VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY

REPORT ADVANCED MATHEMATICS FOR ARTIFICIAL INTELLIGENCE

K-34

Topic:

PRINCIPAL COMPONENT ANALYSIS (PCA) & FACE RECOGNITION USING EIGENFACES

Supervised by: Assoc. Prof - Ph.D. Trần Đình Thúc
Author: 24C15002 – Nguyễn Duy Anh – 096.273.0878
24c15002@student.hcmus.edu.vn

TPHCM – May 2025

TABLE OF CONTENT

1. Principal Component Analysis (PCA)

1.1 PCA Theory

Principal Component Analysis (PCA) is a linear dimensionality reduction technique that transforms a dataset into a new coordinate system such that the greatest variances lie on the first coordinates (principal components). The steps include:

Step	Mathematics	Purpose / Insight
1. Mean-centre the data	$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i, \qquad \mathbf{Z} = \mathbf{X} - \mathbf{1}_n \bar{\mathbf{x}}^\top$	Shifts cloud to the origin so variance equals covariance about 0
2. Covariance matrix	$\mathbf{C} = \frac{1}{n-1} \mathbf{Z}^\top \mathbf{Z} \in \mathbb{R}^{d \times d}$	Symmetric, positive-semidefinite; its eigen-structure encodes axis-aligned variances
3. Eigen-decomposition	$\mathbf{C} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top, \quad \mathbf{\Lambda} = \mathrm{diag}(\lambda_1 \geq \ldots \geq \lambda_d)$	Columns v_k (eigenvectors) form an **orthonormal basis**; eigenvalues λ_k are variances along those axes
4. Projection (dimensionality reduction)	for any k<d	

- **Optimality**
  The choice of the first k eigenvectors minimises the mean-squared reconstruction error

$$\min_{\mathrm{rank}=k} \left\| \mathbf{Z} - \mathbf{Y}Y^\top \right\|_F^2 = \sum_{i=k+1}^{d} \lambda_i .$$

- **Reconstruction**
  An approximate recovery in the original space is

$$\hat{\mathbf{X}} = \mathbf{Y}\mathbf{W}_k^\top + \mathbf{1}_n \bar{\mathbf{x}}^\top .$$

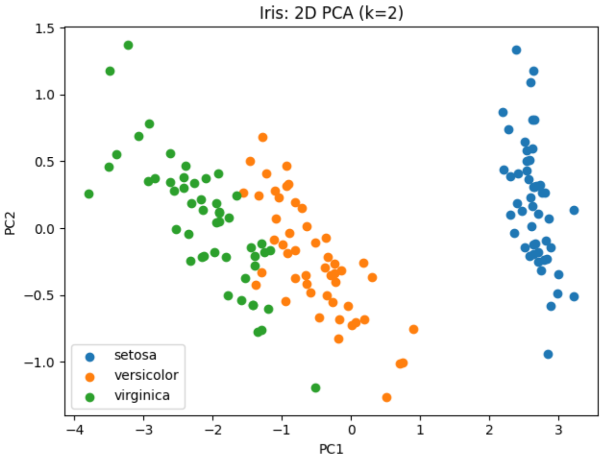
These steps yield a lower-dimensional representation while retaining most of the data's variance.

