BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC QUY NHƠN

NGUYỄN NHẬT NAM

MỘT SỐ PHƯƠNG PHÁP PHÂN TÍCH MA TRẬN VÀ ỨNG DỤNG

KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC

Ngành: Toán ứng dụng
Chuyên ngành: Khoa học dữ liệu

Người hướng dẫn: TS.Lâm Thị Thanh Tâm

Bình Định, 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC QUY NHƠN

MỘT SỐ PHƯƠNG PHÁP PHÂN TÍCH MA TRẬN VÀ ỨNG DỤNG

KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC

Ngành: Toán ứng dụng
Chuyên ngành: Khoa học dữ liệu

Sinh viên thực hiện: NGUYỄN NHẬT NAM
Mã số SV: 4251140002
Lớp, Khóa: Toán ứng dụng K42

Người hướng dẫn: TS.Lâm Thị Thanh Tâm

Bình Định, 2023

Lời cảm ơn

Trước khi đi vào nội dung của khóa luận, em xin bày tỏ lòng biết ơn chân thành tới toàn thể giáo viên của Trường Đại học Quy Nhơn nói chung và của Khoa Toán-Thống kê nói riêng vì đã tận tâm dạy bảo em trong suốt quá trình em theo học tại trường, và đặc biệt là sự định hướng dẫn dắt của TS.Lâm Thị Thanh Tâm đối với đề tài khóa luận này.

Nhân dịp này em cũng xin được gửi những lời cảm ơn đến những người bạn, đặc biệt là những người thân trong gia đình đã luôn giúp đỡ em hết mình, luôn động viên, cổ vũ tinh thần và tạo điều kiện thuận lợi giúp em có thể hoàn thành được khoá luận tốt nghiệp này.

Trong quá trình viết khóa luận này mặc dù đã được chỉnh sửa nhiều lần nhưng không thể tránh khỏi việc thiếu sót. Em xin chân thành cảm ơn nếu nhận được sự góp ý từ các quý thầy cô, anh chị và bạn bè để có thể chỉnh sửa luận văn được tốt hơn.

Quy Nhơn, ngày ... tháng ... năm 2023

Sinh viên thực hiện

Nguyễn Nhật Nam

Mục lục

Lời mở đầu

Trong thời đại công nghệ thông tin nói chung và dữ liệu hơn nói riêng đang ngày càng phát triển mạnh mẽ, việc thu thập và xử lý dữ liệu ngày một trở nên quan trọng và cần thiết. Các phương pháp phân tích dữ liệu như phân rã giá trị kì dị (SVD) và phân tích thành phần chính (PCA) đóng vai trò quan trọng trong việc giải quyết các vấn đề liên quan đến dữ liệu lớn và đa chiều. Những phương pháp này không chỉ giúp giảm kích thước dữ liệu mà còn giúp khai thác thông tin hữu ích từ dữ liệu gốc. SVD và PCA đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực khoa học và kỹ thuật, chẳng hạn như xử lý hình ảnh, phân tích dữ liệu, nhận dạng mẫu, thống kê,...

Trong khóa luận này, chúng tôi tập trung nghiên cứu về các phương pháp phân tích dữ liệu như phân rã giá trị kì dị (SVD) và phân tích thành phần chính (PCA), cũng như một số ứng dụng của chúng trong bài toán thực tế. Mục tiêu chính của khóa luận là tìm hiểu về các tính chất cơ bản và ứng dụng của SVD và PCA trong việc giải quyết các vấn đề liên quan đến dữ liệu.

Đối tượng nghiên cứu trong khóa luận này là phương pháp phân rã giá trị kì dị (SVD) và phân tích thành phần chính (PCA). Ngoài lời cảm ơn, mục lục, lời mở đầu và một số ký hiệu, luận văn được chia làm bốn chương chính:

- Chương 1: Kiến thức cơ bản. Trong chương này, chúng tôi trình bày một số kiến thức cơ sở về đại số tuyến tính, giải tích và thống kê, nhằm chuẩn bị cho việc tìm hiểu về SVD và PCA.

- Chương 2: Phân rã giá trị kì dị (SVD). Trong chương này, chúng tôi giới thiệu về SVD và các tính chất của ma trận liên quan đến SVD.

- Chương 3: Phân tích thành phần chính (PCA). Trong chương này, chúng tôi giới thiệu về PCA, cách tính PCA và mối quan hệ giữa PCA và SVD.

- Chương 4: Ứng dụng của SVD và PCA trong bài toán thực tế. Chương này sẽ trình bày một số ví dụ cụ thể về việc áp dụng SVD và PCA trong các bài toán thực tế như xử lý hình ảnh, nhận dạng khuôn mặt, phân tích dữ liệu và thống kê.

Khóa luận được hoàn thành dưới sự hướng dẫn của TS.lâm Thị Thanh Tâm. Tôi cũng xin chân thành cảm ơn tập thể lớp Toán Ứng Dụng K42, Khoa Toán và Thống kê Trường Đại học Quy Nhơn đã giúp đỡ và tạo mọi điều kiện thuận lợi cho tôi hoàn thành khóa học cùng với khóa luận này.

Cuối cùng, tôi xin chân thành cảm ơn gia đình, bạn bè, những người thân yêu đã luôn quan tâm, giúp đỡ và ủng hộ tôi trong suốt quá trình học tập và thực hiện khóa luận này. Dù đã cố gắng hết sức, nhưng do thời gian và năng lực có hạn, khóa luận không tránh khỏi những thiếu sót và hạn chế. Rất mong nhận được sự góp ý và chỉ bảo của quý thầy cô, quý bạn đồng nghiệp để khóa luận tốt nghiệp này được hoàn thiện hơn. Tôi xin chân thành cảm ơn.

Một số ký hiệu

$\mathbb{R}$	:	Tập hợp các số thực
$\mathbb{R}^n$	:	Tập hợp các véc tơ thực n chiều
$\mathbb{R}^{n\times m}$	:	Tập hợp các ma trận thực cấp $n \times m$
$\|\mathbf{A}\|_F$	:	Chuẩn Frobenius của ma trận $\mathbf{A}$
$\|\mathbf{A}\|_2$	:	Chuẩn 2 của ma trận $\mathbf{A}$
$\mathbf{A}^T$	:	Ma trận chuyển vị của $\mathbf{A}$
$\mathbf{A}^{-1}$	:	Ma trận nghịch đảo của $\mathbf{A}$
$\mathbf{A}^+$	:	Ma trận giả nghịch đảo của $\mathbf{A}$

Chương 1

Một số kiến thức chuẩn bị

1.1 Ma trận

.

Định nghĩa 1.1.1. *[**hieu2019**] Ma trận cỡ $m \times n$ là một bảng gồm mn số thực được sắp xếp thành m dòng và n cột. Ma trận thường được kí hiệu như sau*

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & \cdots & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \cdots & \cdots & \cdots & \vdots \\ a_{m1} & a_{m2} & a_{13} & \cdots & \cdots & \cdots & a_{mn} \end{bmatrix},$$

hoặc

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & \cdots & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \cdots & \cdots & \cdots & \vdots \\ a_{m1} & a_{m2} & a_{13} & \cdots & \cdots & \cdots & a_{mn} \end{pmatrix},$$

hoặc $\mathbf{A} = (a_{ij})_{m\times n}$, trong đó a_{ij} là phần tử của ma trận nằm trên dòng i, cột j, với $i = 1, 2, \ldots, m$ và $j = 1, 2, \ldots, n$.

Khi $m = n$, ta gọi ma trận cỡ $m \times m$ là ma trận vuông cấp m. Các phần tử a_{11}, $a_{22}, \ldots, a_{mm}$ nằm trên một đường thẳng được gọi đường chéo chính của ma trận.

Định nghĩa 1.1.2. *[**hieu2019**] Ma trận đơn vị cấp n là ma trận vuông cấp n có mọi phần tử nằm trên đường chéo chính bằng 1, các phần tử khác bằng 0. Ta ký hiệu ma trận đơn vị cấp n bởi* $\mathbf{I}_n$ *và nó có dạng như sau*

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

Trong trường hợp không cần chú ý đến cấp của ma trận, ta ký hiệu ma trận đơn vị bởi $\mathbf{I}$.

Định nghĩa 1.1.3. *[**an2015**] Ma trận đường chéo là ma trận vuông có các phần tử nằm trên đường chéo chính khác 0, các phần tử nằm ngoài đường chéo chính bằng 0. Ma trận đường chéo có dạng như sau*

$$\mathbf{D} = \begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{mm} \end{bmatrix}$$

Ma trận chỉ có một dòng được gọi là vector dòng. Ma trận chỉ có một cột được gọi là vector cột.

Định nghĩa 1.1.4. *[**hung2020**] Ma trận vuông A cấp n được gọi là ma trận khả nghịch nếu tồn tại một ma trận* $\mathbf{A}'$ *vuông cấp n thỏa mãn* $\mathbf{AA}' = \mathbf{A}'\mathbf{A} = \mathbf{I}_n$*. Ma trận* $\mathbf{A}$ *được gọi là ma trận nghịch đảo của ma trận* $\mathbf{A}$*, và được ký hiệu là* $\mathbf{A}^{-1}$

Định nghĩa 1.1.5. *[**hieu2019**] Cho ma trận* $\mathbf{A} = (a_{ij})_{m\times n}$*. Ma trận chuyển vị của* $\mathbf{A}$*, ký hiệu là* $\mathbf{A}^T$*, có dạng* $\mathbf{A}^T = (a_{ji})_{n\times m}$*.*

Ma trận vuông $\mathbf{A}$ *được gọi là ma trận đối xứng nếu* $\mathbf{A}^T = \mathbf{A}$*, và được gọi là ma trận phản đối xứng nếu* $\mathbf{A}^T = -\mathbf{A}$*.*

Định nghĩa 1.1.6. *[**hieu2019**] Ma trận vuông* $\mathbf{A}$ *được gọi là ma trận trực giao nếu* $\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T = \mathbf{I}$

Nhận xét:

(i) Ma trận trực giao $\mathbf{A}$ là khả nghịch và $\mathbf{A}^T = \mathbf{A}^{-1}$.

(ii) Ma trận $\mathbf{A}$ trực giao khi và chỉ khi các vector cột và các vector hàng của $\mathbf{A}$ tạo thành các hệ trực chuẩn.

Định nghĩa 1.1.7. *[hieu2019] Cho $\mathbf{A}$ là một ma trận vuông cấp n. Định thức của ma trận $\mathbf{A}$, ký hiệu là $\det(\mathbf{A})$ hay $|\mathbf{A}|$ là một giá trị được xác định bằng công thức*

$$\det(\mathbf{A}) = a_{11}\mathbf{A}_{11} + a_{12}\mathbf{A}_{12} + \cdots + a_{1n}\mathbf{A}_{1n}$$

trong đó $\mathbf{A}_{ik} = (-1)^{i+k} \det(\mathbf{M}_{ik})$, với $\mathbf{M}_{ik}$ là ma trận vuông cấp $n-1$ nhận được từ ma trận $\mathbf{A}$ bằng cách bỏ đi dòng thứ i và cột thứ k. Đại lượng $\mathbf{A}_{ik}$ được gọi là phần bù đại số của a_{ik}.

Nhận xét:

- Định thức cấp một: Nếu $\mathbf{A} = (a_{11})$ thì $\det(\mathbf{A}) = a_{11}$.

- Định thức cấp hai: Nếu $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ thì $\det(\mathbf{A}) = ad - bc$.

Định nghĩa 1.1.8. *[thuan2003] Cho $\mathbf{A}$ là ma trận cỡ $m \times n$ bất kì và s là số nguyên thỏa $1 \leqslant s \leqslant \min(m, n)$. Khi đó, các phần tử nằm trên giao của s dòng và s cột của ma trận $\mathbf{A}$ sẽ lập nên các ma trận vuông cấp s, ta gọi đó chính là các ma trận con cấp s của $\mathbf{A}$. Định thức của các ma trận này được gọi là định thức con cấp s của ma trận $\mathbf{A}$.*

Định nghĩa 1.1.9. *[hieu2019] Định thức con cấp cao nhất, khác 0 của ma trận $\mathbf{A}$ được gọi là định thức con cơ sở của ma trận $\mathbf{A}$. Một ma trận $\mathbf{A}$ có thể có nhiều định thức con cơ sở cùng cấp.*

Hạng của ma trận $\mathbf{A}$ là cấp của định thức con cơ sở. Ký hiệu hạng của ma trận $\mathbf{A}$ là $\text{rank}(\mathbf{A})$.

Nhận xét: Cho $\mathbf{A}$ là ma trận cấp $m \times n, \mathbf{B}$ là ma trận cấp $n \times l$.

(i) Nếu $\text{rank}(\mathbf{B}) = n$ thì $\text{rank}(\mathbf{A} \cdot \mathbf{B}) = \text{rank}(\mathbf{A})$.

(ii) Nếu $\text{rank}(\mathbf{A}) = n$ thì $\text{rank}(\mathbf{A}.\mathbf{B}) = \text{rank}(\mathbf{B})$.

Định nghĩa 1.1.10. *[tri2014] Vết của ma trận vuông $\mathbf{A}$ cấp n được xác định bằng tổng các phần tử trên đường chéo chính của ma trận $\mathbf{A}$, và được ký hiệu là Tr(A).*

1.2 Vector riêng- Giá trị riêng

Định nghĩa 1.2.1. *[thuan2003] Cho* $\mathbf{A}$ *là ma trận vuông cấp* n*. Khi đó đa thức bậc* n *của biến* λ *được xác định như sau*

$$P_{\mathbf{A}}(\lambda) = \det(\mathbf{A} - \lambda\mathbf{I}) = \begin{vmatrix} a_{11} - \lambda & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} - \lambda & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} - \lambda \end{vmatrix}$$

được gọi là đa thức đặc trưng của ma trận $\mathbf{A}$*. Các nghiệm của đa thức* $P_{\mathbf{A}}(\lambda)$ *được gọi là các giá trị riêng của ma trận* $\mathbf{A}$*.*
Vector $\mathbf{u} \in \mathbb{R}^n$ *được gọi là vector riêng ứng với giá trị riêng* λ *của ma trận* $\mathbf{A}$ *nếu thỏa* $\mathbf{Au} = \lambda\mathbf{u}$*.*

Nhận xét:

(i) Nếu λ là một giá trị riêng của $\mathbf{A}$ thì $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$. Khi đó hệ phương trình thuần nhất

$$(\mathbf{A} - \lambda\mathbf{I}) \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = 0$$

có vô số nghiệm.

(ii) Mỗi giá trị riêng có thể có nhiều vector riêng.

(iii) Mỗi vector riêng chỉ ứng với một giá trị riêng duy nhất.

(iv) Nếu $\lambda = 0$ là một giá trị riêng của ma trận $\mathbf{A}$ thì $\mathbf{A}$ không khả nghịch. Ngược lại, nếu mọi giá trị riêng của $\mathbf{A}$ đều khác 0 thì ma trận $\mathbf{A}$ khả nghịch.

1.3 Định lí phổ của ma trận đối xứng

Định nghĩa 1.3.1. *[linh2016] Cho* $\mathbf{A}$ *là ma trận đối xứng cấp* n*. Khi đó:*

(i) Với mỗi giá trị riêng thực λ *của* $\mathbf{A}$*, tồn tại một vector riêng tương ứng* $\mathbf{u} \in \mathbb{R}^n$ *sao cho* $\mathbf{Au} = \lambda\mathbf{u}$*.*

(ii) Tồn tại ma trận đường chéo $\mathbf{D}$ *cấp* n *và ma trận trực giao* $\mathbf{U}$ *cấp* n *sao cho* $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{U}^T$*, trong đó các phần tử nằm trên đường chéo chính của* $\mathbf{D}$ *là các giá trị riêng của* $\mathbf{A}$*, và các vector cột của* $\mathbf{U}$ *là các vetor riêng của* $\mathbf{A}$ *tương ứng với các giá trị riêng đó. Tức là, nếu*

$$\mathbf{D} = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$$

và

$$\mathbf{U} = \left[\begin{array}{c|ccc|c} \mathbf{u}^{(1)} & \mathbf{u}^{(2)} & \ldots & \ldots & \mathbf{u}^{(n)} \end{array} \right]$$

thì $\mathbf{A}\mathbf{u}^{(i)} = \lambda_i \cdot \mathbf{u}^{(i)}, \quad i = 1, 2, \ldots, n.$

Chứng minh. Ta sẽ chứng minh định lí này bằng phương pháp quy nạp toán học. Trường hợp $n = 1$, kết quả trên đúng. Giả sử kết quả trên đúng với mọi ma trận có cấp nhỏ hơn hoặc bằng $n - 1$, ta sẽ chứng minh kết quả trên đúng trong trường hợp ma trận $\mathbf{A}$ là ma trận đối xứng cấp n.

