Capítulo 5. Construcción de una Red Neuronal Artificial Asesora

Capítulo 5. Construcción de una Red Neuronal Artificial Asesora

5.1 Construcción de la red

A lo largo de las investigaciones realizadas en diversos laboratorios, se ha demostrado que las redes neuronales artificiales son una herramienta muy eficiente para el reconocimiento de patrones sin intervención directa de un usuario. El éxito de estos sistemas es que están basados en la manera en que el propio cerebro humano procesa la información.

Con la experiencia obtenida por el proyecto ANSSYD se llegó a la conclusion de que la mayoría de los algoritmos de segmentación heurísticos presentan fallas en al menos un 40% de los casos debido a la gran variedad de formas que la letra manuscrita puede tomar. Esto es, algunas personas escriben de manera muy espaciada, otras de manera estrecha, algunos omiten los espacios blancos que caracterizan las letras p, b, e, o, etc., otros presionan muy levemente la pluma sobre el papel y otros lo hacen de manera rigurosa, generando una gran variedad de ancho de tinta.

Es por ello, que el uso de una red neuronal que simule un asesoramiento resulta vital. Si el algoritmo se segmentación presenta alguna falla, la red revisará cada punto de segmentación obtenido y verifica si en verdad es correcto. Lo más recomendable en estos casos, es el tratar de sobresegmentar (en un nivel controlable) la palabra de tal manera que nos aseguremos que los verdaderos puntos de segmentación se encuentran ya

establecidos y la RNA se encargará solamente de eliminar los que son incorrectos [Blumenstein & Verma, 1999].

5.1.1 Conversión de la Red Neuronal (SHELL-BP) al paradigma orientado a objetos.

La Red neuronal que se implantó en el proyecto fue proporcionada por la Dra. Pilar Gómez Gil, la cual forma parte del proyecto "Predicción de Electrocardiogramas y otras señales caóticas utilizando RNA". Es un Shell para utilizar redes neuronales de varios niveles entre nodos con Retro-propagación. Este Shell se encuentra construido bajo el paradigma procedural. Debido a que este proyecto en general se encuentra construido bajo el paradigma orientado a objetos, el Shell se re-escribió bajo este modelo. Otras razones de esta acción fueron el aprovechar las ventajas que el modelado orientado a objetos presenta en cuanto a reutilización de código, ver apéndice A.

La arquitectura de la red neuronal está basada en el Modelo de Retro-propagación a varios niveles. Está diseñada para resolver problemas no lineales como lo es el proceso de segmentación. Se dice que el proceso de reconocimiento de puntos de segmentación no es lineal debido a que no es fácil establecer distinciones entre clases. La figura 5.1 muestra de manera clara esta diferencia.