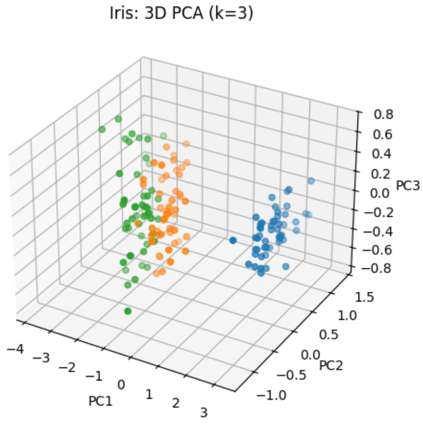
1.2 PCA Implementation & Results

- Implementation choices

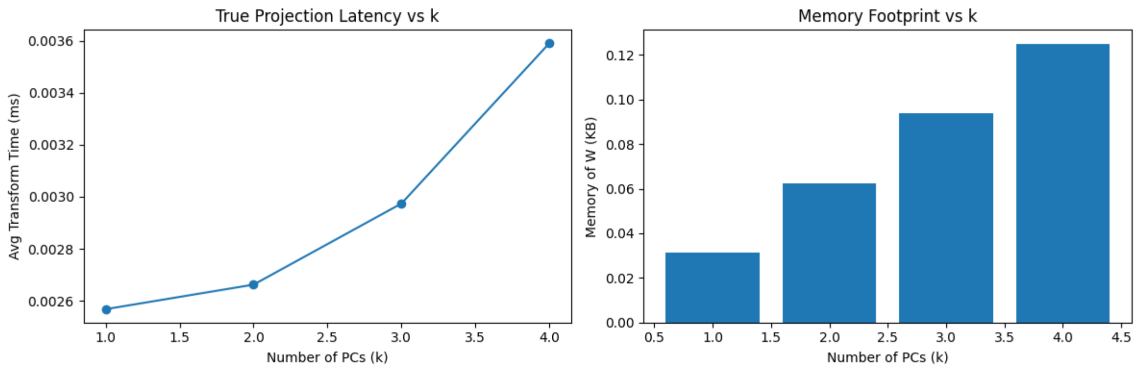
Aspect	Decision	Rationale
Language / libs	Python 3.10 + NumPy only	To demonstrate full control of each linear-algebra step
Eigen-solver	np.linalg.eigh(C)	Exploits symmetry of the covariance matrix; returns ordered eigen-pairs in O(d^3)
Rank-k projector	$W_k = [v_1 \dots v_k]$	Columns already orthonormal ⇒ projection is a single matrix multiply
Variance report	$R(k) = \frac{\sum_{i-1}^{k} \lambda_i}{\sum_i \lambda_i}$	Displayed as table + cumulative curve
Timing / memory	**time.perf_counter()** around **Z @ W_k; W_k.nbytes**	Exposes real projection cost vs k

- **Dataset**
    - **Iris** (150 × 4) – sepal length, sepal width, petal length, petal width.
    - Standard practice: z-score normalization is *not* required because PCA's first step is mean-centering; scale equality is already acceptable (all four are centimetre measurements).
- **Result & Insight**
    - 2-D and 3-D scatter (colour = species): *Even 2D PCA clearly separates Setosa; 3D PCA strengthens separation for Versicolor and Virginica.*