Xét hàm số $p(t) = \det(t\mathbf{I} - \mathbf{A})$. Ta có $p(t)$ là một đa thức bậc n và được gọi là đa thức đặc trưng của ma trận $\mathbf{A}$. Theo Định lí cơ bản của đại số, đa thức $p(t)$ sẽ có n nghiệm là $\lambda_1, \lambda_2, \ldots, \lambda n$, và ta gọi chúng là các giá trị riêng của ma trận $\mathbf{A}$.

Giả sử λ là một giá trị riêng của ma trận $\mathbf{A}$, ta có $\det(\lambda\mathbf{I} - \mathbf{A}) = 0$, tức ma trận $(\lambda\mathbf{I} - \mathbf{A})$ không khả nghịch. Điều này có nghĩa là, tồn tại một vector thực, khác không $\mathbf{u}$ sao cho $\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$. Ta có thể chuẩn hóa vector $\mathbf{u}$ sao cho $\mathbf{u}^T\mathbf{u} = 1$. Khi đó, $\lambda = \mathbf{u}^T\mathbf{A}\mathbf{u}$ là số thực.

Với λ_1 là một giá trị riêng của $\mathbf{A}$ và $\mathbf{u}_1$ là một vector riêng tương ứng. Sử dụng phép trực giao hóa Gramm-Schmidt, ta có thể tìm được ma trận $\mathbf{V}_1$ cấp $n \times (n-1)$ sao cho $[\mathbf{u}_1\mathbf{V}_1]$ là một ma trận trực giao. Ta có $\mathbf{V}_1^T\mathbf{A}\mathbf{V}_1$ là ma trận đối xứng cấp $n - 1$. Khi đó theo giả thiết quy nạp, ta có thể viết $\mathbf{V}_1^T\mathbf{A}\mathbf{V}_1 = \mathbf{Q}_1\mathbf{D}_1\mathbf{Q}_1^T$, trong đó $\mathbf{D}_1 = \operatorname{diag}(\lambda_2, \lambda_3, \ldots, \lambda_n)$ là ma trận đường chéo với các phần tử nằm trên đường chéo chính là $n - 1$ giá trị riêng của $\mathbf{A}$ và $\mathbf{Q}_1$ là ma trận trực giao cấp $(n-1)$ gồm $(n-1)$ vector riêng của $\mathbf{V}_1^T\mathbf{A}\mathbf{V}_1$ tương ứng.

Ta định nghĩa ma trận $\mathbf{U}_1$ cấp $n \times (n-1)$ bởi $\mathbf{U}_1 = \mathbf{V}_1\mathbf{Q}_1$. Khi đó $\mathbf{U} = [\mathbf{u}_1, \mathbf{U}_1]$ là ma trận trực giao. Ta có

$$\mathbf{U}^T\mathbf{A}\mathbf{U} = \begin{pmatrix} \mathbf{u}_1^T \\ \mathbf{U}_1^T \end{pmatrix} \mathbf{A}\left(\mathbf{u}_1 - \mathbf{U}_1\right) = \begin{pmatrix} \mathbf{u}_1^T\mathbf{A}\mathbf{u}_1 & \mathbf{u}_1^T\mathbf{A}\mathbf{U}_1 \\ \mathbf{U}_1^T\mathbf{A}\mathbf{u}_1 & \mathbf{U}_1^T\mathbf{A}\mathbf{U}_1 \end{pmatrix} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \mathbf{D}_1 \end{pmatrix} = \mathbf{D}.$$

Điều này chứng tỏ $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{U}^T$, với $\mathbf{U}$ là ma trận trực giao cấp n và $\mathbf{D} = \text{diag}\left(\lambda_1, \lambda_2, \ldots, \lambda_n\right)$. Đây được gọi là phân tích giá trị riêng của ma trận $\mathbf{A}$ (Eigen value decomposition, EVD). □

1.4 Xác xuất thống kê

Kiến thức trong phần này được tổng hợp từ [**tiep2018**]

1.4.1 Kỳ vọng

Kỳ vọng (expectation) của một biến ngẫu nhiên $\mathbf{x}$ được định nghĩa bởi:

$$\text{E}[\mathbf{x}] = \sum_{\mathbf{x}} \mathbf{x}p(\mathbf{x}) \quad \text{nếu } \mathbf{x} \text{ là rời rạc.} \tag{1.4.1}$$

$$\text{E}[\mathbf{x}] = \int \mathbf{x}p(\mathbf{x})dx \quad \text{nếu } \mathbf{x} \text{ là liên tục.} \tag{1.4.2}$$

Giả sử $f(.)$ là một hàm số trả về một số với mỗi giá trị $\mathbf{x}^*$ của biến ngẫu nhiên $\mathbf{x}$. Khi đó, nếu $\mathbf{x}$ là biến ngẫu nhiên rời rạc, ta có

$$\text{E}[f(\mathbf{x})] = \sum_{\mathbf{x}} f(\mathbf{x})p(\mathbf{x}). \tag{1.4.3}$$

Công thức cho biến ngẫu nhiên liên tục cũng được viết tương tự.
Với xác suất đồng thời, kỳ vọng của một hàm cũng được xác định tương tự:

$$\text{E}[f(x, y)] = \sum_{\mathbf{x},\mathbf{y}} f(\mathbf{x}, \mathbf{y})p(\mathbf{x}, \mathbf{y})dxdy. \tag{1.4.4}$$

Có ba tính chất cần nhớ về kỳ vọng:

1. Kỳ vọng của một hằng số theo một biến ngẫu nhiên $\mathbf{x}$ bất kỳ bằng chính hằng số đó:

$$\text{E}[\alpha] = \alpha. \tag{1.4.5}$$

2. Kỳ vọng có tính chất tuyến tính:

$$\mathrm{E}[\alpha\mathbf{x}] = \alpha\mathrm{E}[\mathbf{x}], \tag{1.4.6}$$

$$\mathrm{E}[f(\mathbf{x}) + g(\mathbf{x})] = \mathrm{E}[f(\mathbf{x})] + \mathrm{E}[g(\mathbf{x})]. \tag{1.4.7}$$

3. Kỳ vọng của tích hai biến ngẫu nhiên độc lập bằng tích kỳ vọng của chúng:

$$\mathrm{E}[f(\mathbf{x})g(\mathbf{y})] = \mathrm{E}[f(\mathbf{x})]\mathrm{E}[g(\mathbf{y})]. \tag{1.4.8}$$

Khái niệm kỳ vọng thường đi kèm với khái niệm *phương sai* (variance) trong không gian một chiều và *ma trận hiệp phương sai* (covariance matrix) trong không gian nhiều chiều.

1.4.2 Phương sai

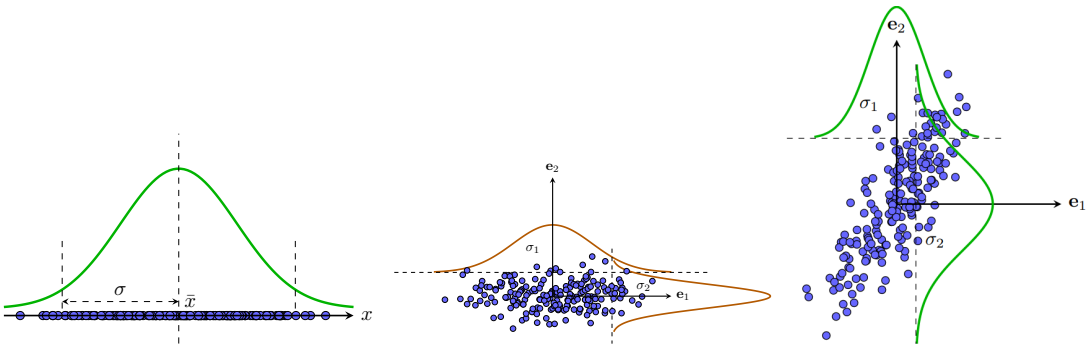
Cho N giá trị $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$. Kỳ vọng và phương sai của bộ dữ liệu này được tính theo công thức

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n = \frac{1}{N}\mathbf{x}\mathbf{1}, \tag{1.4.9}$$

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})^2, \tag{1.4.10}$$

với $\mathbf{x} = [x_1, x_2, \ldots, x_N]$, và $\mathbf{1} \in \mathbb{R}^N$là vector cột chứa toàn phần tử 1. Kỳ vọng đơn giản là trung bình cộng của toàn bộ các giá trị. Phương sai là trung bình cộng của bình phương khoảng cách từ mỗi điểm tới kỳ vọng. Phương sai càng nhỏ, các điểm dữ liệu càng gần với kỳ vọng, tức các điểm dữ liệu càng giống nhau. Phương sai càng lớn, dữ liệu càng có tính phân tán. Ví dụ về kỳ vọng và phương sai của dữ liệu một chiều có thể được thấy trong Hình 1.1a.

Căn bậc hai của phương sai, σ còn được gọi là *độ lệch chuẩn* (standard deviation) của dữ liệu.



(a) (b) (c)

Hình 1.1: Ví dụ về kỳ vọng và phương sai. (a) Dữ liệu trong không gian một chiều. (b) Dữ liệu trong không gian hai chiều mà hai chiều không tương quan. Trong trường hợp này, ma trận hiệp phương sai là ma trận đường chéo với hai phần tử trên đường chéo là σ_1, σ_2, đây cũng chính là hai trị riêng của ma trận hiệp phương sai và là phương sai của mỗi chiều dữ liệu. (c) Dữ liệu trong không gian hai chiều có tương quan. Theo mỗi chiều, ta có thể tính được kỳ vọng và phương sai. Phương sai càng lớn thì dữ liệu trong chiều đó càng phân tán. Trong ví dụ này, dữ liệu theo chiều thứ hai phân tán nhiều hơn so với chiều thứ nhất.

1.4.3 Ma trận hiệp phương sai

Cho N điểm dữ liệu được biểu diễn bởi các vector cột $\mathbf{x}_1, \ldots, \mathbf{x}_N$, khi đó, vector kỳ vọng và ma trận hiệp phương sai của toàn bộ dữ liệu được định nghĩa là

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n, \tag{1.4.11}$$

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T = \frac{1}{N}\hat{\mathbf{X}}\hat{\mathbf{X}}^T. \tag{1.4.12}$$

Trong đó $\hat{\mathbf{X}}$ được tạo bằng cách trừ mỗi cột của $\mathbf{X}$ đi $\bar{\mathbf{x}}$:

$$\hat{\mathbf{x}}_n = \mathbf{x}_n - \bar{\mathbf{x}}. \tag{1.4.13}$$

Một vài tính chất của ma trận hiệp phương sai:

1. Ma trận hiệp phương sai là một ma trận đối xứng, hơn nữa, nó là một ma trận.

2. Mọi phần tử trên đường chéo của ma trận hiệp phương sai là các số không âm. Chúng chính là phương sai của từng chiều dữ liệu.

3. Các phần tử ngoài đường chéo $s_{ij}, i \neq j$ thể hiện sự tương quan giữa thành phần thứ i và thứ j của dữ liệu, còn được gọi là hiệp phương sai. Giá trị này có thể dương, âm hoặc bằng không. Khi nó bằng không, ta nói rằng hai thành phần i, j trong dữ liệu là *không tương quan.*

4. Nếu ma trận hiệp phương sai là ma trận đường chéo, ta có dữ liệu hoàn toàn không tương quan giữa các chiều.

Ví dụ về sự tương quan của dữ liệu được cho trong Hình 1.1b và 1.1c.

Chương 2

PHÂN TÍCH GIÁ TRỊ KÌ DỊ

Trong chương này, chúng tôi trình bày cơ sở của phân tích giá trị kì dị của ma trận, thuật toán phân tích giá trị kì dị, và trình bày một số tính chất của ma trận liên quan đến phân tích giá trị kì dị của nó.

2.1 Giới thiệu

Phân tích giá trị kì dị (SVD) là một trong những phân tích rất quan trọng của ma trận, có nhiều ứng dụng trong khoa học và kĩ thuật. Dạng cổ điển nhất của phân tích giá trị kì dị được phát hiện trong lĩnh vực Hình học vi phân. Vào năm 1873 và 1874 , hai nhà toán học Eugenio Beltrami và Camille Jordan đã độc lập đưa ra giá trị kì dị của dạng song tuyến tính[**stewart1993**]. Năm 1889, James Joshept Sylvester đã đưa ra phân tích giá trị kì dị của ma trận vuông. Đến năm 1915, nhà toán học Autonne phát hiện ra phân tích giá trị kì dị dưới dạng phân tích cực. Năm 1936, hai nhà toán học Carl Eckart và Gale Young đã lần đầu tiên chứng minh phân tích giá trị kì dị đối với ma trận hình chữ nhật thực và ma trận vuông phức[**eckart1936**]. Trong chương này, chúng tôi sẽ tìm hiểu về phân tích giá trị kì dị cho một ma trận hình chữ nhật thực $\mathbf{A}$ cỡ $m \times n$.

2.2 Phân tích giá trị kì dị

Để phát biểu và chứng minh định lý phân tích giá trị kì dị, chúng ta sẽ cần định lí sau đây.

Định lý 2.2.1. *[lay1994] Cho A là một ma trận có cấp $m \times n$. Khi đó, mọi giá trị riêng của ma trận $\mathbf{A}^T\mathbf{A}$ đều không âm.*

Chứng minh. Dễ thấy $\mathbf{A}^T\mathbf{A}$ là một ma trận đối xứng. Theo định lí 1.3.1, mọi giá trị riêng của $\mathbf{A}^T\mathbf{A}$ đều là số thực. Với mỗi giá trị riêng λ của $\mathbf{A}^T\mathbf{A}$, giả sử $\mathbf{v}$ là véc tơ riêng tương ứng. Khi đó

$$\mathbf{A}^T\mathbf{A}\mathbf{v} = \lambda\mathbf{v} \tag{2.2.1}$$

Để ý rằng ta có thể chọn $\mathbf{v}$ là véc tơ đơn vị, i.e. $\|\mathbf{v}\| = 1$. Với cách chọn như vậy, ta nhận được

$$\|\mathbf{A}\mathbf{v}\|^2 = \langle \mathbf{A}\mathbf{v}, \mathbf{A}\mathbf{v}\rangle = (\mathbf{A}\mathbf{v})^T(\mathbf{A}\mathbf{v}) = \mathbf{v}^T\mathbf{A}^T\mathbf{A}\mathbf{v} = \lambda\mathbf{v}^T\mathbf{v} = \lambda\|\mathbf{v}\|^2 = \lambda \tag{2.2.2}$$

Do vậy, $\lambda \geqslant 0$. □

Ta có định nghĩa và cách xác định các giá trị kì dị của một ma trận như sau

Định nghĩa 2.2.1. Cho ma trận $\mathbf{A}$ cỡ $m \times n$ bất kì ($m \leqslant n$) và $\lambda_1, \lambda_2 \ldots, \lambda_r$, $1 \leqslant r \leqslant \min(m, n)$, lần lượt là các giá trị riêng của ma trận $\mathbf{A}^T\mathbf{A}$. Các giá trị $\sigma_i = \sqrt{\lambda_i}, 1 \leqslant i \leqslant r$, được gọi là các giá trị kì dị của ma trận $\mathbf{A}$.

Định lí sau cho ta sự tồn tại của SVD của một ma trận bất kỳ

Định lý 2.2.2. *[lay1994] Mọi ma trận* A *cỡ $m \times n$ bất kì đều có phân tích SVD có dạng*

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T \tag{2.2.3}$$

trong đó $\mathbf{U}$ *và* $\mathbf{V}$ *lần lượt là các ma trận trực giao cấp m và n,*
$\mathbf{D}$ *là ma trận đường chéo cỡ $m \times n$ có dạng*

$$\mathbf{D} = \begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \dots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & \sigma_r & 0 & \dots & 0 \\ 0 & \dots & \dots & 0 & 0 & \dots & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{D}_{r\times r} & 0 & \dots & 0 \\ 0 & \dots & \dots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \dots & \dots & 0 \end{bmatrix}$$

Chứng minh. Vì $\mathbf{A}^T\mathbf{A}$ là ma trận đối xứng nên theo Định lí 2.2.1 n giá trị riêng $\lambda_1, \dots, \lambda_n$ của nó đều không âm. Do đó tồn tại $r \leqslant n$ sao cho $\lambda_1 \geqslant \lambda_2 \geqslant \lambda_r > 0$ và $\lambda_j = 0$ với $j > r$. Khi đó ma trận $\mathbf{A}$ có các giá trị kì dị là $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_r \geqslant 0$, và $\sigma_j = 0$ với $j > r$.

Chọn $\mathbf{v}_i \in \mathbb{R}^n, i = 1, 2, \dots, n$, là các vetor riêng tương ứng với λ_i sao cho $\mathbf{v}_i$ là các vetor đơn vị. Đặt $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n)$, với $\mathbf{v}_i$ là các vector cột. Khi đó $\mathbf{V}$ là ma trận trực giao.

