Computación y Sistemas Vol. 6 No. 1 pp. 038 - 050
© 2002, CIC - IPN. *ISSN 1405-5546* Impreso en México

Reconocimiento Eficiente de Caracteres Alfanuméricos Provenientes de Mapas Ráster por Medio de Clasificadores Neuronales

Efficient Alphanumeric Character Recognition from Raster Maps by Means of Neural Classifiers

Aurelio Velázquez [1], Humberto Sossa [2] y Serguei Levachkine [2]
[1] Instituto Mexicano del Petróleo
Eje Central Lázaro Cárdenas 152, México D. F. 07730, México
[2] Centro de Investigación en Computación-IPN
Av. Juan de Dios Bátiz esquina con M. Othón de Mendizábal
Colonia Nueva Industrial Vallejo, México, D. F. 07738, México
E-mail: [avela, hsossa, palych]@cic.ipn.mx

Artículo recibido en Mayo 05, 2001; aceptado en Agosto 02, 2002

Resumen

Las Redes Neuronales Artificiales (RNA), desde sus orígenes en los años 50's, han sido usadas ampliamente en muchos campos. Uno de estos campos es el reconocimiento automatizado de texto, donde los caracteres pueden presentarse en una gran variedad de tamaños, tipos, inclinaciones, o incluso en cursiva o subrayado. En este trabajo se describe una nueva manera para el aprendizaje de clasificadores de tipo neuronal que pueden ser usados con éxito para reconocer casi cualquier tipo de carácter alfanumérico entre los tamaños de 8 a 25 puntos, en presencia de traslaciones y rotaciones, así como el ruido introducido por los píxeles distorsionados durante el proceso de rotación y barrido, por ejemplo. Como arquitectura de red se usó el perceptrón multicapa y como paradigma de aprendizaje se usó una versión modificada del de propagación hacia atrás (Backpropagation). El conjunto resultante de RNA fue verificado en dos escenarios: con 18,432 muestras generadas en forma sintética y 1025 muestras obtenidas de mapas ráster escaneados en color. Los porcentajes de eficiencia respectivos fueron de 99.82 y 95.512%.

Palabras clave: Redes neuronales artificiales, Reconocimiento de texto, Épocas, Paradigma de propagación hacia atrás, Red de alimentación hacia adelante, Aprendizaje supervisado.

Abstract

Since the earlier 50's Artificial Neural Networks (RNA) have been widely used in many fields. One of these is fields is text recognizing, where characters might appear in great variety of sizes, types, rotations. In this work a simple technique to train neural classifiers is described. Once trained the set of classifiers can be used to recognized almost any type of character from 8 to 25 points, in presence of translations, rotations, and noise introduced by the scanning process. As architecture the classical multiplayer perceptron was used. As learning paradigm a modified back-propagation version was proposed. The set of trained RNA was tested in two scenarios: with 18, 432 syntactically generated patterns, and with 1025 patterns obtained form color scanned cartographic maps. The performance percentages were, respectively, of 99.82% and 95.512%.

Keywords: Artificial Neural Networks, Text Recognition, Epochs, Backpropagation Paradigm, Feed-forward Network, Supervised Learning.

1 Introducción

Los mapas impresos, en distintos medios, han sido la forma más común desde tiempos remotos para representar *información geográfica*, Los mapas impresos son un medio fácil para simbolizar el entorno natural, en una forma analógica, mediante líneas y dibujos que pueden mostrar los rasgos hechos por la naturaleza y por el hombre. Utilizando "capas" de información, se agregan los contornos, los límites estatales, las redes hidráulica, ferroviaria, etc., así como los puntos de interés.

Para poder identificar claramente el contenido del mapa, éste es acompañado por una gran variedad de leyendas para identificar cada evento. Este trabajo está orientado para analizar mapas cartográficos con leyendas en español.

