COMPUTING FEATURE MATRICES USING PCA-SVD HYBRID METHOD ON SMALL-SCALE SYSTEMS

Le Tien Hung*, Vu Minh Trong, Phan Viet Thanh, Nguyen Le Van
Military Technical Academy, Vietnam
*Corresponding author: letienhung@lqdtu.edu.vn

(Received: April 22, 2024; Revised: August 12, 2024; Accepted: September 27, 2024)
DOI: 10.31130/ud-jst.2024.218E

Abstract - The task of performing feature extraction from input matrices is a well-known problem in biometric recognition. This paper aims to develop an effective method for reduction and decomposition on large matrices with low required computational resources and fast processing times. Our contribution is to design a PCA-SVD hybrid method that divides the feature extraction into two phases: PCA-based size reduction and SVD-based decomposition. In our method, PCA is first applied to a large matrix to extract its important components. The size of the reduced matrix is defined based on the characteristics of the original matrix and the computational capacity of the hardware system, which allows SVD to be applied later. As a result, our method can effectively handle large matrices, leading to significant performance improvements for biometric recognition applications on small computers.

Key words - Feature extraction; biometric recognition; PCA; SVD; eigenvalue; Raspberry Pi.

1. Introduction

Principal Component Analysis (PCA) [1] and Singular Value Decomposition (SVD) [2] are pivotal in processing and analyzing biometric data, offering a balance between computational efficiency and recognition accuracy. Through dimensionality reduction and feature enhancement, these techniques play crucial roles in developing robust biometric recognition systems capable of operating effectively in diverse conditions.

PCA using eigenvalue decomposition (ED) is a linear transformation technique that projects high-dimensional data onto a lower-dimensional space while preserving the maximum amount of variance. It works by finding the principal components of the data, which are the linear combinations of the original features that capture the most variation in the data. PCA can be applied to a data set comprising n vectors $x_1,\dots, x_\mathrm{n} \in R^p$ and in turn returns a new basis for R^p whose elements are terms the principal components. This process is often referred to as Eigenface in facial recognition technologies [3]. It is important that the method is completely data-dependent, that is, the new basis is only a function of the data.

SVD, on the other hand, is a more general matrix factorization technique that can be used to factorize any matrix into three parts: $A=UDV^\mathrm{T}$ such that D is diagonal and U and V are orthogonal. SVD can be used to reduce the dimensionality of a dataset by truncating the diagonal matrix D, which effectively removes the least important singular values and corresponding columns from the original data matrix

PCA complexity involves computing the covariance matrix, which is O(np^2), and then finding its eigenvectors, typically O(p^3), so the total cost is O(np^2+p^3). SVD complexity is generally O(min{np^2, n^2p}), directly decomposing the original matrix without first calculating the covariance matrix. SVD is also preferred for its numerical stability and efficiency, especially when n or p is large.

Recently, SVD has been preferred over ED, and many versions of PCA are based on SVD [4]. Several relevant studies have been presented, including various improved versions of SVD [5], and the integration of PCA and SVD [6] within a unified system for applications ranging from medical imaging to speech recognition and pattern matching. However, when the matrix $\boldsymbol{A}$ becomes large, performing SVD can be very time-consuming, as expected on small-scale systems with limited computing resources. It has been observed through rigorous experimentation and analysis that the efficiency of computational methods is closely linked to the nature of the input data and the computational capacity of the hardware used. Therefore, finding a universally applicable solution across different system architectures remains a challenge.

This paper proposes a hybrid method of PCA and SVD to efficiently reduce the size of feature matrices in biometric recognition systems on small computers. The approach is divided into two steps: using PCA to subtract the mean samples from each row of the large feature matrices, and then performing SVD on the resulting matrices. This strategy retains the useful features for the matching process and significantly reduces the computational resources required. We rigorously investigate the relationship between PCA and SVD, analyze important parameters such as accuracy and running time, and experimentally demonstrate the optimized performance of specific systems.

