Trabajo práctico 1
Principal Component Analysis

1. Introducción

En diversos contextos y aplicaciones resulta crucial reducir el tamaño de los datos a procesar sin comprometer significativamente información relevante. Este objetivo se puede lograr eliminando los elementos redundantes que no están aportando información útil, pero que, sin embargo, incrementan innecesariamente el tamaño de los datos. Algunas de las razones que justifican la compresión de datos pueden ser:

- Ahorro de espacio de almacenamiento: uso más eficiente de recursos de memoria.
- Transmisión de datos: sean archivos, audio o video en tiempo real, la compresión permite un menor uso del ancho de banda del medio de comunicación.
- Reducción de dimensionalidad: en aplicaciones de aprendizaje automático, es común que las entradas de los modelos (features), como ser imágenes, audio, texto, etc, requieran la compresión de datos. Esto permite limitar el sobreajuste del modelo, tener una mayor eficiencia computacional en el entrenamiento y una mayor concentración de las características más predictivas eliminando aquellas irrelevantes para mejorar la precisión.

Una de las técnicas de procesamiento que permite la reducción del tamaño, manteniendo intacta la mayor parte de la información, se denomina *Análisis de Componentes Principales*, que se detalla en la siguiente sección.

1.1. Breve resumen sobre PCA

El *análisis de componentes principales*, o PCA (Principal Component Analysis), es una técnica estadística utilizada para reducir la dimensionalidad de un conjunto de datos al transformarlo en un nuevo conjunto de variables no correlacionadas llamadas *componentes principales*. Estas componentes capturan la mayor parte de la variabilidad presente en los datos originales, lo que facilita su análisis y visualización. De esta manera, pueden seleccionarse y eliminarse aquellas componentes que poseean poca variabilidad, ya que éstas no aportan demasiada información.

Pensemos en un vector aleatorio $\mathbf{X} = [X_1\ X_2\ ...\ X_L]^T$, de media $\boldsymbol{\mu}_X = \mathbb{E}[\mathbf{X}]$ y matriz de covarianza $C_X = \mathbb{E}[(\mathbf{X} - \boldsymbol{\mu}_X)(\mathbf{X} - \boldsymbol{\mu}_X)^T]$. Esta matriz se puede diagonalizar para obtener la matriz de autovectores V y de autovalores asociados Λ, Ecuaciones 1 y 2. Si se proyecta el vector $\mathbf{X}$ en el espacio de autovectores de C_X, se obtiene un nuevo vector $\mathbf{Y} = V^T\mathbf{X}$, cuya matriz de covarianza $C_\mathbf{Y} = \Lambda$ es diagonal y está compuesta por los autovalores de C_X. Esto implica que las componentes del vector $\mathbf{Y} = [Y_1\ Y_2\ ...\ Y_L]^T$ están descorrelacionadas, siendo las varianzas de cada componente $\sigma^2_{Y_i} = \lambda_i$. Si identificamos aquellas componentes con menor

varianza, puede reducirse el tamaño de los datos eliminando esas componentes y concentrar así la información relevante del vector en las componentes principales.

Para saber qué componentes descartar, pueden fijarse distintos criterios. Una posibilidad es especificar el tamaño final que tendrán los datos reducidos, a costa de que los errores por pérdidas varíen dependiendo de la naturaleza del vector original. La forma más simple de hacer esto es definiendo una matriz U que contenga los K autovectores principales de V asociados a los autovalores de mayor peso. Luego, si proyectamos $\mathbf{X}$ en el espacio de U, obtenemos el vector $\widehat{\mathbf{Y}} = U^T\mathbf{X}$ que conserva casi la misma información que $\mathbf{X}$ pero con un tamaño menor (dado que $\dim(\widehat{\mathbf{Y}}) = K < \dim(\mathbf{X})$). Para revertir el proceso y recuperar el vector original (asumiendo cierto error al ser una compresión con pérdidas) se aplica la transformación inversa $\widehat{\mathbf{X}} = U\widehat{\mathbf{Y}}$.

$$C_X = V\Lambda V^T \tag{1}$$

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \lambda_L \end{bmatrix} \tag{2}$$

