Tecnológico de Monterrey

Análisis de Componentes Principales para Reducción de Dimensiones en Reconocimiento Facial

Santiago Mora Cruz[1] and Gabriel Reynoso Escamilla[1]

[1] *Tecnologico de Monterrey, Escuela de Ingeniería y Ciencias*

Noviembre de 2023

Abstract— The Principal Component Analysis (*PCA*) is a statistical method aimed at reducing the dimensionality of a dataset by analyzing the variance and covariance of its variables. This report illustrates dimensionality reduction using a dataset that contains images of faces and applying *PCA* to it. Through computational modeling in Python, it transforms the dataset to smaller dimensions, graphically visualizing the reduction in variance. The primary objectives are to explore the uses of *PCA*, optimize storage in a facial recognition database, compare effectiveness based on retained variance, and contrast recursive *PCA* analysis with standard *PCA*. Results show that *PCA* efficiently preserves facial features with fewer dimensions, making it a valuable preprocessing tool. Additionally, recursive *PCA* retains slightly more variance, emphasizing its potential in specific applications. This research contributes to understanding the practical implications of *PCA* for facial recognition databases and broader applications.

Keywords—Principal component analysis, Dimensionality reduction, Facial recognition, Optimization

I. Introducción

El análisis de componentes principales es un método estatístico que busca reducir la dimensión de una base de datos por medio de transformaciones lineales [1]. Para mantener la mayor cantidad de información, se analiza la varianza y covarianza del conjunto de variables [2]. Una manera de ilustrar la reducción de dimensión es mediante las gráficas de una base de datos aletaorios, en primer lugar se elaboró un *dataframe* con 200 muestras dependientes de 3 variables, todas aleatorias, este es el primer gráfico [Fig. 1]

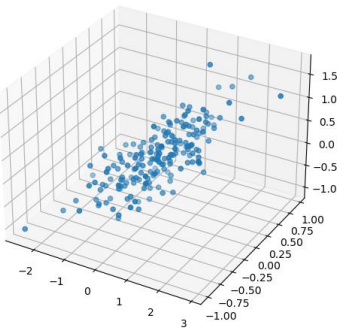

Fig. 1: Gráfica de 200 muestras, en tres dimensiones

Después se utilizan funciones de *scikitlearn* (ver sección II. b. Modelación Computacional) para reducir dicha base de datos a dos y una dimensiones guardandolas como nuevos *dataframes* de esta manera se puede visualizar la reducción en dimensiones y varianza [Fig. 2].

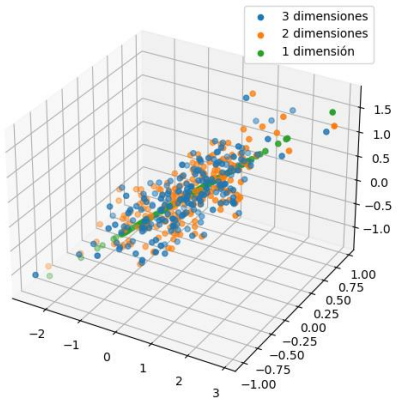

Fig. 2: Gráfica en tres dimensiones con la base original y sus reducciones a dos y una dimensión.

Por ultimo se vuelven a graficar ahora en dos dimensiones, en este gráfico se puede apreciar la perdida de varianza al reducir la dimensionalidad [Fig. 3].

Es importante recalcar que el *PCA* no es el fin del análisis de la base de datos, es un medio utilizando para investigaciones o proyectos más grandes.

Este programa tiene como objetivo principal:

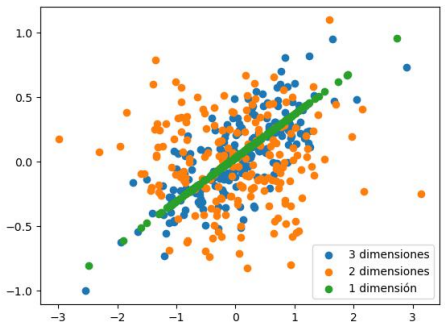

Fig. 3: Gráfica en dos dimensiones con la base original y sus reducciones a dos y una dimensión.