Với $\sigma_i, 1 \leqslant i \leqslant r$, là các giá trị kì dị dương của $\mathbf{A}$, đặt $\mathbf{u}_i = \frac{\mathbf{A}\mathbf{v}_i}{\sigma_i}, i = 1, 2, \dots, r$. Khi đó $\mathbf{u}_i$ là vector đơn vị trong $\mathbb{R}^m$.

Xây dựng ma trận $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_m)$, với $\mathbf{u}_j = 0, r < j \leqslant m$. Ta có $\mathbf{U}$ là ma trận trực giao.

Ta chứng minh $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$, hay $\mathbf{A}\mathbf{V} = \mathbf{U}\mathbf{D}$.

Ta có

$$\begin{aligned} \mathbf{A}\mathbf{V} &= \mathbf{A}\left[\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\right] = \left[\mathbf{A}\mathbf{v}_1, \mathbf{A}\mathbf{v}_2, \dots, \mathbf{A}\mathbf{v}_n\right] \\ &= \left[\mathbf{A}\mathbf{v}_1, \mathbf{A}\mathbf{v}_2, \dots, \mathbf{A}\mathbf{v}_r, 0, \dots, 0\right] \\ &= \left[\sigma_1\mathbf{u}_1, \sigma_2\mathbf{u}_2, \dots, \sigma_r\mathbf{u}_r, 0, \dots, 0\right] = \left[\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_m\right] \cdot \operatorname{diag}\left(\sigma_1, \sigma_2, \dots, \sigma_r\right) \\ &= \mathbf{U}\mathbf{D} \end{aligned}$$

Vậy $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$

Trong trường hợp $\operatorname{rank}(\mathbf{A}) = r < n$, thì SVD của $\mathbf{A}$ có dạng "chặt cụt" như sau:

$$\mathbf{A} = \mathbf{U}_r\mathbf{D}_r\mathbf{V}_r^T$$

trong đó $\mathbf{U}_r, \mathbf{V}_r$ lần lượt là ma trận được tạo bởi r cột đầu tiên của $\mathbf{U}, \mathbf{V}$, và $\mathbf{D}_r$ là ma trận con cấp $r \times r$ được tạo bởi r hàng đầu tiên và r cột đầu tiên của $\mathbf{D}$. Khai triển như trên được gọi là phân tích SVD "chặt cụt" của $\mathbf{A}$. □

Định lý 2.2.3. *[linh2016] (**Về dạng khai triển của phân tích giá trị kì dị**) Cho* $\mathbf{A}$ *là ma trận cỡ* $m \times n$ *bất kì. Khi đó*

$$\mathbf{A} = \mathbf{UDV}^T = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_r \mathbf{u}_r \mathbf{v}_r^T \tag{2.2.4}$$

trong đó σ_i *là các giá trị kì dị dương của ma trận* $\mathbf{A}, \mathbf{u}_i$ *là các vector kì dị trái và* $\mathbf{v}_i$ *là các vector kì dị phải của ma trận* $\mathbf{A}$*.*

Chứng minh. Ta có

$$\mathbf{UDV}^T = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m] \begin{bmatrix} \mathbf{D}_{r\times r} & \ldots & 0 \\ \vdots & \ldots & \vdots \\ 0 & \ldots & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_n^T \end{bmatrix}$$

$$= [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r, \mathbf{u}_{r+1}, \ldots, \mathbf{u}_m] \begin{bmatrix} \mathbf{D}_{r\times r} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_r^T \\ \mathbf{v}_{r+1}^T \\ \vdots \\ \mathbf{v}_n^T \end{bmatrix}$$

$$= [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r] \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_r \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_r^T \end{bmatrix} + \begin{bmatrix} \mathbf{u}_{r+1} & \mathbf{u}_{r+2} & \ldots & \mathbf{u}_m \end{bmatrix} [0] \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_n^T \end{bmatrix}$$

$$= [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r] \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_r \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_r^T \end{bmatrix}$$

$$= [\sigma_1 \mathbf{u}_1, \ldots, \sigma_r \mathbf{u}_r] \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_r^T \end{bmatrix} = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_r \mathbf{u}_r \mathbf{v}_r^T.$$

□

2.3 Thuật toán tìm SVD của một ma trận

Cho $\mathbf{A}$ là ma trận cỡ $m \times n$, với $m \geq n$. Để tìm SVD của ma trận $\mathbf{A}$, chúng ta thực hiện các bước sau:

- Bước 1. Tính ma trận $\mathbf{A}^T\mathbf{A}$ và giải phương trình $\det\left(\mathbf{A}^T\mathbf{A} - \lambda\mathbf{I}\right) = 0$ để tìm các giá trị riêng $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$ của ma trận $\mathbf{A}^T\mathbf{A}$. Từ đó suy ra các giá trị kì dị của $\mathbf{A}$ là $\sigma_i = \sqrt{\lambda_i}, i = \overline{1,n}$ và $\mathbf{D} = \text{diag}\left(\sigma_1, \sigma_2, \ldots, \sigma_n\right)$

- Bước 2. Tương ứng với mỗi giá trị riêng λ_i, tìm véctơ riêng $\mathbf{v}_i \in \mathbb{R}^n$ sao cho $\left(\mathbf{A}^T\mathbf{A} - \lambda\mathbf{I}\right) v_i = 0$. Từ đó tìm được ma trận trực giao $\mathbf{V}$ cấp n chứa các véctơ kì dị phải của $\mathbf{A}$.

- Bước 3. Xác định các véctơ kì dị trái của $\mathbf{A}$ theo công thức

$$\mathbf{u}_i = \frac{1}{\sigma_i}\mathbf{A}\mathbf{v}_i$$

  Bổ sung $n-r$ vector $\mathbf{u}_{r+1}, \ldots, \mathbf{u}_n$ vào hệ $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r\}$ sao cho $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n\}$ lập thành một cơ sở trực chuẩn của $\mathbb{R}^n$. Từ đó nhận được ma trận trực giao $\mathbf{U}$ chứa các véctơ kì dị trái của $\mathbf{A}$, và

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

  là phân tích SVD của ma trận $\mathbf{A}$.

Cài đặt thuật toán trên PYTHON3 có thể xem Hình 2.1

```python
def svd(A):
    # Compute the eigenvectors and eigenvalues of A.T @ A
    w, V = LA.eig(np.nan_to_num(A.T @ A))
    # Calculate sigma values (singular values)
    sigma = np.sqrt(np.abs(w))
    # Sort singular values and corresponding columns
    sorted_indices = np.argsort(sigma)[::-1]
    sigma = sigma[sorted_indices]
    V = V[:, sorted_indices]
    # Create the diagonal matrix D using singular values
    D = np.diag(sigma)
    # Calculate the U matrix
    U = np.zeros((A.shape[0], len(sigma)))
    for i in range(len(sigma)):
        if np.isclose(sigma[i], 0):
            U[:, i] = 0
        else:
            U[:, i] = (1 / sigma[i]) * A @ V[:, i]
    return U, D, V.T
```

Hình 2.1: Cài đặt thuật toán SVD trên PYTHON3

Ví dụ

Tìm SVD của ma trận $\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$.

Lời giải:

Bước 1: Tìm các giá trị kì dị củ ma trận $\mathbf{A}$

Ta có

$$\mathbf{A}^T\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

Giải phương trình $\det\left(\mathbf{A}^T\mathbf{A} - \lambda\mathbf{I}\right) = 0$, ta tìm đượccác giá trị riêng λ của $\mathbf{A}^T\mathbf{A}$ là $\lambda_1 = 2, \lambda_2 = 1$. Do đó các giá trị kì dị của $\mathbf{A}$ là $\sigma_1 = \sqrt{2}, \sigma_2 = 1$. Suy ra

$$\mathbf{D} = \begin{bmatrix} \sqrt{2} & 0 \\ 0 & 1 \end{bmatrix}$$

Bước 2: Tìm ma trận $\mathbf{V}$

Giải phương trình $\left(\mathbf{A}^T\mathbf{A} - \lambda\mathbf{I}\right)\mathbf{v} = 0$ ta tìm được các véctơ riêng tương ứng là $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ và $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

Từ đó suy ra

$$\mathbf{V}^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Bước 3: Tìm ma trận $\mathbf{U}$

Ta có

$$\mathbf{u}_1 = \frac{1}{\sigma_1}\mathbf{A}\mathbf{v}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix}$$

và

$$\mathbf{u}_2 = \frac{1}{\sigma_2}\mathbf{A}\mathbf{v}_1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Do đó

$$\mathbf{U} = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 \\ 0 & 1 \end{bmatrix}$$

Vậy phân tích SVD của ma trận $\mathbf{A}$ là

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Bên dưới là ví dụ về SVD trên PYTHON3 Hình 2.2.

```python
# Example usage:
A = np.array([[1, 0],
              [1, 0],
              [0, 1]])
U, D, Vt =svd(A)
print("SVD:")
print("U:", U)
print("D:", D)
print("Vt:", Vt)
```

```
SVD:
U: [[0.70710678 0.        ]
 [0.70710678 0.        ]
 [0.         1.        ]]
D: [[1.41421356 0.        ]
 [0.         1.        ]]
Vt: [[1. 0.]
 [0. 1.]]
```

Hình 2.2: Ví dụ SVD trên PYTHON3

2.4 Một số tính chất của ma trận liên quan đến SVD của nó

Định lý 2.4.1. *[linh2016] Cho ma trận* $\mathbf{A}$ *cỡ* $m \times n$ *bất kì, có phân tích SVD là*

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

Khi đó, hạng của $\mathbf{A}$ *đúng bằng số các giá trị kì dị dương của* $\mathbf{A}$.

Chứng minh. Giả sử r là số các giá trị kì dị dương của $\mathbf{A}$. Đặt $\mathbf{U}_r = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r]$, $\mathbf{V}_r = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r]$. Do tính chất của hạng của ma trận và tính trực giao của $\mathbf{U}, \mathbf{V}$ ta suy ra

$$\operatorname{rank}(\mathbf{U}_r) = \operatorname{rank}\left(\mathbf{U}_r\mathbf{U}_r^T\right) = \operatorname{rank}(\mathbf{I}_r) = r,$$
$$\operatorname{rank}\left(\mathbf{V}_r^T\right) = \operatorname{rank}\left(\mathbf{V}_r^T\mathbf{V}_r\right) = \operatorname{rank}(\mathbf{I}_r) = r.$$

Do đó

$$\operatorname{rank}(\mathbf{A}) = \operatorname{rank}\left(\mathbf{U}\mathbf{D}\mathbf{V}^T\right) = \operatorname{rank}\left(\mathbf{U}_r\mathbf{D}\mathbf{V}_r^T\right) = \operatorname{rank}\left(\mathbf{D}\mathbf{V}_r^T\right) = \operatorname{rank}(\mathbf{D}) = r$$

□

Định lý 2.4.2. *[linh2016] Cho* $\mathbf{A}$ *là một ma trận có cỡ* $m \times n$*. Giả sử* $\mathbf{A}$ *có phân tích* SVD *dưới dạng khai triển là*

$$\mathbf{A} = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \cdots + \sigma_r \mathbf{u}_r \mathbf{v}_r^T$$

Khi đó, với mỗi số nguyên dương $k < r$*, ma trận*

$$\mathbf{A}_k = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T$$

có hạng $k, \operatorname{rank}(\mathbf{A}_k) = k$*.*

Chứng minh. Dễ thấy rằng ma trận $\mathbf{A}_k$ có thể viết ở dạng

$$\mathbf{A}_k = \left(\sigma_1 \mathbf{v}_1^T + \cdots + \sigma_k \mathbf{v}_k^T\right) \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_k \end{bmatrix}$$

Ta có

$$\begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_k \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_k \end{bmatrix}^T = \mathbf{I}_k \text{ và } \operatorname{rank} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_k \end{bmatrix} = \operatorname{rank} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_k \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_k \end{bmatrix}^T = k.$$

Xét $\operatorname{rank}\left(\sigma_1 \mathbf{v}_1^T, \sigma_2 \mathbf{v}_2^T, \ldots, \sigma_k \mathbf{v}_k^T\right)$. Ta có

$$\left(\mathbf{v}_1^T, \mathbf{v}_2^T, \ldots, \mathbf{v}_k^T\right)(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k) = \mathbf{I}_k$$

Do đó

$$\left(\sigma_1 \mathbf{v}_1^T, \ldots, \sigma_k \mathbf{v}_k^T\right)(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k) = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_k \end{bmatrix} \mathbf{I}_k = \begin{bmatrix} \sigma_1 & 0 & \ldots & 0 \\ 0 & \sigma_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \sigma_k \end{bmatrix} = \mathbf{B}$$

Vì $\text{rank}(\mathbf{B}) = k = \text{rank}\,(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k)$ nên

$$\text{rank}\left(\sigma_1 \mathbf{v}_1^T, \sigma_2 \mathbf{v}_2^T, \ldots, \sigma_k \mathbf{v}_k^T\right) = k$$

Từ đó, ta suy ra $\text{rank}\,(\mathbf{A}_k) = k$. □

Định lý 2.4.3. *[linh2016] Cho* $\mathbf{A}$ *là ma trận cỡ* $m \times n$ *bất kì và* $\sigma_1, \sigma_2, \ldots, \sigma_r$ *là các giá trị kì dị dương của* $\mathbf{A}$*. Khi đó* $\|\mathbf{A}\|_F = \sqrt{\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_r^2}$.

Chứng minh. Giả sử $\mathbf{Q}$ là ma trận trực giao cấp m. Khi đó

$$\begin{aligned}\|\mathbf{QA}\|_F^2 &= \|[\mathbf{Qa}_1, \ldots, \mathbf{Qa}_n]\|_F^2 \\ &= \|\mathbf{Q_1}\|_F^2 + \cdots + \|\mathbf{Qa}_n\|_F^2 \\ &= \|\mathbf{a}_1\|_F^2 + \cdots + \|\mathbf{a}_n\|_F^2 = \|\mathbf{A}\|_F^2.\end{aligned}$$

Giả sử $\mathbf{A}$ có phân tích SVD là $\mathbf{A} = \mathbf{UDV}^T$. Vì $\mathbf{U}$ và $\mathbf{V}$ là các ma trận trực giao nên

$$\|\mathbf{A}\|_F^2 = \left\|\mathbf{UDV}^T\right\|_F^2 = \left\|\mathbf{DV}^T\right\|_F^2 = \left\|\left(\mathbf{DV}^T\right)^T\right\|_F^2 = \left\|\mathbf{VD}^T\right\|_F^2 = \left\|\mathbf{D}^T\right\|_F^2 = \sigma_1^2+\sigma_2^2+\cdots+\sigma_r^2$$

Vậy

$$\|\mathbf{A}\|_F^2 = \sqrt{\sigma_1^2 + \cdots + \sigma_r^2}$$

với $\sigma_i, i = 1, 2, \ldots, r$ là các giá trị kì dị của ma trận $\mathbf{A}$. □

Chương 3

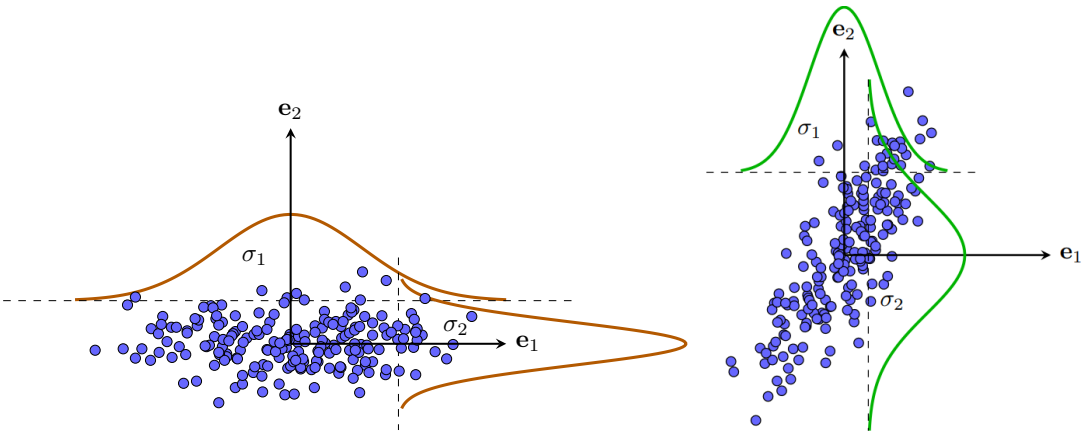
Phân tích thành phần chính

3.1 Giới thiệu

Phép phân tích thành phần chính (Principal Components Analysis - PCA) là một phương pháp phân tích ma trận đa biến, giúp giảm số chiều của dữ liệu bằng cách tìm ra các thành phần chính của ma trận đó. Ý tưởng của PCA là chuyển đổi dữ liệu ban đầu từ không gian có số chiều cao sang không gian có số chiều thấp hơn, giúp cho việc xử lý và phân tích dữ liệu dễ dàng hơn. Phép phân tích thành phần chính được đưa ra vào năm 1901 bởi Karl Pearson và đã được phát triển và ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau như nhận dạng hình ảnh, phân tích tín hiệu, dự báo kinh tế và thị trường tài chính.