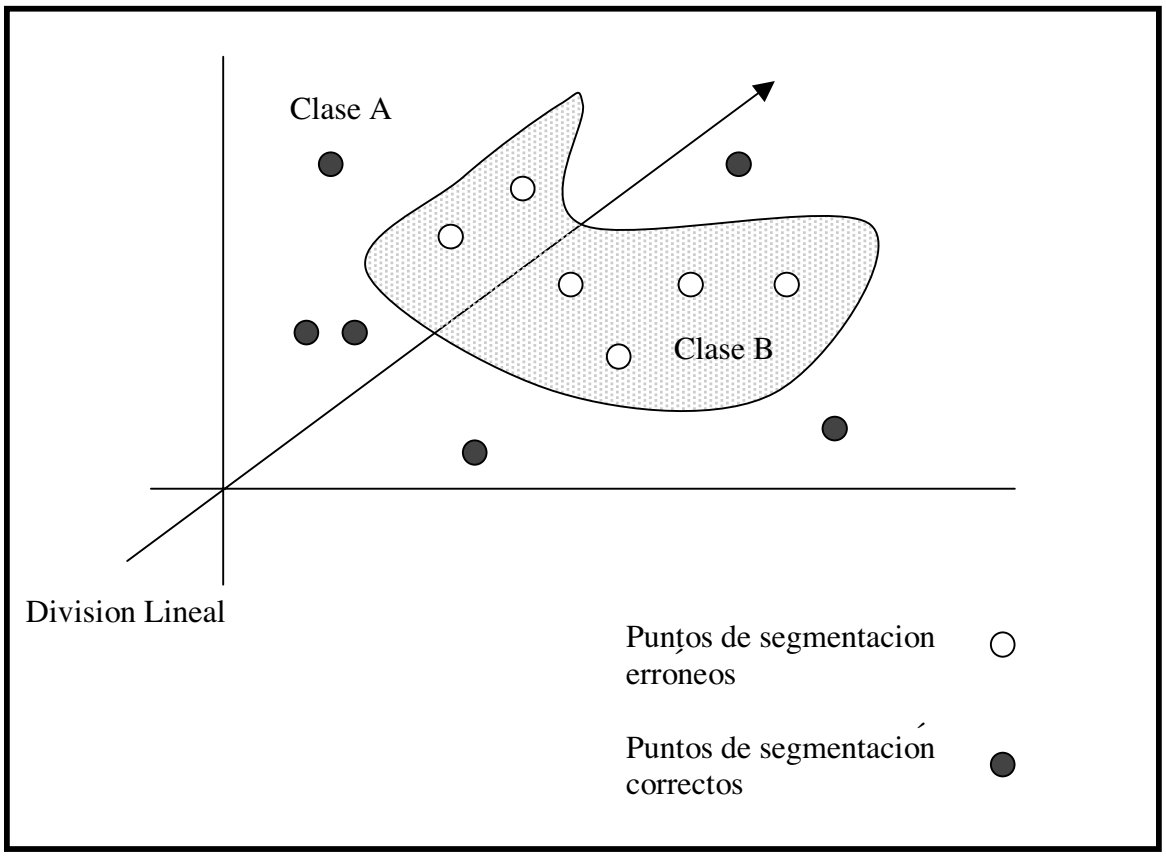

Figura 5.1 El reconocimiento de puntos de segmentacion no es un problema lineal

5.1.2 Topología de Red

La Topología de una Red Neuronal Artificial es una estructura determinada, de número de niveles, de número de nodos por nivel y la conexión entre ellos. La escritura general de las redes neuronales está conformada por un nivel de entrada, al menos un nivel intermedio y un nivel de salida.

Debido a que no existe una metodología exacta que indique la manera de realizar las conexiones o el número de niveles escondidos óptimos para obtener resultados

óptimos [Linares & Spínola, 2000], en el proyecto actual se realizaron pruebas sobre distintas topologías para evaluar sus resultados.

Una manera de medir el éxito o fracaso de una topología de red neuronal artificial es el observar el error total por barrida que se va obteniendo durante el aprendizaje. Este error se obtiene al comparar los resultados obtenidos por la red y la respuesta ideal esperada. Si los resultados distan en gran medida de los esperados, el porcentaje de error se incrementa, de lo contrario va decreciendo paulatinamente. Las barridas representan el número de veces en que la red neuronal ejecuta el proceso de aprendizaje. En teoría, entre más barridas realice la red, mejor entrenada se encontrará. Sin embargo, ciertas topologias no ofrecen este comportamiento ya que su porcentaje de error puede disminuir en la i-ésima barrida y aumentar en la barrida i+1. Esto demuestra que la topología no es adecuada al problema ya que la red no minimiza su porcentaje de error, sino que varía dependiendo de la barrida.

La topología más adecuada a nuestro problema es a dos niveles escondidos. En el nivel de entrada se establecieron 108 nodos, en el primer nivel escondido de 130 y en el segundo también. El nivel de salida representa los diferentes tipos de soluciones a los que puede llegar la red. En nuestro caso, el nivel de salida está compuesto de un solo nodo, que indica si el punto de segmentación analizado es correcto o incorrecto (0 ó 1). En las secciones posteriores se explicará el por qué se escogieron las dimensiones de los niveles de entrada e intermedios.

5.2 Creación de un archivo de entrenamiento

Para que la red pueda ser entrenada es necesario contar con un archivo de entrenamiento que contenga los patrones de entrada así como las salidas esperadas después de la evaluación de la red.

La red neuronal es entrenada de la siguiente manera; primero lee el archivo de entrenamiento que contiene los patrones de entrada (es decir, los elementos sobre los cuales establecerá una decisión final). Una vez realizada esta acción, el algoritmo asigna aleatoriamente pesos a cada una de las conexiones que conforman la red. A continuación compara los resultados obtenidos con los resultados deseados. Como es obvio, la diferencia entre lo obtenido y lo esperado será muy grande en la primera barrida, por tanto modificará los pesos de los nodos y ejecutará el proceso de aprendizaje de nueva cuenta. La siguiente barrida será más exitosa ya que los pesos no serán asignados aleatoriamente sino que se modificarán los obtenidos en la barrida anterior. De Nuevo comparará los resultados y modificará los pesos. Este proceso continuará hasta que el error entre lo esperado y obtenido sea mínimo.

HAWOST utiliza un archivo de entrenamiento basado en ventanas, utilizando la propuesta presentada por [Blumenstein & Verma, 1999]. Cada ventana contendrá un conjunto de píxeles correspondientes al punto de segmentación.

5.2.1 Generación de un vector de ventanas

Los puntos de segmentación arrojados por el algoritmo heurístico son manualmente analizados de tal manera que las coordenadas en x sean catalogadas en clases de puntos "correctos" e "incorrectos" [Blumenstein & Verma, 1999]. La herramienta proporciona una forma muy sencilla de llevar a cabo esta tarea. Como se muestra en la figura 4.9, primero se debe escoger la opción de dibujo de puntos de segmentación. A continuación se podrán dibujar puntos a lo largo de la imagen dando *clicks* sobre ella, tomando en cuenta solo las coordenadas en x.