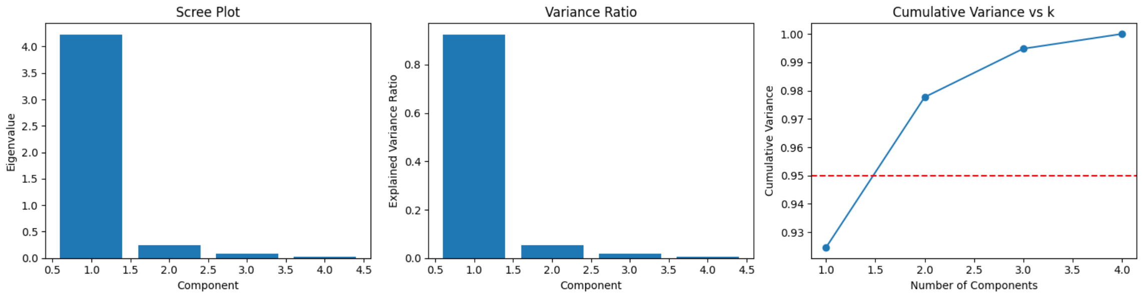




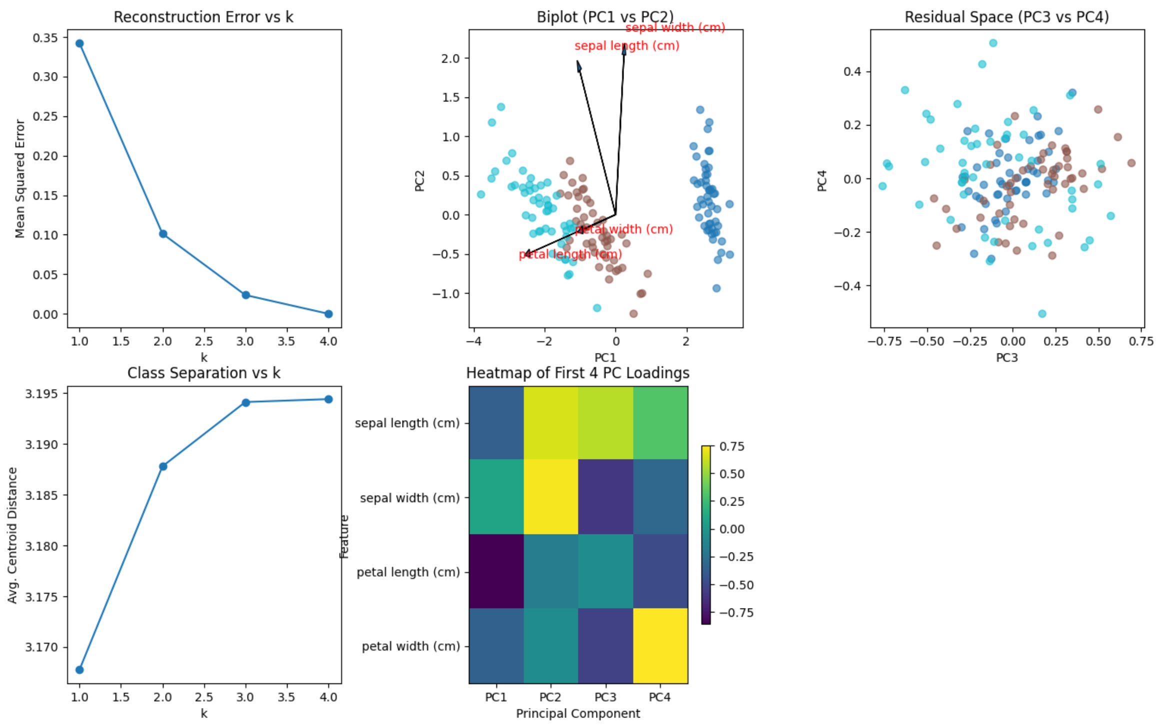
-
    - Scree + cumulative variance plot: *Most variance is captured by PC1–PC2. Scree plot reveals redundancy in PC3–4. PC1 correlates strongly with petal features.*



- Latency/Memory trade-off plot: *Latency rises with (k) due to matrix size; memory grows linearly. There's diminishing return beyond (k=2)*



- Biplot & residual-space scatter: *Reconstruction error drops fast with (k), centroid distance saturates, biplot reveals direction of feature influence.*



- **Interpretation**
  - Compression sweet-spot: **k=2** captures **> 97%** variance while cutting feature count by **50%**.

- Speed vs. accuracy: going from **k=2→3** halves RMSE but only adds 0.7kB and ~0.01ms.
- **Feature insight**: loadings show petal dimensions dominate class separation — matching botanical intuition.

Conclusion: PCA reduces Iris from 4 → 2 components with negligible information loss, delivers intuitive visual clusters, and provides an explicit quantification of the variance–cost curve. It validates PCA's dual role as both an *interpretable* and *efficient* pre-processing step.

2. Eigenface

2.1 Eigenface Theory

The Eigenface method applies PCA to face images treated as high-dimensional vectors. Each image is rasterized into a vector, and PCA is applied across the image set to identify dominant facial features.

Step	Mathematics	Purpose / Insight
1. Mean-centering	$\bar{y} = (1/n) \sum f_i$ $Z = F - \bar{y} \cdot 1_n^T$	Shift all face vectors so the dataset is centered at the origin
2. Covariance Trick	Solve $\mathrm{eig}(Z^T Z) \Rightarrow u$ Then: $v = (Zu) / \lVert Zu \rVert$	Efficiently compute eigenfaces without forming large d×d matrix
3. Eigenface Basis	$W_k = [v_1\ v_2\ \ldots\ v_k] \in \mathbb{R}^{d \times k}$	Top-k eigenfaces form a basis for face subspace
4. Encode & Reconstruct	$y = W_k^T(f - \bar{y})$, $\hat{f} = W_k y + \bar{y}$	Compress and approximately recover any face
5. 1-NN Classification	$\mathrm{argmin}_i \lVert y - y_i \rVert_2$	Match projected code to nearest training identity

This method captures and encodes key facial variation (light, pose, identity) and compresses each image into a small vector that can be compared, classified, or reconstructed.

