DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 2

16/6/2015 Redes Neuronales Artificiales

Integrante	LU	Correo electrónico
Landini, Federico Nicolás	034/11	federico91_fnl@yahoo.com.ar
Rama Vilariño, Roberto Alejandro	490/11	bertoski@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Índice

1. Introducción

En el presente trabajo se busca atacar un mismo problema utilizando dos modelos distintos de redes neuronales no supervisadas. El problema que intentamos resolver es la clasificación de distintas empresas en categorías a partir de una breve descripción. Dicha descripción estará representada por una *bag of words*, que convierte un texto a un vector en donde cada posición representará la cantidad de apariciones de una palabra en particular. Si bien esto nos permite una representación espacial de un texto, perdemos la información del orden de las palabras.

El primero de los modelos se lo definirá con la finalidad de que permita reducir la alta dimensionalidad de las entradas a sólo 3 dimensiones. Esto lo lograremos utilizando las reglas basadas en aprendizaje Hebbiano de Oja y Sanger[2], con lo que obtendremos, a traves del entrenamiento de la red neuronal, una base de componentes principales. La misma nos permitirá proyectar una entrada del espacio de alta dimensionalidad a un espacio de 3 dimensiones.

El segundo modelo estará basado en aprendizaje competitivo[2]. En este tipo de aprendizaje sólo una unidad de salida, o una por grupo, está activa a la vez. Las unidades de salida compiten por ser aquellas en activarse y el objetivo de estas redes es clusterizar o categorizar los datos de entrada. Entradas similares deberían clasificarse en la misma categoría, por lo que deberían activar la misma unidad de salida. En nuestro trabajo la red realizará un mapeo de características auto-organizado utilizando el algoritmo de Kohonen.

Para resolver los problemas consideramos distintas configuraciones de redes para cada problema. Luego, evaluamos su rendimiento con el fin de comparar los resultados y elegir la configuración que mejor se adaptaba a los conjuntos de datos.

Durante las pruebas decidimos utilizar la técnica de cross-validation [2], con el fin de realizar un análisis más robusto respecto del rendimiento. Esta técnica nos permitió tener una idea más real del nivel de generalidad de los resultados obtenidos con las redes.

2. Desarrollo

2.1. Reducción de dimensionalidad

El propósito de la reducción de dimensiones es intentar lidiar con el conocido "curse of dimensionality". El mismo es un fenómeno que se da en espacios de grandes dimensiones donde los datos son esparsos y clasificar se vuelve un problema grave para cualquier método que requiera significancia estadística. Una de las formas de reducir la dimensionalidad del espacio es a través del análisis de componentes principales.

2.1.1. Análisis de componentes principales

El análisis de componentes principales es un proceso estadístico que usa una transformación ortogonal que convierte observaciones en un conjunto de variables linearmente no correlacionadas llamadas componentes principales. La transformación se define de modo que la primera componente principal tiene la mayor varianza entre todas las variables y cada una de las siguientes componentes tiene la mayor varianza posible entre todas las ortogonales a las componentes anteriores. Luego, el objetivo es calcular una cierta cantidad de componentes principales que contengan la mayor información respecto de los datos para poder expresarlos en función de dichos vectores pertenecientes a una base ortonormal.

2.1.2. Regla de Oja1

La regla de Oja es una modificación a la regla de aprendizaje Hebbiano que previene la divergencia. Mediante la modificación, el vector de pesos se aproxima a una longitud constante $|w| = 1$, sin la necesidad de realizar ninguna normalización. Por otra parte, el vector w ciertamente se acerca al autovector C con el autovalor δ_{max} más alto, es decir el autovector principal. Esta regla también es llamada "Oja1", ya que solamente nos da el primer autovector o componente principal.

Como en este trabajo nos interesan las primeras tres componentes principales, no analizaremos la eficacia de esta regla, si no que iremos con su version M.

2.1.3. Reglas de Oja y Sanger

Como expresamos anteriormente, es deseable tener una red de M-salidas que extraiga las primeras M componentes principales. Sanger y Oja diseñaron redes neuronales de una sola capa y feed-forward para análisis de componentes principales que se encargan de llevar a cargo esta tarea.

La regla de aprendizaje de Sanger es:

$$\Delta w_{ij} = \eta V_i (x_i - \sum_{k=1}^{i} V_k w_{kj})$$

Mientras que la de Oja es:

$$\Delta w_{ij} = \eta V_i (x_i - \sum_{k=1}^{M} V_k w_{kj})$$

Donde V es el vector obtenido al multiplicar la instancia de entrada por la matriz de pesos, M es la cantidad de neuronas y η es el learning rate.

La única diferencia entre ambas es el límite superior de la sumatoria. En ambos casos los w_i vectores convergen a vectores unitarios ortogonales, tales que $w_i^T w_j = \delta_{ij}$. En el caso de la regla de Sanger, los vectores de pesos se aproximan exactamente a las primeras M componentes principales, y es por esto que es mas útil en aplicaciones ya que extrae los mismos en orden. Como nota de color, si algún algoritmo es utilizado en cerebros de seres vivientes, probablemente sea más parecido a la regla de Oja: no hay ninguna ventaja obvia para un ser en tener la información ordenada en base a las componentes principales.

De todas formas, en ambos casos la salida proyecta un vector de entrada x_i en el espacio de las M componentes.

2.1.4. Criterios de convergencia

Para determinar que el algoritmo convergió a vectores ortogonales correspondientes a las primeras componentes principales, utilizamos dos criterios diferentes. Por un lado, calcular el producto entre los vectores obtenidos en cada iteración y ver que el resultado sea menor a un epsilon pequeño. Si todos los productos de los vectores entre sí son pequeños entonces se puede decir que los mismos son suficientemente ortogonales y entonces dejar de iterar. El otro criterio considerado observa los valores de los δW. Si dicha matriz tiene una norma Frobenius menor a un epsilon pequeño entonces se asume que el aprendizaje que se puede obtener de allí es mínimo y por ende no tiene sentido continuar iterando.

2.1.5. Pseudocódigo

A modo de dejar bien asentada la implementación, la presentamos a continuación en formato de pseudocódigo. Sin embargo, no daremos mayores explicaciones ya que no presenta ninguna innovación.

function HEBBIANO(D, M, Ep_m, $regla$, γ, α)
  $Ep = 0$
  $N =$ cols(D)
  $W =$ 0,1∗randn(cols(D), $M_1 * M_2$)
  $\epsilon =$ 0,001
  suma$\Delta = 0$
  salir = false
  while $Ep < Ep_{max} \wedge \neg$salir **do**
    for $x \in D$ **do**
      if $regla = sanger$ **then**
        $\tilde{x} = (W$∗triagS(y'∗unos$(1, M)))$
        $r = unos(M, 1) * x - \tilde{x}$
        $\Delta W = \gamma * r. * (y'$∗unos$(1, N))$
      else
        $\Delta W = \gamma * (x - y * W')' * y$
      end if
      $W+ = \Delta W$
    end for
    if $\alpha \neq 0$ **then**
      $\gamma = Ep^{-\alpha}$
    end if
    suma$\Delta = ||\Delta W||_{Frobenius}$
    if criterio $= p$ **then**
      salir = salidaPesos($suma\Delta$. ϵ)
    else
      salir = salidaOrtogonal(W. ϵ)
    end if
    $Ep+ = 1$
  end while
  return W
end function

function SALIDAPESOS($suma\Delta$. ϵ)
  if suma$\Delta < \epsilon$ **then**
    return true
  end if
  return false
end function

function SALIDAORTOGONAL(W. ϵ)
  $j_1 = 1$
  while $j_1 <$#cols$(W) - 1$ **do**
    $v_1 =$fila$_{j_1}(W)$
    $j_2 = j_1 + 1$
    while $j_2 <$#cols$(W) - 1$ **do**
      $v_2 =$fila$_{j_2}(W)$
      if $|v_1' * v_2| > \epsilon$ **then**
        return false
      end if
      $j_2 = j_2 + 1$
    end while
    $j_1 = j_1 + 1$
  end while
  return true
end function

2.2. Mapeo de características auto-organizado

Los mapas de características auto-organizados sirven para generar una representación de un espacio de muestras de gran dimensionalidad en otra de menor preservando propiedades topológicas del espacio de entrada. De alguna manera podría decirse que los SOMs (Self-Organizing Maps) son la versión en dos dimensiones de PCA (Principal Component Analysis).

2.2.1. Pseudocódigo

A modo de dejar bien asentada la implementación, la presentamos a continuación en formato de pseudocódigo. Sin embargo, no daremos mayores explicaciones ya que no presenta ninguna innovación.

function SoM(D, M_1, M_2, Ep_{max}, γ, σ, aj)
  $Ep = 0$
  N = cols(D)
  W = 0,1∗randn(cols(D), $M_1 * M_2$)
  ΔW =zeros(cols(D), $M_1 * M_2$)
  while $Ep < Ep_{max}$ **do**
    for $x \in D$ **do**
      $\tilde{y}$ =resta(x,W)
      j^* = indiceDeMinimo($\tilde{y}$)
      Di = dists(j^*, $M_1 * M_2$, M_2, σ)
      ΔW = correccion(Di, $x' - W$,γ)
      $W+ = \Delta W$
    end for
    $Ep+ = 1$
  end while
  if aj **then**
    $\gamma = Ep^{-\frac{1}{2}}$
    $\sigma = \frac{1}{2} * M_2 * Ep^{-\frac{1}{3}}$
  end if
  return W
end function

function DISTS(j^*, M, M_2, σ)
  $P(j, M_2) = (j/M_2, j \bmod M_2)$
  for $jin[1, M]$ **do**
    $C(j) = e^{-\frac{||P(j)-P(j^*)||^2}{2\sigma^2}}$
  end for
  return C
end function

function CORRECCION(Di, $\tilde{y}$,γ)
  for $i \in [1, filas(\tilde{y})]$ **do**
    for $j \in [1, cols(\tilde{y})]$ **do**
      $\Delta W(i, j) = \gamma * Di(j) * \tilde{y}$
    end for
  end for
  return ΔW
end function

function RESTA(x, W)
  for $j \in [1, cols(W)]$ **do**
    $\tilde{y}_j = ||x^t - \text{col}_j(W)||$
  end for
  return $\tilde{y}$
end function

2.2.2. Mapa de dominantes

Durante la experimentación con esta red nos surgió la necesidad de crear una manera de analizar el comportamiento de la misma. Para esto, decidimos generar nueve "mapas" de neuronas, uno correspondiente a cada clase. En cada mapa, se encuentran registradas las cantidades de veces que se activó cada neurona para dicha clase, registrando las activaciones para cada entrada en el conjunto de datos.

Una vez completados estos nueve mapas generamos un último mapa de neuronas. Este mapa contendrá en cada posición un número entre 0 y 9. El número 0 indicará que dicha neurona no se activó para ningún test, mientras que la presencia de un número entre 1 y 9 indicará que dicha neurona se activó más veces para esa clase, es decir que esa clase "domina" a la neurona.

Presentamos el pseudocódigo con el fin de esclarecer el método:

function MAPADOMINANTES(W, D, M_1, M_2)
  MP = zeros(M_1, M_2, 9)
  MD = zeros(M_1, M_2)
  for $x \in D$ **do**
    $\tilde{y}$ =resta(x,W)
    j^* = indiceDeMinimo($\tilde{y}$)
    $MC(j^*/M_2, j^* \bmod M_2) += 1$
  end for
  for $i \in [1, M_1]$ **do**
    for $j \in [1, M_2]$ **do**
      $MD(i, j)$ = indiceMax($MC(i,j,:)$)
    end for
  end for
  return MD
end function

2.2.3. Autoajuste de los parámetros de aprendizaje

Para la experimentación decidimos utilizar learning rate y σ autoajustables según la época. La opción de autoajuste consiste en: $\alpha = t^{-0,5}$ y $\sigma = \frac{M_2}{2} t^{-\frac{1}{3}}$ donde α es el learning rate, t es la época actual y σ es el coeficiente que varía las amplitudes del efecto de propagación.

3. Resultados

Para analizar los distintos métodos decidimos generar 9 folds con los datos. Teniendo un conjunto provisto de 900 datos, cada entrenamiento se realizó con 800 entradas y la validación con las 100 restantes. Además, en el caso de reducción de dimensiones, cada experimento fue repetido 5 veces a fin de tener diferentes corridas de cada caso y observar si se producían diferentes resultados con cada una.

3.1. Reducción de dimensiones

Para analizar este modelo se consideraron los dos criterios de parada mencionados anteriormente: componentes principales ortogonales y ΔW nulo. Para cada una de esas opciones se usaron las reglas de Oja y de Sanger de modo de poder probar las distintas combinaciones y comparar sus resultados.

En cada experimento se realizó un gráfico en el cual cada instancia se indica en un espacio tridimensional correspondiente a las tres componentes principales con diferentes colores según la clase. Allí, cada clase de entre las posibles (1 al 9) recibe un color y las instancias de entrenamiento son representadas con un círculo mientras que las de validación se indican con un triángulo. El objetivo entonces es observar cuál es la distribución de las instancias en ese espacio.

El límite de cantidad de épocas elegido fue 500 considerando que es un número lo suficientemente grande para que el método converja. En todos los casos, si no se llega a que los vectores sean ortogonales o bien que ΔW sea pequeño entonces el criterio de corte es alcanzar ese límite máximo de épocas. Además, el valor de epsilon elegido fue 0,001.