Một trong những ứng dụng quan trọng nhất của phép phân tích thành phần chính là giảm số chiều của dữ liệu. Khi dữ liệu có số chiều lớn, việc phân tích và xử lý trở nên phức tạp và tốn nhiều thời gian. Đồng thời cho phép giảm số chiều của dữ liệu bằng cách xác định các thành phần chính của ma trận, giúp cho việc xử lý và phân tích dữ liệu trở nên đơn giản và nhanh chóng hơn.

Ngoài ra, phép phân tích này còn được sử dụng để giảm tác động của nhiễu trong dữ liệu, phát hiện các mối quan hệ giữa các biến đầu vào và giúp tối ưu hóa các thuật toán máy học.Với những ứng dụng và lợi ích của mình, phép phân tích thành phần chính đã trở thành một công cụ quan trọng trong phân tích dữ liệu và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau. Nội dung trong chương này được tổng hợp chính từ các nguồn [**my2020**], [**tiep2018**] và [**hyvarinen2009**]



(a) (b)

Hình 3.1: Ví dụ về phương sai của dữ liệu trong không gian hai chiều. (a) Phương sai của chiều thứ hai (tỉ lệ với độ rộng của đường hình chuông) nhỏ hơn phương sai của chiều thứ nhất. (b) Cả hai chiều có phương sai đáng kể. Phương sai của mỗi chiều là phương sai của thành phần tương ứng được lấy trên toàn bộ dữ liệu. Phương sai tỉ lệ thuận với độ phân tán của dữ liệu.

3.2 Ý tưởng

Giả sử dữ liệu ban đầu là $\mathbf{x} \in \mathbb{R}^m$ và dữ liệu đã được giảm chiều là $\mathbf{z} \in \mathbb{R}^r$ với $r < m$. Cách đơn giản nhất để giảm chiều dữ liệu từ m về $r < m$ là chỉ cần giữ lại r phần tử quan trọng nhất. Có hai câu hỏi được đặt ra ở đây:

(i) Làm thế nào để xác định tầm quan trọng của mỗi chiều dữ liệu?

(ii) Nếu tầm quan trọng của các chiều dữ liệu là như nhau, ta cần bỏ đi chiều nào?

Giả sử vector dữ liệu ban đầu $\mathbf{x} \in \mathbb{R}^D$ được giảm chiều trở thành $\mathbf{z} \in \mathbb{R}^K$ với $K < D$. Một cách đơn giản để giảm chiều dữ liệu từ D về $K < D$ là chỉ giữ lại K phần tử quan trọng nhất. Có hai câu hỏi lập tức được đặt ra. Thứ nhất, làm thế nào để xác định tầm quan trọng của mỗi phần tử? Thứ hai, nếu tầm quan trọng của các phần tử là như nhau, ta cần bỏ đi những phần tử nào?

Để trả lời câu hỏi thứ nhất, hãy quan sát Hình 3.1a. Giả sử các điểm dữ liệu có thành phần thứ hai (phương đứng) giống hệt nhau hoặc sai khác nhau không đáng kể (phương sai nhỏ). Khi đó, thành phần này hoàn toàn có thể được lược bỏ, và ta ngầm

hiểu rằng nó sẽ được xấp xỉ bằng kỳ vọng của thành phần đó trên toàn bộ dữ liệu. Ngược lại, nếu áp dụng phương pháp này lên chiều thứ nhất (phương ngang), lượng thông tin bị mất đi đáng kể do sai số xấp xỉ quá lớn. Vì vậy, lượng thông tin theo mỗi thành phần có thể được đo bằng phương sai của dữ liệu trên thành phần đó. Tổng lượng thông tin là tổng phương sai trên toàn bộ các thành phần. Lấy một ví dụ về việc có hai camera được đặt dùng để chụp cùng một người, một camera phía trước và một camera đặt trên đầu. Rõ ràng, hình ảnh thu được từ camera đặt phía trước mang nhiều thông tin hơn so với hình ảnh nhìn từ phía trên đầu. Vì vậy, bức ảnh chụp từ phía trên đầu có thể được bỏ qua mà không làm mất đi quá nhiều thông tin về hình dáng của người đó.

Câu hỏi thứ hai tương ứng với trường hợp Hình 3.1b. Trong cả hai chiều, phương sai của dữ liệu đều lớn; việc bỏ đi một trong hai chiều đều khiến một lượng thông tin đáng kể bị mất đi. Tuy nhiên, nếu xoay trục toạ độ đi một góc phù hợp, một trong hai chiều dữ liệu có thể được lược bở vì dữ liệu có xu hướng phân bố xung quanh một đường thẳng.

Phân tích thành phần chính (principle component analysis, PCA) là phương pháp đi tìm một phép xoay trục toạ độ để được một hệ trục toạ độ mới sao cho trong hệ mới này, thông tin của dữ liệu chủ yếu tập trung ở một vài thành phần. Phần còn lại chứa ít thông tin hơn có thể được lược bỏ.

3.3 Phân tích thành phần chính

Về mặt hình thức, cho ma trận dữ liệu $\mathbf{X} \in \mathbb{R}^{m \times n}$ với mỗi giá trị đã được chuẩn hóa sao cho mỗi cột có giá trị trung bình là 0. PCA của $\mathbf{X}$ với r thành phần là tìm ma trận trực giao $\mathbf{A} \in \mathbb{R}^{m \times r}$ và ma trận $\mathbf{B} \in \mathbb{R}^{n \times r}$ sao cho $\mathbf{X}$ có thể biểu diễn được dưới dạng ma trận

$$\mathbf{X} = \mathbf{A}\mathbf{B}^T + \mathbf{E}$$

trong đó $\|\mathbf{E}\|^2$ đạt giá trị nhỏ nhất. Ma trận $\mathbf{A}$ được gọi là ma trận thành phần, và các cột của nó là các thành phần chính. Ma trận $\mathbf{B}$ được gọi là ma trận tải, các tải trọng là các trọng số cho phép tái cấu trúc các biến ban đầu dưới dạng tổ hợp tuyến

tính của các thành phần chính. Cặp $(\mathbf{A}, \mathbf{B})$ được gọi là nghiệm PCA. Ngoài ra, PCA được trình bày theo một cách khác dưới dạng véctơ như sau:

$$\mathbf{X} = \sum_{i=1}^{r} \mathbf{a}_i \mathbf{b}_i^T + \mathbf{E}$$

Diều này cho thấy PCA là xấp xỉ $\mathbf{X}$ với tổng của r ma trận có hạng 1. Mục tiêu của PCA là làm giảm tối thiểu $\|\mathbf{E}\|^2 = \left\|\mathbf{X} - \mathbf{A}\mathbf{B}^T\right\|^2$. Vì rank $\left(\mathbf{A}\mathbf{B}^T\right) \leq r$ nên $\mathbf{A}\mathbf{B}^T$ là SVD "chặt cụt" của $\mathbf{X}$, nghĩa là nếu $\mathbf{U}_r\mathbf{D}_r\left(\mathbf{V}_r\right)^T$ là SVD "chặt cụt" của $\mathbf{X}$ thì $\mathbf{A} = \mathbf{U}_r$ và $\mathbf{B}^T = \mathbf{D}_r\left(\mathbf{V}_r\right)^T$. Số lượng thành phần tối thiểu phù hợp là số giá trị kì dị khác 0 của $\mathbf{X}$, hay là hạng của $\mathbf{X}$. Vì vậy, không cần thiết phải lấy số thành phần r lớn hơn số lượng biến n. Trên thực tế, r thường được lấy nhỏ hơn nhiều so với n.

Do thứ tự giảm dần của các giá trị kì dị của $\mathbf{X}$ trong SVD, thành phần chính đầu tiên giải thích phương sai nhiều nhất có thể và từng thành phần chính tiếp theo giải thích phương sai với ràng buộc là nó trực giao (không tương quan với các thành phần trước đó). Khi đó tổng phương sai được giải thích là $\mathrm{Tr}\left(\mathbf{B}\mathbf{B}^T\right) = \mathrm{Tr}\left(m^{-\frac{1}{2}}\mathbf{D}_r^2\right)$ với $\mathrm{Tr}\left(\mathbf{D}_r^2\right)$ là tổng bình phương của r giá trị kì dị lớn nhất của $\mathbf{X}$.

3.4 Tìm các thành phần chính của bài toán PCA thông qua SVD

Xét một vector $\mathbf{x}$ bất kì. Thành phần chính là tổ hợp tuyến tính $\mathbf{s} = \sum_{i=1}^{m} \mathbf{w}_i\mathbf{x}_i$ có chứa càng nhiều phương sai của dữ liệu đầu vào càng tốt. Như vậy, thành phần chính đầu tiên được định nghĩa bằng trực giác là tổ hợp tuyến tính của các biến quan sát, trong đó có phương sai lớn nhất.

Chúng ta cần đưa ra ràng buộc cho chuẩn của vectơ $\mathbf{w} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m)$. Để đơn giản, chúng ta ràng buộc $\mathbf{w}$ có chuẩn bằng 1 , tức là

$$\|\mathbf{w}\| = \sqrt{\sum_{i=1}^{m} \mathbf{w}_i^2} = 1$$

Các ràng buộc khác về giá trị chuẩn của $\mathbf{w}$ chúng ta có thể đưa về ràng buộc trên. Chú ý rằng phương sai của một tổ hợp tuyến tính bất kì đều có thể được tính thông qua

ma trận hiệp phương sai của dữ liệu. Xét một tổ hợp tuyến tính $\mathbf{w}^T\mathbf{x} = \sum_{i=1}^{m} w_i x_i$. Giả sử giá trị trung bình bằng 0 , tức là $E\{\mathbf{x}\} = 0$. Khi đó

$$\begin{aligned} E\left\{\left(\mathbf{w}^T\mathbf{x}\right)^2\right\} &= E\left\{\left(\mathbf{w}^T\mathbf{x}\right)\left(\mathbf{w}^T\mathbf{x}\right)\right\} \\ &= E\left\{\mathbf{w}^T\left(\mathbf{x}\mathbf{x}^T\right)\mathbf{w}\right\} \\ &= \mathbf{w}^T E\left\{\mathbf{x}\mathbf{x}^T\right\} \\ &= \mathbf{w}^T\mathbf{C}\mathbf{w} \end{aligned}$$

trong đó $\mathbf{C} = E\left\{\mathbf{x}\mathbf{x}^T\right\}$ là ma trận hiệp phương sai. Vì vậy, bài toán cơ bản PCA được xác định như sau:

$$\max_{\mathbf{w}:\|\mathbf{w}\|=1} \mathbf{w}^T\mathbf{C}\mathbf{w}$$

Vì $\mathbf{C}$ là ma trận đối xứng nên theo định lí phổ của ma trận đối xứng, tồn tại ma trận trực giao $\mathbf{U} \in \mathbb{R}^{m\times n}$ và ma trận đường chéo $\mathbf{D} = \operatorname{diag}(\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^{n\times n}$ sao cho $\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{U}^T$, trong đó $\lambda_1, \ldots, \lambda_n$ là các giá trị riêng của $\mathbf{C}$, và các vector cột của $\mathbf{U}$ là các véctơ riêng của $\mathbf{C}$ ứng với giá trị riêng đó. Thực hiện đổi biến $\mathbf{v} = \mathbf{U}^T\mathbf{w}$. Khi đó ta nhận được

$$\mathbf{w}^T\mathbf{C}\mathbf{w} = \mathbf{w}^T\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{w} = \mathbf{v}^T\mathbf{D}\mathbf{v} = \sum_{i=1}^{n} \mathbf{v}_i^2\lambda_i$$

Vì $\mathbf{U}$ trực giao nên $\|\mathbf{v}\| = \|\mathbf{w}\|$, do đó, $\mathbf{v}$ có ràng buộc $\|\mathbf{v}\| = 1$. Tiếp tục thực hiện phép đổi biến $\mathbf{m}_i = \mathbf{v}_i^2, i = 1, \ldots, n$. Khi đó ràng buộc $\|\mathbf{v}\| = 1$ tương đương với ràng buộc $\mathbf{m}_i \geq 0$ và $\sum_{i=1}^{n} \mathbf{m}_i = 1$. Bài toán được chuyển sang dạng

$$\max \sum_{i=1}^{n} \mathbf{m}_i\lambda_i, \text{ với } \mathbf{m}_i \geq 0, \sum_{i=1}^{n} m_i = 1.$$

Rõ ràng, bài toán cho thấy giá trị lớn nhất tìm được khi $\mathbf{m}_i$ tương ứng với λ_i lớn nhất bằng 1 và các $\mathbf{m}_i$ còn lại bằng 0. Kí hiệu i^* là chỉ số của giá trị riêng lớn nhất. Trở lại biến $\mathbf{w}$, điều này tương đương với $\mathbf{w}$ bắt đầu từ vectơ riêng thứ i^*, tức là cột thứ i^* của $\mathbf{U}$. Như vậy, thành phần chính đầu tiên được tìm một cách dễ dàng thông qua phân tích giá trị riêng.

Do các vectơ riêng của ma trận đối xứng là trực giao nên việc tìm thành phần chính thứ hai đồng nghĩa với việc tối đa hóa phương sai sao cho $\mathbf{v}_{i^*}$ vẫn bằng 0. Điều

này thực sự tương đương với việc tạo $\mathbf{w}$ trực giao mới cho vectơ riêng đầu tiên. Như vậy, chúng ta có bài toán tối ưu hóa tương tự nhưng với ràng buộc $\mathbf{m}_{i^*} = 0$. Rõ ràng, nghiệm của bài toán tối ưu đó thu được khi $\mathbf{w}$ bằng với vector riêng tương ứng với giá trị riêng lớn nhất thứ hai. Logic này áp dụng cho thành phần chính thứ k.

Do đó, tất cả các thành phần chính có thể được tìm thấy bằng cách đặt các vector riêng $\mathbf{u}_i$ với $i = 1, \ldots, n$ trong $\mathbf{U}$ sao cho các giá trị riêng giảm dần. Chúng ta hãy giả sử $\mathbf{U}$ được định nghĩa như vậy. Khi đó thành phần chính thứ i có dạng

$$\mathbf{s}_i = \mathbf{u}_i^T \mathbf{x}$$

Lưu ý rằng tất cả các λ_i đều không âm đối với ma trận hiệp phương sai.

3.5 Tính duy nhất nghiệm của PCA

Định lý 3.5.1. *Nếu* $(\mathbf{A}, \mathbf{B})$ *là một nghiệm của mô hình PCA thì* $(\mathbf{AQ}, \mathbf{BQ})$ *cũng là một nghiệm của mô hình PCA, với* $\mathbf{Q}$ *là ma trận trực giao cấp* r*.*

Lúc này, $\mathbf{Q}$ *được gọi là phép quay trực giao.*

Chứng minh. Giả sử $(\mathbf{A}, \mathbf{B})$ là một nghiệm của mô hình PCA. Với $\mathbf{Q}$ là một ma trận trực giao cấp r, tức là $\mathbf{QQ}^T = \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_r$, ta có

$$(\mathbf{AQ})(\mathbf{AQ})^T = \mathbf{AQQ}^T\mathbf{A}^T = \mathbf{AI}_r\mathbf{A}^T = \mathbf{AA}^T = \mathbf{I}_m$$

và

$$(\mathbf{AQ})^T(\mathbf{AQ}) = \mathbf{Q}^T\mathbf{A}^T\mathbf{AQ} = \mathbf{Q}^T I_r\mathbf{Q} = \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_r$$

Suy ra $(\mathbf{AQ})$ là ma trận trực giao cỡ $m \times r$. Mặt khác, ta có

$$(\mathbf{AQ})(\mathbf{BQ})^T = \mathbf{AQQ}^T\mathbf{B}^T = \mathbf{AI}_r\mathbf{B}^T = \mathbf{AB}^T$$

và

$$\left\|\mathbf{X} - \mathbf{AQQ}^T\mathbf{B}^T\right\|^2 = \left\|\mathbf{X} - \mathbf{AB}^T\right\|^2$$

Vậy $(\mathbf{AQ}, \mathbf{BQ})$ là một nghiệm của mô hình PCA. □

Từ Định lí 3.5.1, ta có nhận xét sau:

Nhận xét:

(i) Nghiệm $(\mathbf{A}, \mathbf{B})$ của PCA không duy nhất.

(ii) Phép quay trực giao $\mathbf{Q}$ sẽ cho ta ma trận tải có cấu trúc đơn giản hơn, do đó các nhân tố sẽ được diễn giải dễ dàng hơn.

3.6 Thuật toán tìm PCA của một ma trận

Giả sử $\mathbf{X}$ là ma trận cỡ $m \times n$, với $m \geq n$. Để tìm PCA của ma trận $\mathbf{X}$, chúng ta thực hiện các bước sau:

- Bước 1: Chuẩn hóa dữ liệu sao cho $E\{\mathbf{x_i}\} = 0$ với $\mathbf{x_i}$ là vector hàng của của ma trận $\mathbf{X}$

- Bước 2: Tìm SVD "chặt cụt" của ma trận $\mathbf{X}$, ta được $\mathbf{X} = \mathbf{U}_r\mathbf{D}_r\left(\mathbf{V}_r\right)^T$ với $r \leq n$.