Para aplicar este método en un caso práctico, los datos que deseamos procesar (imágenes, audio, etc) deben segmentarse y definir una colección de M vectores $\mathbf{X}_m \in \mathbb{R}^L$, donde $m = 0, 1, ..., M-1$, representando cada uno de ellos una realización particular del vector aleatorio $\mathbf{X}$. Luego de esto, puede aplicarse el método PCA para compresión y descompresión de los datos como se resume a continuación:

Compresión

1. Estimar la media $\boldsymbol{\mu}_{\mathbf{X}}$ del vector aleatorio de acuerdo a la Ecuación 3.
2. Estimar la matriz de covarianza $C_{\mathbf{X}}$ de acuerdo a la Ecuación 4.
3. Diagonalizar la matriz de covarianza $C_{\mathbf{X}}$ para obtener las matrices V y Λ.
4. Definir una matriz U que contenga únicamente los K autovectores (columnas) de V asociados a los autovalores principales.
5. Obtener $\widehat{\mathbf{Y}}_m = U^T\mathbf{X}_m$ proyectando todos los vectores $\mathbf{X}_m$ en el espacio de U.
6. Guardar la colección de vectores $\widehat{\mathbf{Y}}_m$ y la matriz U.

Descompresión

1. Abrir la colección de vectores $\widehat{\mathbf{Y}}_m$ y la matriz U.
2. Volver a transformar cada vector reducido al espacio original aplicando la transformación inversa $\widehat{\mathbf{X}}_m = U\widehat{\mathbf{Y}}_m$.

1.2. PCA aplicado a señales de audio

Para las señales de audio digital, la compresión permite reducir el tamaño o cantidad de bytes que se necesitan para representar esa señal, sin afectar apreciablemente la calidad percibida del sonido. Si bien en la práctica los estándares de compresión normalmente utilizados son MP3, AAC, FLAC, etc, basados en otras técnicas más eficientes, en este trabajo práctico utilizaremos PCA aplicado a compresión de audio como caso de estudio para poder analizar las propiedades del método.

1.2.1. Digitalización de una señal de audio

Una señal de audio digital se representa comúnmente con secuencia de muestras discreta, llamémosla $x(n)$, donde cada muestra almacena la amplitud de la señal en un punto específico en el tiempo. Estas muestras se toman a intervalos regulares a lo largo del tiempo y se almacenan o procesan digitalmente para su reproducción o análisis. El formato de la señal y la frecuencia de muestreo dependerán de la aplicación. En este trabajo utilizaremos archivos de audio en formato WAV (sin compresión), digitalizados a una tasa de muestreo de 44100 Hz y resolución de 16 bits. En el apéndice se muestran algunas funciones en Python y Matlab para la lectura y reproducción de estos archivos.

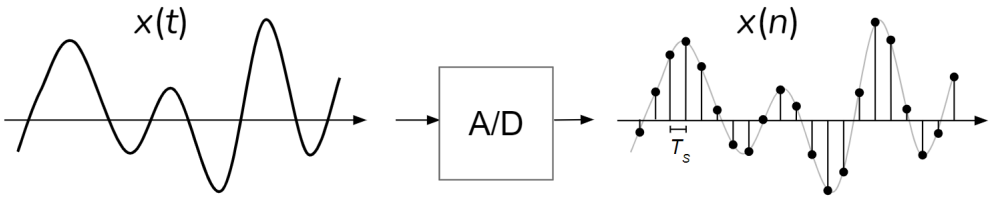

Figura 1: Digitalización de una señal para una tasa de muestreo $f_s = 1/T_s$

1.2.2. Realizaciones del vector X

Para aplicar el método PCA a una señal de audio $x(n)$ en el dominio del tiempo, debemos primero definir cuáles serán los vectores aleatorios. Para ello segmentamos la señal en M tramos, obteniendo en cada tramo una secuencia de corta duración $x_m(n) = x(n - mL)w_L(n)$, donde $w_L(n)$ es una ventana rectangular de L muestras de largo y $m = 0, 1, ..., M - 1$ es el número segmento ventaneado, como se observa en la Figura 2. De esta forma, las muestras temporales contenidas en cada ventana se utilizarán directamente como las componentes de cada vector $\mathbf{X}_m$. Es decir, con la señal de audio completa podremos obtener la colección de realizaciones de $\mathbf{X}$, una por cada tramo de señal, que representarán los M experimentos aleatorios con los que se estimarán la media y covarianza C_X.