3.1.1. Resumen de resultados

En todos los casos

- A es "Cantidad de veces con corte por máximo de épocas"
- B es "Cantidad de veces con corte por ortogonalidad"
- C es "Corte por épocas, norma Frobenius de matriz delta pesos mínima entre las repeticiones"
- D es "Corte por épocas, norma Frobenius de matriz delta pesos máxima entre las repeticiones"
- E es "Corte por épocas error promedio entre las repeticiones"
- F es "Corte por ortogonalidad cantidad de épocas mínima entre las repeticiones"
- G es "Corte por ortogonalidad cantidad de épocas máxima entre las repeticiones"
- H es "Corte por ortogonalidad cantidad de épocas promedio entre las repeticiones"

Usando 500 épocas como máximo

Fold	A	B	C	D	E	F	G	H
1	4	1	0.004782	0.004782	0.004782	18	18	18
2	4	1	0.004761	0.004761	0.004761	17	17	17
3	4	1	0.004757	0.004757	0.004757	15	15	15
4	5	0	0.003081	0.003081	0.003081	–	–	–
5	2	3	0.004759	0.004759	0.004759	11	17	14.33
6	4	1	0.004767	0.004767	0.004767	17	17	17
7	4	1	0.004805	0.004805	0.004805	13	13	13
8	3	2	0.004777	0.004777	0.004777	17	27	22
9	4	1	0.004733	0.004733	0.004733	5	5	5

Cuadro 1: Resultados para los 9 folds, usando ortogonalidad como criterio de parada y usando la regla de Sanger. Las cantidades y promedios son sobre todas las repeticiones hechas.

Fold	A	B	C	D	E	F	G	H
1	0	5	–	–	–	22	27	23,6
2	0	5	–	–	–	20	30	26
3	0	5	–	–	–	23	31	25,6
4	0	5	–	–	–	10	30	21,4
5	0	5	–	–	–	18	27	22,8
6	0	5	–	–	–	21	27	23
7	0	5	–	–	–	13	27	20,4
8	1	4	0.004718	0.004718	0.004718	22	30	25,5
9	0	5	–	–	–	1	25	17,4

Cuadro 2: Resultados para los 9 folds, usando ortogonalidad como criterio de parada y usando la regla de Oja. Las cantidades y promedios son sobre todas las repeticiones hechas.

Fold	A	B	C	D	E	F	G	H
1	5	0	0.004782	0.004782	0.004782	–	–	–
2	5	0	0.004761	0.004761	0.004761	–	–	–
3	5	0	0.004757	0.004757	0.004757	–	–	–
4	5	0	0.004767	0.004767	0.004767	–	–	–
5	5	0	0.004759	0.004759	0.004759	–	–	–
6	5	0	0.004767	0.004767	0.004767	–	–	–
7	5	0	0.004805	0.004805	0.004805	–	–	–
8	5	0	0.004777	0.004777	0.004777	–	–	–
9	5	0	0.004733	0.004733	0.004733	–	–	–

Cuadro 3: Resultados para los 9 folds, usando ΔW como criterio de parada y usando la regla de Sanger. Las cantidades y promedios son sobre todas las repeticiones hechas.

Fold	A	B	C	D	E	F	G	H
1	5	0	0.004730	0.004730	0.004730	–	–	–
2	5	0	0.004724	0.004724	0.004724	–	–	–
3	5	0	0.004712	0.004712	0.004712	–	–	–
4	5	0	0.002974	0.002974	0.002974	–	–	–
5	5	0	0.004703	0.004703	0.004703	–	–	–
6	5	0	0.004718	0.004718	0.004718	–	–	–
7	5	0	0.004750	0.004750	0.004750	–	–	–
8	5	0	0.004718	0.004718	0.004718	–	–	–
9	5	0	0.004699	0.004699	0.004699	–	–	–

Cuadro 4: Resultados para los 9 folds, usando ΔW como criterio de parada y usando la regla de Oja. Las cantidades y promedios son sobre todas las repeticiones hechas.

Dado que en muchos de los casos el corte se produjo por alcanzarse el máximo de épocas y que las distintas repeticiones produjeron un error idéntico, decidimos repetir la experimentación pero ahora con una cantidad de épocas máxima igual a 250 para ver si de esa manera se convergía a los mismos errores pero con una menor cantidad de épocas.

Usando 250 épocas como máximo

Fold	A	B	C	D	E	F	G	H
1	5	0	0.004782	0.004782	0.004782	–	–	–
2	2	3	0.004761	0.004761	0.004761	13	21	16
3	4	1	0.004757	0.004757	0.004757	16	16	16
4	5	0	0.003081	0.003081	0.003081	–	–	–
5	3	2	0.004759	0.004759	0.004759	14	18	16
6	4	1	0.004767	0.004767	0.004767	18	18	18
7	4	1	0.004805	0.004805	0.004805	19	19	19
8	4	1	0.004777	0.004777	0.004777	12	12	12
9	5	0	0.004733	0.004733	0.004733	–	–	–

Cuadro 5: Resultados para los 9 folds, usando ortogonalidad como criterio de parada y usando la regla de Sanger. Las cantidades y promedios son sobre todas las repeticiones hechas.

Fold	A	B	C	D	E	F	G	H
1	0	5	–	–	–	19	26	23,2
2	0	5	–	–	–	21	30	24
3	0	5	–	–	–	16	26	22
4	0	5	–	–	–	19	29	24,6
5	0	5	–	–	–	14	27	21,8
6	0	5	–	–	–	19	25	22,6
7	0	5	–	–	–	18	34	25,6
8	0	5	–	–	–	22	28	24,8
9	0	5	–	–	–	22	29	24,8

Cuadro 6: Resultados para los 9 folds, usando ortogonalidad como criterio de parada y usando la regla de Oja. Las cantidades y promedios son sobre todas las repeticiones hechas.

Fold	A	B	C	D	E	F	G	H
1	5	0	0.004761	0.004782	0.0047778	–	–	–
2	5	0	0.004757	0.004761	0.0047606	–	–	–
3	5	0	0.004757	0.004757	0.004757	–	–	–
4	5	0	0.003081	0.003081	0.003081	–	–	–
5	5	0	0.004759	0.004759	0.004759	–	–	–
6	5	0	0.004767	0.004767	0.004767	–	–	–
7	5	0	0.004805	0.004805	0.004805	–	–	–
8	5	0	0.004777	0.004777	0.004777	–	–	–
9	5	0	0.004733	0.004733	0.004733	–	–	–

Cuadro 7: Resultados para los 9 folds, usando ΔW como criterio de parada y usando la regla de Sanger. Las cantidades y promedios son sobre todas las repeticiones hechas.

Fold	A	B	C	D	E	F	G	H
1	5	0	0.004730	0.004730	0.004730	–	–	–
2	5	0	0.004724	0.004724	0.004724	–	–	–
3	5	0	0.004712	0.004712	0.004712	–	–	–
4	5	0	0.002974	0.002974	0.002974	–	–	–
5	5	0	0.004703	0.004703	0.004703	–	–	–
6	5	0	0.004718	0.004718	0.004718	–	–	–
7	5	0	0.004750	0.004750	0.004750	–	–	–
8	5	0	0.004718	0.004718	0.004718	–	–	–
9	5	0	0.004699	0.004699	0.004699	–	–	–

Cuadro 8: Resultados para los 9 folds, usando ΔW como criterio de parada y usando la regla de Oja. Las cantidades y promedios son sobre todas las repeticiones hechas.

Observando las Tablas 3 y 4 es posible ver que en todos los casos los errores cometidos al llegar al máximo de épocas permitido es el mismo para todas las repeticiones de los experimentos. Observando los valores es posible ver que casi todos los valores están en torno a 0.0047 y 0.0048 y sólo uno en torno a 0.0029. Dado que el valor de epsilon tomado es 0.001, en todos los casos el criterio de corte efectivo acaba siendo el máximo de épocas. Dado el learning rate escogido

(0.001) y las operaciones entre instancias de entrada y pesos, este error (correspondiente a la norma Frobenius de la matriz de delta pesos) es el mínimo alcanzado y por ende se puede decir que el algoritmo convergió. Es claro entonces que el epsilon elegido es muy pequeño para los valores en los que converge la red con estas instancias independientemente de si la regla utilizada es la de Oja o la de Sanger. Comparando ambas reglas, es posible ver que los valores con la regla de Oja son levemente menores.

Dado que con 500 épocas en todos los casos se llegó a que la red convergiese, decidimos repetir la experimentación usando como tope 250 épocas. Observando las Tablas 7 y 8 es posible ver que los valores de convergencia son prácticamente los mismos que usando 500 épocas.

En relación a las Tablas 1 y 2 se desprende que el criterio de ortogonalidad entre los vectores obtenidos permite converger en pocas épocas, si es que esto es posible. Las diferencias entre las dos reglas son notorias en este caso donde la regla de Sanger, salvo algunos folds, muy pocas veces sobre las 5 repeticiones converge. Por otro lado, usando la regla de Oja, sólo en un caso se llegó al límite máximo de épocas. Sin embargo, cuando se converge por ortogonalidad usando la regla de Sanger es, en promedio, usando apenas 15,16625 épocas mientras que si se usa la regla de Oja es, en promedio, usando 22,85555 épocas. Esto deja ver que en los casos en los que Sanger converge, si bien apenas un cuarto de los casos, lo hace mucho más rápidamente que Oja. Sin embargo, este último converge en casi el 100 % de los casos.

Considerando además las Tablas 5 y 6 donde se repitieron los experimentos con un tope de cantidad máxima de épocas igual a 250, es posible ver que los resultados se repiten: Sanger converge algunas veces mientras que Oja converge siempre pero con una cantidad de épocas en promedio mayor que Sanger.

3.1.2. Gráficos espaciales de algunos resultados

A continuación presentamos algunos ejemplos de resultados que se repitieron entre los distintos folds. Dado que la cantidad de gráficos correspondientes a todas las instancias es muy alta decidimos mostrar apenas los casos relevantes y que muestran patrones repetidos en general. En los gráficos se decidió mostrar apenas una porción del espacio (cubo con lados en -4 y 4 en las tres dimensiones) con el objetivo de poder comparar los distintos gráficos. Esto implica que en algunos casos algunas instancias no se puede ver dado que estaban ubicadas fuera del espacio representado, sin embargo, consideramos que esto no era un problema dado que eran muy pocas instancias y de este modo podemos comparar sin problemas todos los gráficos.

En las explicaciones diremos, por ejemplo, “clase azul” para referirnos a la clase distinguida en los gráficos con el colro azul. Esto claramente no es correcto pues una clase no es un color pero nos expresaremos de esta manera para dar mayor claridad al texto. A continuación indicamos qué color corresponde a cada clase.

1. Clase 1: magenta
2. Clase 2: cian
3. Clase 3: rojo
4. Clase 4: verde
5. Clase 5: azul
6. Clase 6: amarillo
7. Clase 7: marrón

8. Clase 8: violeta
9. Clase 9: azul oscuro

Particularidades con regla de Oja y criterio de ortogonalidad

En las siguientes Figuras (1, 2, 3 y 4) presentamos cuatro casos observados al usar la regla de Oja con el criterio de ortogonalidad. Las características observadas en ellos se repiten en varias de las repeticiones con los distintos folds y lo que buscamos mostrar con estos ejemplos son las diferencias observadas en todos los casos.

En la Figura 1 se puede ver como, si bien las instancias se encuentran posicionadas relativamente juntas se pueden divisar algunas porciones del espacio donde algunas clases ocupan lugares más reducidos como por ejemplo las clases azul, y magenta. Otras clases se encuentran superpuestas entre sí como las verde, marrón y amarilla y por otro lado la celeste y la roja. Las clases restantes se encuentran más difuminadas en el espacio y sin conformar cúmulos definidos. Estos comportamientos fueron los más repetidos a lo largo de todas las experimentaciones.

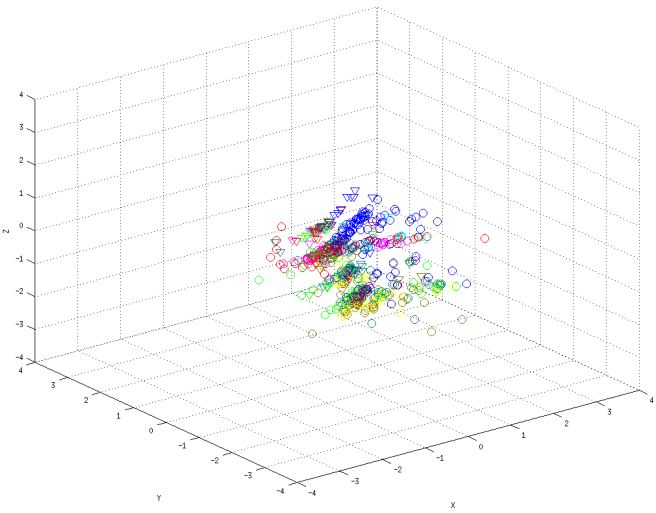
En la Figura 2 vemos un caso no tan común en las experimentaciones realizadas pero también particular. Allí puede verse cómo nuevamente las clases azul y magenta ocupan porciones del espacio más alejadas pero en este caso bien delimitadas. Al igual que en el caso antes discutido, las clases verde, marrón y amarilla ocupan prácticamente los mismos lugares pero esta vez un poco más alejadas del resto de las clases. En cuanto a las clases roja y celeste, nuevamente forman cúmulos en las mismas regiones pero la diferencia en este caso es que el resto de las clases también se aglomeran en esos mismos lugares, mostrando menos dispersión que en el caso anterior.

