istanbul Turkey, September 2010.

[27] Z. Abu-Aisheh, R. Raveaux, and J.-Y. Ramel, "Efficient k-nearest neighbors search in graph space," *Pattern Recognition Letters*, vol. 134, pp. 77–86, 2020.

[28] M. Karasuyama and H. Mamitsuka, "Manifold-basedsimilarity adaptation for label propagation," *Advances in Neural Information Processing Systems*, vol. 26, pp. 1547–1555, 2013.

[29] D. Xu, W. Cheng, B. Zong et al., "Deep co-clustering," in *Proceedings of the 2019 SIAM International Confer-ence on Data Mining. SIAM*, pp. 414–422, Houston, TX, USA, March 2019.