Muchos de los mapas actualmente existentes se encuentran impresos en papel. Se requiere de mucho espacio físico para almacenar tanta información. El manejo de la misma se hace cada vez más difícil también. Esto exige la digitalización computarizada de dicha información, para que pueda ser manejada bajo un Sistema de Información Geográfica (SIG). Para lograrlo se requiere, sin embargo, separar cada mapa en sus diferentes capas. Una de estas capas es la de caracteres alfanuméricos.

Los caracteres alfanuméricos impresos en los mencionados mapas vienen en distintos tamaños, tipos, orientaciones, colores, en cursiva o subrayados, y entremezclados con otros objetos. En la figura 1 se muestran algunos ejemplos.

Todos estos factores hacen que su identificación tenga un alto grado de dificultad [Chhabra, 1997 y Levachkine, 2000].

Lo anterior exige del diseño de algoritmos robustos para aislar (segmentar) conjuntos de píxeles y, una vez aislados estos, utilizar clasificadores especialmente diseñados para reconocer, con gran certidumbre, a cada carácter.

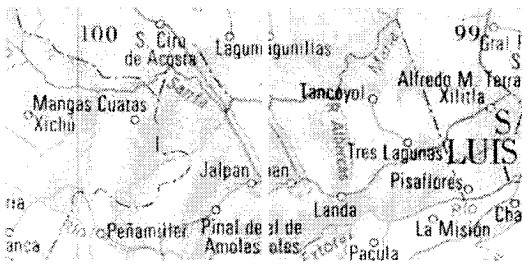


Figura 1. Un mapa conteniendo a la vez caracteres alfanuméricos y otros objetos

Para que el reconocimiento de caracteres sea utilizado en la producción de mapas vectoriales a partir de mapas ráster, se necesita un porcentaje de aciertos mayor al 90%.

En [Myers *et al*, 1995], por ejemplo, Myers y sus colegas usan conocimiento acerca del contexto para extraer de un mapa en forma automatizada texto y algunos otros componentes. Aprovechan el principio de que los elementos y sus relaciones pueden ser reconocidos con facilidad si el algoritmo sabe que está buscando y lo hace en una búsqueda directa. En su método, Myers y sus colegas formulan primero hipótesis, luego usan el conocimiento *a priori* asociado con los mapas para verificar si dichas hipótesis son ciertas.

Por otro lado, en [Wenyin *et al* 1997], Wenyin y sus colegas primero definen dos clases de caja: la de carácter y la de texto (que es una cadena de cajas de carácter). Éstas son primeramente segmentadas de acuerdo a las características de los trazos del carácter (las que son cortas, esto es, la máxima razón entre la longitud y el ancho es 10). Las cajas de caracteres cerradas son luego agrupadas para formar o, una cadena horizontal o, una cadena vertical de acuerdo a la razón de la longitud y el ancho de la caja. Sólo se permite texto orientado en forma vertical u horizontal.

Así mismo en [Luo *et al* 1997], Luo y sus colegas utilizan operaciones morfológicas direccionales, para separar los caracteres que se encuentran junto con las líneas, en los mapas o en los gráficos. Para esto usan ocho elementos estructurantes, para cuatro trayectorias en las direcciones (horizontal, vertical y diagonal izquierda y diagonal derecha). La segmentación de los caracteres, la llevan a cabo en tres pasos: primero borran los caracteres contenidos en la imagen. A continuación restauran los bordes de las líneas que quedaron en la imagen y, por último, restan la imagen de líneas de la imagen original para obtener la imagen con los caracteres. Los planos que manejan, sólo están formados con líneas y caracteres.

En [Deseilligny *et al* 1997], los autores proponen una estrategia denominada de "Pelar la Cebolla", quitando los elementos reconocidos (mediante un "borrado no a ciegas"). Primeramente, reconocen y eliminan los elementos de los que se tengan datos externos. Enseguida, las áreas texturizadas. Después las cadenas de caracteres y de demás símbolos. Luego los rasgos representados con formas variables, y finalmente las redes. La taza de error en que se encuentran sus procesos es del 5%.