En la Figura 3 podemos ver que a diferencia de los casos anteriores la segunda componente principal separa notablemente a las instancias en dos partes como puede observarse en 3 d. Si bien la vista en perspectiva indica que todas las instancias están mucho más juntas que en los casos anteriores, puede verse cómo según la segunda componente, para valores negativos se agrupan las clases amarilla, verde y marrón mientras que en valores cercanos al 0 y positivos están las demás. En este caso las clases azul y magenta no se encuentran tan diferenciadas por su posición de las demás pero nuevamente las clases roja y celeste aparecen en cúmulos bien demarcados aunque agrupadas también con las clases azul oscuro y violeta.

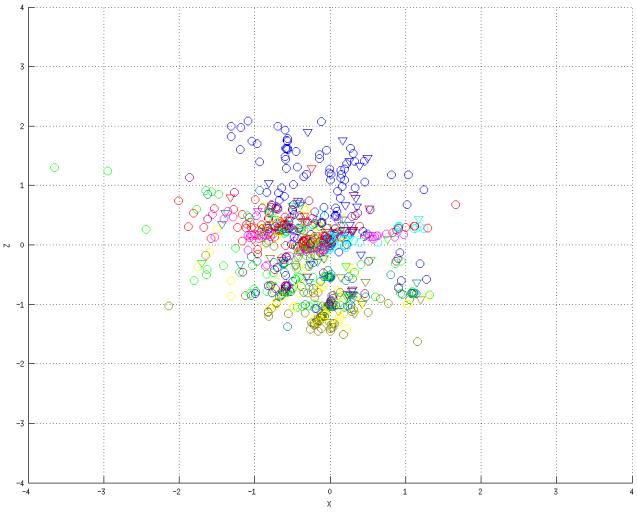
Ya en la Figura 4 tenemos otra disposición donde sólo la clase azul aparece bien demarcada en el espacio mientras que el resto de las clases están mucho más juntas entre sí. Se vuelve a repetir que las clases amarilla, verde y marrón estén juntas y que las roja y celeste ocupen los mismos lugares. En este caso, además la clase magenta aparece más cerca de estas últimas mientras que las demás clases aparecen en zonas intermedias.

Recordando que para la regla de Oja y el criterio de ortogonalidad casi siempre se produjo el corte por ortogonalidad y analizando los resultados espaciales de las instancias podemos afirmar que en general la separación producida por las tres componentes principales no es demasiado útil desde lo visual. Apenas dos clases aparecen en general más separadas de las demás, la azul y la magenta (5 y 1 respectivamente). Sin embargo, hay otras clases que se ubican formando cúmulos (clases roja, celeste, verde, amarilla y marrrón) de modo que utilizando alguna técnica de clasificación como KNN sería posible caracterizarlas correctamente aunque obviamente sería necesario realizar experimentos con esto para corroborarlo.

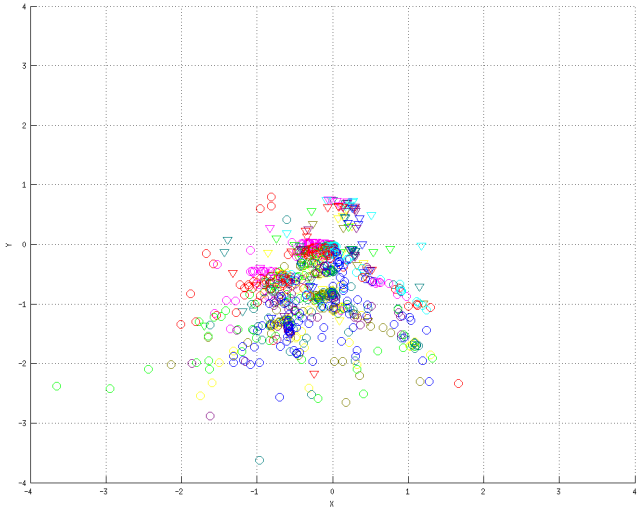
Considerando ahora las instancias de validación (triángulos en los gráficos), salvo raras excepciones no se ven instancias de validación demasiado alejadas de las instancias de entrenamiento correspondientes a la misma clase (mismo color). Esto indica que las tres primeras componentes principales permiten agrupar de manera bastante eficaz a las distintas clases. Es claro que la apreciación visual de los gráficos no es suficiente para determinar esto ya que sería necesario aplicar posteriormente alguna técnica de clasificación pero auspicia posibles buenos resultados.



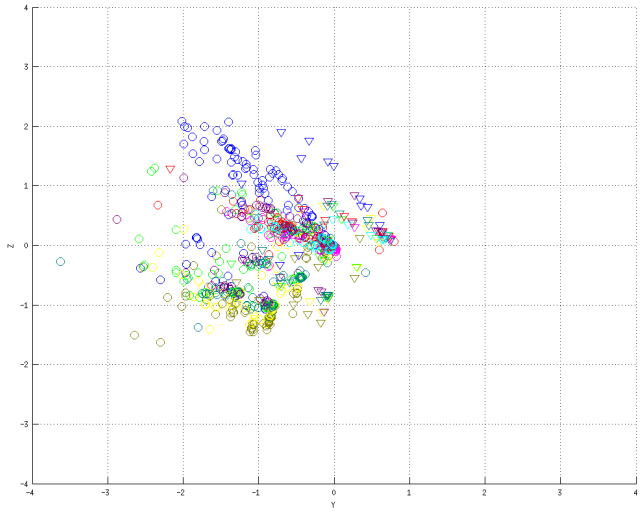
(a) Vista en perspectiva.



(b) Plano X-Z.

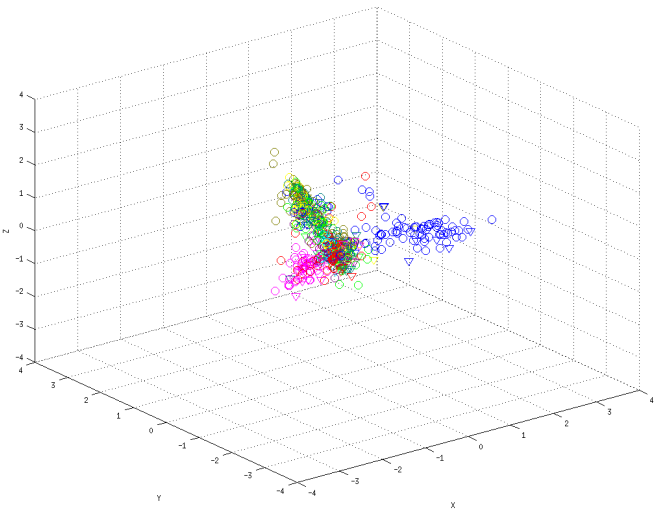


(c) Plano X-Y.

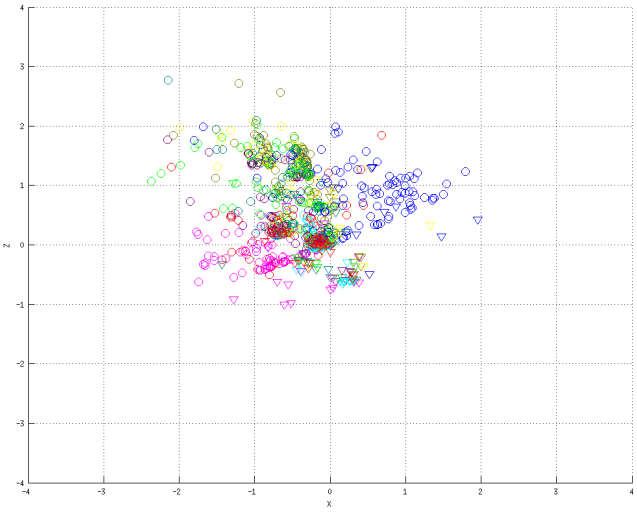


(d) Plano Y-Z.

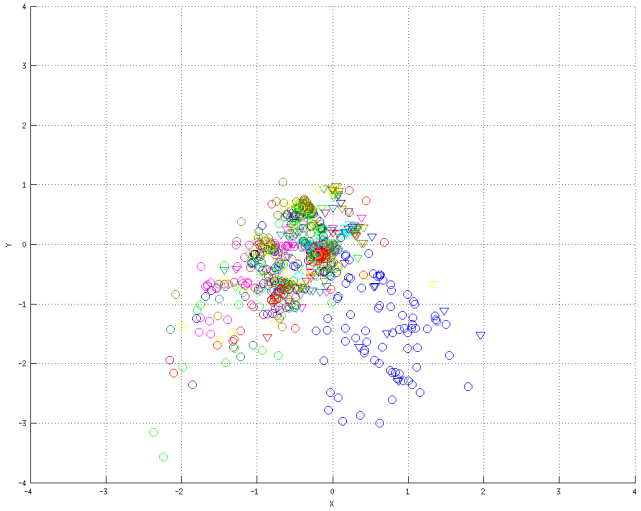
Figura 1: Gráfico espacial para el Fold 1 usando ortogonalidad como criterio de parada y la regla de Oja con learning rate 0.001 en la repetición 1.



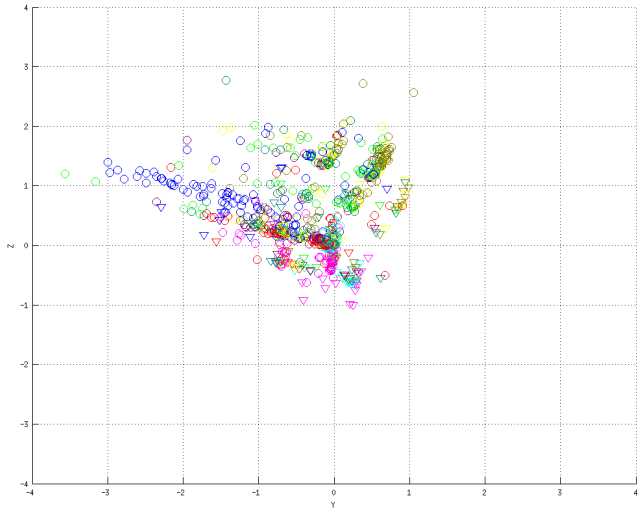
(a) Vista en perspectiva.



(b) Plano X-Z.

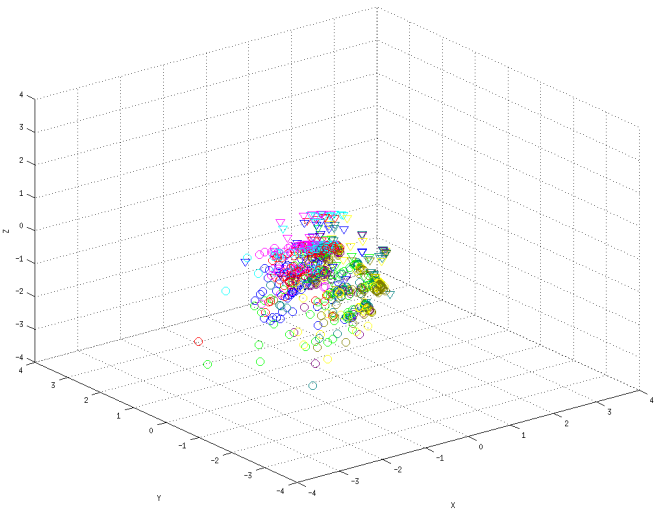


(c) Plano X-Y.

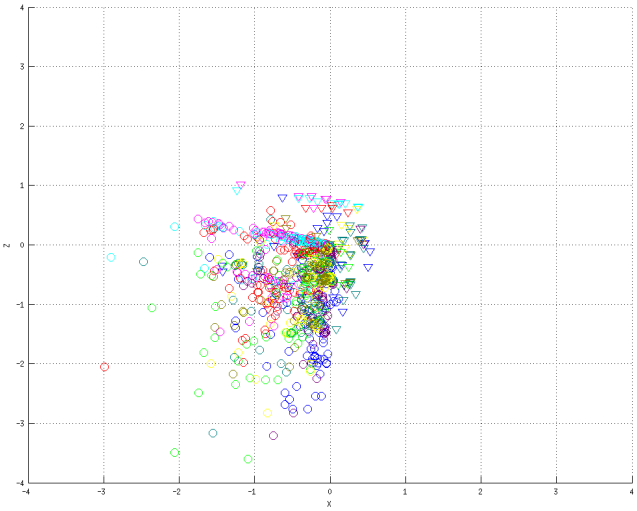


(d) Plano Y-Z.

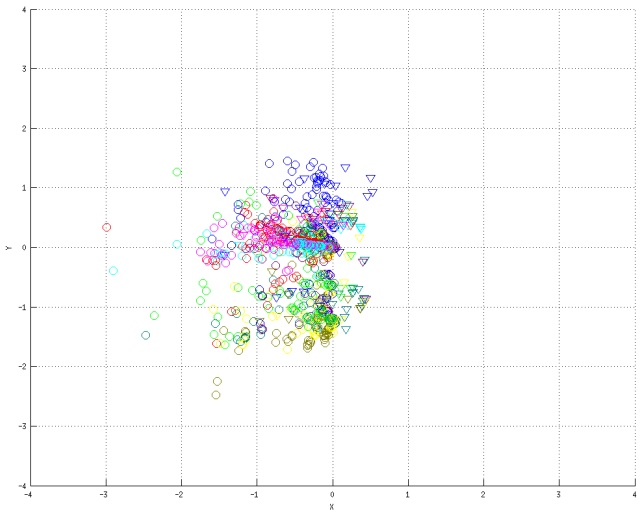
Figura 2: Gráfico espacial para el Fold 4 usando ortogonalidad como criterio de parada y la regla de Oja con learning rate 0.001 en la repetición 3.