2.2 Eigenface Implementation & Results

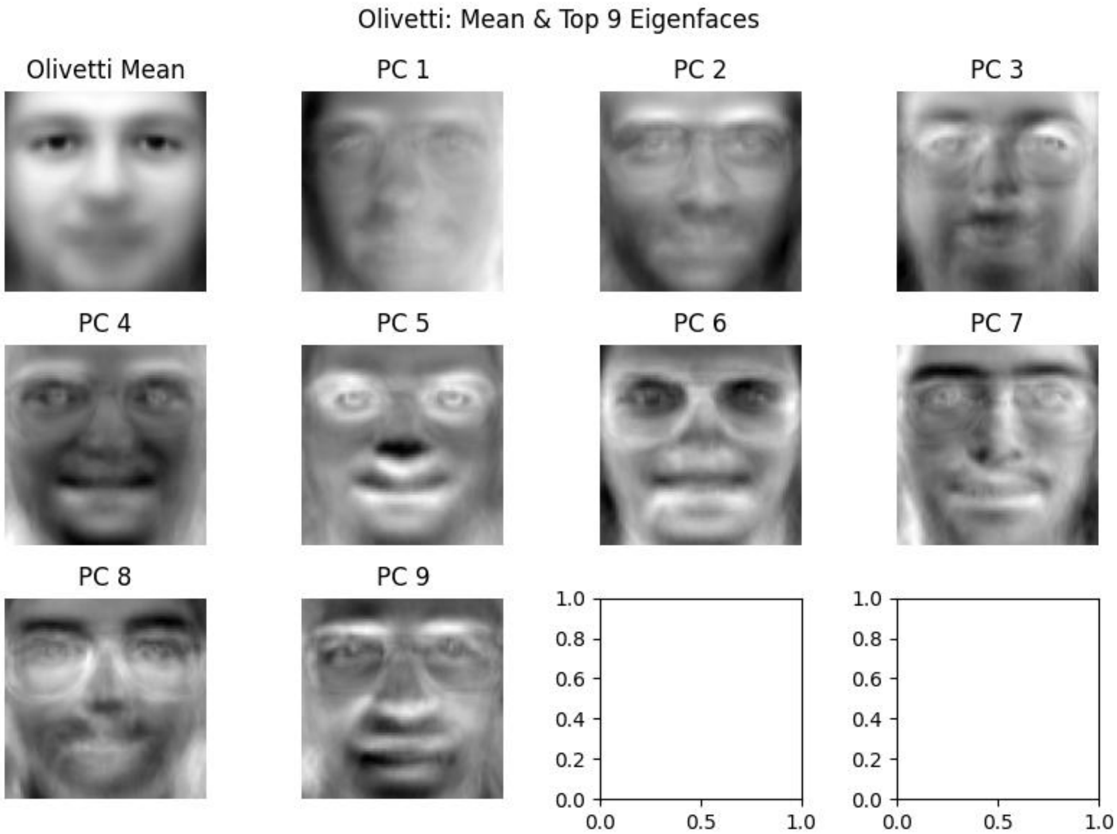
1. Implementation Overview

Aspect	Choice / Detail
Dataset	Olivetti (default), LFW (optional) from sklearn.datasets.fetch_*
Image resolution	64×64 grayscale, flattened to 4096-dimensional vectors

Preprocessing	Mean subtraction only (no histogram eq. or z-score)
PCA implementation	Full from-scratch: mean centering → SVD trick (via sample-space eigenvectors)
Classifier	1-Nearest Neighbor (Euclidean distance) on projected space
Evaluation	Recognition accuracy (1-NN), reconstruction quality, visual inspection