2. Related works

2.1. Principal Component Analysis and Singular Value Decomposition

We begin by formally introducing two computational schemes. Let $X \in R^{n\times p}$ be a data matrix of interest, consisting of n observations $x_1,\dots, x_n \in R^p$. In other words, each row corresponds to an observation. The goal of PCA is to pro ject this high-dimensional data from R^p onto a lower-dimensional space R^k where $k < p$, where a low-dimensional embedding will be denoted as $Y \in R^{n\times k}$. For the sake of simplicity, we will assume that $p < n$ throughout the paper.

In this part, we turn our attention to detailing the two schemes of PCA mentioned earlier. The first scheme involves the use of eigendecomposition of an empirical covariance matrix. This approach is likely the most straightforward way to both understand the nature of PCA and implement the method using modern computing platforms. Note that $1_n = [1, \dots, 1] \in R^n$ represents a vector of length n whose entries are all 1's, and $\bar{x} = \sum_{i=1}^{n} x_i/n$ denotes a mean vector.

2.1.1. PCA by eigenvalue decomposition of an empirical covariance matrix

The algorithmic details of this traditional PCA algorithm are described below.

– Step 1: Compute an empirical covariance matrix $D \in R^{p\times p}$,

$$D = \frac{1}{n-1}(X - 1_n\bar{x}^T)^T(X - 1_n\bar{x}^T)$$

– Step 2: Apply eigendecomposition to D,

$$DV = V\Lambda \quad \rightarrow \quad Dv_j = \lambda_j v_j \text{ for } j = 1, \dots, p.$$

The eigenvalues are ordered, i.e., $\lambda_1 \geq \dots \geq \lambda_p$ and we assume the same ordering for eigenvectors $v_1, \dots, v_p$.

– Step 3: Assemble the projection matrix $V_{1:k}$ by taking the first k eigenvectors,

$$V_{1:k} = [v_1, \dots, v_k] \in R^{p\times k}$$

– Step 4: Compute the low-dimensional embedding

$$Y = XV_{1:k} \in R^{n\times k}$$

A major drawback of this scheme is its infeasibility for high dimensionality when p is large. Computing the empirical covariance matrix becomes impractical. For example, in the R programming environment [7], a dimensionality of p = 104 results in a covariance matrix that consumes approximately 190 MB of memory. If p is increased to 5×10, the memory requirement increases to 18.6 GB in the standard dense matrix format. These limitations underscore the need for an efficient and feasible alternative, leading to the use of SVD.

2.1.2. PCA by SVD

The algorithmic details of this alternative approach are described below.

– Step 1: Center the data matrix,

$$\bar{X} = X - 1_n\bar{x}^T$$

– Step 2: Apply SVD to $\bar{X} = UDV^T$

Note that left and right singular vectors u_i, v_j are ordered according to the descending order of singular values $\sigma_1 \geq \dots \geq \sigma_p$, respectively.

– Step 3: Assemble the projection matrix $V_{1:k}$ by taking the first k right singular vectors, $V_{1:k} = [v_1, \dots, v_k] \in R^{p\times k}$.

– Step 4: Compute the low-dimensional embedding $Y = XV_{1:k} \in R^{n\times k}$.

Given that the empirical covariance of centered X is $X^TX/(n-1)$, it becomes apparent that the eigenvectors of an empirical covariance matrix are equivalent to the right singular vectors of the original data matrix.

2.2. Computational capacity of small computers

Concurrent with technological advancements, Internet of Things (IoT) computing devices have been integrated extensively across various sectors, including biometric recognition systems. These devices are valued for their compactness, flexibility, and cost-effectiveness. However, IoT devices are often limited by their CPU capacity, RAM, and ability to perform Floating-point Operations Per Second (FLOPs), which can reduce their data processing capabilities for complex tasks [8]. For example, a typical IoT device may be equipped with a CPU operating at approximately 1 GHz, have between 256 MB to 512 MB of RAM, and be capable of performing between 1 to 10 GFLOPs. This comparison emphasizes the importance of optimizing data dimensionality reduction algorithms to match the computational capabilities of these devices.