En [Frischknecht *et al* 1997], los autores identifican los elementos en los mapas topográficos, mediante la comparación, con un conjunto de plantillas generadas, de los tipos de letras capturadas por barrido y de los patrones de tipo símbolo, formando con ellos, una biblioteca como una base de conocimiento gráfica. Estas plantillas, tienen asociadas otras plantillas alternas, como la 'i' que tiene a 'r', 'j' y 'l'. Para procesar el mapa completo, definen una secuencia de búsqueda. Primero, buscan los elementos más grandes, seguidos por los más pequeños. Seleccionan secuencias óptimas, que minimizan el riesgo de una clasificación falsa debido a la similitud de ciertas letras, ejemplo: la letra 'H' es buscada antes que la letra 'T' y la letra 'I'; y la letra 'T' es buscada antes que la letra 'I'. Giran la plantilla, de -90° a 90° con incrementos de 2°, en su tamaño original y, después la escalan para los rasgos de distinto tamaño. Para cada posición de la plantilla transformada, el valor del color de la plantilla original es procesado utilizando este factor de escala. Obtienen resultados que oscilan entre el 88.5% y 96.4%.

Finalmente, en [Levachkine 2000], el autor presenta una metodología compuesta de tres etapas. Durante la etapa de preprocesado se busca adecuar la imagen de entrada para un mejor manejo de los elementos que la conforman a través de técnicas conocidas pero eficientes de tratamiento digital de imágenes. En la etapa de procesado se usa un algoritmo denominado de "caterpilar" (oruga), que permite extraer los rasgos identificados para su futura vectorización. Finalmente, durante una etapa de postprocesamiento se corrigen posibles errores a través del uso de propiedades topológicas de los elementos en las capas vectoriales y la correlación espacial de las capas vectoriales correspondientes. Para corregir la capa de caracteres, se utilizan, como conocimiento adicional, diccionarios toponímicos.

Un enfoque muy usado actualmente en el reconocimiento de caracteres y otros patrones (ver por ejemplo [Ansari, 1993a, Ansari 1993b, Hwang, 1992, Kim, 1992, Mitzias, 1994, Pal, 1993, Raman, 1995, Tsai, 1996, Wang, 1996]) que pudiera usarse para separar la capa de caracteres de un mapa es el enfoque neuronal.

Para que los clasificadores neuronales escogidos puedan ser usados eficientemente deben ser entrenados satisfactoriamente. Esto no solo implica usar el tipo y número de muestras apropiado, sino también la manera de aprendizaje con el objetivo de lograr buenos porcentajes de reconocimiento.

En este trabajo se describe una técnica simple pero eficiente para el aprendizaje de clasificadores neuronales, redes neuronales (RNA) en este caso.

En lugar de utilizar caracteres obtenidos directamente de los mapas que son escasos y algunos poco representativos, como se menciona en la literatura [Frischknecht, 1997], se utilizan caracteres "sintéticos" provenientes de un procesador de texto. Con esto se garantiza que las RNA, tendrán un conjunto de muestras de aprendizaje mucho más completo, contemplando todos los caracteres del alfabeto

con distintos tamaños y diferentes inclinaciones que no pueden ser encontradas fácilmente en los mapas temáticos.

Unos caracteres son "aprendidos por la RNA" más rápido que otros, por lo que se propone una modificación al paradigma de Retropropagación, para que "estudie" más veces los caracteres que más tarda en aprender, lográndose con ello un menor tiempo de aprendizaje [Velázquez, 2002].

El conjunto de clasificadores resultante, como veremos más adelante, puede ser usado en el reconocimiento de casi cualquier tipo de carácter alfanumérico entre los tamaños de 8 a 25 puntos, en presencia de traslaciones y rotaciones y el ruido introducido por el proceso de rotación, por ejemplo. El desempeño del conjunto de RNA ha sido también verificado con éxito en el reconocimiento de caracteres provenientes de mapas ráster escaneados en color, lográndose en ambos casos altos porcentajes de eficiencia.

