Tarea 2 IPD-477

Eduardo Soto

10 de diciembre de 2018

*

1. Pregunta 1

En primer lugar es necesario generar el conjunto de imágenes $r(x, y)$ mediante la fórmula:

$$r(x, y) = G_w(x, y)\,(G_m(x, y) * \omega(x, y)) \tag{1}$$

Con G_w, G_m funciones gaussianas bidimensionales y $\omega(x, y)$ una matriz de ruido gaussiano.

En el caso de la convolución, esta es equivalente a utilizar un filtro gaussiano, o enfoque gaussiano, dicha función ya se encuentra implementada en la librería de *scipy*. Para el caso de la ventana gaussiana G_w, se hace uso de una función de multiplicación de matrices del módulo *numpy*.

El desarrollo y las etapas de generación de las imágenes $r(x, y)$ pueden verse en la Figura 1.

Una vez generadas 2000 imágenes de este tipo se procede a entrenar una red mediante el algoritmo Hebbiano. Para esto se inicializan la matriz de los pesos W y el vector de salidas Y. Tal que, dado la matriz de entradas R, con vectores de entrada r, se tiene:

$$Y = W^T r \tag{2}$$

Por lo que se utiliza un dato junto con los pesos generados de forma aleatoria para inicializar Y.

Luego se aplica en recursión para cada dato r el algoritmo Hebbiano descrito en la publicación, de forma matricial, es decir, en cada iteración calcular:

$$\Delta W(t) = \beta\left(Y(t)r^T - LT(Y(t)Y(t)^T)W(t)\right) \tag{3}$$

1. Detalle de resultados

$$W(t+1) = W(t) + \Delta W(t) \tag{4}$$

$$Y(t+1) = W(t+1)^T r \tag{5}$$

Adicionalmente se requiere normalizar la matriz de pesos en cada iteración a modo de que su magnitud no aumente hasta el infinito.

Los resultados de los pesos se pueden encontrar en la carpeta *weigths* y pueden verse algunos en la Figura 2. Puede apreciarse que los pesos de las salidas se concentraron el la silueta central, lo que aproxima bastante bien la cadena de imágenes esperada.

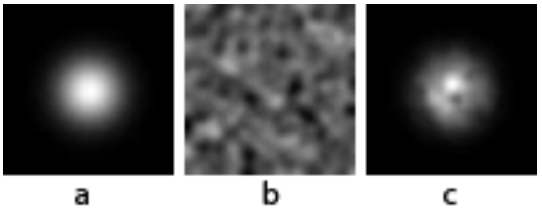

Figura 1: (a),Ventana Gaussiana. (b) Ruido Gaussiano convolucionado con función gaussiana. (c) Imágen final

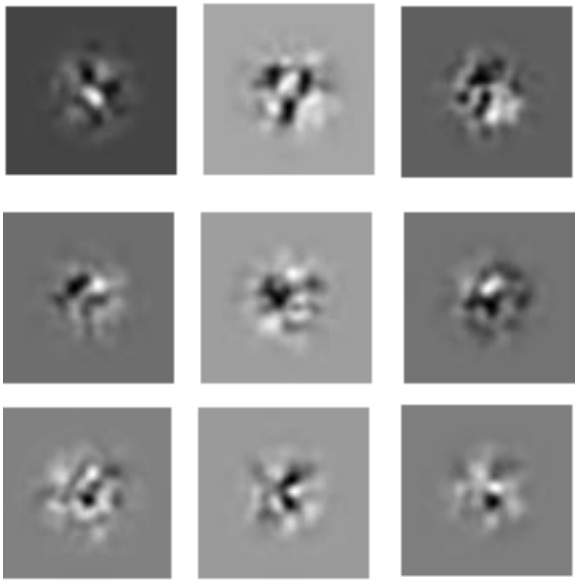

Figura 2: Ejemplos de pesos para entradas $r(x, y)$

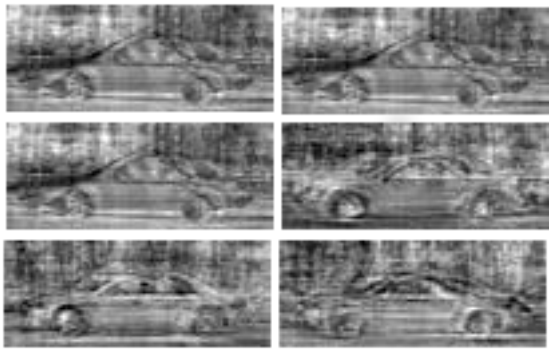

Figura 3: Ejemplos de pesos para entradas imágenes *CarData*

2. Pregunta 2

Se sigue el mismo procedimiento que la sección anterior pero esta vez se modifican los criterios dimensionales y las entradas pasan a ser el banco de imágenes de automóviles.

Los resultados de los pesos se encuentran en la carpeta *Carweigths* y algunos pueden verse en la Figura 3.

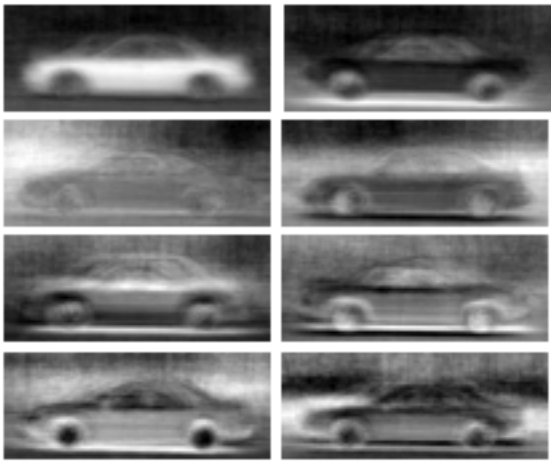

Figura 4: Ejemplos de componentes principales para *CarData*

3. Pregunta 3

El análisis de componentes principales se efectúa en MATLAB luego de que en Python los resultados esperados no pudieron ser obtenidos, intentando de diversas formas. Se presume que puede ser causa de los métodos numéricos utilizados en el cálculo de los valores propios de las matrices de datos.

El algoritmo de implementado en el script de MATLAB se encarga de calcular la matriz de covarianza y obtener de esa matriz los vectores propios los cuales, ordenados por sus respectivos valores propios, corresponden a los componentes principales de los datos en orden.

Los 16 componentes principales obtenidos como vector se transforman a matriz y posteriormente a imagen. Los resultados se encuentran en la carpeta *PCA16*. Un ejemplo de algunos obtenidos se aprecia en la Figura 4.

Se aprecian similitudes entre las imágenes de los pesos obtenidos por el algoritmo Hebbiano y lo obtenido mediante PCA aunque en el caso de este último método se aprecian siluetas de mayor contraste. Esto puede deberse a que no haya existido convergencia completa en el cálculo recursivo y porque los métodos numéricos utilizados en Python y en MATLAB difieren entre sí.