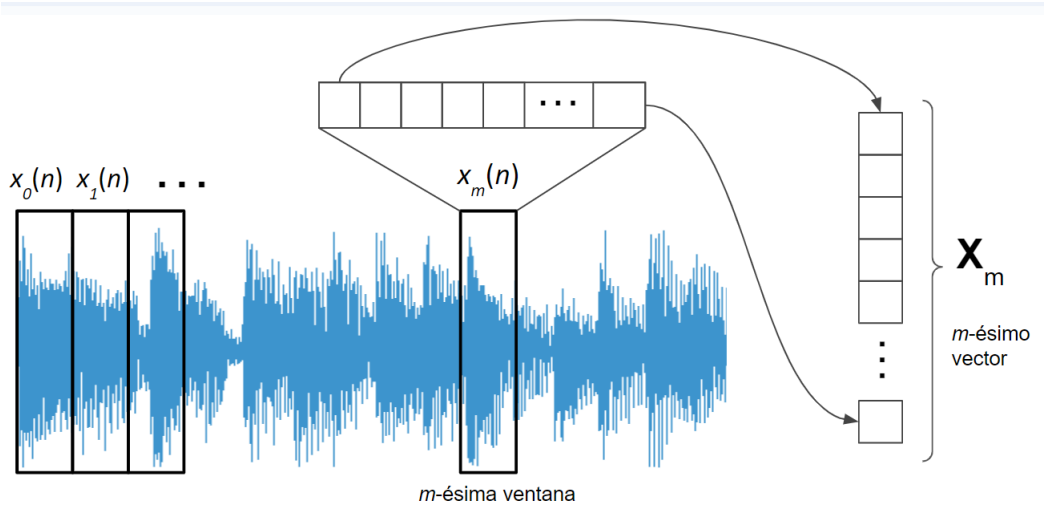

Figura 2: Segmentación de la señal en ventanas $x_m(n)$ cuyas muestras forman los vectores $\mathbf{X}_m$.

1.3. Métricas y estimadores

Media y covarianza En las Ecuaciones 3 y 4 se definen los estimadores para la media y covarianza respectivamente, de un vector aleatorio $\mathbf{X}$ con M realizaciones $\mathbf{X}_m$.

$$\widehat{\boldsymbol{\mu}}_X = \frac{1}{M} \sum_{m=0}^{M-1} \mathbf{X}_m \tag{3}$$

$$\widehat{C}_X = \frac{1}{M} \sum_{m=0}^{M-1} (\mathbf{X}_m - \widehat{\boldsymbol{\mu}}_X)(\mathbf{X}_m - \widehat{\boldsymbol{\mu}}_X)^T \tag{4}$$

MSE (Mean Square Error) El error cuadrático medio permite medir el desempeño en términos de la energía promedio del error, Ecuación 5, donde $x(n)$ y $\hat{x}(n)$ rerpesentan la señal original y la reconstruida luego de todo el proceso de compresión y descompresión

$$MSE = \frac{1}{N} \sum_{n=0}^{N-1} (x(n) - \widehat{x}(n))^2 \tag{5}$$

CR (Compression Rate) En este contexto es apropiado definir la tasa de compresión (CR) que cuantifica la proporción de datos que se reduce al comprimir, Ecuación 6.

$$CR = \left(1 - \frac{\text{cantidad de componentes principales (K)}}{\text{cantidad de componentes totales (L)}}\right) \times 100\,\% \tag{6}$$

2. Ejercicios

Ejercicio 1: Correlación

En este ejercicio vamos a analizar gráficamente el concepto de correlación entre componentes de un vector aleatorio generado desde la señal de audio.

(a) Abrir la señal de audio **audio_01_2024a.wav**, normalizarla dividiéndola por su valor eficaz $x(n) = \text{audio}(n)/\text{RMS}(\text{audio(n)})$. Luego segmentar la señal para guardar la colección de vectores $\mathbf{X}_m$ con $L = 2$. Nota: definimos vectores de dimensión 2 con el propósito de visualizar la correlación entre las dos componentes del vector. Luego este análisis se podrá generalizar a mayores dimensiones.