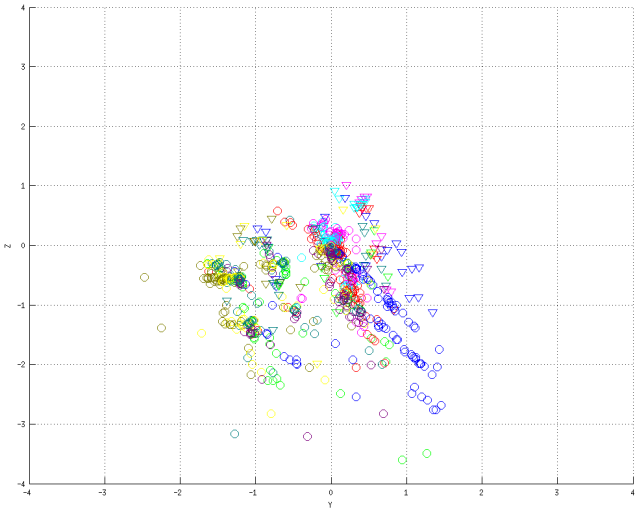
(a) Vista en perspectiva.



(b) Plano X-Z.

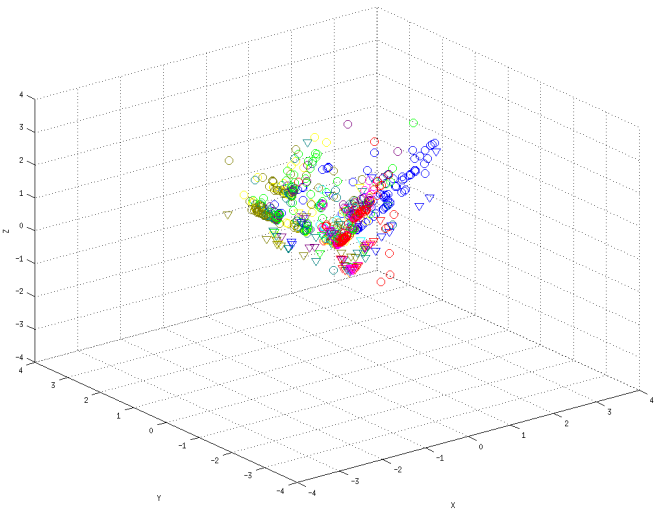


(c) Plano X-Y.

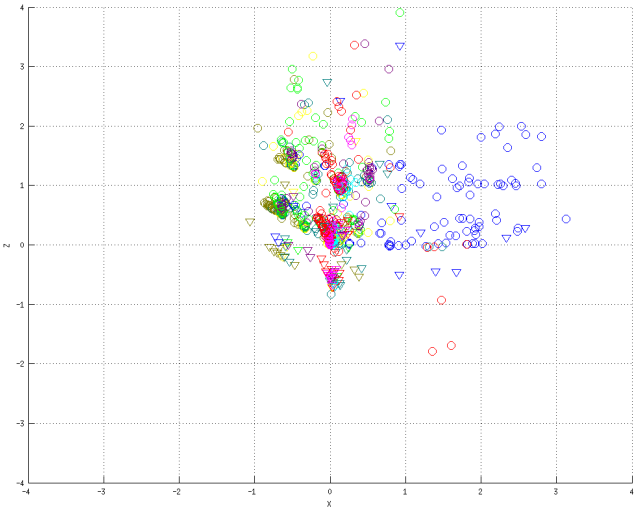


(d) Plano Y-Z.

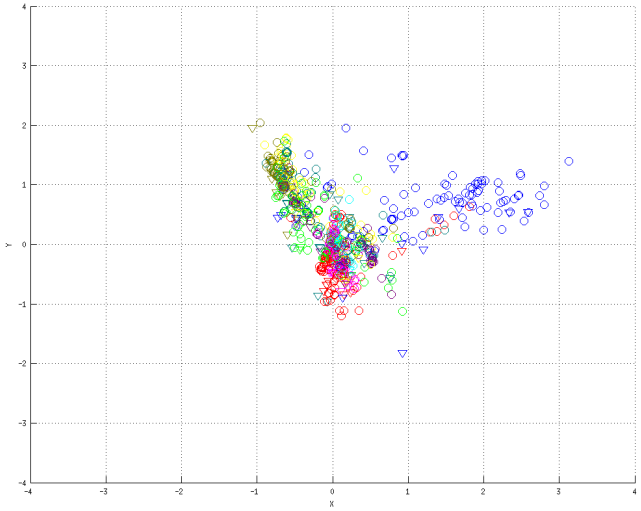
Figura 3: Gráfico espacial para el Fold 5 usando ortogonalidad como criterio de parada y la regla de Oja con learning rate 0.001 en la repetición 2.



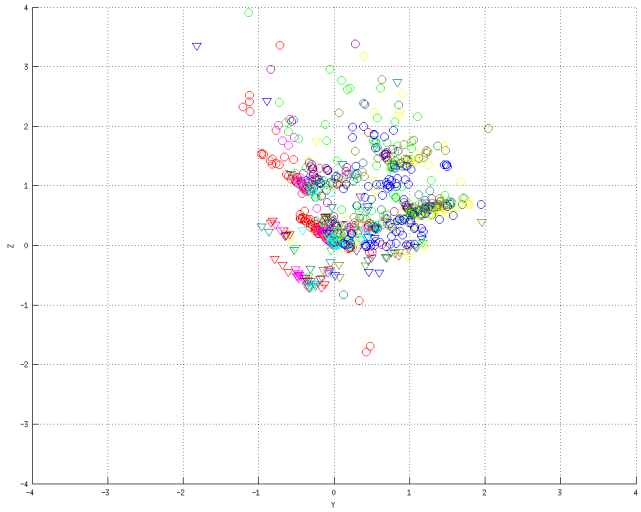
(a) Vista en perspectiva.



(b) Plano X-Z.



(c) Plano X-Y.



(d) Plano Y-Z.

Figura 4: Gráfico espacial para el Fold 7 usando ortogonalidad como criterio de parada y la regla de Oja con learning rate 0.001 en la repetición 4.

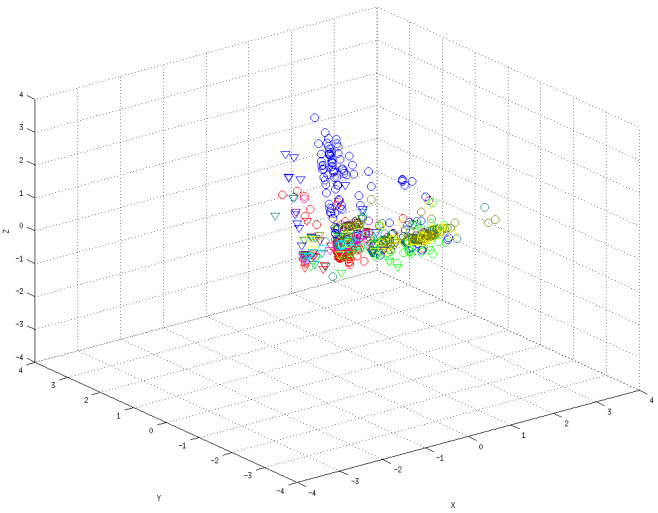
Particularidades con regla de Sanger y criterio de ortogonalidad

De manera similar al caso anterior, decidimos mostrar algunos de los resultados que de alguna manera representan a la mayoría. Vale aclarar que no se observaron diferencias significativas entre los casos en los cuales se convergió por criterio de ortogonalidad y por máximo de épocas.

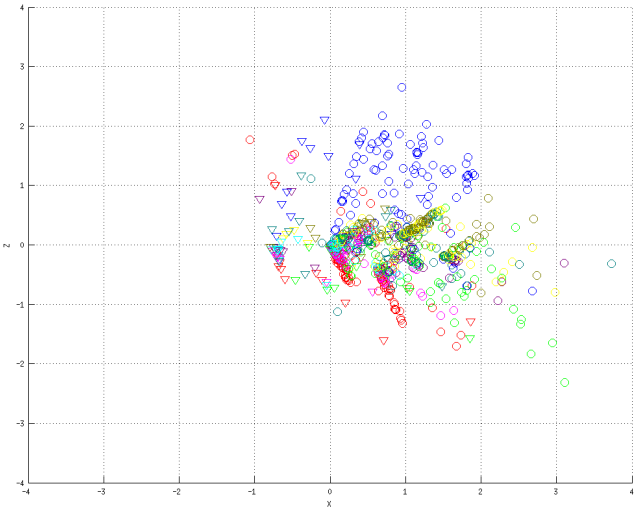
Por un lado, en la Figura 6 se ve cómo las distintas clases no se encuentran espacialmente muy dispersas. Apenas la clase azul está un poco alejada de las demás. El resto de las clases aparecen en pequeños cúmulos como la roja, violeta, magenta, verde, amarilla y marrón. Las restantes están un poco más dispersas.

Por otro lado, en la Figura 5 se ve a simple vista cómo la clase azul ocupa un lugar más alejado del resto de las clases mientras que las otras clases aparecen más juntas entre sí. De todos modos es posible ver algunos cúmulos como el ocupado por las clases verde, amarilla y marrón. La clase roja, a diferencia del caso analizado anteriormente, aparece más dispersa.

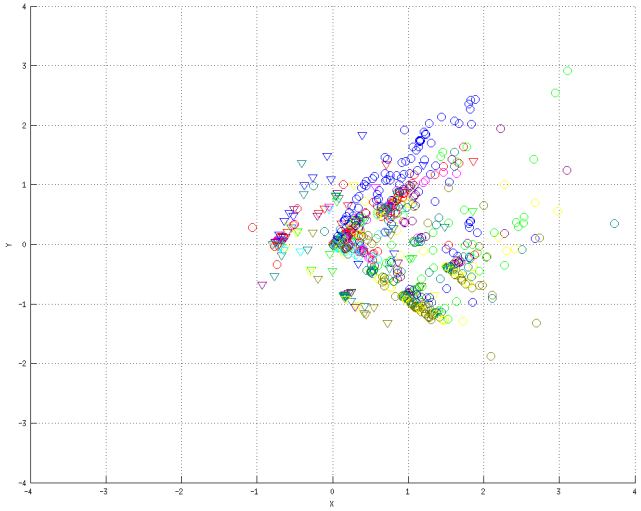
De manera análoga a lo observado al usar la regla de Oja, es posible ver que salvo casos particulares, tanto las instancias de entrenamiento como las de validación ocupan los mismos lugares en el espacio por lo que es posible que al realizar una clasificación posterior, el uso de las tres componentes principales permitan obtener buenos resultados. Al igual que antes, desde lo visual no se pueden diferenciar zonas claras para cada color aunque sí algunos cúmulos ocupados específicamente por ciertas clases.



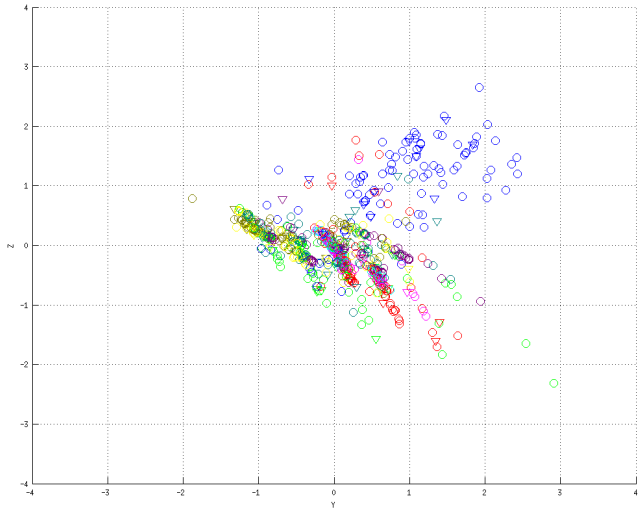
(a) Vista en perspectiva.



(b) Plano X-Z.

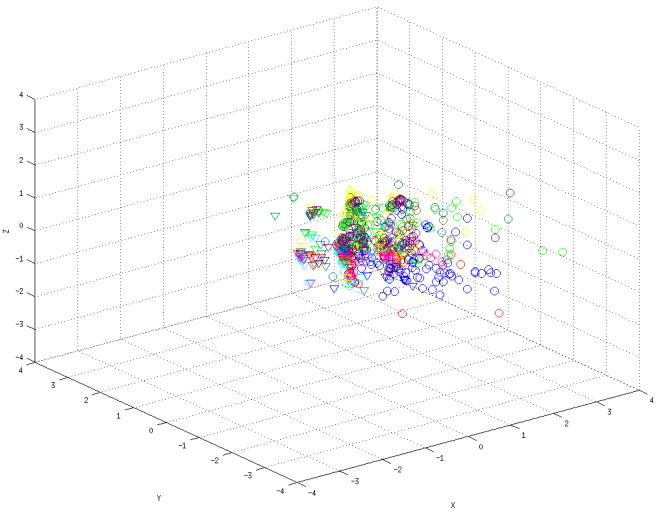


(c) Plano X-Y.