- Bước 3: Tính ma trận $\mathbf{A}$ và $\mathbf{B}$ theo công thức sau:

$$\mathbf{A} = \mathbf{U}_r, \quad \mathbf{B}^T = \mathbf{D}_r\left(\mathbf{V}_r\right)^T$$

- Bước 4: Nếu nghiệm $(\mathbf{A}, \mathbf{B})$ chưa tốt thì chọn phép quay $\mathbf{Q}$, với $\mathbf{Q}$ là ma trận trực giao cấp r, ta tìm được nghiệm của mô hình PCA là $(\mathbf{AQ}, \mathbf{BQ})$

Cài đặt thuật toán trên PYTHON3 Hình 3.2

```python
def pca(X, k):
    # Center the data
    X_centered = X -np.mean(X,axis=1).reshape(-1,1)
    # Perform SVD on the centered data
    U, D, Vt = svd(X_centered)
    # Compute the transformed matrices
    A = U
    Bt = D@Vt
    # Reduce the dimensionality to k dimensions
    A_k = U[:,:k]
    Bt_k = D[:k,:k]@Vt[:k,:]
    return A_k, Bt_k
```

Hình 3.2: Cài đặt thuật toán PCA trên PYTHON3

Ví dụ

Ví dụ ta tìm PCA của ma trận

$$\mathbf{x} = \begin{bmatrix} 2 & 3 & 5 & 1 \\ 1 & 4 & 6 & 2 \\ 3 & 6 & 8 & 4 \\ 4 & 7 & 9 & 3 \end{bmatrix}$$

với $r = 3$ thành phần chính.

Lời giải:

- Bước 1: Chuẩn hóa ma trận $\mathbf{X}$ bằng các trừ từng vector cột đi vector trung bình của toàn bộ dữ liệu để được dữ liệu chuẩn hoá:

$$\hat{\mathbf{x}}_n = \mathbf{x}_n - \bar{\mathbf{x}} \tag{3.6.1}$$

với $\bar{\mathbf{x}} = [2.75, 3.25, 5.25, 5.75]$.

- Bước 2: Tìm SVD của ma trận $\bar{\mathbf{x}}$ Ta có

$$\bar{\mathbf{X}} = \begin{bmatrix} 2-2.75 & 3-2.75 & 5-2.75 & 1-2.75 \\ 1-3.25 & 4-3.25 & 6-3.25 & 2-3.25 \\ 3-5.25 & 6-5.25 & 8-5.25 & 4-5.25 \\ 4-5.75 & 7-5.75 & 9-5.75 & 3-5.75 \end{bmatrix} = \begin{bmatrix} -0.75 & 0.25 & 2.25 & -1.75 \\ -2.25 & 0.75 & 2.75 & -1.25 \\ -2.25 & 0.75 & 2.75 & -1.25 \\ -1.75 & 1.25 & 3.25 & -2.75 \end{bmatrix}$$

$$\bar{\mathbf{X}} = \mathbf{UDV}^T = \begin{bmatrix} -0.371 & -0.5 & -0.78254 & 0 \\ -0.49256 & 0.5 & -0.08596 & 0 \\ -0.49256 & 0.5 & -0.08596 & 0 \\ -0.61412 & -0.5 & 0.61062 & 0 \end{bmatrix} \cdot \begin{bmatrix} 7.66339 & 0 & 0 & 0 \\ 0 & 1.41421 & 0 & 0 \\ 0 & 0 & 0.52196 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\cdot \begin{bmatrix} 0.46578 & -0.20868 & -0.72288 & 0.46578 \\ -0.70711 & 0 & 0 & 0.70711 \\ -0.18179 & 0.84051 & -0.47692 & -0.18179 \\ 0.5 & 0.5 & 0.5 & 0.5 \end{bmatrix}$$

- Bước 3: Tính ma trận $\mathbf{A}$ và $\mathbf{B}$

$$\mathbf{A}_r = \begin{bmatrix} -0.371 & -0.5 & -0.78254 \\ -0.49256 & 0.5 & -0.08596 \\ -0.49256 & 0.5 & -0.08596 \\ -0.61412 & -0.5 & 0.61062 \end{bmatrix} \tag{3.6.2}$$

$$\mathbf{B}_r^T = \begin{bmatrix} 3.56945 & -1.59923 & -5.53968 & 3.56945 \\ -1 & 0 & 0 & 1 \\ -0.09489 & 0.43871 & -0.24893 & -0.09489 \end{bmatrix} \tag{3.6.3}$$

Ta thấy nghiệm $(\mathbf{A}, \mathbf{B})$ đã đủ tốt cụ thể có thể xem thử nghiệm trên PYTHON3 Hình 3.3.

Ngoài ra ta có thể sắp xỉ lại ma trân ban đầu theo công thức sau

$$\mathbf{X} \approx \mathbf{A}_r \mathbf{B}_r^T + \bar{\mathbf{x}} \tag{3.6.4}$$

Với một điểm dữ liệu mới $\mathbf{v} \in \mathbb{R}^D$ sẽ được giảm chiều bằng PCA theo công thức

$$\mathbf{w} = \mathbf{A}_r^T (\mathbf{v} - \bar{\mathbf{x}}) \in \mathbb{R}^K \tag{3.6.5}$$

```
A = np.array([[2 , 3 , 5 , 1],
[1 , 4 , 6 , 2],
[3 , 6 , 8 , 4],
[4 , 7 , 9 , 3]])
```

```
A_k, Bt_k = pca(A,3)
```

```
np.around(A_k, decimals=6)
```

```
array([[-0.370996, -0.5     , -0.782535],
       [-0.492556,  0.5     , -0.085956],
       [-0.492556,  0.5     , -0.085956],
       [-0.614116, -0.5     ,  0.610624]])
```

```
np.around(Bt_k, decimals=6)
```

```
array([[ 3.569453, -1.599228, -5.539678,  3.569453],
       [-1.      ,  0.      , -0.      ,  1.      ],
       [-0.09489 ,  0.438713, -0.248933, -0.09489 ]])
```

```
A_k@Bt_k + np.mean(A,axis=1).reshape(-1,1)
```

```
array([[2., 3., 5., 1.],
       [1., 4., 6., 2.],
       [3., 6., 8., 4.],
       [4., 7., 9., 3.]])
```

Hình 3.3: Thử nghiệm thuật toán PCA trên PYTHON3

Chương 4

Một số ứng dụng SVD và PCA

4.1 Ứng dụng của SVD trong bài toán xấp xỉ hạng thấp tốt nhất của ma trận

Về lí thuyết, khi làm việc với ma trận, ta cần nắm được toàn bộ những thông tin về ma trận đó. Tuy nhiên, trong thực tế có những ma trận cấp rất lớn với nhiều phần tử, lượng thông tin cần xử lí nhiều khiến ta rất khó khăn khi tiếp cận. Vấn đề đặt ra là, làm thế nào để ta cần loại bỏ, giảm tải bớt những thông tin, số liệu không cần thiết mà vẫn đảm bảo giữ lại những số liệu, những thông tin đặc trưng của ma trận, nhằm giúp cho công việc nghiên cứu được thuận tiện và đơn giản hơn. Nói cách khác, ta cần tìm một ma trận có hạng nhỏ hơn xấp xỉ với ma trận đã cho. Một trong những công cụ có thể giúp ta tìm được ma trận xấp xỉ đó chính là SVD. Trong chương này, chúng ta sẽ cùng tìm hiểu về ứng dụng của SVD trong bài toán xấp xỉ hạng thấp tốt nhất của ma trận.

Cho $\mathbf{A}$ là một ma trận cấp $m \times n$ bất kì, $\text{rank}(\mathbf{A}) = r$. Giả sử $\mathbf{A}$ có phân tích SVD dạng khai triển là

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

Với mỗi $k \in \{1, 2, \ldots, r\}$, đặt

$$\mathbf{A}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

Khi đó, $\text{rank}\,(\mathbf{A}_k) = k$. Hơn nữa, $\mathbf{A}_k$ chính là ma trận xấp xỉ tốt nhất có hạng bằng k của $\mathbf{A}$ theo cả hai chuẩn là chuẩn Frobenius và chuẩn 2. Ta có hai định lí quan trọng sau

Định lý 4.1.1. *[**eckart1936**]Định lí Eckart- Young, 1936*

Với mọi ma trận $\mathbf{B}$ *cỡ* $m \times n$ *và* $\text{rank}(\mathbf{B}) \leqslant k$*, ta có*

$$\|\mathbf{A} - \mathbf{B}\|_F \geqslant \|\mathbf{A} - \mathbf{A}_k\|_F .$$

Để chứng minh Định lý Eckart- Young, ta cần Bổ đề sau

Bổ đề 4.1.1. *Giả sử* $\mathbf{A}$ *là một ma trận cấp* $m \times n$ *bất kì,* $\text{rank}(\mathbf{A}) = r, \mathbf{A}$ *có phân tích SVD dạng khai triển là*

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

Khi đó với vector $\mathbf{x} \in \mathbb{R}^n$ *bất kì và* $1 \leqslant k \leqslant r$*, ta có*

$$\|\mathbf{A}\mathbf{x}\|_F^2 \leqslant \sigma_k^2 \|\mathbf{x}\|_F^2 + \sum_{i=1}^{k} \sigma_i^2 \left(\mathbf{v}_i^T \mathbf{x}\right)^2 - \sigma_k^2 \sum_{i=1}^{k} \left(\mathbf{v}_i^T \mathbf{x}\right)^2 .$$

Chứng minh. Theo phân tích SVD dạng khai triển của $\mathbf{A}$ ta có

$$\mathbf{A}\mathbf{x} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathcal{V}_i^T \mathbf{x} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \left(\mathcal{V}_i^T \mathbf{x}\right) = \sum_{i=1}^{r} \sigma_i \left(\mathcal{V}_i^T \mathbf{x}\right) \mathbf{u}_i$$

Do $(\mathbf{u}_1 \cdot \mathbf{u}_2, \ldots, \mathbf{u}_m)$ là cơ sở trực chuẩn trong $\mathbb{R}^m$ và $\sigma_1 \geqslant \sigma_2 \cdots \geqslant \sigma_r$ nên

$$\|\mathbf{A}\mathbf{x}\|_F^2 = \sum_{i=1}^{r} \sigma_i^2 \left(\mathcal{V}_i^T \mathbf{x}\right)^2 \leqslant \sum_{i=1}^{k} \sigma_i^2 \left(\mathcal{V}_i^T \mathbf{x}\right)^2 + \sigma_k^2 \sum_{i=k+1}^{r} \left(\mathcal{V}_i^T \mathbf{x}\right)^2 .$$

Mặt khác

$\sigma_k^2 \sum_{i=1}^{k} \left(\mathcal{V}_i^T \mathbf{x}\right)^2 + \sigma_k^2 \sum_{i=k+1}^{r} \left(\mathcal{V}_i^T \mathbf{x}\right)^2 = \sigma_k^2 \sum_{i=1}^{r} \left(\mathcal{V}_i^T \mathbf{x}\right)^2 \leqslant \sigma_k^2 \sum_{i=1}^{n} \left(\mathcal{V}_i^T \mathbf{x}\right)^2 = \sigma_k^2 \left\|\mathbf{V}^T \mathbf{x}\right\|_F^2 = \sigma_k^2 \|\mathbf{x}\|_F^2$

Do đó,

$$\sigma_k^2 \sum_{i=k+1}^{r} \left(\mathcal{V}_i^T \mathbf{x}\right)^2 \leqslant \sigma_k^2 \|\mathbf{x}\|_F^2 - \sigma_k^2 \sum_{i=1}^{k} \left(\mathcal{V}_i^T \mathbf{x}\right)^2$$

Vậy

$$\|\mathbf{A}\mathbf{x}\|_F^2 \leqslant \sigma_k^2 \|\mathbf{x}\|_F^2 + \sum_{i=1}^{k} \sigma_i^2 \left(\mathcal{V}_i^T \mathbf{x}\right)^2 - \sigma_k^2 \sum_{i=1}^{k} \left(\mathcal{V}_i^T \mathbf{x}\right)^2$$

□

Tiếp theo, chúng ta sẽ chứng minh Định lý Eckart-Young.

Chứng minh. Định lý Eckart-Young. Giả sử ma trận $\mathbf{B}$ có khai triển SVD dạng

$$\mathbf{B} = \sum_{i=1}^{k} \mathbf{x}_i \mathbf{y}_i^T$$

trong đó $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$ là hệ trực giao, $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ là hệ trực chuẩn. Ta có $\|\mathbf{A} - \mathbf{B}\|_F^2 = \mathrm{Tr}\left((\mathbf{A} - \mathbf{B})\left(\mathbf{A} - \mathbf{B}^T\right)\right) = \mathrm{Tr}\left(\mathbf{A}\mathbf{A}^T - \mathbf{A}\mathbf{B}^T - \mathbf{B}\mathbf{A}^T + \mathbf{B}\mathbf{B}^T\right)$. Mặt khác

$$\begin{aligned}
\mathbf{B}\mathbf{B}^T - \mathbf{A}\mathbf{B}^T - \mathbf{B}\mathbf{A}^T &= \left(\sum_{i=1}^{k} \mathbf{x}_i \mathbf{y}_i^T\right)\left(\sum_{i=1}^{k} \mathbf{y}_i \mathbf{x}_i^T\right) - A \sum_{i=1}^{k} \mathbf{y}_i \mathbf{x}_i^T - \left(\sum_{i=1}^{k} \mathbf{x}_i \mathbf{y}_i^T\right) \mathbf{A}^T \\
&= \sum_{i=1}^{k} \mathbf{x}_i \mathbf{x}_i^T - \sum_{i=1}^{k} \mathbf{x}_i \mathbf{y}_i^T \mathbf{A}^T - \sum_{i=1}^{k} \mathbf{A}_i \mathbf{x}_i^T + \sum_{i=1}^{k} \mathbf{A}_i \mathbf{y}_i^T \mathbf{A}_T - \sum_{i=1}^{k} \mathbf{A}_i \mathbf{y}_i^T \mathbf{A}^T \\
&= \sum_{i=1}^{k} \mathbf{x}_i \left(\mathbf{x}_i^T - \mathbf{y}_i^T \mathbf{A}^T\right) - \sum_{i=1}^{k} \mathbf{A}\mathbf{y}_i \left(\mathbf{x}_i^T - \mathbf{y}_i^T \mathbf{A}^T\right) - \sum_{i=1}^{k} \mathbf{A}\mathbf{y}_i \mathbf{y}_i^T \mathbf{A}^T \\
&= \sum_{i=1}^{k} \left(\mathbf{x}_i - \mathbf{A}\mathbf{y}_i\right)\left(\mathbf{x}_i^T - \mathbf{y}_i^T \mathbf{A}^T\right) - \sum_{i=1}^{k} \mathbf{A}\mathbf{y}_i \mathbf{y}_i^T A^T \\
&= \sum_{i=1}^{k} \left(\mathbf{x}_i - \mathbf{A}\mathbf{y}_i\right)\left(\mathbf{x}_i - \mathbf{y}_i \mathbf{A}\right)^T - \sum_{i=1}^{k} \left(\mathbf{A}\mathbf{y}_i\right)\left(\mathbf{a}\mathbf{y}_i\right)^T.
\end{aligned}$$

Vì

$$\sum_{i=1}^{k} \mathrm{Tr}\left(\left(\mathbf{x}_i - \mathbf{A}\mathbf{y}_i\right)\left(\mathbf{x}_i - \mathbf{y}_i \mathbf{A}\right)^T\right) = \sum_{i=1}^{k} \|\mathbf{x}_i - \mathbf{A}\mathbf{y}_i\|_F^2 \geqslant 0$$

nên suy ra

$$\begin{aligned}\|\mathbf{A}-\mathbf{B}\|_F^2 &= \operatorname{Tr}\left(\mathbf{A}\mathbf{A}^T\right) + \sum_{i=1}^{k}\operatorname{Tr}\left(\left(\mathbf{x}_i-\mathbf{A}\mathbf{y}_i\right)\left(\mathbf{x}_i-\mathbf{y}_i\mathbf{A}\right)^T\right) - \sum_{i=1}^{k}\|\mathbf{A}\mathbf{y}_i\|_F^2 \\ &\geqslant \operatorname{Tr}\left(\mathbf{A}\mathbf{A}^T\right) - \sum_{i=1}^{k}\|\mathbf{A}\mathbf{y}_i\|_F^2 .\end{aligned} \tag{4.1.1}$$