Table 1. *Computational capabilities of CPUs on IoT devices[8]*

CPU	GFLOPS /W	GFLOPS	Average Power	Max Power	Type
Raspberry Pi Zero 2 W	1.46	5.10	3.5	4.8	ARMv7
Raspberry Pi 4B (4GB)	1.35	9.69	7.2	8.2	ARMv7/8
Raspberry Pi 3	0.813	3.62	4.3	4.8	ARMv7/8
BEagle V	0.68	5.3	8.0	9.0	RISCV64
Dragonboar D	0.450	2.10	4.7	5.7	ARMv8
Raspberry Pi 2	0.432	1.47	3.4	3.6	ARMv7
Raspberry Pi Zero	0.236	0.319	1.3	1.4	ARMv6
Raspberry Pi A+	0.223	0.218	1.0	1.0	ARMv6
Cubieboard2	0.194	0.861	4.4	4.6	ARMv7
Pandaboard-ES	0.163	0.951	5.8	6.5	ARMv7
Raspberry Pi B	0.073	0.213	2.9	3.0	ARMv6

Efficient decomposition techniques are crucial in the era of big data. Li [9] recorded the CPU time for various PCA and SVD methods when predicting labels for items in the test dataset at different dimensionality reduction rates. Figure 1 shows the time consumed by these methods with different dimensionality reduction rates p when applied to the EEG dataset [10]. The PCA methods required less time for classification than the SVD methods on large matrices.

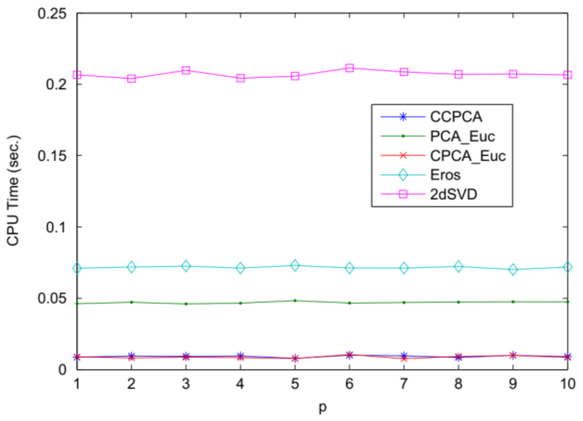
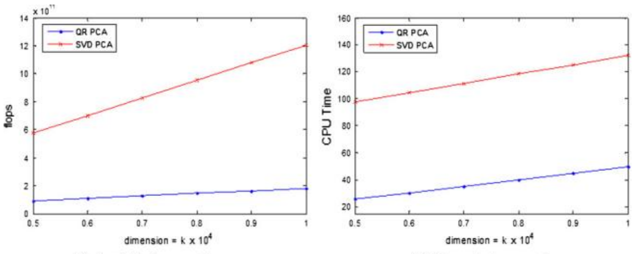


Figure 1. *CPU time costs of various methods according to different dimensionality reduction rates [9]*

Although there are alternative methods to SVD that offer enhanced computational efficiency, such as versions of PCA that employ QR decomposition [10] (see Figure 1), the implementation of SVD remains imperative in biometric recognition systems to ensure accuracy in recognition processes (as depicted in Table 2) [11].



(a) Analytical comparison (b) Numerical comparison

Figure 2. *A comparison of CPU time from [10]*

Table 2. *Performance Comparison of PCA and SVD under various noise conditions [11]*

Noise Conditions	PCA (%)	SVD (%)
Gauss	96.75	98.5
Salt and Pepper	63.75	64.75
Exponential	79.00	82.50
Weibull	67.00	73.50
Beta	100	100

3. PCA-SVD hybrid method

Both PCA (using eigenvalue decomposition) and SVD (using matrix factorisation, which has numerous versions, including the PCA eigenvalue decomposition version) can assist in the evaluation and ranking (by matrix diagonal) of matrix elements according to their importance in the recognition process. This enables the elimination of less significant components, thus reducing the size of the feature matrix, while ensuring the retention of the most crucial components, thereby maintaining recognition accuracy.

The proposed method for reducing the dimensionality of the feature vector from n to k involves two sequential steps. First, an improved PCA method, such as [10], is used to reduce the dimensionality of the feature space from n to k' (where $k' > k$) by leveraging eigenvalue decomposition. Then, SVD is applied to obtain the optimal $k\times k$ feature matrix from the reduced k'-dimensional feature space. This approach effectively mitigates the exponential increase in FLOPs associated with SVD computations, while maintaining the accuracy of the recognition results.