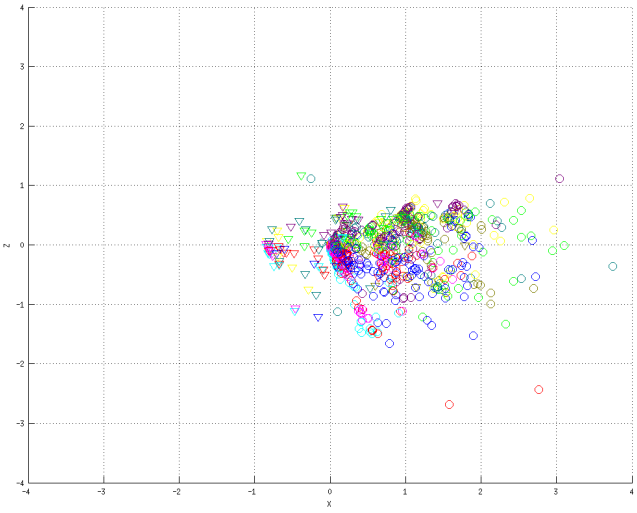


(d) Plano Y-Z.

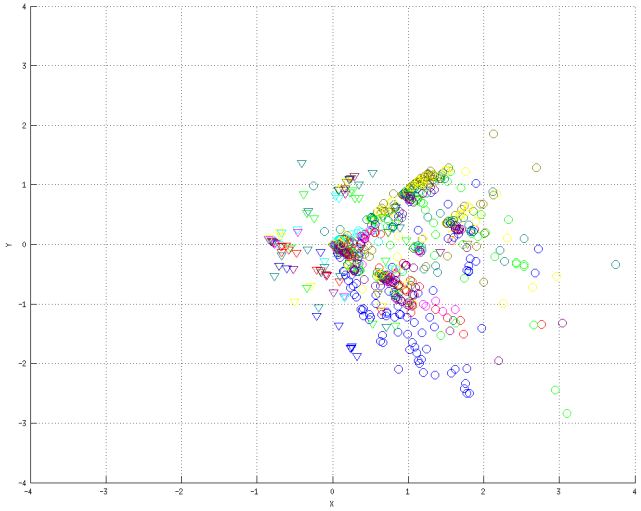
Figura 5: Gráfico espacial para el Fold 1 usando ortogonalidad como criterio de parada y la regla de Sanger con learning rate 0.001 en la repetición 4.



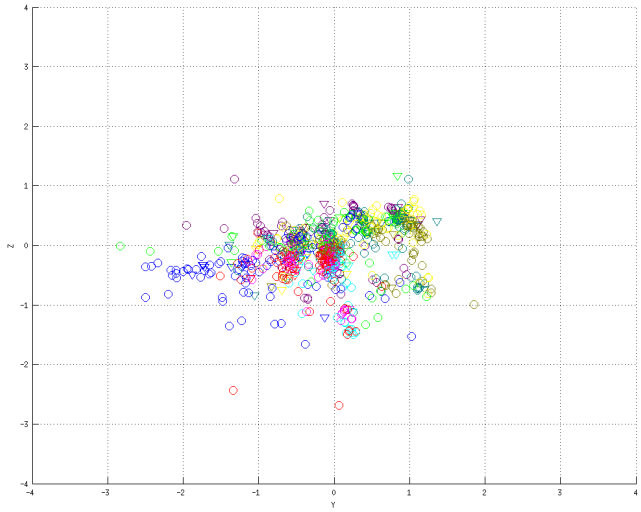
(a) Vista en perspectiva.



(b) Plano X-Z.



(c) Plano X-Y.



(d) Plano Y-Z.

Figura 6: Gráfico espacial para el Fold 2 usando ortogonalidad como criterio de parada y la regla de Sanger con learning rate 0.001 en la repetición 4.

Particularidades con regla de Oja y criterio de ΔW

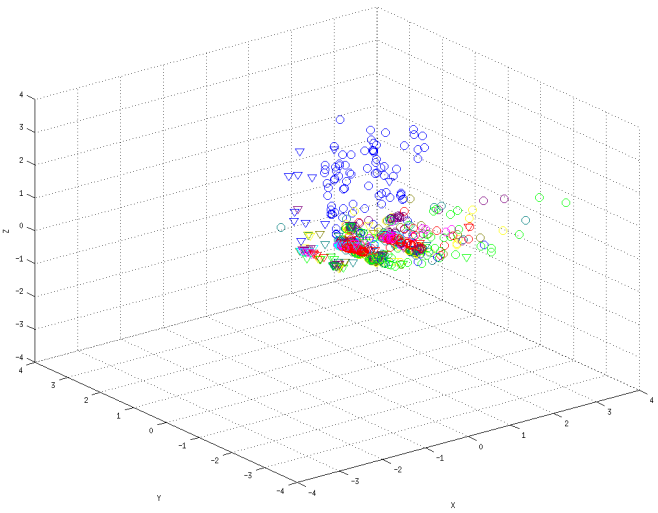
A continuación presentamos resultados correspondientes al uso de la regla de Oja con el criterio de la matriz de delta de pesos. Recordando de las Tablas 4 y 8 que en casi todos los casos se produjo el corte por alcanzar un valor muy bajo en la matriz de pesos.

En la Figura 7 se observa el patrón más veces repetido entre los experimentos. En dicho gráfico se ve que las instancias de la clase azul ocupan una porción alejada del resto de las instancias mientras que el resto se ubican mucho más juntas. En ellas hay varios cúmulos donde se encuentran las clases roja y celeste juntas por un lado rodeadas de la magenta y un poco más alejadas las verde, amarilla y marrón. Las clases violeta y azul oscuro aparecen junto a los cúmulos pero más esparcidas en el espacio.

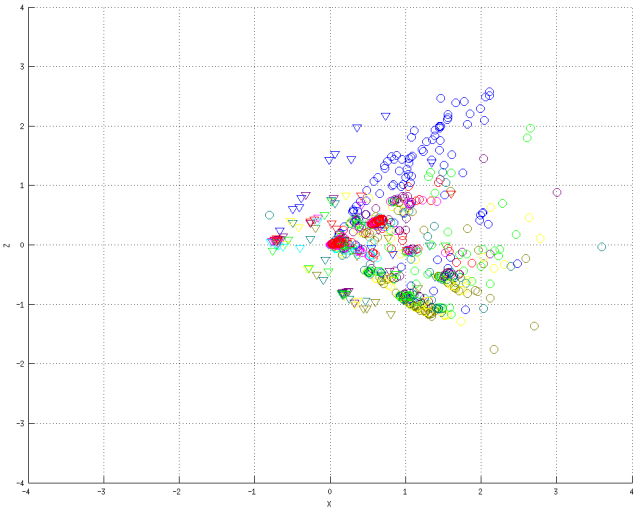
La Figura 8 presenta un caso particular en el cual varias instancias se ubican formando casi una línea recta sobre el eje X. Al igual que en los casos anteriores la clase azul ocupa un lugar más alejado que el resto de las clases. Principalmente las clases roja y magenta tienen a sus instancias ubicadas casi en línea recta indicando que el mayor peso para esas instancias es respecto de una de las componentes. Observando el conjunto de entrenamiento y los resultados no se ve ninguna característica a simple vista de modo que intuímos que el resultado tiene que ver con el orden de entrenamiento y los pesos iniciales de la matriz de pesos.

De manera similar al caso anterior, las clases amarilla, verde y marrón se ubican en los mismos cúmulos y por otro lado las instancias de la clase celeste aparecen bien concentradas. El resto de las clases aparecen distribuídas sin seguir ningún patrón fácilmente identificable en los gráficos.

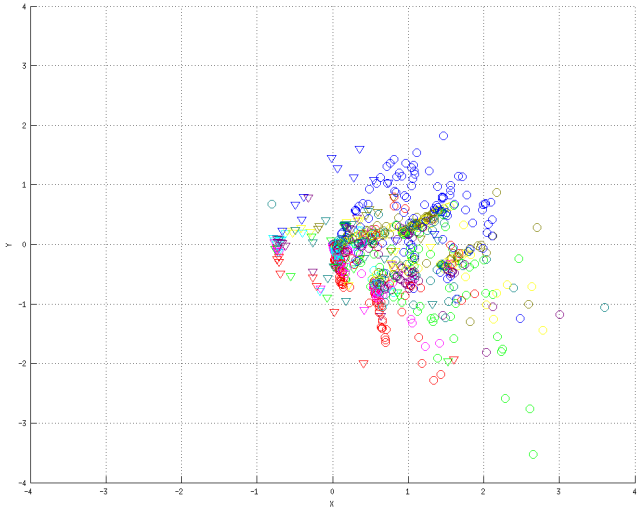
Considerando las instancias de validación, en los gráficos no se ven muchas de ellas alejadas de las de entrenamiento. De manera visual, esto indica que la separación en tres dimensiones de esta técnica permite diferenciar bastante bien las clases. Es claro que no es posible asegurar este comportamiento de esta manera por lo que sería necesario aplicar posteriormente alguna técnica de clasificación para analizar cuantitativamente los resultados. La principal ventaja de este conjunto de parámetros es que en una cantidad de épocas mucho menor se alcanzaron resultados muy similares a las combinaciones discutidas anteriormente.



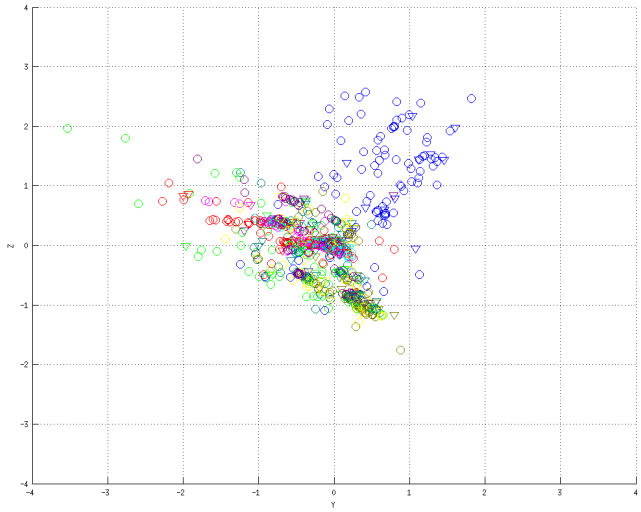
(a) Vista en perspectiva.



(b) Plano X-Z.

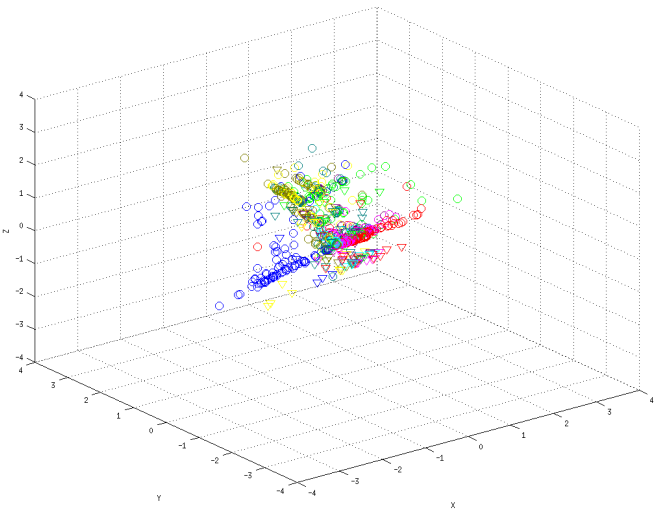


(c) Plano X-Y.

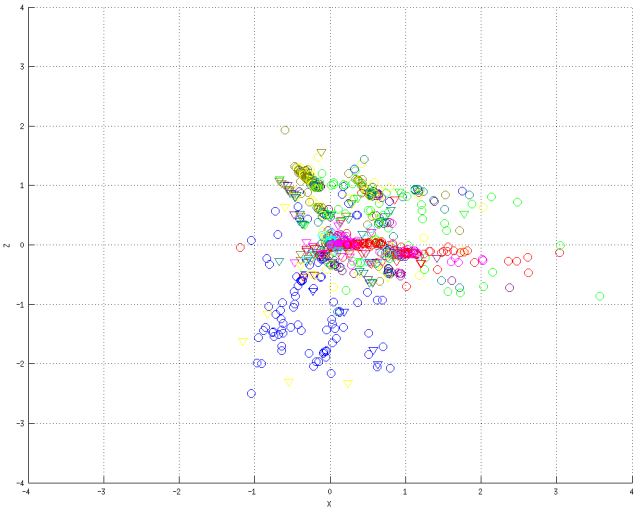


(d) Plano Y-Z.

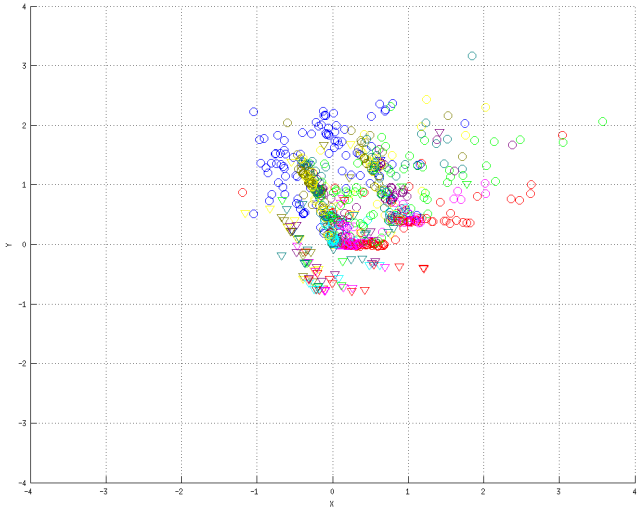
Figura 7: Gráfico espacial para el Fold 1 usando ΔW como criterio de parada y la regla de Oja con learning rate 0.001 en la repetición 2.