1.1 Organización del Trabajo

El resto del trabajo está organizado como sigue. En la sección 2 se dan algunos pormenores sobre redes neuronales. Estos son necesarios básicamente para facilitar la lectura del resto del trabajo. En la sección 3 se describe en detalle cada una de las etapas de la técnica propuesta, mientras que en la sección 4 se muestran un par de experimentos donde el desempeño de los RNA entrenadas es verificado. Finalmente, en la sección 5 se dan las conclusiones y directivas para el trabajo futuro.

2 Fundamentos de Redes Neuronales

En esta sección se dan algunos aspectos básicos sobre las redes neuronales. Estos son útiles para una más fácil lectura del resto del manuscrito.

2.1 Aspectos Generales

El cerebro humano está formado de una enorme cantidad de *neuronas*: 10^{11} neuronas. Una neurona está conectada con varias vecinas y éstas, a su vez, están conectadas con sus propias vecinas y, mediante sus conexiones, transmiten en paralelo la información almacenada que les corresponde, junto con la de otras neuronas. La unión de la información, transmitida por un gran conjunto de neuronas, forma lo que conocemos como el *Conocimiento*.

Tomando como base esta característica del cerebro humano, se han podido simular Redes Neuronales Artificiales (RNA), formando tantas conexiones como sean necesarias, siendo la comunicación un valor aportado por cada conexión (*Pesos*). Estos pesos son ajustados durante la etapa denominada de aprendizaje. Durante esta etapa se le alimentan valores de aprendizaje, se procesan y los resultados obtenidos son comparados con los resultados esperados. Si difieren, los esperados con los obtenidos, se *ajustan* los pesos de cada conexión entre las neuronas hasta que entreguen un resultado dentro de un *umbral* propuesto. En este caso de dice que la red ha sido entrenada en forma supervisada.

Aunque la historia de las Redes Neuronales Artificiales data de los inicios de los 50's, se han estado usando con solidez desde principio de los 80's y en 1988 DARPA *Neural Network Study* [DARPA, 1988] emitió una lista de aplicaciones de Redes Neuronales, empezando en 1984 con un "ecualizador de canal adaptable", un reconocedor limitado de palabras, un monitor de procesos, un clasificador de señales de sonar y hasta un sistema de análisis de riesgo.

Se han estado diseñando Redes Neuronales para aplicarlas en campos tan vastos como: el reconocimiento de patrones, la identificación de parámetros de procesos, la visión por computadora y el control de sistemas. Actualmente, se siguen diseñando y entrenando redes para diversas aplicaciones, tales como: aerospaciales, automotrices, bancarias, de defensa, electrónicas, de entretenimiento, de finanzas, de seguros, de manufactura, médicas, petroleras, robóticas, de voz, de seguridad, de telecomunicaciones, de transportes y otras aplicaciones como el reconocimiento de caracteres [Hilera, 2000, DARPA, 1988].

Los métodos de *aprendizaje supervisado* son los que más se usan, aunque se pueden desarrollar redes que se entrenen sin ser supervisadas, es decir, *con aprendizaje no supervisado*, con el objeto de identificar grupos de datos. Incluso, redes que pueden ser *diseñadas directamente* como las redes de *Hopfield*. También existen los algoritmos de construcción de redes, que normalmente empiezan con un nodo y van agregando nodos y conexiones hasta que aprende del conjunto de aprendizaje, entre ellas se encuentran Tiling [Mezard *et al* 1989] y NetLines [Torres 1998]. Las redes neuronales pueden ser una herramienta muy útil para la industria, la educación, el entretenimiento y la investigación.