1. Expandir en los usos del Análisis de Componentes Principales, demostrando que se puede optimizar el uso de almacenamiento en una base de datos con fotografías de rostros de personas mediante la reducción de dimensionalidad haciendo uso del Análisis de Componentes Principales.
2. Comparar la efectividad de la reconstrucción de las imágenes a partir del número de componentes a los que se reduce, evaluando la varianza explicada acumulada que se conserva al aplicar la transformación de datos mediante el uso de Análisis de Componentes Principales.
3. Comparar la efectividad de un transformación de Análisis de Componentes Principales regular y una transformación recursiva, es decir una transformación de Análisis de Componentes Principales aplicada a un objeto al que previamente se le aplicó la misma transformación. Asimismo, comparar la efectividad de la transformación dependiendo del número de iteraciones que se realicen.

A lo largo de las secciones de metodogía y de análisis de resultados se ilustrará como es que se transformó la base de datos utilizada para indagar en estos objetivos.

II. Metodología

a. Matemáticas

Supongamos que tenemos una base de datos con C componentes y M mediciones, se necesitaría un número D de componentes para reproducir la variabilidad total de un sistema, esta variabilidad puede ser reproducida por una cantidad menor M de componentes manteniendo la mayor información, remplazando una base de datos de M mediciones de D componentes por una reducida de N mediciones con C componentes, esto puede ser logrado mediante el Análisis de Componentes Principales. Para explorar la varianza y covarianza de este conjunto de componentes, podemos realizar una serie de combinaciones lineales [2].

Tenemos un conjunto de vectores $X = x_1, x_2, ..., x_n$ con $n = 1, 2, ..., i$ con dimensión D, su matriz de covarianza Σ es

$$\begin{bmatrix} \text{Var}(x_1) & \text{Cov}(x_1,x_2) & ... & \text{Cov}(x_1,x_n) \\ \text{Cov}(x_2,x_1) & \text{Var}(x_2) & ... & \text{Cov}(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(x_n,x_1) & \text{Cov}(x_n,x_2) & ... & \text{Var}(x_n) \end{bmatrix} \quad (1)$$

y sus eigenvalores $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_D$.

Consideramos las siguientes combinaciones:

$$Y_1 = a_1^T X Y_2 = a_2^T X \vdots Y_D = a_D^T X$$

Buscamos reducir este espacio a un subespacio con una dimension M en el que $M < D$ buscando que este subespacio mantenga la máxima varianza. La varianza de las combinaciones es la siguiente:

$$\begin{aligned} \text{Var} &= \frac{1}{D}\sum_{i=1}^{D}(u_i^T x_n - u_i^T \bar{x})^2 \\ &= \frac{1}{D}\sum_{i=1}^{D}(u_i^T (x_n - \bar{x}))^2 \\ &= \frac{1}{D}\sum_{i=1}^{D}(u_i^T (x_i - \bar{x})(x_i - \bar{x})^T u_i) \\ &= u_i^T \Sigma u_i \end{aligned} \quad (2)$$

Por lo tanto

$$\text{Var}(Y_i) = a_i^T \Sigma a_i$$

$$\text{Cov}(Y_i, Y_k) = a_i^T \Sigma a_k$$

para $i, k = 1, 2, ..., D$

El primer componente será la combinación linear $a_1^T X$ que maximize

$$\text{Var}(Y_1) \text{ sujeto a } a_1^T a_1 = 1$$

El segundo componente será la combinación linear $a_2^T X$ que maximize

$$\text{Var}(Y_2) \text{ sujeto a } a_2^T a_2 = 1 \text{ y } \text{Cov}(Y_1, Y_2)$$

Hasta el paso i, una combinación linear $a_i^T X$ que maximize

$$\text{Var}(Y_i) \text{ sujeto a } a_i^T a_i = 1 \text{ y } \text{Cov}(Y_i, Y_2 k)$$

donde $k < i$.