Tiếp theo, ta đánh giá số hạng $\sum_{i=1}^{k}\|\mathbf{A}\mathbf{y}_i\|_F^2$ trong bất đẳng thức 4.1.1. Theo Bổ đề 4.1.1 ta có

$$\|\mathbf{A}\mathbf{y}_i\|_F^2 \leqslant \sigma_k^2 + \sum_{i=1}^{k}\sigma_i^2\left(\mathcal{V}_i^T\mathbf{y}_i\right)^2 - \sigma_k^2\sum_{i=1}^{k}\left(\mathcal{V}_{\mathcal{V}}^T\mathbf{y}_i\right)^2 .$$

Lấy tổng theo chỉ số j ta có bất đẳng thức

$$\begin{aligned}\sum_{j=1}^{k}\left\|\mathbf{A}\mathbf{y}_j\right\|^2 &\leqslant k\sigma_k^2 + \sum_{j=1}^{k}\left[\sum_{i=1}^{k}\sigma_i^2\left(\mathcal{V}_i^T\mathbf{y}_j\right)^2 - \sigma_k^2\sum_{i=1}^{k}\left(\mathcal{V}_i^T\mathbf{y}_j\right)^2\right] \\ &= k\sigma_k^2 + \sum_{j=1}^{k}\left[\sum_{i=1}^{k}\left(\sigma_i^2-\sigma_k^2\right)\left(\mathcal{V}_i^T\mathbf{y}_j\right)^2\right] \\ &= \sum_{i=1}^{k}\left[\sigma_k^2 + \sum_{j=1}^{k}\left(\sigma_i^2-\sigma_k^2\right)\left(\mathcal{V}_i^T\mathbf{y}_j\right)^2\right] \\ &= \sum_{i=1}^{k}\left[\sigma_k^2 + \left(\sigma_i^2-\sigma_k^2\right)\sum_{j=1}^{k}\left(v_i^T y_j\right)^2\right].\end{aligned}$$

Mặt khác,

$$\sum_{j=1}^{k}\left(\mathcal{V}_i^T\mathbf{y}_j\right)^2 \leqslant \sum_{j=1}^{n}\left(\mathcal{V}_i^T\mathbf{y}_j\right)^2 = \left\|\mathcal{V}_i^T\mathbf{Y}\right\|_F^2 = \left\|\mathcal{V}_i^T\right\|_F^2 = 1$$

với $\mathbf{Y}$ là ma trận trực giao có các cột là các vector $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n$. Do đó

$$\sum_{j=1}^{k}\left\|\mathbf{A}\mathbf{y}_j\right\|_F^2 \leqslant \sum_{i=1}^{k}\sigma_k^2$$

Thay vào bất đẳng thức 4.1.1 ta được

$$\|\mathbf{A}-\mathbf{B}\|_F^2 \geqslant \operatorname{Tr}\left(\mathbf{A}\mathbf{A}^T\right) - \sum_{i=1}^{k}\|\mathbf{A}\mathbf{y}_i\|_F^2 \geqslant \|\mathbf{A}\|_F^2 - \sum_{i=1}^{k}\sigma_i^2.$$

Khi $\mathbf{B} = \mathbf{A}_k = \sum_{i=1}^{k}\sigma_i\mathbf{u}_i\mathcal{V}_i^T$, ta có

$$\mathbf{A} - \mathbf{A}_k = \sum_{i=k+1}^{r} \sigma_i \mathbf{u}_i \mathcal{V}_i^T$$

Từ đó suy ra

$$\|\mathbf{A} - \mathbf{A}_k\|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2 = \sum_{i=1}^{r} \sigma_i^2 - \sum_{i=1}^{k} \sigma_i^2 = \|A\|_F^2 - \sum_{i=1}^{k} \sigma_i^2.$$

Điều này chứng tỏ $\mathbf{A}_k$ chính là xấp xỉ hạng k tốt nhất của ma trận $\mathbf{A}$ theo chuẩn Frobenius. □

Bên cạnh đó ta có nhận xét quan trọng về sai số do xấp xỉ một ma trận $\mathbf{A}$ có hạng r bởi $\mathbf{A}_k$ với $k < r$ phần tử như sau:

$$\frac{\|\mathbf{A} - \mathbf{A}_k\|_F^2}{\|\mathbf{A}\|_F^2} = \frac{\sum_{i=k+1}^{r} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}$$

Ở đây, lượng thông tin được định nghĩa là tổng bình phương của giá trị suy biến. Ví dụ, nếu muốn giữ lại ít nhất 90% lượng thông tin trong $\mathbf{A}$, trước hết ta tính $\sum_{j=1}^{r} \sigma_j^2$, sau đó chọn k là số nhỏ nhất sao cho

$$\frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{j=1}^{r} \sigma_j^2} \geq 0.9$$

Định lý 4.1.2. *[linh2016]*

Giả sử $\mathbf{A}$ *là ma trận cỡ* $m \times n$ *bất kì, với* $\text{rank}(A) = r$ *và* $\mathbf{A}$ *có khai triển kì dị SVD là*

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathcal{V}_i^T$$

Khi đó, với mọi ma trận $\mathbf{B}$ *có cỡ* $m \times n$ *bất kì và* $\text{rank}(\mathbf{B}) \leqslant k$*, ta có*

$$\|\mathbf{A} - \mathbf{B}\|_2 \geqslant \sigma_{k+1}$$

Dấu "=" xảy ra khi $\mathbf{B} = \mathbf{A}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathcal{V}_i^T$.

Chứng minh. Giả sử tồn tại ma trận $\mathbf{B}$ có $\text{rank}(\mathbf{B}) \leqslant k$ và $\|\mathbf{A} - \mathbf{B}\|_2 < \sigma_{k+1}$. Vì $\text{rank}(\mathbf{B}) \leqslant k$ nên tồn tại một không gian con $\mathbf{W}$ cỡ $r - k$ sao cho với mọi $\mathbf{w} \in \mathbf{W}$ thì

Bw = 0. Do đó với mọi $\mathbf{w} \in \mathbf{W}$ thì $\mathbf{Aw} = (\mathbf{A} - \mathbf{B})\mathbf{w}$ và

$$\|\mathbf{Aw}\|_2 = \|(\mathbf{A} - \mathbf{B})\mathbf{w}\|_2 \leqslant \|\mathbf{A} - \mathbf{B}\|_2 \|\mathbf{w}\|_2 < \sigma_{k+1}\|\mathbf{w}\|_2.$$

Vì vậy W là không gian con cỡ $r - k$ sao cho

$$\|\mathbf{A}\|_2 < \sigma_{k+1}\|\mathbf{w}\|_2$$

Mặt khác, tồn tại một không gian con $\mathbf{U}$ có chiều là $k+1$ được xây dựng từ $k+1$ giá trị kì dị đầu tiên của ma trận $\mathbf{A}$ sao cho

$$\|\mathbf{Au}\| \geqslant \sigma_{k+1}\|\mathbf{u}\|_2, \quad \forall \mathbf{u} \in \mathbf{U}$$

Khi đó $\dim(\mathbf{U}+\mathbf{W}) = (r-k)+(k+1) = r+1 > r$. Do đó tồn tại $\mathcal{V} \neq \mathbf{0}$ và $\mathcal{V} \in \mathbf{U} \cap \mathbf{W}$. Suy ra $\|\mathbf{A}\mathcal{V}\|_2 < \sigma_{k+1}\|\mathcal{V}\|_2$ và $\|\mathbf{A}\mathcal{V}\| \geqslant \sigma_{k+1}\|\mathcal{V}\|_2$. Điều này không thể xảy ra. Vì vậy giả thiết phản chứng ban đầu là sai.

Vậy với mọi ma trận $\mathbf{B}$ cỡ $m \times n$ bất kì có $\text{rank}(\mathbf{B}) \leqslant k$ thì $\|\mathbf{A} - \mathbf{B}\|_2 \geqslant \sigma_{k+1}$. Khi $\mathbf{B} = \mathbf{A}_k$ thì $\|\mathbf{A} - \mathbf{A}_k\|_2 = \sigma_{k+1}$. Điều này chứng tỏ $\mathbf{A}_k$ chính là xấp xỉ hạng k tốt nhất của $\mathbf{A}$ theo chuẩn 2 . □

4.2 Phân tích SVD trong xử lí ảnh

Như đã nói, phân tích SVD là dạng phân tích có rất nhiều ứng dụng trong lý thuyết và thực tiễn. Một trong những ứng dụng ấn tượng nhất chính là sử dụng SVD trong hiệu chỉnh hình ảnh kĩ thuật số. Nhờ đó hình ảnh được truyền đi một cách hiệu quả bằng vệ tinh, internet, ...

Ý tưởng cơ sở của việc hiệu chỉnh ảnh là làm giảm số lượng thông tin truyền đi mà không làm mất đi những thông tin thực chất. Trong một bức ảnh kĩ thuật số, mỗi điểm ảnh được thể hiện bởi ba giá trị màu: xanh (blue), lục (green), đỏ (red) với các trị số từ 0 đến 255. Như vậy, với một hình ảnh chỉ với độ lớn là 130×130 pixels thì chúng ta phải lưu trữ 3 ma trận (thể hiện màu sắc của các điểm) có cùng độ lớn là 130×130, tức là phải lưu trữ 50700 số. Tuy nhiên, trong thực tế, khi truyền hay lưu trữ thông tin ảnh, ta có thể không cần những hình ảnh, hoặc một số phần của hình ảnh có độ nét quá lớn. Sử dụng phân tích SVD, chúng ta có thể loại bỏ rất nhiều thông tin không cần thiết đó.

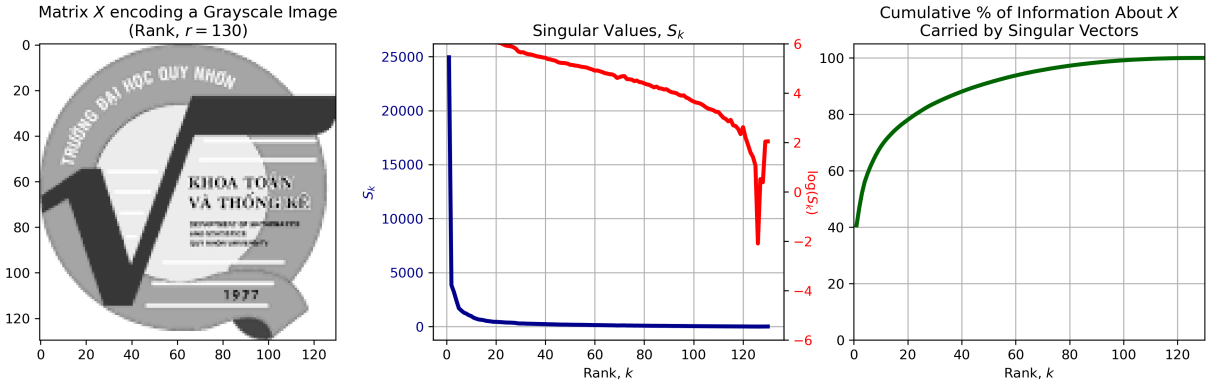
Ví dụ một hình ảnh 130×130 pixels được phân tích thành 3 ma trận $\mathbf{A}, \mathbf{B}, \mathbf{C}$ có cùng độ lớn 130×130. Giả sử $\mathbf{A}$ có phân tích SVD là

$$\mathbf{A} = \sigma_1 \mathbf{u}_1 \mathcal{V}_1^T + \cdots + \sigma_r \mathbf{u}_r \mathcal{V}_r^T$$

Theo như chứng minh ở Định lí 4.1.1 với mỗi giá trị $k < r$ thì $A_k = \sigma_1 \mathbf{u}_1 \mathcal{V}_1^T + \cdots + \sigma_k \mathbf{u}_k \mathcal{V}_k^T$ là xấp xỉ hạng k tốt nhất của $\mathbf{A}$.

Ví dụ, $k = 16$ thì ma trận $\mathbf{A}_k$ thể hiện các dữ liệu của $\mathbf{A}$ tương ứng với 16 giá trị kì dị đầu tiên. Như vậy, ta chỉ cần lưu trữ 16 giá trị kì dị, 16 vector $\mathbf{u}_i, 16z$ vector $\mathcal{V}_i$, tương đương với 4176 số. Tương tự như vậy với hai ma trận $\mathbf{B}$ và $\mathbf{C}$, số lượng các số phải lưu trữ là 12528 số. Rõ ràng phân tích SVD đã giúp giảm tải một lượng thông tin cần lưu trữ một cách đáng kể.

Trong Python, ta có thể hiệu chỉnh độ nét của hình ảnh theo tham số k tùy chọn. Ta xét ví dụ sau để đơn giản ta chỉ xét hình ảnh xám với kích cỡ $130 \times 130 \times 1$ như sau :



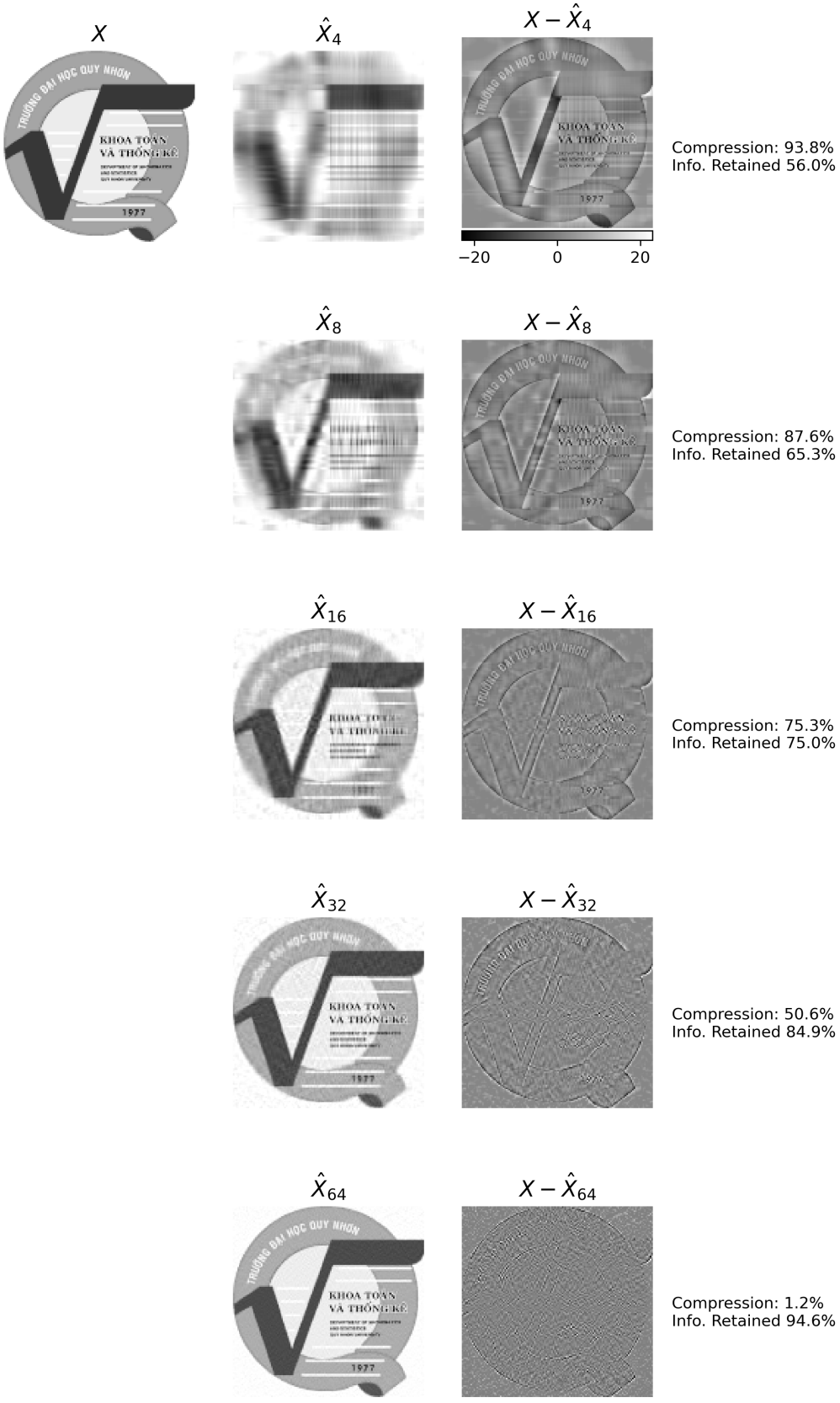
Hình 4.1: Tính quan trọng của các thành phần chính trong ma trận ảnh và mức độ giữ lại thông tin khi ta giảm số lượng các thành phần chính

Đồ thị đầu tiên (bên trái) hiển thị ma trận ảnh gốc, được biểu diễn bằng các giá trị màu xám. Ma trận này thể hiện hình ảnh ban đầu trước bất kỳ xử lý nào.Đồ thị thứ hai (ở giữa) biểu diễn các giá trị kì dị (singular values) của ma trận ảnh. Các giá trị kì dị thể hiện độ lớn của các thành phần chính trong ma trận. Đồ thị hiển thị hai đường: một đường cho các giá trị kì dị ban đầu và một đường cho các giá trị kì dụ được chuyển đổi sang đơn vị logarit (logarithmic scale). Biểu đồ này cho ta cái nhìn

tổng quan về độ quan trọng của các thành phần chính trong ma trận ảnh.Đồ thị cuối cùng (bên phải) biểu diễn tổng hợp thông tin tích lũy bằng cách sử dụng các giá trị kì dị. Nó cho thấy phần trăm thông tin được giữ lại (tích lũy) khi ta giữ lại một số lượng gia tăng các thành phần chính. Đồ thị này giúp ta quyết định xem ta có thể giảm số lượng thành phần chính mà không mất quá nhiều thông tin quan trọng. Cụ thế hơn khi giảm chiều dữ liệu với $k = 4, 16, 32, 64$.. Ta thấy với việc sấp xỉ ma trận ban đầu bằng SVD với $k = 32$ ta đã giảm được dung lương lưu trữ đi khoảng 50% nhưng vẫn không làm giảm đi chất lượng hình ảnh quá nhiều.