2.2 Fundamentos Teóricos

Una neurona cuenta como entrada un escalar simple, digamos p, a esta entrada se le aplica un peso que puede ser w, para dar como resultado un escalar simple dado por n, este resultado puede o no, ser modificado por un valor de tendencia (bias) identificado por b, al resultado n se le aplica una función especial, y da como resultado un escalar simple, representado por a. Este tipo de neuronas se pueden representar con las siguientes expresiones:

$$\begin{aligned} n &= wp,\ a = f(n), \\ n &= wp + b,\ a = f(n). \end{aligned} \tag{1}$$

La función f es llamada *función de transferencia* y puede ser una función tipo paso o sigmoidal. Tanto w como b son los parámetros escalares ajustables de la neurona. El parámetro de tendencia *bias* b, se puede considerar como el peso que se le aplica a una entrada ficticia que tiene un valor constante de uno [Hilera, 2000].

Algunos ejemplos de funciones de transferencia se muestran en la Figura 2.

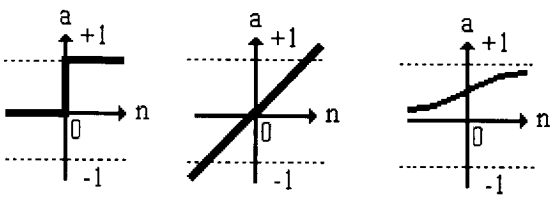


Figura 2. Función de transferencia, de izquierda a derecha, de Escalón, Lineal y Sigmoíde

Una neurona puede tener varias entradas, digamos r, estos valores son representados por el vector $P=\{p_1,..., p_r\}$, a las que se le asocian sus correspondientes pesos en el vector $W=\{w_1,...,w_r\}$. La expresión que representa a esta neurona, y a la que se le incluye bias, es:

$$n = w_1 p_1 + w_2 p_2 + ... + w_r p_r + b;\ a = f(n) \quad (2)$$

En forma matricial el cálculo de n se puede representar como:

$$n = WP + b. \quad (3)$$

Si se combinan más neuronas en el mismo nivel, digamos s neuronas que tendrán s salidas, éstas recibirán los r datos de entrada y, formando lo que se conoce como una *capa*. Los r valores son representados por el vector $P=\{p_1,...,p_r\}$. A la combinación de r entradas con s neuronas se le asocia los pesos correspondientes dados por la matriz:

$$\boldsymbol{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,r} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,r} \\ \cdots & \cdots & \cdots & \cdots \\ w_{s,1} & w_{s,2} & \cdots & w_{s,r} \end{bmatrix}$$

Entonces, se pueden crear redes neuronales que contengan varias capas, n_c, y cada una puede contener cualquier número de neuronas, ${}^{C_i}S$, donde las de la primera capa reciben los datos de entrada. Las salidas de esta primera capa pasan a ser las entradas de la segunda capa y, éstas a su vez, las entradas de la tercer capa, y así sucesivamente hasta que, la última capa de neuronas proporciona los resultados finales obtenidos. A esta última capa se le conoce como *capa de salida*. A todas las demás capas se les denomina *capas ocultas*. Cada neurona de una capa, está conectada con todas las neuronas de la capa siguiente y no están conectadas entre sí, esta arquitectura se conoce como *Alimentación Hacia Adelante*.

De manera formal, una red neuronal multicapa puede definirse como sigue.

Sea $P=\{p_1,...,p_r\}$ el conjunto de valores que forman el vector de los datos de entrada, ${}^{P,C_1}W = \{{}^{p,c_i}w_{i,j}\}$, $\forall\ i=1$, ..., ${}^{c_1}s$, y $j=1$, ..., ${}^{p}r$, el conjunto de escalares, que forman la matriz de los pesos, que conectan a los datos proporcionados como entrada, con la primer capa oculta de la red, y sea ${}^{C}W = \{{}^{k,k+1}w_{i,j}\}$, $\forall\ i=1$, ..., ${}^{k+1}s$, $j=1$, ..., ${}^{k}s$, $C=\{k,k+1\}$, $k=1$, ..., n_c-1, los conjuntos de escalares formando las matrices de los pesos conectando capas ocultas. Por último, sea $\boldsymbol{B} = \{B^i\}$, $\forall\ i=1,...,n_c$, el conjunto de escalares, que forma la matriz de los pesos, de los bias, $B^j = \{b_{i,j}\}$, $\forall\ i=1$, ..., ${}^{k}s$, $j=1$, ..., n_c, $k=1$, ..., n_c, entonces el conjunto de resultados escalares de cada capa, $A^c = \{a_1,...,a_{ks}\}$, se puede definir como:

$$A^1 = f^1\left({}^{P,c_1}WP + B^1\right) \quad (4)$$

que contiene los resultados obtenidos en la primer capa y

$$A^{i+1} = f^{i+1}\left({}^{i,i+1}WA^i + B^{i+1}\right), \forall i = 1,...,n_c - 1 \quad (5)$$

que contienen los resultados del resto de las capas, los resultados finales que se esperan, se encuentran en la última capa, la A^{nc}.

2.3 Aprendizaje

Antes de emplear una RNA, debe ser "entrenada". Al inicio, se recomienda asignar a los pesos valores entre ± 0.5, en forma aleatoria.

Para el aprendizaje, se cuenta con un conjunto de datos iniciales que se denominan *valores de aprendizaje*, estos son alimentados a la red, junto con los resultados esperados y un umbral para el error permitido. Cada vez que la diferencia, en valor absoluto, entre los valores esperados y los valores obtenidos sea mayor que el umbral permitido, se ajustan los pesos.

De esta manera se va entrenando a la RNA. La modificación a los pesos, se debe hacer de tal manera que, los resultados obtenidos se vayan acercado a los valores esperados, hasta alcanzar el umbral permitido. Los ciclos que se tienen que ejecutar durante la etapa de aprendizaje, se denominan *épocas* (epoch) y sirven para medir la eficiencia del aprendizaje.

Para ajustar los pesos de las redes se pueden usar varios paradigmas. El más conocido es el de propagación hacia atrás, del inglés *BackPropagation* [Hilera, 2000], que consiste en ajustar primero, los pesos que conectan a las dos últimas capas, considerando las diferencias de los valores esperados y los valores obtenidos, a continuación se va propagando hacia atrás a los demás pares de capas ocultas, mediante un *delta*, hasta llegar a las que conectan la capa de entrada con la primer capa oculta.

Es conveniente contar con una salida que "identifique" los elementos que no pertenecen al conjunto de caracteres válidos. Este aprendizaje, se inicia proponiendo elementos seleccionados entre los datos que no sean caracteres alfanuméricos y, cuando una figura es identificada como "falsa positiva", ésta puede ser alimentada como muestra "no valida", en un reaprendizaje posterior.

3 La Técnica Propuesta

La técnica propuesta consta de las mismas dos etapas básicas de cualquier técnica conocida, una de aprendizaje y

una de prueba o de generalización. Cada una de estas etapas comprende adicionalmente una serie de pasos. En esta sección de describe con detalle cada uno de dichos pasos.

3.1 Formación del Conjunto de Aprendizaje

Para que la red neuronal pueda reconocer el mayor tipo de caracteres, se le debe de "mostrar o enseñar" una muestra, lo suficientemente representativa, para que pueda "aprender" mediante el ejemplo. Esta es la técnica clásica de aprendizaje de redes usada por la mayoría de los investigadores en la temática del reconocimiento de patrones.

Para formar el conjunto de aprendizaje requerido, se puede seleccionar cualquier procesador de texto que genere caracteres de diferentes tipos, tamaños y diferentes inclinaciones. Uno de estos puede ser "Microsoft® Excel 97". Estos caracteres se pasan a un programa que los represente en forma gráfica, por ejemplo "Paint de Microsoft®", y posteriormente se separan, como se indica a continuación.

3.1.1 Selección de los Caracteres

"Microsoft® Excel 97" proporciona 64 fuentes de caracteres alfanuméricos, dentro de los que se encuentran algunos tipos que no pertenecen al castellano como "symbol", estos se pueden descartar y, de los restantes, se pueden seleccionar los más representativos.