The advantages of our solution include:

– The method employs PCA to reduce data dimensions from n to k' using an approximate eigenvvalue matrix, significantly conserving computational resources.

– The eigenvectors are selected based on the convergence properties of descending eigenvalues to minimize error in determining the best components in SVD later.

– The resulting post-PCA matrix is sparse.

– SVD is then used to further reduce dimensions from k' to k, ensuring an optimal singular matrix.

– Variants of SVD, such as truncated and randomly sampled, are incorporated to enhance processing speed.

We constructed an iris recognition system using a combination of Curvelet transform [12] and the PCA-SVD hybrid method to evaluate its performance. The feature extraction process consists of the following main steps:

– Normalization measures are applied to images to minimize errors caused by intensity differences between two images. This normalization is crucial due to discrepancies in light intensity between different images, which can introduce errors in direct pixel intensity comparisons.

– The system employs the Fast Discrete Curvelet Transform (FDCT) [13] to generate Curvelet coefficient layers from 1 to N, which typically correspond to the dimensions A and B of the image. Normally, N is computed as $N = [\log(min\{A, B\}) - 3]$. The ($N$-1) layer, also known as the fine scale, represents detailed iris image features. However, it has been empirically observed that at this level, the size of the feature vector is significantly large [13]. Therefore, for practicality, only the coefficients from the first layer are selected for iris feature extraction.

– The first layer's coefficients for all images are normalized into row vectors, and iris features are established. The dimensionality of these features is then reduced using the PCA and the important features are selected using SVD. This procedure is applied to datasets of iris images and those requiring recognition, resulting in iris information representation samples.

– The Hamming distance is used to compare the samples.

4. Experimental Results and Discussion

4.1. Recognition accuracy

As previously explained, we select three Curvelet coefficient layers after applying FDCT preprocessing to the images. The coefficients of the first layer contain low-frequency information, which is the primary information of the image. On the other hand, the coefficients of the (N-1) layer represent high-frequency, high-resolution information, containing the finer details of the image. Empirical evidence suggests that utilizing only the first layer's coefficients results in a higher recognition rate without significantly altering the size of the iris feature vector [13]. For recognition purposes, 270 iris images from 27 individuals were selected from the CASIA IrisSyn dataset [14], with each individual contributing 10 images. A random selection of 1 to 9 iris images per individual was used for experimental validation. The experimental procedure is as follows:

– First, feature vectors are generated from low-frequency components and then normalized. Next, features are identified, and dimensionality reduction is achieved through the combined application of PCA and SVD.

– We normalize the low-frequency information into vectors and then normalize the Curvelet coefficients of the second layer to create vectors.

– We formulate a corresponding feature vector for each image. Finally, we perform feature extraction and dimensionality reduction using PCA and SVD.

To evaluate the effectiveness of the recognition process, experiments were conducted using algorithms that integrate Curvelet transform with PCA and Curvelet transform with SVD, employing the same dataset and methodology as described above. The experimental results demonstrate that iris feature extraction using our method yields a higher recognition rate compared to other methods. The same experimental process is repeated for the UBIRIS dataset [15].

Table 3. *Accuracy comparison on UBIRIS and CASIA datasets*

Methods	UBIRIS dataset			CASIA dataset		
	FAR (%)	FRR (%)	Accuracy (%)	FAR (%)	FRR (%)	Accuracy (%)
Daugman [16]	8.08	8.37	91.63	6.78	7.07	92.93
Wavelet [17]	5.21	6.31	93.69	6.54	7.98	92.02
Curvelet [18]	2.21	2.27	97.73	2.01	2.09	97.81
Proposed method	1.20	1.33	98.67	1.84	2.13	97.87

In Table 3, the experiment uses input images from two datasets UBIRIS and CASIA. A total of 450 input images were randomly selected from the UBIRIS database. Each subject designated for recognition is represented by 5 input images captured under suboptimal conditions. The CASIA database included 300 input images, with the first 5 images for each subject undergoing recognition. Preprocessing techniques were not applied to accurately assess recognition quality and eliminate artifacts such as eyelashes, eyebrows, and areas of image reflection or refraction. The identifiable features from the iris images were extracted and transformed into rectangular segments of 360×64 pixels, corresponding to rotational angles of −100°, −50°, 0°, 50°, and 100°. The registration process used 2 input images per subject, resulting in a total of 10 images per subject post-processing. Each subject was left with 3 input images for the testing phase. The evidence suggests that the proposed method improves the metrics used to assess recognition quality.