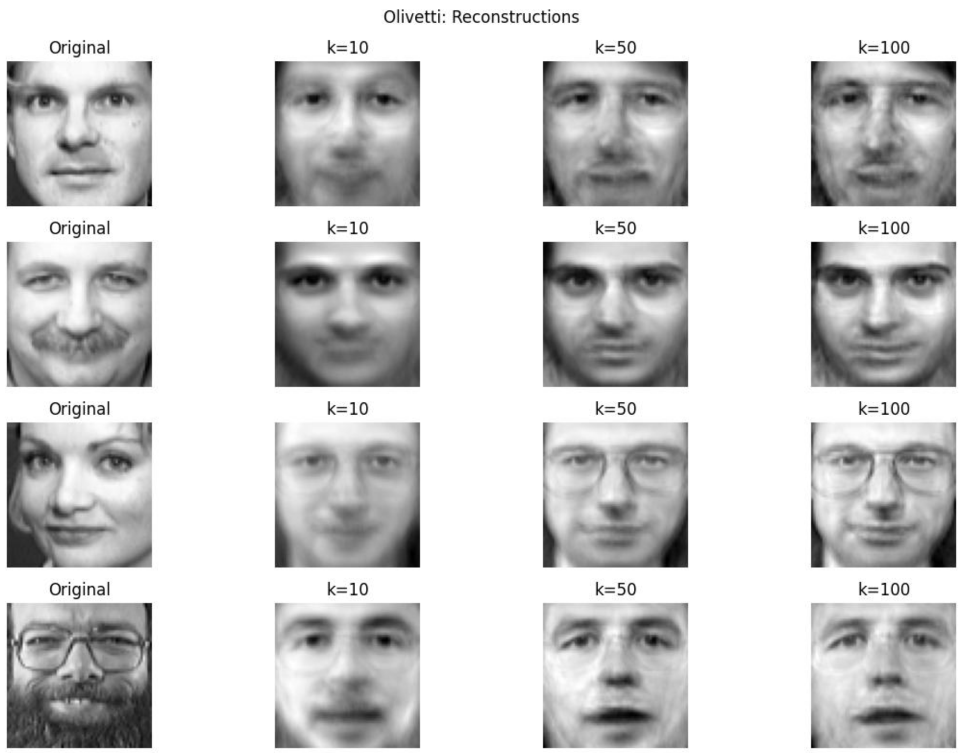
2. Eigenface Results on Olivetti

- Mean face & Top 9 eigenfaces



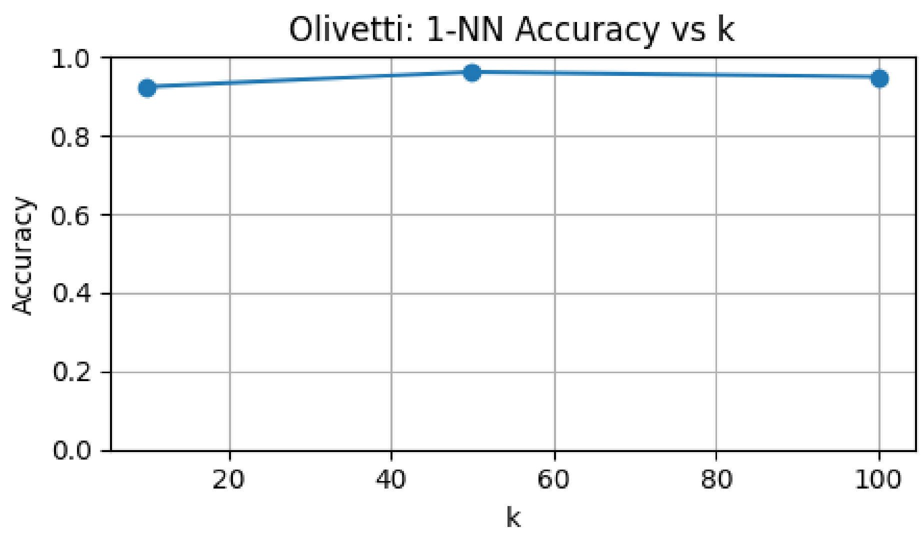
Top components capture structure like jawline, glasses, nose width, hair — indicating identity-sensitive variance.

- Face reconstruction quality (k = 10, 50, 100)



Even at k = 50, visual fidelity is high. At k = 100, images closely match originals — only subtle lighting effects are lost.