Lưu ý:

- Compression là tỉ lệ dung lượng đã giảm được để lưu trữ hình ảnh nén.
- Info. Retained là phần thông tin từ ma trận gốc mà chúng ta vẫn giữ được sau khi nén.



Hình 4.2: Ứng với từng k ta quan sát được mức độ tối ưu trong việc lưu trữ thông tin cũng như là lượng thông tin được giữ lại.

4.3 Eigenface và ứng dụng PCA trong nhận dạng khuôn mặt

Phân tích thành phần chính (PCA - Principal Component Analysis) luôn được biết đến như là một phương pháp thống kê phổ biến dùng để giảm kích thước của dữ liệu trong khi vẫn giữ được những thông tin quan trọng nhất. PCA tìm ra các "thành phần chính" trong dữ liệu - những hướng mà dữ liệu biến đổi nhiều nhất, và dùng chúng để tái biểu diễn dữ liệu trong một không gian ít chiều hơn. Do đó, PCA đóng một vai trò quan trọng trong lĩnh vực nhận dạng khuôn mặt. Nhận dạng khuôn mặt là một bài toán phức tạp do sự đa dạng vô cùng lớn của các khuôn mặt. Mỗi khuôn mặt đều có những đặc điểm duy nhất và khác biệt như kích thước, hình dạng, màu sắc, cấu trúc, đặc điểm về giới tính, tuổi tác, v.v... PCA giúp chúng ta tìm ra những đặc điểm quan trọng nhất (hay còn gọi là "thành phần chính") của khuôn mặt, giúp tối ưu hóa quá trình nhận dạng và giảm bớt chi phí tính toán. Một trong những phương pháp thường được biết đến đó là Eigenface.

Eigenface là một phương pháp nhận dạng khuôn mặt dựa trên kỹ thuật phân tích thành phần chính (PCA). Thuật ngữ "Eigenface" xuất phát từ từ "eigen" trong tiếng Đức, có nghĩa là "đặc trưng" hoặc "riêng". Trong ngữ cảnh này, một Eigenface là một tập hợp các vector riêng, mà từ đó có thể tạo ra hầu hết các khuôn mặt khác nhau.

Phương pháp Eigenface giảm kích thước của dữ liệu hình ảnh khuôn mặt bằng cách chuyển đổi nó thành một tập hợp các thành phần chính. Những thành phần này, được gọi là Eigenfaces, đại diện cho các đặc điểm cơ bản của khuôn mặt, như hình dạng của mắt, mũi, miệng và vị trí tương đối của chúng. Mỗi khuôn mặt trong dữ liệu đều có thể được biểu diễn như một tổ hợp tuyến tính của những Eigenface này. Tôi sẽ thử nghiệm phương pháp Eigenface trên bộ dữ liệu LFW.

Bộ dữ liệu LFW (Labeled Faces in the Wild)[**huang**] là một bộ dữ liệu hình ảnh khuôn mặt phổ biến được sử dụng trong nghiên cứu nhận dạng khuôn mặt. Bộ dữ liệu này bao gồm hình ảnh của hàng trăm người nổi tiếng, với mỗi người được chụp dưới nhiều góc độ và biểu cảm khác nhau. Mỗi hình ảnh trong bộ dữ liệu là một ma trận hai chiều biểu thị giá trị cường độ điểm ảnh có thể hiểu như ma trận $50 \times 37 \times 1$. với 1288 hình. Và được chia thành 7 loại khác nhau ứng với 7 người nổi tiếng.

Khi áp dụng phương pháp Eigenface cho bộ dữ liệu LFW, ta sẽ tạo ra một "khuôn mặt trung bình" cùng với một tập hợp các "Eigenfaces". Sau đó, mỗi khuôn mặt trong bộ dữ liệu có thể được biểu diễn như một tổ hợp tuyến tính của các Eigenfaces, giúp cho việc nhận dạng khuôn mặt trở nên hiệu quả và chính xác hơn.

Đầu tiên ta sẽ làm "phẳng" ma trận đầu vào thành ma trận $(50 \times 37, 1288)$ với mỗi cột là một vector ứng với một khuôn mặt trong tập dữ liệu. Sau đó ta chia dữ liệu đầu vào thành tập huấn luyện và tập kiểm thử với tỉ lể lần lượt là 0.75 và 0.25. Ta nhận được 2 ma trận với kích cỡ là $(1850, 966)$ và $(1850, 322)$

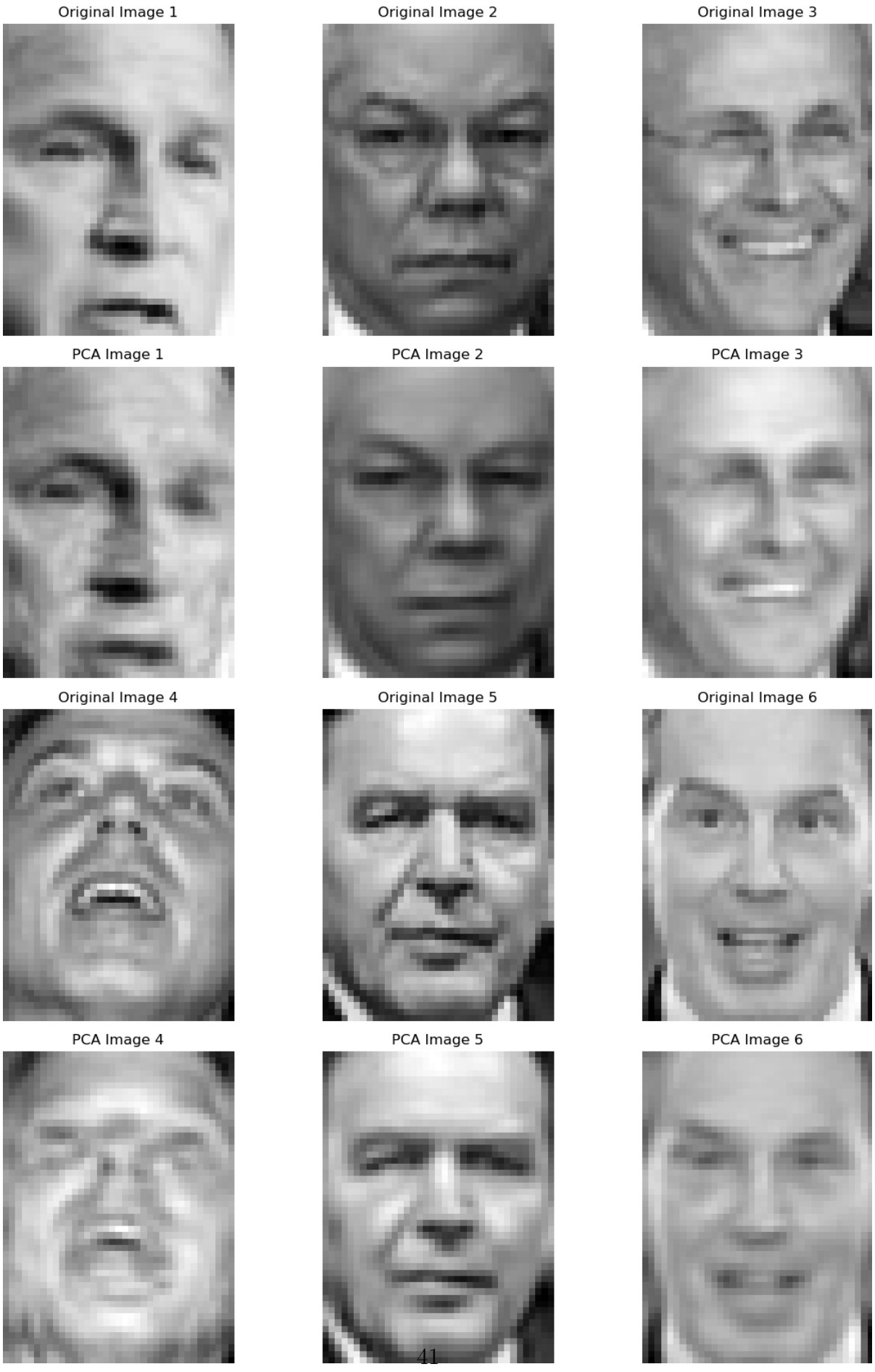
Sau đó, ta đi tìm ma trận $\mathbf{A}_k$ chứa các Eigenfaces và ma trận biểu diễn dữ liệu gốc trên cơ sở của các thành phần chính $\mathbf{B}_k^T$ thông qua PCA.

PCA sẽ giúp chúng ta xác định các thành phần chính (hay còn gọi là Eigenfaces trong ngữ cảnh này) của tập dữ liệu hình ảnh khuôn mặt. Các thành phần này, được chứa trong ma trận $\mathbf{A}_k$, đại diện cho các hướng mà dữ liệu biến thiên nhiều nhất. Điều này giúp tạo ra một không gian mới, với các trục tương ứng với các thành phần chính, để biểu diễn dữ liệu.

Kế tiếp, chúng ta tính toán ma trận $\mathbf{B}_k^T$, mà mỗi cột của nó biểu diễn một hình ảnh khuôn mặt trong không gian của các thành phần chính. Mỗi hình ảnh khuôn mặt giờ đây được biểu diễn bằng một tổ hợp tuyến tính của các Eigenfaces, với hệ số tương ứng nằm trong $\mathbf{B}_k^T$.

Vì vậy, thông qua PCA, chúng ta có thể giảm đáng kể số lượng chiều dữ liệu mà vẫn giữ được phần lớn thông tin cần thiết từ tập dữ liệu hình ảnh khuôn mặt ban đầu. Với $k = 150$ ta có thể xem kết quả sau khi được kết quả ảnh sau khi qua PCA Hình 4.3

Sau đó giảm chiều dữ liệu kiểm thử bằng cách chiếu nó qua ma trận $\mathbf{A}_k$ theo công thức 3.6.5. Bây giờ ta có 966 điểm dữ liệu huấn liệu và 322 điểm dữ liệu kiểm thử. Liệu ta có thể phân biệt được các nét đặc trưng của mỗi khuông mặt chỉ thông qua các vector eigenface chỉ với 150 chiều thay và 1850 như ban đầu. Để trả lời câu hỏi này ta sẽ huấn luyện một bộ phân lớp tương đối nổi tiếng và SVM(Support Vector Machine)[**cristianini2000**]. hình 4.4 và hình 4.5 là kết quả sau khi thử nghiệm với bộ phân loại SVM. Ta thấy bộ phân loại có thể nhận dạng khuôn mặt tốt dựa trên vector eigenface với độ chính xác trùng bình là 80%. Hình 4.6 là kết quả suy luận của bộ phân



Hình 4.3: Hình ảnh ban đầu và hình ảnh sau khi được khôi phục qua PCA.

loại trên tập kiểm thử. Hình 4.4 là báo cáo phân loại (classification report) của một

	precision	recall	f1-score	support
Ariel Sharon	0.62	0.77	0.69	13
Colin Powell	0.77	0.83	0.80	60
Donald Rumsfeld	0.59	0.63	0.61	27
George W Bush	0.88	0.84	0.86	146
Gerhard Schroeder	0.83	0.80	0.82	25
Hugo Chavez	0.75	0.60	0.67	15
Tony Blair	0.73	0.75	0.74	36
accuracy			0.80	322
macro avg	0.74	0.75	0.74	322
weighted avg	0.80	0.80	0.80	322

Hình 4.4: Báo cáo phân loại sau khi huận luyện bộ phân loại SVM.

mô hình phân loại, được tính toán trên bộ dữ liệu kiểm thử (test set). Dựa vào dữ liệu bạn cung cấp, có vẻ như mô hình đã được huấn luyện để phân loại ảnh của các chính trị gia khác nhau.

Báo cáo bao gồm các thông số đo lường hiệu suất của mô hình như sau:

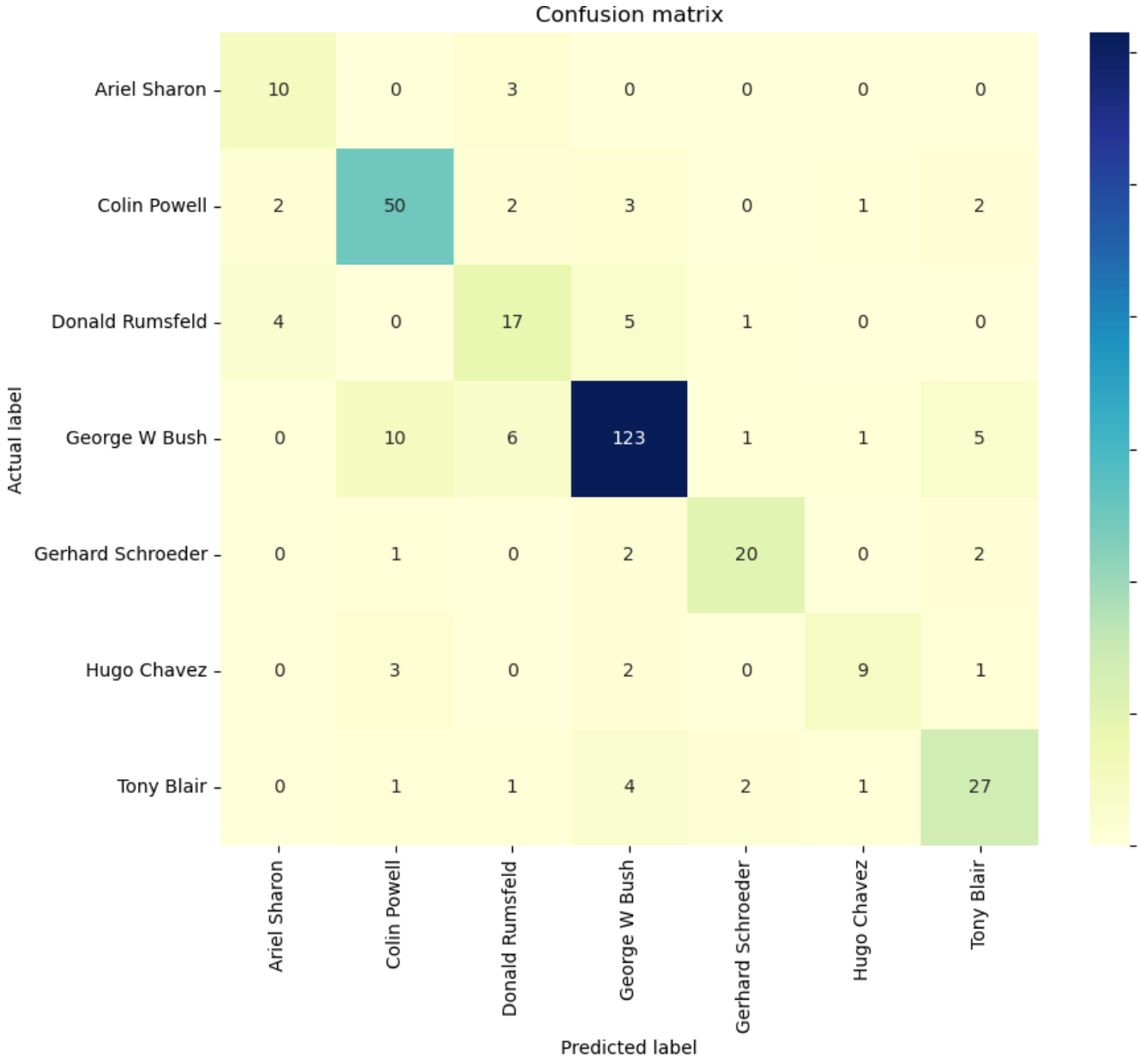
- Precision: Độ chính xác cho từng lớp. Độ chính xác là tỉ lệ giữa số mẫu dự đoán đúng và tổng số mẫu dự đoán cho một lớp cụ thể.
- Recall: Độ phủ cho từng lớp. Độ phủ là tỉ lệ giữa số mẫu dự đoán đúng và tổng số mẫu thực sự thuộc về lớp đó.
- F1-score: Trung bình điều hòa của độ chính xác và độ phủ. Điểm F1 càng cao thì mô hình càng tốt.
- Support: Số lượng mẫu thực sự có trong tập kiểm thử cho mỗi lớp.