Para dicha maximización se utiliza el multiplicador de Lagrange resultando en la siguiente expresión:

$$u_i^T \Sigma u_i + \lambda (1 - u_i^T u_i)$$

la cual se deriva con respecto a u

$$\frac{d}{du} = 2\Sigma u_i - \lambda 2 u_i$$

igualamos la expresión a cero para encontrar el máximo

$$\frac{d}{du} = 2\Sigma u_i - \lambda 2 u_i = 0$$

$$\Sigma u_1 = \lambda u_i$$

donde λ es un eigenvalor de la matriz de covarianza, por lo tanto el eigenvalor máximo es la varianza máxima sujeta a $a_i^T a_i = 1$ para el componente principal [2].

Tecnológico de Monterrey

b. Modelación Computacional

Se utilizó el lenguaje de programación *Python* [3] para aplicar el Análisis de Componentes Principales a la base de datos con la que se trabajó: *Labeled Faces in the Wild* [4]. Para esto, se hizo uso de las librerías *numpy* [5], *matplotlib* [6] y *scikitlearn* [7] inspirándonos en el trabajo realizado en el libro *Python Data Science Handbook* [8].

La base de datos consiste en una serie de 13,000 fotografías de rostros de dimensión de alrededor de 3000 componentes cada una. En lo que consiste el Análisis de Componentes Principales que se desarrolló en el programa es en, para cada transformación a N componentes, realizar el siguiente procedimiento general:

1. Crear una variable `pcaN` con el número de componentes al que se deseé transformar usando la función predeterminada de la librería `scikitlearn`:

   `pcaN = PCA(N)`

2. Se ajusta esta variable a los datos con los que se trabajará utilizando la función correspondiente:

   `pcaN.fit(people.data)`

3. Mediante la transformación `pcaN` definida anteriormente, se extraen los componentes de los objetos dentro de la base de datos y se guardan en la variable `componentesN`:

   `componentesN = pcaN.transform(people.data)`

4. Una vez que se tienen estos componentes, se reconstruyen los objetos dentro de la base de datos haciendo la transformación inversa de `pcaN` y se guardan en la variable `proyeccionN`:

   `proyeccionN = pcaN.inverse_transform(componentesN)`

5. Para cuantificar la efectividad de cada tranformación a cierto número de componentes se evalúa la varianza explicada acumulada, es decir la varianza total que se retiene de la base de datos original. Esto se desarrolló en el programa mediante la función:

   `np.cumsum(pcaN.explained_variance_ratio_)`

Una vez definidos los métodos generales que se utilizaron para transformar la base de datos haciendo uso del Análisis de Componentes Principales, los siguientes son los métodos particulares que se llevaron a cabo en el programa.

1. Reducción a 300 componentes

Para ejemplificar el uso de la transformación de la base de datos mediante el uso del Analisis de Componentes Principales en su forma más simple, se hizo una reducción de componentes de los objetos de la base de datos a 300 componentes, para observar como se ven las reconstrucciones de estos objetos a partir de menos del 10% de los componentes. Después de la transformación, se evaluó la varianza explicada acumulada para ver cuánta varianza de la total retenían los componentes [Fig. 4]. En la sección de Análisis de Resultados se puede observar como se ve esta transformación en las caras de la base de datos.

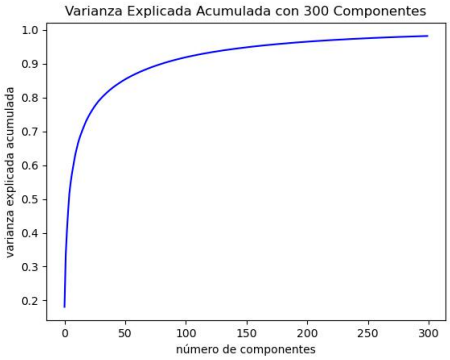

Fig. 4: Varianza explicada acumulada de 300 componentes

2. Comparación entre 50, 100, 200 y 300 componentes

Con el fin de comparar las transformaciones de los objetos de la base de datos a 50, 100, 200 o 300 componentes principales, se aplicó la transformación a estos objetos con cada una de estas cantidades de componentes. Una vez realizadas estas transformaciónes, se calculó la varianza explicada acumulada correspondiente a cada transformación [Fig. 5]. En la sección de Análisis de Resultados se puede observar esta comparación en las caras de la base de datos.