Debido a que HAWOST es una herramienta que le facilita la tarea al usuario, es posible eliminar un punto de segmentación mal establecido. Solo basta con seleccionarlo y posteriormente usar la opción de eliminar punto de segmentación.

Para establecer si un punto de segmentación es correcto, es necesario seleccionarlo y verificar que su color pase de azul a verde. Esto significa que si un punto de segmentación está en verde, la red neuronal lo debe catalogar como erróneo y aprender que este tipo de puntos no deben ser considerados correctos.

Para cada punto de segmentación en particular (dada su coordenada en x), una matriz de píxeles es extraída y guardada en un archivo. Un proceso de extracción es ejecutado a continuación, partiendo a la matriz del punto de segmentación en ventanas pequeñas del mismo tamaño y analizando la densidad de píxeles blancos y negros. De esta forma, en lugar de presentar un vector muy grande de píxeles a la red neuronal, sólo

se le presenta un vector de densidad de ventanas. Como ejemplo, si una ventana de dimensiones 5x5 contiene 12 pıxeles blancos, entonces se le asignara un valor de 0.48 (numero de pıxeles negros/25) y sera anadida a un vector cuya posicion representara a esa ventana. Un ejemplo de este extractor se presenta en la figura 5.2. Acompanando a cada vector de ventanas se le anadira la salida esperada y se guardara en el archivo de entrenamiento.

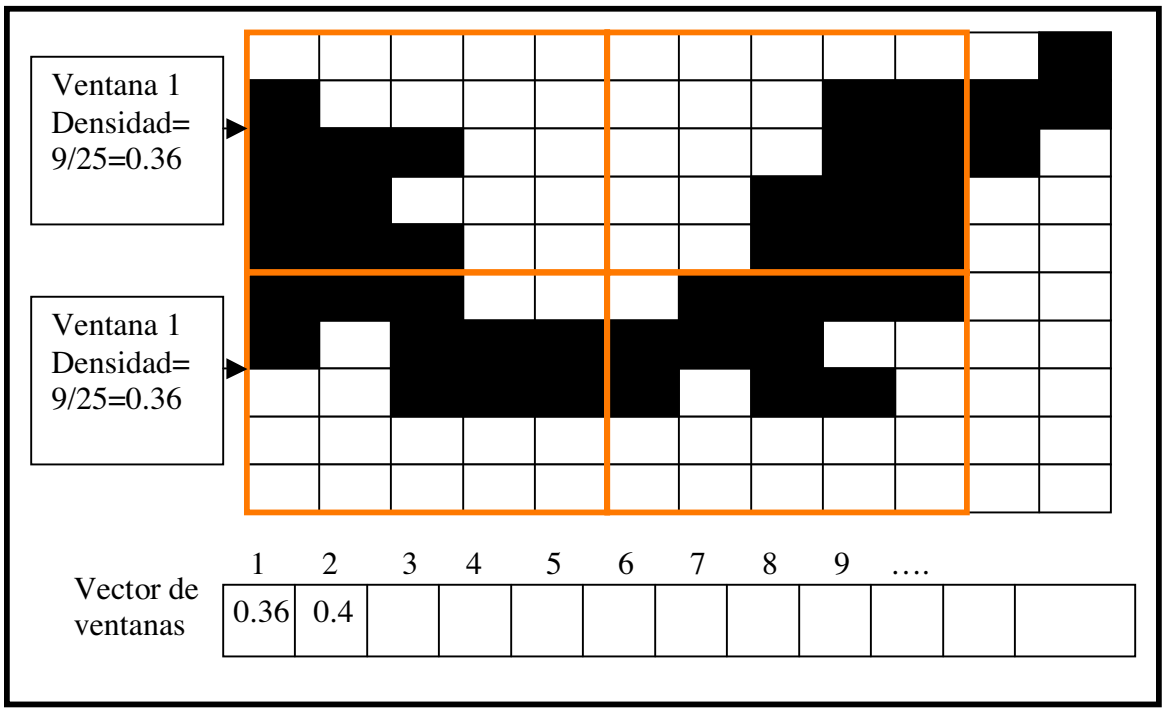

Figura 5.2 Generacion del vector de ventanas

5.2.2 Estructura del archivo de entrenamiento

El archivo de entrenamiento debera estar conformado por un conjunto de patrones, cada uno seguido por la salida esperada, es decir, si se trata o no de un punto de

segmentación correcto. Cada patrón será un vector de ventanas donde cada una representa la densidad de píxeles negros en ella.

Además de esto, cada vector correcto deberá estar intercalado por un vector de un punto de segmentación incorrecto. Esta restricción se debe a que se ha demostrado que las redes neuronales aprenden de manera correcta si un patrón correcto va seguido por uno incorrecto y así sucesivamente. De otra manera la red se sesgaría hacia un punto en que sólo aprendería a reconocer los ciertos o viceversa. Sin embargo, el usuario solamente debe preocuparse de establecer un número equivalente entre correctos o incorrectos. La herramienta se encargará de intercalarlos a través de un algoritmo sencillo.