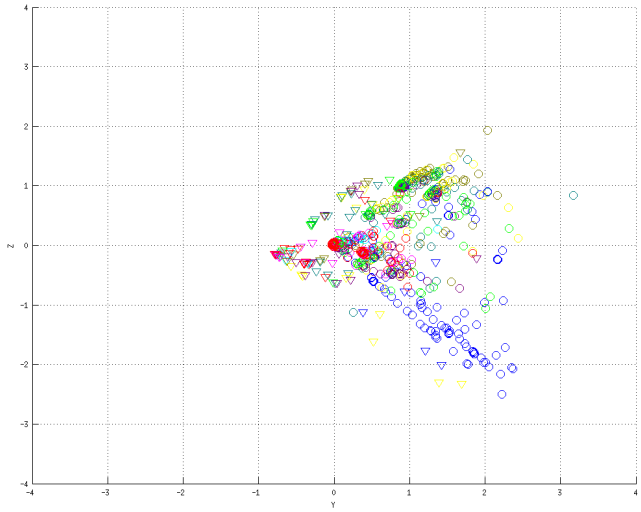
(a) Vista en perspectiva.



(b) Plano X-Z.



(c) Plano X-Y.



(d) Plano Y-Z.

Figura 8: Gráfico espacial para el Fold 6 usando ΔW como criterio de parada y la regla de Oja con learning rate 0.001 en la repetición 3.

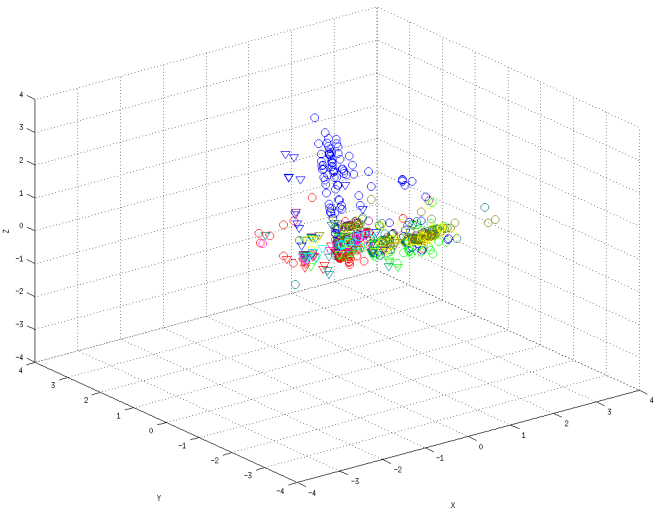
Particularidades con regla de Sanger y criterio de ΔW

Los resultados usando la regla de Sanger con el criterio de delta pesos son, tal vez, los más homogéneos en el sentido que todos los gráficos tienen características similares. Como se puede ver en las Figuras 9 y 10, si bien la ubicación espacial cambia, la posición relativa entre las

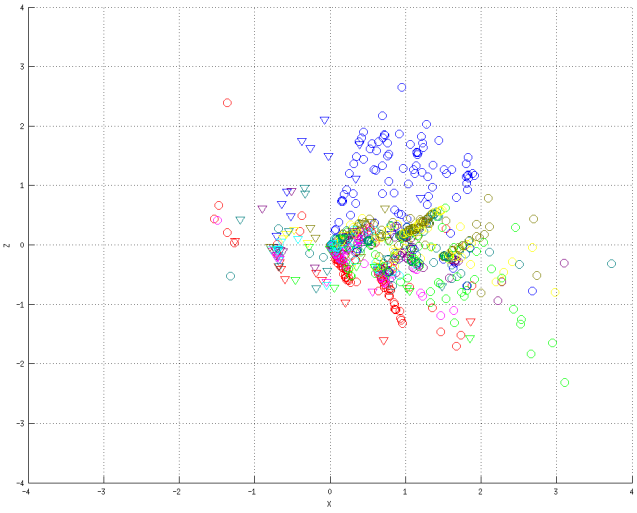
instancias de las distintas clases es bastante similar.

En general, las instancias de la clase azul vuelven a aparecer alejadas del resto como en los casos anteriores. De manera similar las clases roja y celeste ocupan lugares similares y por otro lado lo mismo sucede con las amarilla, marrón y verde. Bajo esta combinación de parámetros sí sucede que la clase magenta aparece más dispersa, algo no tan común en los otros casos. Para el resto de las clases, las dispersiones son similares a casos anteriores ya discutidos.

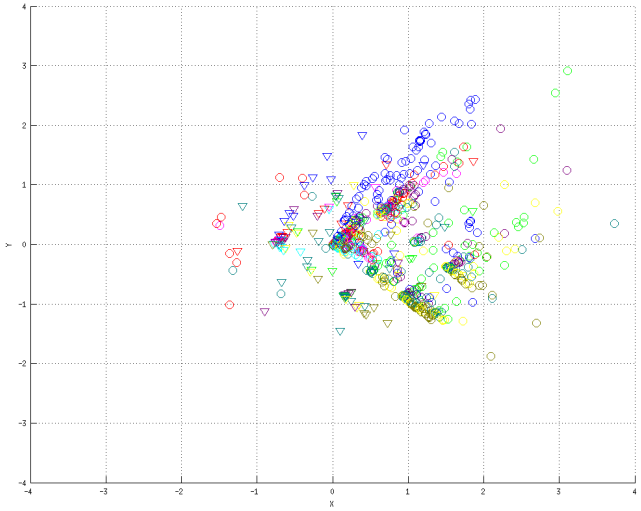
En este caso sucede también que las instancias de validación se encuentran cerca de las de entrenamiento correspondientes a la misma clase de modo que, al igual que antes, es un buen indicio para una futura clasificación.



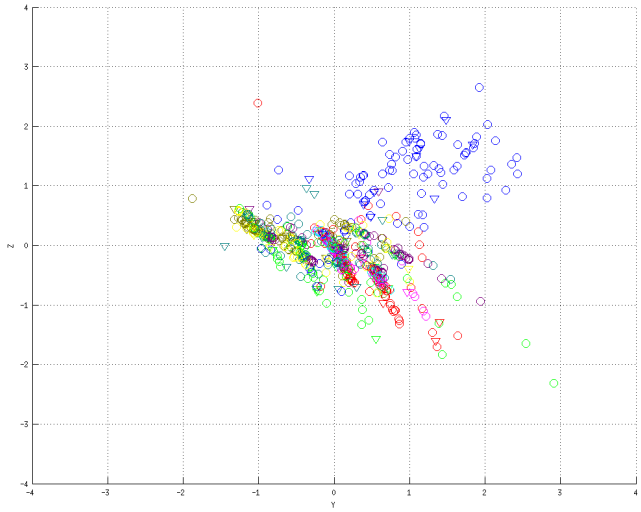
(a) Vista en perspectiva.



(b) Plano X-Z.

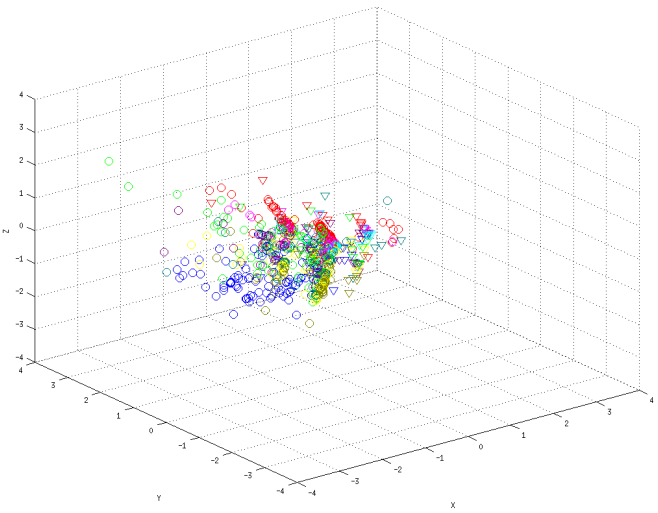


(c) Plano X-Y.

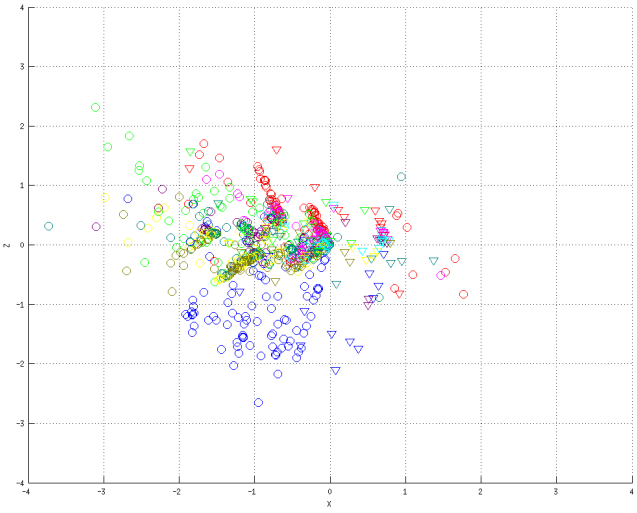


(d) Plano Y-Z.

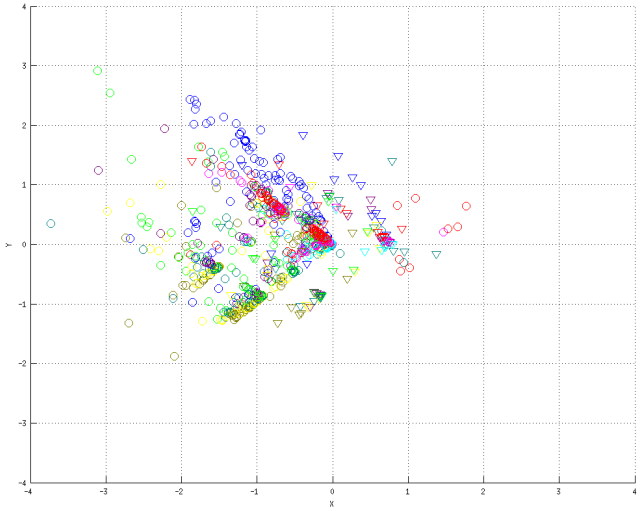
Figura 9: Gráfico espacial para el Fold 1 usando ΔW como criterio de parada y la regla de Sanger con learning rate 0.001 en la repetición 3.



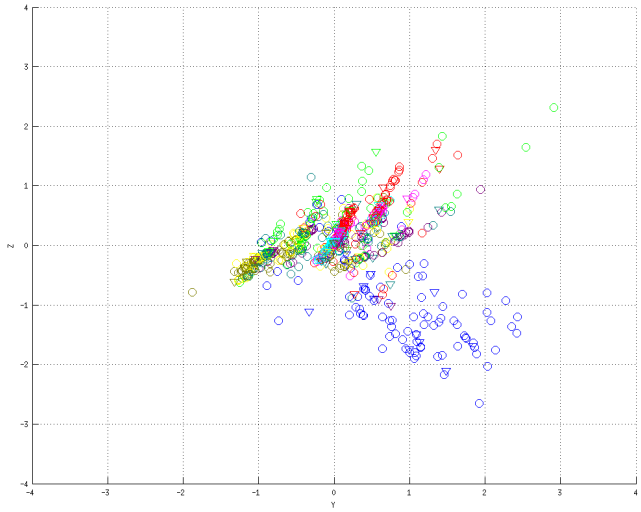
(a) Vista en perspectiva.



(b) Plano X-Z.



(c) Plano X-Y.



(d) Plano Y-Z.

Figura 10: Gráfico espacial para el Fold 1 usando ΔW como criterio de parada y la regla de Sanger con learning rate 0.001 en la repetición 5.

Dados los resultados obtenidos en la experimentación y considerando que en general se obtuvieron los máximos niveles de convergencia (dado que espacialmente los resultados con el máximo de épocas fueron similares a cuando se cortó por el criterio, ya sea por pesos o por ortogonalidad), decidimos analizar la velocidad de convergencia al usar un método adaptativo.

Para ello utilizamos diferentes funciones para modificar el learning rate, sin embargo, los

resultados obtenidos no fueron buenos. En todos los casos, o bien el algoritmo alcanzó los criterios de convergencia (ortogonalidad o pesos) con 1 época arrojando resultados incorrectos o bien el learning rate no permitía la convergencia y acabábamos obteniendo valores Not a Number en la matriz de pesos. En el primer caso consideramos que se debe a que ante la primera época, luego de actualizar el learning rate los pesos se modifican de modo que satisfacen trivialmente los criterios. En el segundo caso se debe a que el learning rate es demasiado grande y hace que la matriz de pesos aumente y dada la realimentación positiva por los valores de los pesos entonces al actualizarlos éstos crecen cada vez más.

Las opciones consideradas fueron:

- $\eta = epoca^{-\frac{1}{2}}$
- $\eta = \alpha^{-epoca}$
- $\eta = epoca^{-\alpha}$

donde α fue un parámetro variado en valores mayores y menores a 1 y *epoca* es la época actual.

3.2. Mapeo de características

Para analizar los resultados de las métricas decidimos generar 9 folds distintos para luego promediar los resultados de los mismos. No vimos la necesidad de repetir los tests a diferencia del análisis anterior, ya que la red de kohonen da resultados mas estables. Cada fold estará compuesto por un set de entrenamiento y uno de validación con instancias seleccionadas al azar.