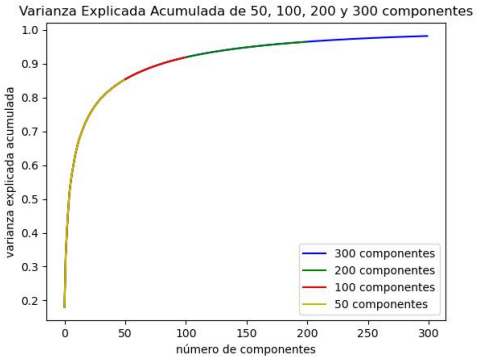

Fig. 5: Comparación en la varianza explicada acumulada de 50, 100, 200 y 300 componentes

3. Comparación entre transformación PCA normal y recursiva

Se define una transformación *PCA* recursiva como una transformación de un objeto a N componentes principales que se reconstruye mediante la transformación *PCA* inversa, y que luego se le vuelve a aplicar una transformación *PCA* de ahora M componentes principales, siendo M menor que N. Así, en la sección de código correspondiente se hizo una transformación recursiva de los objetos de la base de datos de, primero, 600 componentes principales y después ese mismo a 200 componentes. Por simplicidad, se le llamará transformación de 600-200 componentes. Se hizo una segunda transformación recursiva, ahora de 1000 a 750 a 500 a 200 componentes principales. Por simplicidad, se le llamará transformación de 1000-200 componentes.

Estas dos reconstrucciones se compararon con una transformación simple a 200 componentes principales para observar cuál retiene más varianza, y cuánta más varianza se retiene [Fig. 6]. De nuevo, en la sección de Análisis de Re-

sultados se puede observar esta comparación en las caras de la base de datos.

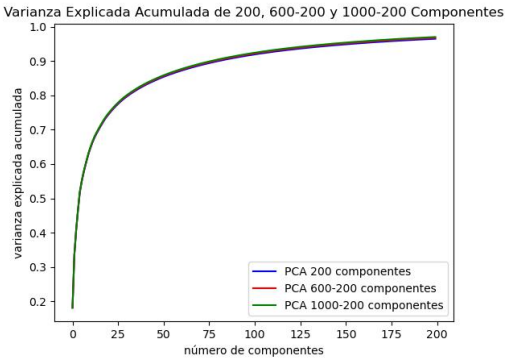

Fig. 6: Comparación en la varianza explicada acumulada de 200, 600-200 y 1000-200 componentes

III. Análisis de Resultados

1. Reducción a 300 componentes

La figura 7 [Fig. 7] compara las imágenes originales del conjunto de datos con el que se trabajó (≈3000 componentes), con las imágenes reconstruidas por el programa a partir de 300 componentes principales.

En esta visualización se aprecia la eficiencia de la reducción de dimensionalidad. Las imágenes de la fila inferior tienen solamente ≈10% de los componentes de la fila superior y, a pesar de esto, contienen suficiente información para se pueda reconocer a simple vista a los individuos de la imagen. Esto implica que se puede hacer mucho más eficiente el proceso de entrenar a un algoritmo de *machine learning*, ya que se entrenaría con datos de 300 dimensiones en vez de 3000. Esto también significa que se podrían almacenar 10 veces más imágenes de 300 dimensiones en vez de 3000, haciendo mucho más eficients el almacenamiento de los datos.

Fig. 7: Imágenes originales de la base de datos vs. Reconstrucción a partir de 300 componentes principales

2. Comparación entre 50, 100, 200 y 300 componentes

La figura 8 [Fig. 8] compara las recosntrucciones de las imágenes hechas a partir de 300, 200, 100 y 50 componentes, respectivamente.

