DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Redes Neuronales

Trabajo práctico 2

Resumen

Entrenamiento de Redes Neuronales

Integrante	LU	Correo electrónico
Negri, Franco	893/13	franconegri2004@hotmail.com
?, ?	?/?	?@?.com

Palabras claves:

TP

Índice

1. Introduccion

En este trabajo, nos disponesmos a utlizar diferentes tecnicas de aprendisaje no supervisado para clasificar textos con ciertas caracteristicas. Los mismos consisten en descripciones de diversas companías y nuestro objetivo será lograr clasificar cada una en una categoria correspondiente. Contamos, ademas, con las verdaderas categorias de cada compañía para realizar una validación de los datos.

2. Desarrollo

2.1. PCA

EL primer modelo utilizado para intentar clasificar los datos será el de Analisis de Componentes Principales. Para ello utilizamremos dos algoritmos basados en aprendisaje Hebbiano y reduciremos las instancias de entrenamiento a 3 dimenciones. Lo que esperamos observar es que aquellas instancias que pertenecen a una misma clase de empresa se encuentran cercas unas de otras, pudiendo observar ñuvesde puntos bien definidas.

2.1.1. Implementación

En particular los algoritmos utilizados serán los de Oja y $Sanger$. Teniendo una complegidad computacional identica y siendo los algoritmos muy similares, lo distintivo entre estos dos metodos es que $Sanger$ ordenará las componentes prinsipales de mayor a menor de acuerdo a sus autovalores mientras que Oja no.

El pseodocodigo utilizado para aprendisaje del algoritmo Oja será:

1: Para toda instancia de entrenamiento, x
2: $y = x.W$
3: $\tilde{x} = y.W^T$
4: $\Delta W = learning\_rate((x - \tilde{x})^T.y$

Algorithm 1: Algoritmo De Oja

Mientras que el de $Sanger$

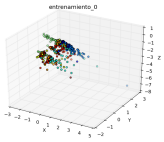
Utilizando el paquete numpy de python es posible traducir este codigo de manera casi exacta y de esa manera aprovechar las optimizaciones matriciales que se realizan sobre los datos.

2.1.2. Experimentación

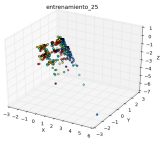
Para la experimentación entrenamos la red con parte del set de datos que nos fue entregado

1: U = Matriz Triangular Superior Con 1s
2: Para toda instancia de entrenamiento, x
3: $y = x.W$
4: $\tilde{x} = W(y^T.U)$
5: $\Delta W = learning\_rate((x^T - \tilde{x}).y$

Algorithm 2: Algoritmo De Sanger

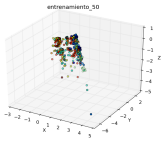


(a) A subfigure

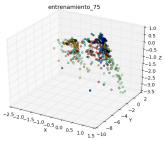


(b) A subfigure

Figura 1: A figure with two subfigures



(a) A subfigure



(b) A subfigure

Figura 2: A figure with two subfigures

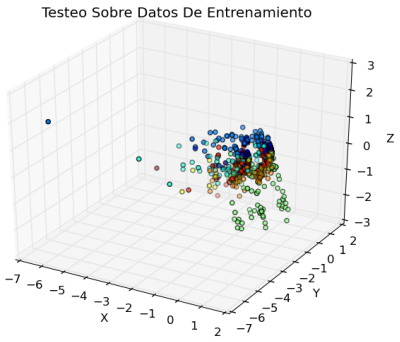
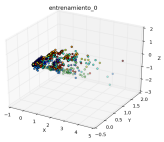
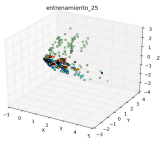


Figura 3: A figure

Sanger

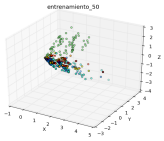
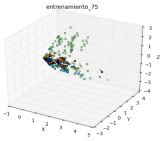
2.2. Mapeo de Características

En este apartado construiremos un modelo de mapeo de caracteristicas auto-organizado con la intención de clasificar los documentos en un arreglo de dos dimenciones. Para ello utilizaremos el algoritmo de Kohonen sobre los datos de entrenamiento y una vez que la red haya convergido, graficaremos los las respuestas obtenidas en el plano.

(a) A subfigure (b) A subfigure

Figura 4: A figure with two subfigures

(a) A subfigure (b) A subfigure

Figura 5: A figure with two subfigures

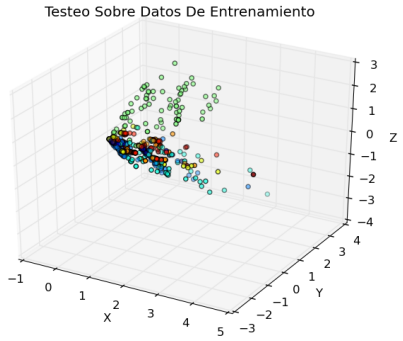


Figura 6: A figure

Convergencia Kohonen

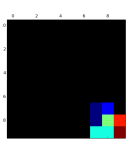
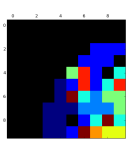
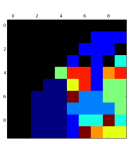
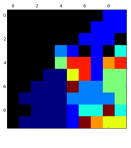
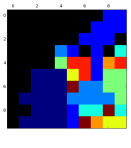






Figura 7: 0 % Figura 8: 25 % Figura 9: 50 % Figura 10: 75 % Figura 11: 100 %

3. Detalles De ejecución

3.1. Detalles de ejecución

Para correr el programa simplmente debe ejecutarse:

python main.py numero_ejercicio

Donde *numero_ejercicio* puede ser $1, 2, 3$ siendo oja, sanger y kohonen respectivamente.

Ademas se cuenta con distintos flags opcionales:

- -i Ruta al archivo de entrenamiento
- -o Ruta al archivo donde guardar la red
- -n Ruta al red a utilizar
- -t Ruta al archivo contra el que testear
- -g Graficar Resultados

Por default el programa buscará el archivo de entrenamiento en la carpeta raiz donde se esta ejecutando el programa, en caso de no brindarse un archivo de testing se partirá este mismo en dos partes, se entrenará con una de ellas y se testeará sobre la otra.

En caso de tener habilitado un entorno grafico, el flag $-g$ mostrará en una ventana los resultados de manera visual.

4. Concluciones