En este caso, el análisis de este modelo se vuelve algo particular. Para realizar el mismo, decidimos utilizar dos métricas: la primera con el objetivo de obtener alguna noción de desempeño de la red, y la segunda para conocer que tan distanciados estaban los distintos grupos de activación luego del entrenamiento.

3.2.1. Distancia a dominantes (DD)

Representará la distancia relativa media entre los dominantes del set de validación y el set de activación. Para generar su cálculo, tomamos cada uno de los dominantes del mapa de validación y buscamos el dominante más cercano en el mapa de entrenamiento, en el mejor caso esta distancia es cero.

$$DD_abs(DTr, DTe, W) = dM(mAct(DTr, W), mAct(DTe, W))$$

Obteniendo todas las distancias entre los distintos dominantes calcularemos su media y luego la normalizaremos dividiéndola por la distancia máxima que puede existir; esto es $||(M_1, M_2)||$, es decir la distancia entre las esquinas de la matriz de pesos.

$$DD_rel(DTr, DTe, W, M_1, M_2) = \frac{DD_abs(DTr, DTe, W)}{||(M_1, M_2)||}$$

Donde DTr es el dataset de entrenamiento, DTe es el dataset de validación, W es la matriz de pesos y M_1, M_2 son las dimensiones de la última. Se supone que $mAct$ calcula los mapas de dominantes para los datos otorgados y que dM calcula la distancia media entre dichos dominantes. En otra variación de la metrica, dM puede calculado utilizando distancia Manhattan.

3.2.2. Factor de cruce (FC)

Esta métrica nos dará una noción de cuántas neuronas se activan con más de una clase. Esto nos sirve para observar de qué forma la red neuronal está respondiendo a distintos impulsos. Inferimos que es deseable que este factor sea bajo, ya que lo ideal sería que las distintas clases activen grupos de neuronas disjuntos.

Para calcular este factor, nos armaremos un mapa de la red neuronal en donde tendremos registradas las cantidades de veces que se activó una neurona en las 9 clases distintas. En un escenario ideal, ningún número debería ser mayor a 1, ya que indicaría que una neurona se activó para más de una clase.

Una vez calculado este mapa sumaremos todos los valores de la matriz, le restaremos la cantidad de neuronas que no eran cero dentro del mapa (por lo que en el caso ideal el factor nos daría cero) y lo dividiremos por esa misma cantidad.

$$FD(D, W) = \frac{sumActCruz(D, W) - noSonCero(D, W)}{noSonCero(D, W)}$$

Donde D representa los datos a utilizar y W la matriz de pesos de la red neuronal ya entrenada. Se supone que $sumActCruz$ internamente calcula las activaciones, al igual que $noSonCero$ calcula la cantidad de neuronas que se activaron.

3.2.3. Experimentación preliminar

Para decidir qué tamaños de redes utilizar durante la experimentación, decidimos realizar un análisis preliminar viendo cómo la red neuronal "cubría" el conjunto de entrenamiento con distinta cantidad de neuronas.

Para esto, realizamos un análisis de componentes principales de los datos y trasladamos los mismos a un espacio planar utilizando las dos primeras componentes como base del nuevo espacio. En un espacio más grande cada neurona se mueve en una dimensión mayor, pero ya que Kohonen tiene una naturaleza geométrica, suponemos que no hace falta una cantidad mayor de neuronas para cubrir de forma eficiente los datos en un espacio de mayor dimensionalidad; dicho de otra forma, la relación se mantiene independiente de la dimensionalidad. Es por eso que este análisis nos pareció relevante como criterio de elección de las arquitecturas.

Durante este análisis preliminar, utilizamos el modo de auto-ajuste con 1000 épocas de entrenamiento. Analizamos arquitecturas cuadradas de neuronas, desde 5x5 hasta 50x50 con incrementos de 5 neuronas en cada paso y un experimento final con una red de 100x100 neuronas. Por lo que, realizamos un total de 11 experimentos preliminares.

A continuación vemos los gráficos al realizar la transformación usando las dos primeras componentes principales sobre el Fold 9. Sin embargo se obtuvieron resultados similares para los demás. Dado que no aportaba nada distinto mostrarlos a todos decidimos colocar los gráficos de sólo uno de ellos.

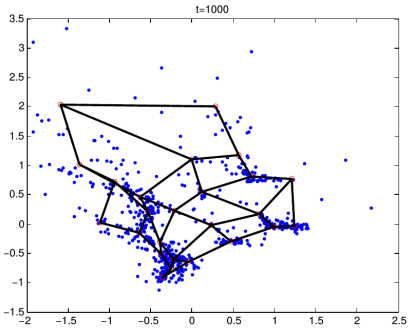


Figura 11: 25 (5x5) neuronas

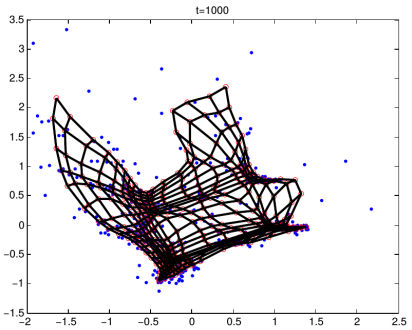


Figura 12: 400 (20x20) neuronas

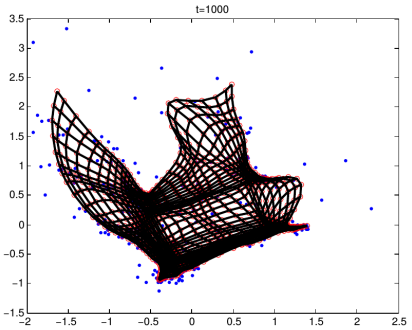


Figura 13: 1225 (35x35) neuronas

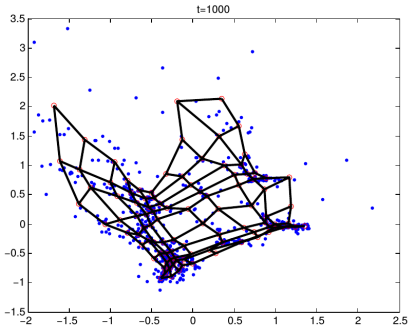


Figura 14: 100 (10x10) neuronas

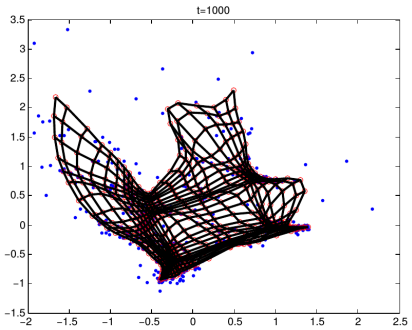


Figura 15: 625 (25x25) neuronas

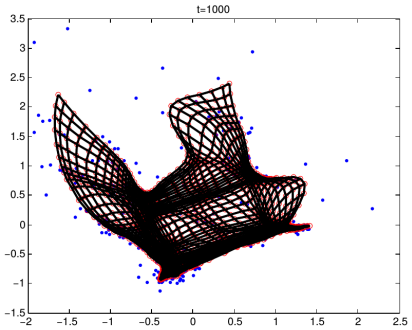


Figura 16: 1600 (40x40) neuronas

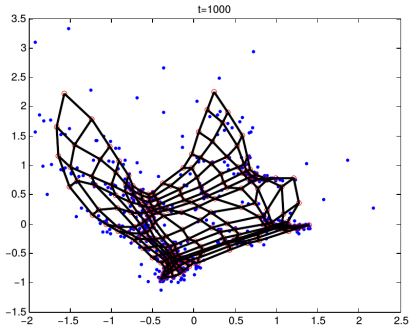


Figura 17: 225 (15x15) neuronas

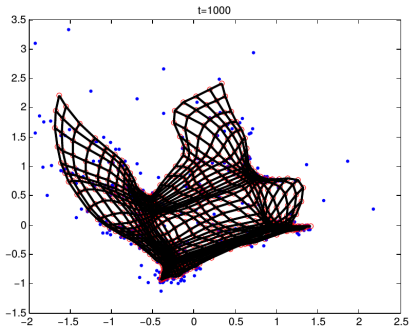


Figura 18: 900 (30x30) neuronas

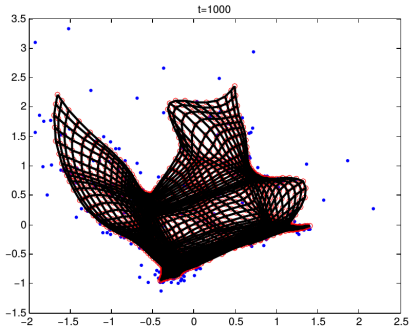


Figura 19: 2025 (45x45) neuronas

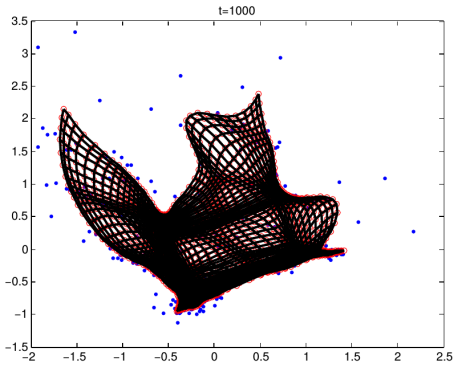


Figura 20: 2500 (50x50) neuronas

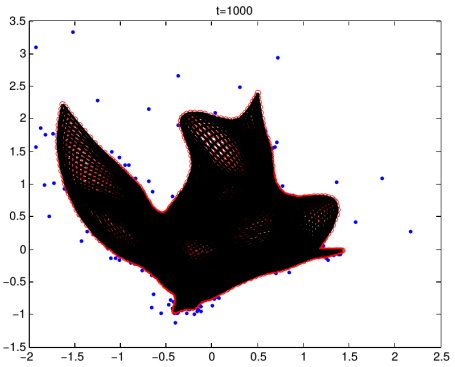


Figura 21: 1000 (100x100) neuronas

Como se puede observar en los gráficos obtenidos, las diferencias más significativas en la forma que adquiere la red se pueden notar entre las redes de 5x5 hasta 30x30. Desde 30x30 en adelante no se notan diferencias mayores en la topología de la red, si no que, expresado de alguna forma, es un incremento en la "resolución" de la misma.

3.2.4. Elección de cantidad de épocas

Para decidir con cuantas épocas realizar la experimentación analizamos como la red de 30x30 (a la que encontramos como más representativa) iba cambiando su forma al pasar las épocas. Mostramos a continuación los gráficos correspondientes.

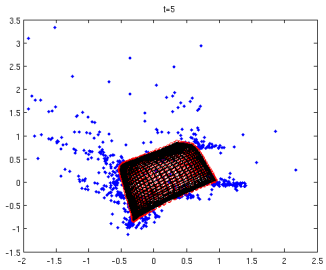


Figura 22: 5 épocas

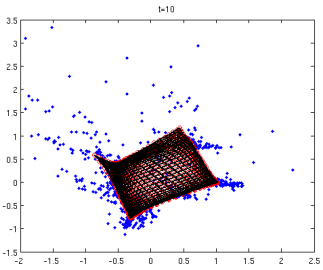


Figura 25: 10 épocas

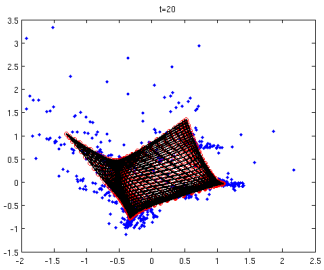


Figura 28: 20 épocas

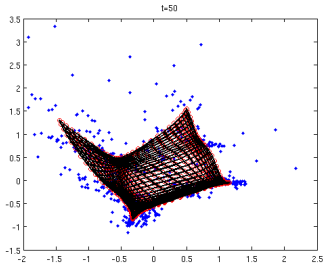


Figura 23: 50 épocas

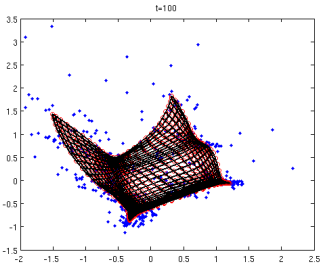


Figura 26: 100 épocas

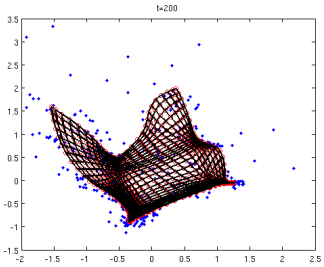


Figura 29: 200 épocas

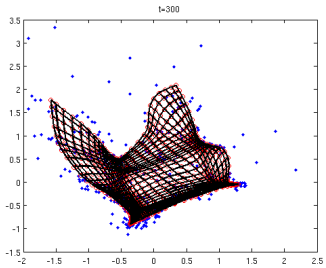


Figura 24: 300 épocas

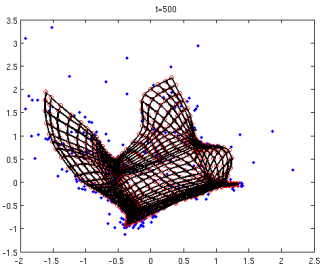


Figura 27: 500 épocas

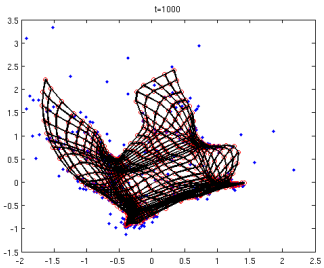


Figura 30: 1000 épocas

Se puede observar que a partir de las 200 épocas procede más como un ajuste fino, por lo que no hay cambios demasiado significativos en la topología. Esto principalmente puede darse por la técnica de autoajuste utilizada en la red. De todas formas, encontramos el limite de 300 épocas como un número apropiado para la experimentación.