4.2. Running time

The simulation process was conducted using the MATLAB programming language on a workstation with a 2.66 GHz Intel Xeon quad-core CPU and 32 GB of RAM. Table 4 presents the average computation time for the feature extraction and matching process of 300 iris images. PCA and SVD calculations were performed using the Eigen library [19], which automatically selects appropriate decomposition techniques for the input matrix.

The results show that the proposed method's computation time is equivalent to or lower than that of most other methods.

Table 4. *Computational time comparison*

Methods	Extracting (ms)	Matching (ms)	Total time (ms)
Daugman [16]	302.81	9.45	312.26
Wavelet [17]	126.48	11.28	138.08
Curvelet [18]	43.40	6.71	50.11
Proposed method	27.02	5.21	32.23

Enhancements in time efficiency for both desktop and Raspberry Pi 4B (CPU Cortex A72, 4GB RAM version) systems are presented in Table 5. The reference value (coefficient equal to 1) for comparison is the computation time for PCA on sparse matrices.

Table 5. *Time improvement on workstation and Raspberry Pi 4B*

Scenarios	Workstation Xeon		Raspberry Pi 4B	
	Running time (s)	Increasing rate	Running time (s)	Increasing rate
PCA (sparse matrix)	0.005	1.0	0.029	1.000
PCA (dense matrix)	0.006	1.2	0.035	1.207
Our method (sparse matrix)	0.007	1.4	0.037	1.276
Our method (dense matrix)	0.007	1.4	0.039	1.345
SVD (sparse matrix)	0.014	2.8	0.068	2.345
SVD (dense matrix)	0.015	3.0	0.068	2.345

Our method significantly reduces the execution time required for dimensionality reduction, particularly when processing large feature spaces.

4.3. Discussion

It is important to note that determining the optimal dimensionality k' when using PCA and SVD together is inherently complex due to its dependence on the computational performance of the system.

Table 6. *Running time of SVD on Raspberry Pi 4B*

Sigular values	Running time (s)
130	0.018
280	0.190
394	0.514
599	1.388
675	2.193
1,000	5.809
1,200	8.771
1,810	28.825
1,960	35.790
2,560	81.444
3,887	275.080
4,000	Out of memory

For example, Table 6 shows that the Raspberry Pi 4B (4GB RAM version) can efficiently perform SVD on matrices with dimensions up to k'=4,000. Based on the experiments on the Raspberry Pi 4B, we chose k'=130 to balance the runtime and accuracy of the matching process. The value of k' can be adjusted, depending on the characteristics of the input data and the computational capacity of the system.

5. Conclusion

In this paper, we introduce a PCA-SVD hybrid method for feature matrix reduction on small computers, which is characterized by both computational cost and time efficiency. The PCA-SVD hybrid method employs two phases for feature matrix reduction, effectively and flexibly handling large matrices. In the first phase, a PCA algorithm is applied to reduce the size of the original matrices so that they can be decomposed using SVD in the next phase with limited computational resources. The new PCA-SVD hybrid method ensures that the large matrices can be reduced while still preserving the important features, which significantly improves its performance, especially when applied to small-scale systems.