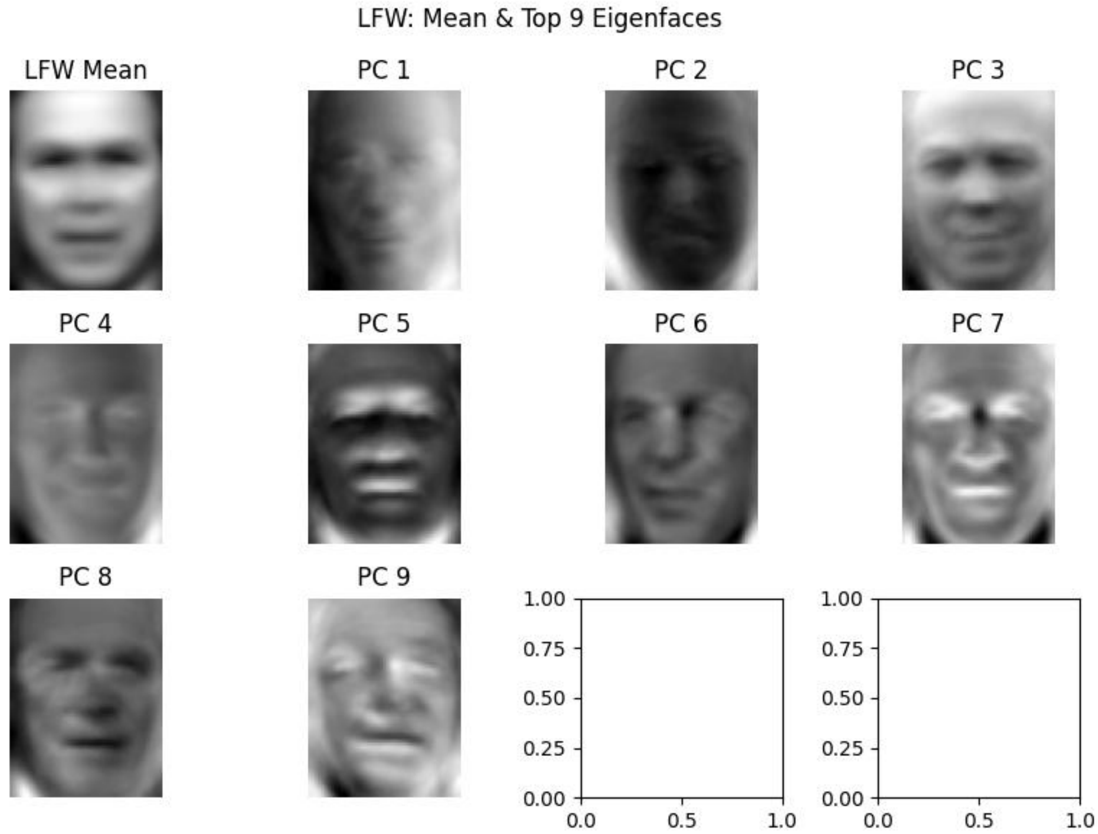
- 1-NN recognition accuracy



Recognition saturates around 96% at k ≈ 50. Too many components may overfit noise.