En esta visualización se aprecia la importancia de tener una retención significativa de la varianza al aplicar una transformación *PCA* a un conjunto de datos. Se resalta la importancia de retener, al menos, el 95% de la varianza de los datos, ya que en las reconstrucciones de 100 y 50 componentes se observan imagenes de rostros muy difíciles o imposibles de reconocer. Comparando las reconstrucciones de 300 y 200 componentes se observa que, a pesar de que en ambas los rostros son bastante reconocibles, ciertas características como las gafas correspondientes a la quinta y sexta cara se aprecian con mucha más facilidad reteniendo el 2% de la varianza adicional que retiene la reconstrucción de 300 componentes sobre la de 200, por lo que es importante tener en cuenta qué tantos detalles de estos se quieren retener o se pueden perder sin sacrificar tanta información al hacer una transformación *PCA* a un conjunto de datos con el que se va trabajar.

Fig. 8: Comparación entre reconstrucciónes a partir de 50, 100, 200 y 300 componentes principales

3. Comparación entre transformación PCA normal y recursiva

La figura 9 [Fig. 9] compara las reconstrucciones de las imágenes hechas a partir de 200 componentes, de 600 componentes a 200 componentes con una transformación *PCA* recursivo de dos iteraciones y de 1000 a 750 a 500 a 200 componentes con una transformación *PCA* recursivo de 4 iteraciones.

A pesar de que en esta visualización no se puede observar a simple vista, con respecto a la transformación *PCA* simple de 200 componentes, la transformación*PCA* recursiva de 2 iteraciónes conservó ≈0.3% más varianza explicada acumulada y la de 4 iteraciones ≈0.5% más varianza explicada acumulada. A pesar de que estos análisis recursivos requirieron más tiempo para completarse, es cierto que conservaron más varianza que el simple utilizando la misma cantidad de componentes principales, por lo que se puede concluir que es más eficiente una transformación *PCA* recursiva relacionando específicamente el número de componentes y la varianza explicada acumulada. Dicho esto, para este conjunto de datos este incremento mínimo de varianza no representó una diferencia significativa, por lo que depende mucho del conjunto de datos con el que se trabaje si vale la pena el mayor tiempo de análisis que representa la transformación *PCA* recursiva con respecto a la normal.

Fig. 9: Comparación entre reconstrucciónes a partir de 200, 600-200 y 1000-200 componentes principales

Tecnológico de Monterrey

IV. Conclusión

Una vez finalizado el análisis de los resultados obtenidos a partir del procedimiento que se realizó, es pertinente comparar los objetivos principales planteados al inicio del documento con lo obtenido. Se demostró, para esta base de datos, que la reducción de dimensiones mediante el Análisis de Componentes Principales es una herramienta útil para hacer más eficientes los procesos de almacenamiento o de uso de la base de datos para entrenar modelos y algoritmos, ya que se conserva una cantidad de varianza que permite reconstruir, casi en su totalidad, la imagen analizada a partir de sus componentes principales. También se analizó la proporción de varianza que se debe retener para tener una reconstrucción reconocible, y se concluyó que esta debe superar al menos 95%, pero depende de la cantidad de detalles que se quieran conservar y de el uso que se le vaya a dar a las reconstrucciones realizadas. Finalmente, se concluyó que el Análisis de Componentes Principales recursivo efectivamente conserva más varianza en la misma cantidad de componentes comparado a un análisis simple. Aunque para el caso específico de esta base de datos el incremento en varianza fue muy pequeño, sería pertinente hacer el mismo análisis con otro tipo de datos y comparar los resultados.

A. Apéndice

Los archivos de código utilizados para llevar a cabo esta investigación, el archivo `.zip` que contiene los archivos utilizados para crear el reporte final, y el Data Availability Statement están disponibles en [Este repositorio].

References

[1] K. K and V. K, "A novel mutual information based pca approach for face identification." *Multimedia Tools and Applications: An International Journal*, pp. 1 – 17, 2023. [Online]. Available: http://0-search.ebscohost.com.biblioteca-ils.tec.mx/login.aspx%3fdirect%3dtrue%26db%3dedssjs%26AN%3dedssjs.EFC5A824%26lang%3des%26site%3deds-live%26scope%3dsite

[2] D. W. Richard Johnson, *Applied Multivariate Statistical Analysis: Pearson New International Edition*. Pearson Education Limited, 2014.

[3] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[4] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007.

[5] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[6] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[8] J. VanderPlas, *Python Data Science Handbook*. O'Reilly Media, Inc., 2016.