(b) Hacer un gráfico de dispersión para ver gráficamente la correlación entre las dos componentes del vector $\mathbf{X}_m = [X_0^{(m)}\ X_1^{(m)}]^T$ (muestras vecinas en el dominio del tiempo). Graficar también los histogramas de ambas componentes $X_0^{(m)}$ y $X_1^{(m)}$. Sugerencia: para una mejor comprensión de los resultados, puede ubicar el histograma de $X_0^{(m)}$ en la parte posterior del gráfico y el histograma de $X_1^{(m)}$ a la derecha, respecto del gráfico de dispersión.

(c) Obtenga la matriz de covarianza C_X del vector aleatorio definido previamente y luego las matrices de autovectores V y de autovalores asociados Λ.

(d) Obtenga un conjunto de vectores Y_m cuyas componentes están descorrelacionadas. Luego haga un gráfico de dispersión de Y y los histogramas asociados a cada elemento de los vectores resultantes de forma análoga al punto (b). Observe los resultados y saque conclusiones. Note que solo se pide proyectar en el espacio de autovectores, pero no aplicamos la reducción de componentes aún.

Ejercicio 2: Compresión

En este caso vamos a aplicar la compresión mediante PCA con un vector aleatorio de mayor dimensión. Vamos a fijar un tamaño de ventana $L = 1323$ muestras (correspondiente a 30 ms de señal a 44100 Hz).

(a) Aplicar el método PCA: Asumiendo $CR = 70\,\%$, obtener la colección de vectores $\widehat{\mathbf{Y}}_m$ de componentes principales. Utilizar como señal el archivo **audio_02_2024a.wav**.

(b) Volver a transformar los vector $\widehat{\mathbf{Y}}_m$ al espacio original y reconstruir la señal de audio. Reproducir las señales original y reconstruida para evaluar subjetivamente las diferencias.

(c) Repita los puntos anteriores pero para tasas de compresión $CR = 90\,\%$, $CR = 95\,\%$. Analice nuevamente los resultados al reproducir las señales reconstruidas luego de la compresión.

Ejercicio 3: MSE vs CR

(a) Aplicar la compresión seguida de la descompresión para los archivos **audio_01_2024a.wav**, **audio_02_2024a.wav** y **audio_03_2024a.wav**, utilizando diferentes tasas (recuerde normalizar cada señal). Suponga los siguientes casos: $CR = \{10\,\%,\ 20\,\%, ..., 90\,\%\}$. Haga un gráfico MSE vs CR para cada audio y compárelos en simultáneo.

(b) Qué puede concluir acerca de la calidad de la reconstrucción observando el MSE entre los distintos audios?

3. Conclusiones

Como conclusiones, elabore un resumen breve y conciso comentando características que considere relevantes del método propuesto en este trabajo y los resultados obtenidos, así como dificultades encontradas y cómo fueron abordadas.

4. Apéndice

4.1. Matlab

```
% Abrir el archivo WAV
[data, fs] = audioread('archivo.wav');
```

```
% Reproducir señal de audio
sound(data, fs)
```

```
% Obtener autovectores (V) y autovalores (D) de una matriz A
[V, D] = eig(A);
```

4.2. Python

```
import soundfile as sf
# Abrir el archivo WAV
data, fs = sf.read("archivo.wav")
```

```
import sounddevice as sd
# Reproducir señal de audio
sd.play(data, fs)
sd.wait()
```

```
import numpy as np
# Obtener autovectores (V) y autovalores (D) de una matriz A
D, V = np.linalg.eig(A)
```

5. Normas y material entregable

- **Informe**: debe ser conciso y comentar los resultados solicitados. El informe debe entregarse en formato PDF (**no se aceptarán otros formatos**) y con nombre: `TP1_GXX.PDF` (donde `XX` es el número de grupo). No debe agregarse código en el informe.
- **Código**: Los archivos de código utilizados deben ser en formato `.m` de Matlab/Octave (o alternativamente `.py` si usara lenguaje Python). El código debe incluirse junto al informe en un archivo ZIP (con mismo nombre que el informe) que deberá subirse al campus.

Referencias

[1] Steven M. Kay. Intuitive probability random processes using MATLAB. Springer 1951.