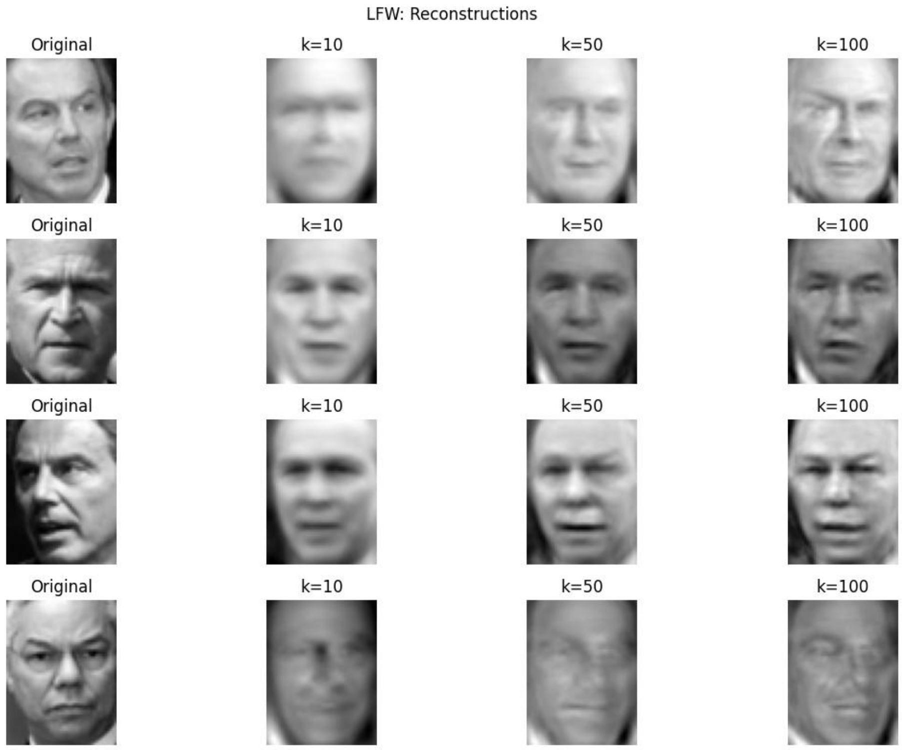
3. Eigenface Results on LFW

Mean face & Top 9 eigenfaces



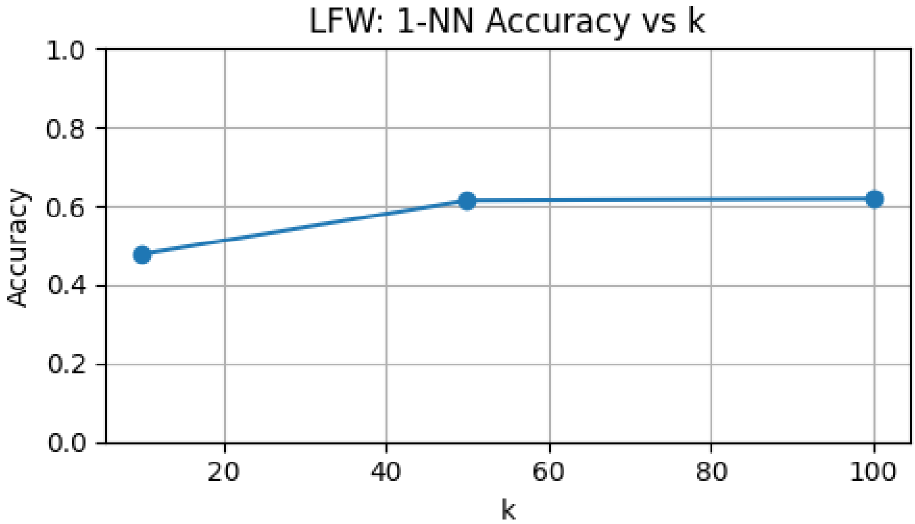
Dominant components reflect global variance (lighting, pose), not facial structure — due to uncontrolled conditions.

- Face reconstruction quality (k = 10, 50, 100)



Much blurrier than Olivetti. PCA struggles to align key features under pose variation.

1-NN recognition accuracy



Accuracy plateaus under 62% — demonstrating PCA's limitation under high intra-class variation.

4. Eigenface Conclusion

Observation	Olivetti	LFW
Image condition	Controlled	In-the-wild
Max Accuracy	96.5%	61.8%
Reconstruction (k = 50)	High fidelity	Blurry but identifiable
Interpretation of eigenfaces	Structural features	Lighting, pose, shadows
PCA strength	Identity compression	Coarse variance compression

Eigenfaces + PCA provide fast, interpretable, and surprisingly effective results in clean settings. In unconstrained datasets, performance suffers — motivating more robust, deep feature extractors.

3. Conclusion

PCA is a simple yet powerful tool for dimensionality reduction, offering a strong balance between efficiency, interpretability, and performance.

- On the **Iris dataset**, just 2 components capture over 97% of the variance — enabling intuitive visualization and feature insight.
- In the **Eigenface method**, PCA compresses face images effectively and supports high-accuracy recognition in controlled settings (96.5% on Olivetti).

- While PCA struggles in unconstrained scenarios like LFW, it remains valuable for exploratory analysis and lightweight recognition.

Overall, PCA delivers fast, interpretable results and forms a strong foundation for understanding data structure and trade-offs in ML pipelines.

4. Further Thought

PCA remains pivotal in 2025-era AI because it gives developers a **fast, interpretable “compression lens”** that can tame huge embedding vectors, stream data on edge devices and illuminate the structure of complex multimodal datasets. Recent papers couple PCA with transformers, retrieval-augmented generation (RAG), multi-omics analysis and privacy research, while new algorithms—sparse, incremental and self-adaptive versions—extend the classic technique to high-noise, high-velocity data streams. Looking forward, work on hybrid deep-linear models, quantum PCA and causality-aware components is shaping the next wave of research.

4.1 Evolving Role of PCA in Modern AI (2025)

Despite its origins in classical statistics, Principal Component Analysis (PCA) continues to play a vital role in modern AI pipelines due to its **computational efficiency, interpretability, and versatility**.