Đối với mỗi chỉ số này, báo cáo cũng cung cấp một giá trị trung bình ("macro avg") và giá trị trung bình có trọng số ("weighted avg") cho tất cả các lớp:

- "Macro avg" là trung bình đơn giản của các số liệu trên tất cả các lớp.
- "Weighted avg" là trung bình có trọng số của các số liệu trên tất cả các lớp, trong đó trọng số cho mỗi lớp là số lượng mẫu trong lớp đó.

- Cuối cùng, "accuracy" là tỉ lệ tổng thể của các dự đoán đúng so với tổng số mẫu.

Dựa trên báo cáo, mô hình phân loại này hoạt động tốt nhất cho lớp "George W Bush" với điểm F1 là 0.86, và kém nhất cho lớp "Donald Rumsfeld" với điểm F1 là 0.61. Độ chính xác tổng thể của mô hình là 0.80, có nghĩa là mô hình đã dự đoán đúng 80% các mẫu trong tập kiểm thử.



Hình 4.5: Ma trận lỗi(confusion matrix)

Ma trận lỗi(hình4.5) là một cách hữu ích để trực quan hóa hiệu suất của bộ phân loại. Mỗi hàng của ma trận tương ứng với lớp thực tế, và mỗi cột tương ứng với lớp

dự đoán.

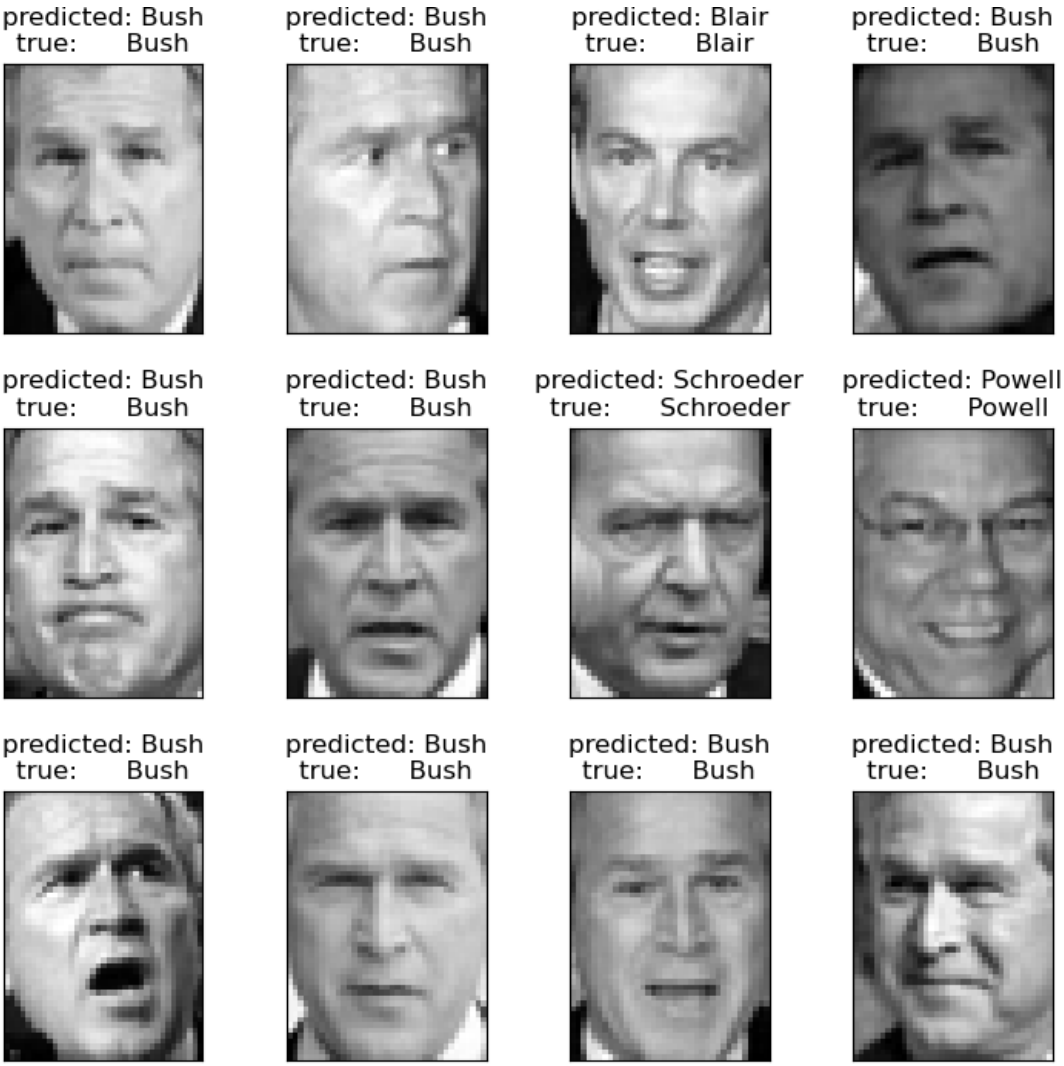
Giá trị tại vị trí (i, j) là số lượng mẫu mà lớp thực tế là i nhưng mô hình dự đoán là lớp j. Do đó, các phần tử trên đường chéo chính của ma trận tương ứng với số lượng các dự đoán chính xác. Ta có các lớp sau:

- Ariel Sharon
- Colin Powell
- Donald Rumsfeld
- George W Bush
- Gerhard Schroeder
- Hugo Chavez
- Tony Blair

Khi đó, các con số trong ma trận sẽ tương ứng với:

- 10 hình ảnh của Ariel Sharon đã được dự đoán chính xác, 3 hình ảnh bị nhầm lẫn với Donald Rumsfeld.
- Colin Powell có 50 hình ảnh được dự đoán chính xác, nhưng 2 hình ảnh bị nhầm lẫn với Ariel Sharon, 2 hình ảnh bị nhầm với Donald Rumsfeld, 3 hình ảnh bị nhầm với George W Bush, và còn lại bị nhầm lẫn với Hugo Chavez và Tony Blair.
- ... và tương tự cho các lớp khác.

Một số hình ảnh từ tập dữ liệu kiểm thử, với mỗi hình ảnh được hiển thị cùng với dự đoán của mô hình và nhãn thực sự của hình ảnh.



Hình 4.6: kết quả suy luận của bộ phân loại trên tập kiểm thử

4.4 Nghiên cứu về ứng dụng SVD trong kiến trúc Transformer

Trong thời gian gần đây với sự bùng nổi của ChatGPT(GPT-3.5) và GPT-4 với lõi là kiến trúc Transformer[**vaswani2017**] đã gây được sự chú ý với các nhà nghiên cứu nói riêng và những người yêu thích công nghệ nói chung. Nhưng trước đó các mô hình với kiến trúc này đã trở nên phổ biến trong xử lý ngôn ngữ tự nhiên (NLP) cho nhiều vấn đề như dịch thuật[**ott2018**], phân loại văn bản, trả lời câu hỏi và nhiều ứng dụng khác[**raffel2019**]. Tuy vậy, số lượng tham số trong các mô hình với kiến trúc Transformer hàng đầu đã tăng đáng kể, từ 340 triệu trong BERT-Large ban đầu lên đến 175 tỷ trong GPT-3[**brown2020**]. Mặc dù những mô hình lớn này mang lại kết quả ấn tượng trên nhiều loại nhiệm vụ, việc huấn luyện và triển khai chúng lại gặp nhiều khó khăn. Ví dụ, mô hình BERT-Large[**Devlin2019**] ban đầu mất 4 ngày để huấn luyện trên 16 Cloud TPUs, và GPT-3 tiêu thụ hàng chục lần nhiều petaflops/ngày so với GPT-2[**radford2019**], mô hình tiền nhiệm của nó. Triển khai các mô hình Transformer vào các ứng dụng thực tế cũng đòi hỏi nhiều tài nguyên và quá trình rút gọn hoặc nén dữ liệu phức tạp[**hinton2015**].

Rào cản chính đối với hiệu suất của các mô hình Transformer là cơ chế tự chú ý. Trong cơ chế này, biểu diễn của mỗi token được cập nhật bằng cách chú ý đến tất cả các token khác trong lớp trước đó. Thao tác này quan trọng để giữ thông tin dài hạn và giúp Transformer vượt trội hơn các mô hình tuần tự trong xử lý chuỗi dài. Tuy nhiên, việc chú ý đến tất cả các token ở mỗi lớp tạo ra độ phức tạp $O(n^2)$ với độ dài chuỗi. Vì vậy, câu hỏi đặt ra là liệu có thể tối ưu hóa mô hình Transformer để tránh thao tác bậc hai này hay thao tác này là yêu cầu bắt buộc để duy trì hiệu suất mạnh mẽ?

Các nghiên cứu trước đây đã đề xuất một số kỹ thuật để cải thiện hiệu suất của cơ chế tự chú ý. Một kỹ thuật phổ biến là áp dụng thưa thớt vào các lớp chú ý[**child2019**, **qiu2019**, **beltagy2020**] bằng cách cho mỗi token chỉ chú ý đến một tập hợp con của các token trong chuỗi. Điều này giảm độ phức tạp tổng thể của cơ chế tự chú ý xuống còn $O(n\sqrt{n})$[**child2019**]. Tuy nhiên, như đã chỉ ra trong[**qiu2019**], phương pháp này gặp phải sự giảm hiệu suất lớn với ít lợi ích về hiệu quả, ví dụ như giảm 2% với chỉ tốc

độ tăng 20%. Gần đây, mô hình Reformer[**kitaev2020**] đã sử dụng băm nhạy cục bộ (LSH) để giảm độ phức tạp tự chú ý xuống còn O(n log(n)). Tuy nhiên, trong thực tế, lợi ích về hiệu quả của Reformer chỉ xuất hiện trên các chuỗi có độ dài lớn hơn 2048. Hơn nữa, cách tiếp cận băm nhiều vòng của Reformer thực tế tăng số lượng các thao tác tuần tự, làm giảm lợi ích hiệu quả cuối cùng của nó.

Vào cuối năm 2020 nhóm các nhà nghiên cứu AI ở Facebook đã giới thiệu một phương pháp mới là Linformer[**wang2020**] và chứng minh rằng cơ chế tự chú ý có thể được xấp xỉ bằng một ma trận có hạng thấp thông qua SVD. Tiếp đó nhóm tác giả còn khai thác phát hiện này để đề xuất một cơ chế tự chú ý mới, giảm độ phức tạp tổng thể của tự chú ý từ $O(n^2)$ xuống $O(n)$ cả về thời gian và không gian. Độ phức tạp thời gian trên mỗi lớp và số lượng phép toán tuần tự cho các kiến trúc khác nhau được thể hiện ở bảng dưới.

Kiến trúc mô hình	Độ phức tạp về thời gian trên mỗi lớp	Số phép toán tuần tự
Recurrent	$O(n)$	$O(n)$
Transformer	$O\left(n^2\right)$	$O(1)$
Sparse Tansformer	$O(n\sqrt{n})$	$O(1)$
Reformer	$O(n\log(n))$	$O(\log(n))$
Linformer	$O(n)$	$O(1)$

Kết luận

Qua quá trình thực hiện khóa luận, tôi đã có cơ hội tìm hiểu và nghiên cứu sâu về các khái niệm và kỹ thuật liên quan đến Phân tích giá trị kì dị (SVD) và Phân tích thành phần chính (PCA). Khóa luận đã đạt được những kết quả sau:

1 Kiến thức chuẩn bị: Khóa luận đã trình bày một cách tổng quát về các kiến thức cơ bản về Ma trận, Vector riêng, Giá trị riêng, và Định lí phổ của ma trận đối xứng. Đồng thời, khóa luận cũng đã giới thiệu về các khái niệm cơ bản trong Xác xuất thống kê bao gồm Kỳ vọng, Phương sai và Ma trận hiệp phương sai.

2 Phân tích giá trị kì dị (SVD): Khóa luận đã giới thiệu về khái niệm SVD, cách thức tính toán SVD của một ma trận, cũng như một số tính chất quan trọng của ma trận liên quan đến SVD.

3 Phân tích thành phần chính (PCA): Khóa luận đã trình bày về khái niệm PCA, ý tưởng chính của PCA, cách thức tìm các thành phần chính của bài toán PCA thông qua SVD, tính duy nhất của nghiệm của PCA, và cũng đã giới thiệu về thuật toán tìm PCA của một ma trận.

4 Một số ứng dụng của SVD và PCA: Khóa luận đã giới thiệu về một số ứng dụng thực tế của SVD và PCA, bao gồm việc sử dụng SVD trong bài toán xấp xỉ hạng thấp tốt nhất của ma trận, ứng dụng của SVD trong xử lý ảnh, và cách sử dụng PCA trong bài toán nhận dạng khuôn mặt thông qua phương pháp Eigenface.

Mặc dù chúng tôi đã cố gắng trình bày khóa luận một cách đầy đủ và chi tiết, nhưng khó tránh khỏi những thiếu sót do hạn chế về thời gian và trình độ. Chúng tôi rất mong nhận được những ý kiến đóng góp từ Quý Thầy, Cô và các bạn để khóa luận có thể hoàn thiện hơn.

Tài liệu tham khảo

[1] Trần Hồng An, *Giáo trình đại số tuyến tính*, Tài liệu lưu hành nội bộ Trường Dại học Công Nghệ Thông Tin, (2015).

[2] Lê Thanh Hiếu, *Bài giảng đại số tuyến tính*, Tài liệu lưu hành nội bộ, (2019).

[3] Nguyễn Hữu Việt Hưng, *Giáo trình đại số tuyến tính 1,2*, Nhà xuất bản Đại học Quốc gia Hà Nội (2000).

[4] Nguyễn Trần Nhật Linh, *Phân tích giá trị kì dị và một số ứng dụng*, Luận văn thạc sĩ (2016).

[5] Nguyễn THị Ái My, *Một số mô hình phân tích thành phần chính ba chiều*, Luận văn thạc sĩ (2020).

[6] Nguyễn Duy Thuận, Phi Mạnh Ban, Nông Quốc Chinh, *Dại số tuyến tính*, Nhà xuất bản Đại học Sư phạm, (2003)

[7] Vũ Hữu Tiệp, *Machine Learning cơ bản*, Nhà xuất bản Khoa học và Kỹ thuật, (2018).

[8] Nguyễn Đình Trí (Chủ biên), Tạ Văn Đĩnh, Nguyễn Hồ Quỳnh *Toán học cao cấp Tập II* Nhà xuất bản Giáo dục Việt Nam.(2014).

[9] Cristianini, N., Shawe-Taylor *An Introduction to Support Vector Machines and other kernel-based learning methods*, Cambridge University Press, Cambrigde, 2000.

[10] Eckart, C.; Young, G. *The approximation of one matrix by another of lower rank.* Psychometrika. 1 (3): 211–8. doi:10.1007/BF02288367. S2CID 10163399.(1936)

[11] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*

[12] Aapo Hyvärinen, *Principal component analysis*, Based on material from the book Natural Image Statistics to be published by Springer-Verlag in 2009.

[13] David C. Lay, *Linear algebra and its applications*, Addison-Wesley, Reading, MA, 1994.

[14] Stewart G. W., *On the early history of the singular value decomposition*, SIAM Review, 35, p. 551-566, 1993.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, lukasz Kaiser, and Illia Polosukhin. *Attention is all you need.* In Advances in neural information processing systems, pp. 5998–6008,2017.

[16] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. *Scaling neural machine translation.* In Proceedings of the Third Conference on Machine Translation: Research Papers, 2018.

[17] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. *Exploring the limits of transfer learning with a unified text-to-text transformer.* arXiv preprint arXiv:1910.10683, 2019.

[18] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, Hao Ma *Linformer: Self-Attention with Linear Complexity* arXiv preprint arXiv:2006.04768, 2020.

[19] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. *Language models are few-shot learners.* arXiv preprint arXiv:2005.14165, 2020.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *Bert: Pre-training of deep bidirectional transformers for language understanding.* In Proceed-

ings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies 2019

[21] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. *Language models are unsupervised multitask learners.* OpenAI Blog, 2019.

[22] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the knowledge in a neural network.* arXiv preprint arXiv:1503.02531, 2015.

[23] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. *Generating long sequences with sparse transformers.* arXiv preprint arXiv:1904.10509, 2019.

[24] Jiezhong Qiu, Hao Ma, Omer Levy, Scott Wen-tau Yih, Sinong Wang, and Jie Tang. *Blockwise self-attention for long document understanding.* arXiv preprint arXiv:1911.02972, 2019.

[25] Iz Beltagy, Matthew E Peters, and Arman Cohan. *Longformer: The long-document transformer.* arXiv preprint arXiv:2004.05150, 2020.

[26] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. *Reformer: The efficient transformer.* In International Conference on Learning Representations, 2020