Aprendizaje no supervisado

Pablo Chehade
pablo.chehade@ib.edu.ar
Redes Neuronales, Instituto Balseiro, CNEA-UNCuyo, Bariloche, Argentina, 2023

EJERCICIO 1: NO CONTROLADO RESPECTO A NOTION

Se entrenó de manera no supervisadauna red neuronal lineal de una sola capa con cuatro entradas y una salida. Los datos de entrada presentan una distribución gaussiana con matriz de correlación Σ

$$\Sigma = \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

El autovector asociado al mayor autovalor de Σ es $\vec{v} = \frac{1}{2}(1,1,1,1)$.

A partir de un vector de pesos inicial $\vec{w} = (w_1, w_2, w_3, w_4)$, donde cada componente w_j adopta un valor aleatorio no mayor a 0,01, se aplicó la regla de Oja:

$$\Delta w_j = \eta V(\xi_j - V w_j).$$

Aquí, η representa la tasa de aprendizaje, V es la salida de la red y ξ_j es la componente j del dato de entrada ξ. Bajo estas condiciones, se espera que el vector de pesos $\vec{w}$ tienda al autovector $\vec{v}$.

En la figura 1a, se ilustra la evolución del módulo de los pesos, las modificaciones Δw_j y la diferencia entre w_j y v_j, para $\eta = 0{,}001$. Tras un período transitorio, se observa que las componentes de $\vec{w}$ tienden hacia el valor 0,5, indicando convergencia hacia $\vec{v}$. Dependiendo de las condiciones iniciales, el vector $\vec{w}$ tiende a alinearse en la dirección $+\vec{v}$ o $-\vec{v}$. Debido a esto se graficó el módulo de cada componente. Posteriormente, en estado estacionario, los pesos continúan modificándose. Por ello, en la figura 1b, se procesan los datos anteriores realizando un promedio en cada paso de tiempo de los 200 pasos adyacentes. Esto resulta en un comportamiento más suave y una reducción notable en las variaciones de Δw_j de hasta un orden de magnitud. Luego de este procesamiento de los datos se confirma nuevamente que $\vec{w}$ tiende a $\vec{v}$

Por último, en la figura 1c, se evolucionó el sistema con $\eta = 0{,}01$. El sistema muestra una convergencia más rápida y variaciones Δw_j de mayor magnitud. Además, se determinó que existe un valor superior límite para η, más allá del cual el sistema tiende a diverger.

EJERCICIO 2

Se entrenpo una red neuronal con dos neuronas de entrada y 10 de salida empleando el algoritmo de Kohonen. Las neuronas de salida están dispuestas sobre una línea.

Los datos de entrada se generaron con la distribución

$$P(\xi) =$$

Se comenzó con pesos pequeños y se actualizaron mediante el algoritmo

$$\delta_w j =$$

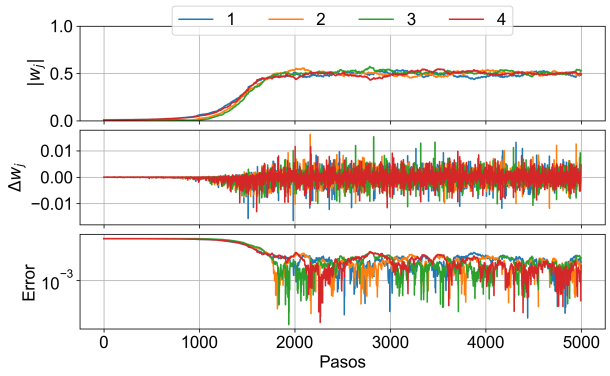
donde η es la tasa de aprendizaje y $\Gamma(i, i^*)$ es la función de vecindad dada por

$$\Gamma(i, i^*) =$$

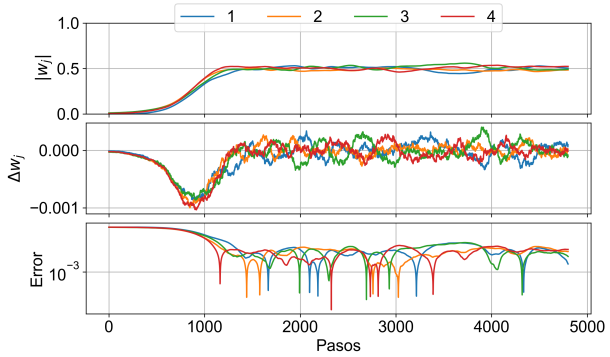
APÉNDICE

A continuación se desarrolla el código empleado durante este trabajo implementado en Python.

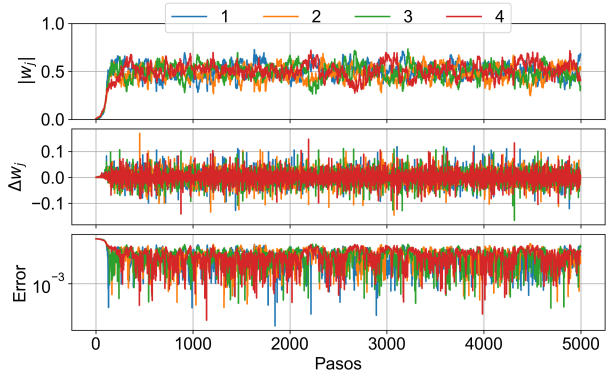
```
#Import libraries
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import tensorflow as tf
```



(a)



(b)



(c)

Figura 1: Evolución del módulo de los pesos w_j, las modificaciones Δw_j y la diferencia entre w_j y v_j durante el entrenamiento de la red neuronal. En a se empleó una tasa de aprendizaje de $\eta = 0{,}001$. En b se empleó la misma tasa de aprendizaje pero además se procesaron los datos promediando en cada paso de tiempo sobre los 200 pasos adyacentes. En c se empleó una tasa de aprendizaje de $\eta = 0{,}01$.