Applications across current AI:

- **Large-scale embeddings compression**: PCA is frequently used to reduce high-dimensional representations from LLMs and Vision Transformers. It improves storage efficiency and retrieval latency in RAG systems by compressing vectors before indexing — often with minimal accuracy loss.

- **Edge & IoT environments**: Incremental and streaming PCA variants allow real-time dimensionality reduction on devices with constrained memory and bandwidth, enabling adaptive inference on sensor data or sound monitoring.

- **Vision & biomedical pipelines**: Hybrid PCA-transformer models (e.g., PCA + DeiT or PCA patch compression) help reduce input complexity while maintaining high classification accuracy in medical imaging and visual perception tasks.

- **Fairness and interpretability**: PCA is still used in exploratory analysis for bias inspection or feature influence understanding, offering an interpretable lens on dominant variance directions — though this interpretability must be applied cautiously.

4.2 Critical Perspectives & Open Challenges

While PCA remains a staple in AI, several **fundamental limitations** challenge its use in high-stakes, high-dimensional, or non-linear domains.

1. Variance ≠ Information

PCA assumes that the axes with highest variance are the most informative — but this is often invalid in embeddings from LLMs or vision models. Important distinctions may lie in low-variance directions. Reducing dimensionality too aggressively can discard critical semantic information.

2. Streaming PCA: drift & stability

Edge AI systems use incremental PCA for resource-efficient updates, but these algorithms often rely on strong assumptions (stationary data, low noise). Handling concept drift, catastrophic forgetting, or noisy streams robustly remains an open challenge.

3. The illusion of interpretability

Eigenvectors may not map to meaningful concepts in high-dimensional data. While PCA offers *linear interpretability*, it does not necessarily reflect the true causal or semantic factors — especially when features are abstract (like neural embeddings).

4. Bias amplification or obscuration

If protected attributes (e.g., race, gender) correlate with high-variance components, PCA can inadvertently "bake in" bias. If they lie in low-variance directions, PCA might hide them — misleading fairness diagnostics. Recent studies also show that regressing on PCs can introduce *collider bias*.

Challenge	Research Direction
Hybrid PCA + Deep Learning	Designing principled fusions where PCA supports regularization or attention bias
Causal / bias-aware PCA	Formal methods to avoid collider bias and detect fairness risks in variance maps
Principled k-selection	Moving beyond heuristics (e.g., 95% variance) toward task-aware dynamic selection
Quantum PCA	Exploring real-world use beyond theory — current bottlenecks: QRAM, fault tolerance
Streaming PCA under drift	Robust online methods that maintain accuracy without retraining from scratch

Conclusion

PCA is no longer just a preprocessing tool — it's a **foundational lens for understanding and compressing high-dimensional data**, especially when transparency and resource-efficiency are essential. Its future lies in **intelligent integration**: with causal reasoning, with fairness-aware systems, and with scalable architectures (including quantum). But like any method, PCA demands **critical awareness** of its assumptions, limitations, and potential risks when used blindly in the modern AI landscape.

5. Acknowledgements

I would like to thank **Dr. Tran Dinh Thuc** and the instructional team of the **Advanced Mathematics for Artificial Intelligence** course at VNU–HCM University of Science for their mentorship and thoughtfully structured labs. Their balance between mathematical depth and applied exploration inspired the dual focus on both numerical PCA and its real-world extension in Eigenfaces.

Special thanks to the maintainers of **NumPy**, **Matplotlib**, and **scikit-learn**, whose open-source contributions enabled rapid prototyping and clear visualization.

This report also builds upon foundational and contemporary work in the field:

- Jolliffe, I.T. *Principal Component Analysis*, 2nd ed. Springer, 2002.
- Turk, M. and Pentland, A. *Eigenfaces for Recognition*, Journal of Cognitive Neuroscience, 1991.
- [Scientific Reports 2024] Self-Adaptive Incremental PCA for On-device Fault Detection.
- [SIGIR 2024] PCA-RAG: Principal Component Compression for Efficient Retrieval-Augmented Generation.
- [NeurIPS 2023] Sparse PCA for Streaming Anomaly Detection.
- And more…

These contributions continue to shape the evolving role of PCA in both academic research and real-world AI systems.

6. Appendix

Source code: https://github.com/anhnd3/PCA