REFERENCES

[1] Pearson, "LIII. On lines and planes of closest fit to systems of points in space", *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, Nov. 1901. https://doi.org/10.1080/14786440109462720

[2] G. W. Stewart, "On the Early History of the Singular Value Decomposition", *SIAM Review*, vol. 35, no. 4, pp. 551–566, Dec. 1993. https://doi.org/10.1137/1035134

[3] M. Turk and A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, Jan. 1991. https://doi.org/10.1162/jocn.1991.3.1.71

[4] D. Kim and K. You, "PCA, SVD, and Centering of Data", *arXiv:2307.15213v2*, Jul. 2023. https://doi.org/10.48550/arXiv.2307.15213

[5] P. C. Hansen, "The truncatedSVD as a method for regularization", *BIT Numerical Mathematics*, vol. 27, no. 4, pp. 534–553, Dec. 1987. https://doi.org/10.1007/BF01937276

[6] M. Mateen, J. Wen, Nasrullah, S. Song, and Z. Huang, "Fundus Image Classification Using VGG-19 Architecture with PCA and SVD", *Symmetry*, vol. 11, no. 1, p. 1, Dec. 2018. https://doi.org/10.3390/sym11010001

[7] R Core Team, "An Introduction to R", *r-project.org*, Feb. 29, 2024. [Online]. Available: https://cran.r-project.org/doc/manuals/r-release/R-intro.html [Accessed Apr. 10, 2024].

[8] The Weaver Computer Engineering Research Group, "The GFLOPS/W of the various machines in the VMW Research Group", *maine.edu*, 2024. [Online]. Available: https://web.eece.maine.edu/~vweaver/group/green_machines.html. [Accessed Apr. 11, 2024].

[9] H. Li, "Accurate and efficient classification based on common principal components analysis for multivariate time series", *Neurocomputing*, vol. 171, pp. 744–753, Jan. 2016. https://doi.org/10.1016/j.neucom.2015.07.010

[10] A. Sharma, K. K. Paliwal, S. Imoto, and S. Miyano, "Principal component analysis using QR decomposition", *International Journal of Machine Learning and Cybernetics*, vol. 4, no. 6, pp. 679–683, Dec. 2013. https://doi.org/10.1007/s13042-012-0131-7

[11] S. Noushath, A. Rao, and G. H. Kumar, "SVD Based Algorithms for Robust Face and Object Recognition in Robot Vision Applications", in *Proceedings of the 24th ISARC, Kochi, India*, Sep. 2007. https://doi.org/10.22260/ISARC2007/0079

[12] R. Vyas, T. Kanumuri, G. Sheoran, and P. Dubey, "Efficient iris recognition through curvelet transform and polynomial fitting", *Optik*, vol. 185, pp. 859–867, May 2019. https://doi.org/10.1016/j.ijleo.2019.04.015

[13] E. Candès, L. Demanet, D. Donoho, and L. Ying, "Fast Discrete Curvelet Transforms", *Multiscale Modeling & Simulation*, vol. 5, no. 3, pp. 861–899, Jan. 2006. https://doi.org/10.1137/05064182X

[14] Institute of Automation-Chinese Academy of Science, "CASIA v3.0 Iris Image Database", *cbsr.ia.ac.cn*, Dec. 14, 2010. [Online]. Available: http://www.cbsr.ia.ac.cn/english/IrisDatabase.asp. [Accessed Apr. 11, 2024].

[15] H. Proenca, S. Filipe, R. Santos, J. Oliveira, and L. A. Alexandre, "The UBIRIS.v2: A Database of Visible Wavelength Iris Images Captured On-the-Move and At-a-Distance", *IEEE Trans Pattern Anal Mach Intell*, vol. 32, no. 8, pp. 1529–1535, Aug. 2010. https://doi.org/10.1109/TPAMI.2009.66

[16] J. Daugman, "New Methods in Iris Recognition", *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 5, pp. 1167–1175, Oct. 2007. https://doi.org/10.1109/TSMCB.2007.903540

[17] M. Haindl and M. Krupička, "Unsupervised detection of non-iris occlusions", *Pattern Recognit Lett*, vol. 57, pp. 60–65, May 2015. https://doi.org/10.1016/j.patrec.2015.02.012

[18] J. Sun, Z.-M. Lu, and L. Zhou, "Iris recognition using curvelet transform based on principal component analysis and linear discriminant analysis", *Journal of Information Hiding and Multimedia Signal Processing*, vol. 5, pp. 567–573, Apr. 2014.

[19] G. Guennebaud and B. Jacob, "SVD module", *eigen.tuxfamily.org*, Apr. 21, 2022. [Online]. Available: https://eigen.tuxfamily.org/dox/group__SVD__Module.html [Accessed Aug. 03, 2024].