3.2.5. Experimentación y análisis de resultados

En base a la experimentación preliminar, decidimos experimentar con las redes de 5x5, 10x10, 15x15, 20x20, 30x30 y 50x50, con autoajuste activado y tomando las métricas de distancia a dominantes, absoluta y relativa, con distancia euclidiana y Manhattan; y la métrica de factor de cruce. Se presentan los resultados obtenidos luego de realizar los promedios entre los 9 folds.

Tam	DD_{rel}	DD_{abs}	$DD_{rel}(M)$	$DD_{abs}(M)$	FC
5x5	0.055	0.392	0.044	0.445	2.421
10x10	0.033	0.466	0.027	0.542	0.972
15x15	0.024	0.503	0.019	0.584	0.499
20x20	0.022	0.615	0.018	0.720	0.328
30x30	0.018	0.759	0.015	0.899	0.186
50x50	0.017	1.201	0.014	1.432	0.108

Cuadro 9: Resultados de experimentación de la red de Kohonen para distintas arquitecturas

Como se puede observar, efectivamente ocurre una reducción en DD_{rel}, $DD_{rel}(M)$ y FC a medida que aumentamos el tamaño de la red, mientras que DD_{abs} y $DD_{abs}(M)$ crecen. Las reducciones en los valores ocurren porque, en términos de las métricas elegidas, se genera una mejor clusterización a medida que aumentamos el tamaño de la red; es decir que los dominantes se vuelven relativamente más cercanos a las instancias que representan y hay una mejor separación entre los distintos clusters a nivel de activación de neuronas.

Por otro lado, las distancias absolutas crecen porque existe una cantidad mayor de neuronas en el medio, lo que nos puede estar indicando una cantidad mayor de neuronas que no se activan para ninguna clase.

Observemos qué ocurre si representamos los mapas de dominantes como mapas de calor. En este caso la clase 0 representa aquellas neuronas que no se activaron para ninguna clase.

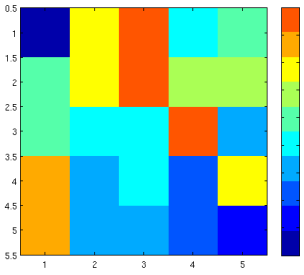


Figura 31: 5x5

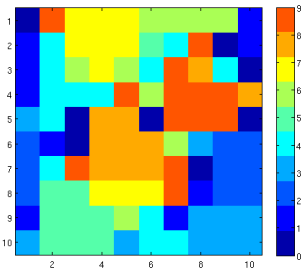


Figura 33: 10x10

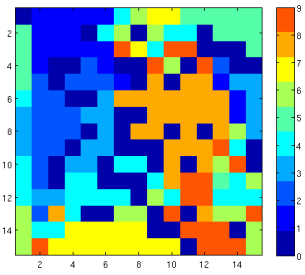


Figura 35: 15x15

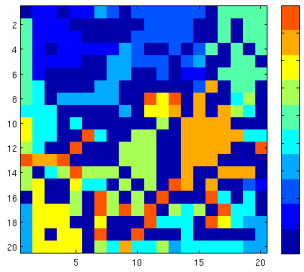


Figura 32: 20x20

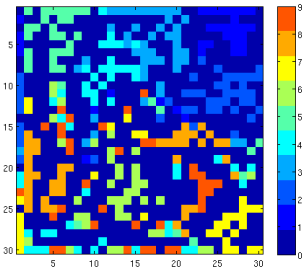


Figura 34: 30x30

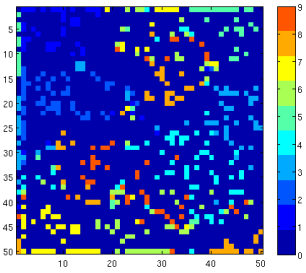


Figura 36: 50x50

Se puede observar que los mapas concuerdan con los resultados analíticos obtenidos. Desde un principio se puede ver una clara clusterización de la red, y a medida que aumentamos el tamaño de la misma la cantidad de neuronas sin activación crece.

Dados estos resultados creemos que la red de 15x15 es una buena arquitectura para modelar el problema, ya que genera una buena clusterización y tiene un factor de cruce aceptable, mientras que hay pocas neuronas que no se activan.

4. Conclusiones

Considerando lo discutido en la sección de Resultados, haremos una breve recopilación de las conclusiones alcanzadas basadas en la experimentación realizada.

4.1. Reducción de dimensiones

En relación a la ubicación espacial de las instancias al utilizar las tres primeras componentes principales calculadas, no se vieron malos resultados en el sentido de instancias totalmente dispersas o instancias de validación muy alejadas de las instancias de entrenamiento. En líneas generales, todas las instancias de experimentación analizadas presentaban características similares: algunas clases bien clusterizadas y otras tal vez menos concentradas espacialmente. Sin embargo, dado que con ninguno de los casos analizados se obtuvieron separaciones espaciales considerables entre todas las clases no podemos afirmar que alguna combinación de parámetros haya funcionado mejor. Para afirmar esto, dados los resultados obtenidos sería necesario realizar un posprocesamiento de los resultados, por ejemplo aplicar k vecinos más cercanos sobre las instancias de validación para ver si con la transformación obtenida a partir de las componentes principales calculadas se puede clasificar bien.

Dado que los resultados espacialmente son similares haremos referencia a cuál método resulta mejor en función de la cantidad de épocas que necesitan para alcanzar los resultados obtenidos. Dado que la combinación de ortogonalidad usando Oja converge en casi todos los casos analizados en una cantidad baja de épocas consideraremos que los mejores resultados fueron obtenidos con esta combinación. Sin embargo, queda pendiente analizar más en profundidad las características bajo las cuales la regla de Sanger converge dado que las veces que lo hizo, en general fue en menos épocas que Oja. Por otro lado, observando los resultados al usar el criterio de matriz de delta pesos nula es claro que con 500 y 250 épocas de máximo se alcanzó un nivel de convergencia con resultados espaciales similares. Por ende, sería necesario analizar a partir de qué época se alcanza esa convergencia. Por razones de tiempo planteamos ese análisis como trabajo futuro ya que implicaría realizar varias corridas sobre el set de datos para hallar el punto en el cual se empiezan a obtener los números observados para la norma Frobenius de la matriz de pesos. Otra posibilidad sería tratar de obtener la época en la cual se alcanza un valor similar, por ejemplo 0.005 aunque dependiendo del caso esto puede ser una aproximación burda.

En relación a las preguntas planteadas en el enunciado se podría decir que dada la distribución espacial de las instancias no queda claro que se pueda hacer una clasificación precisa con todas las clases. Si bien algunas como las representadas con color azul y magenta aparecen normalmente más distanciadas de las demás y otras como las celeste y roja aparecen más concentradas, es posible que al aplicar un algoritmo de clasificación los resultados no sean muy buenos. Algo a tener en cuenta es que realizar clasificación apenas considerando 3 componentes principales sobre instancias con más de 800 parámetros parece demasiado ambicioso. Sería razonable analizar los resultados considerando más componentes principales y observar si allí se obtienen buenos resultados. Obviamente en esos casos no será posible hacer una representación visual como en este caso pero analizando, por ejemplo, knn sobre los vectores de la transformación se podrá observar si los resultados mejoran.

En relación a la calidad de estos métodos para la clasificación de los documentos sería necesario analizarlos con una métrica cuantitativa (y no solamente basarnos en la apreciación cualitativa de los gráficos) sin embargo dada la poca diferenciación obtenida espacialmente pareciera que no son la mejor técnica a utilizar. Vale aclarar que la experimentación con las técnicas no fue

exhaustiva y completa por cuestiones de tiempo por lo que aún quedan muchas posibilidades para explotar en los parámetros de los métodos.

En cuanto a las mejoras o ampliaciones, algo a tener en cuenta es agregar un método numérico para la clasificación de nuevas instancias dado el entrenamiento. Para ello se puede utilizar algoritmos como k-means y dadas esas implementaciones algo a tener en cuenta es analizar los resultados con más componentes principales. Otro aspecto a considerar es tratar de hallar un criterio adaptativo para el learning rate que permita converger más rápidamente ya que con los casos analizados no fue posible realizar este aprendizaje más rápidamente.

4.2. Mapeo de características

En general, se vieron resultados aceptables para todas las arquitecturas elegidas. La arquitectura que da resultados que no son tan aceptables es aquella más chica, de 5x5, ya que su factor de cruce es demasiado alto y por lo tanto puede que no dé una buena clasificación si se utiliza la red para ello. A partir de 10x10, el factor de cruce disminuye drásticamente por lo que los resultados pasan a ser significativos.

Creemos que una arquitectura de más de 30x30 es excesiva para este problema, ya que comienzan a verse demasiadas neuronas que no se activan para ninguna clase, lo que significa que la cantidad de neuronas puede reducirse para obtener resultados similares y así disminuir el tiempo de entrenamiento.

4.3. Posibles trabajos futuros

En la reducción de dimensiones queda pendiente evaluar como criterio de corte para la matriz de delta pesos ver que dos matrices de épocas consecutivas sean prácticamente iguales ya que, como se vio en la experimentación, no se llega a que la matriz de delta pesos sea nula.

En cuanto al mapeo de características, quedan muchas experimentaciones por las cuales seguir investigando. Por ejemplo, experimentar con parámetros fijos ya que solamente lo hicimos con auto-ajustables, con distintas funciones de distancia y actualización, con arquitecturas no cuadradas de neuronas; con una arquitectura de salida distinta, con tal de lograr un clasificador y con ello verificar precisión, etc. Otra línea interesante a investigar es la posibilidad de efectuar una reducción de dimensionalidad antes de entrenar la red para realizar la clusterización en un espacio que tenga menos dimensiones, y que por lo tanto lleve menos tiempo, verificando que la proyección del conjunto de entrenamiento en el subespacio nuevo no altera los resultados de la red neuronal.

5. Opciones de uso

Ambos problemas fueron resueltos usando Matlab. Para ejecutarlos describimos a continuación las opciones posibles.

5.1. Reducción de dimensionalidad

Para la ejecución del código hay dos archivos .m, uno de ellos el que utilizamos para generar todos los resultados de manera automática: main.m el cual varía las repeticiones, criterio de parada, regla y folds. El otro, usarHebbiano.m, permite correr una única instancia si se le dan los siguientes parámetros:

- criterioParada: es 'p' o 'o' según se quiera corte por delta pesos u ortogonalidad
- regla: es 's' o 'M' según se quiera usar Sanger u Oja (M)
- learningRate: es el valor de learningRate a usar
- alpha: es el valor en la fórmula learningRate = epoca$\hat{}$(-alpha) para actualizar el learning rate. Si alpha es 0 entonces esa fórmula no se usa
- calcularPesos: es un bool que cuando es true indica que se calcule la matriz de pesos mientras que cuando es false indica que se lea una matriz ya creada
- cantEpocas: es la cantidad de épocas máxima que se admite (para tener otro criterio de corte)
- trainFilename: es el nombre (path completo) del archivo donde están las instancias de entrenamiento
- testFilename: es el nombre (path completo) del archivo donde están las instancias de validación
- weightsFilename: es el nombre (path completo) del archivo donde se guardará (o desde donde se leerá si calcularPesos=false) la matriz de pesos
- datosFilename: es el nombre del archivo donde se escribirá la información sobre la corrida
- figureFilename: es el nombre del .fig que se creará con el gráfico de las instancias

5.2. Mapeo de características

Al igual que en el punto anterior, main.m realiza la experimentación detallada en el informe. Mientras que el archivo usarKohonen.m, permite correr una única instancia si se le dan los siguientes parámetros:

- learningRate: es el valor de learningRate a usar
- sigma: es el valor que decide el rango de neuronas afectadas durante la actualización
- autoajuste: ajusta dinámicamente sigma y learningRate a medida que avanzan las épocas
- M1: es la cantidad de filas de la matriz de neuronas
- M2: es la cantidad de columnas de la matriz de neuronas
- calcularPesos: es un bool que cuando es true indica que se calcule la matriz de pesos mientras que cuando es false indica que se lea una matriz ya creada
- cantEpocas: es la cantidad de épocas máxima a utilizar
- trainFilename: es el nombre (path completo) del archivo donde están las instancias de entrenamiento
- testFilename: es el nombre (path completo) del archivo donde están las instancias de validación
- weightsFilename: es el nombre (path completo) del archivo donde se guardará (o desde donde se leerá si calcularPesos=false) la matriz de pesos
- figureTrainFilename: es el nombre del mapa de dominantes .fig que se creará con las activaciones de las instancias del archivo de entrenamiento
- figureTrainFilename: es el nombre del mapa de dominantes .fig que se creará con las activaciones de las instancias del archivo de validación
- datosFilename: es el nombre del archivo donde se escribirá la información sobre la corrida

Referencias

[1] John A. Hertz, Anders S. Krogh, Richard G. Palmer, *Introduction To The Theory Of Neural Computation*, Westview Press, June 24, 1991.

[2] Simon O. Haykin, *Neural Networks and Learning Machines*, Prentice Hall, 3rd Edition, November 28, 2008.

[3] Jeff T. Heaton, *Introduction to Neural Networks for Java*, Heaton Research, Inc. November 1, 2005.