Los tamaños que maneja "Microsoft® Excel 97" dependen del tipo de carácter, por ejemplo, el "Times New Roman" ofrece tamaños que van desde 8 hasta 72 puntos. Estos no son continuos, pero permite seleccionar los números faltantes (por ejemplo el número 15, para representar caracteres con 15 puntos).

"Excel" permite inclinar el texto (en nuestro caso cada carácter), en cualquier ángulo comprendido entre -90° y +90°, grado por grado. Se puede optar por utilizar los 180° y, con un giro de 180°, contemplar los cuatro cuadrantes o, considerar sólo 90° y con tres giros de 90° abarcar los 360°.

3.1.2 Selección de las fuentes

El siguiente paso es seleccionar las fuentes. En nuestro caso se seleccionaron las nueve siguientes: "Abadi MT Condensed", "Arial", "AvantGarde", "CG Times", "Courier New", "Times New Roman", "Univers Condensed", "Univers" y "Verdana".

En la Figura 3 se listan los nombres de las fuentes, los números del 1 al 0 con el tipo que les corresponden y las letras 'a' y 'A'. Se puede apreciar la gran diferencia que existe en el ancho de algunos tipos con respecto a otros y la forma de su representación.

Como se mencionó anteriormente, se contempla una cantidad prácticamente ilimitada de tamaños. Para este trabajo se seleccionaron 18 tamaños, que abarcan de los 8 puntos a los 25 puntos, pero si se necesita contemplar tamaños mayores, es muy sencillo incorporarlos.

"Excel" proporciona una fuente especial para tamaños pequeños, que precisamente se llama "Small Fonts", y contempla los siguientes tamaños en "puntos": 2.5, 3, 4, 5.5, 6 y 7.

Esta fuente no fue seleccionada, ya que la representación de los caracteres que son girados a cualesquiera ángulos es distorsionada completamente, lo que hace prácticamente imposible su identificación, aún desde la perspectiva humana. Se puede hacer la selección de este tipo, para diseñar una red neuronal que sólo contemple caracteres con una inclinación de 0° , 90°, 180° y 270°, ya que se pueden realizar giros de 90° sin distorsionarlos.

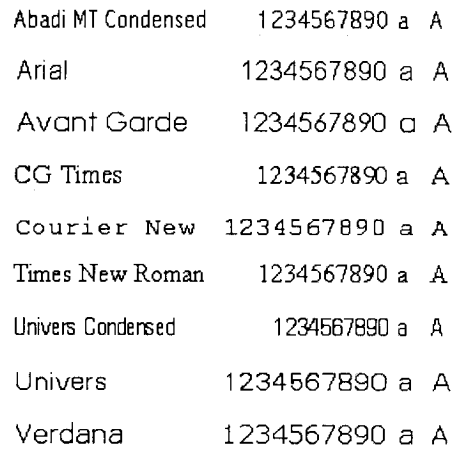


Figura 3. Las nueve fuentes de caracteres y los diez números en cada fuente, junto a las letras 'a' y 'A'

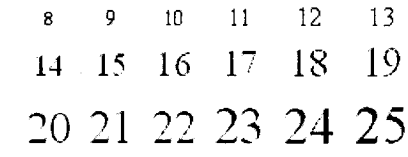


Figura 4. Los 18 tamaños que fueron seleccionados del 8 al 25 en puntos

3.1.2 Selección del Tamaño

3.1.3 Selección del Ángulo de Inclinación

Se optó por la selección de ángulos que abarquen 90°, con un incremento de 10°, estos son: -10°, 0°, 10°, 20°, 30°, 40°, 50°, 60° y 70°.

La Figura 4 muestra el tamaño, en puntos (indicado por el número en cada marco), y el aspecto que alcanzan los caracteres en la fuente Times New Roman.

La Figura 5 muestra los caracteres 'A', '1' y 'a' del tipo "Times New Roman" de 16 puntos, que fueron girados con los 9 ángulos contemplados. Un carácter girado a 80° es

equivalente a uno girado -10°, cuando se le aplica un giro de 90° en el sentido de las manecillas del reloj.

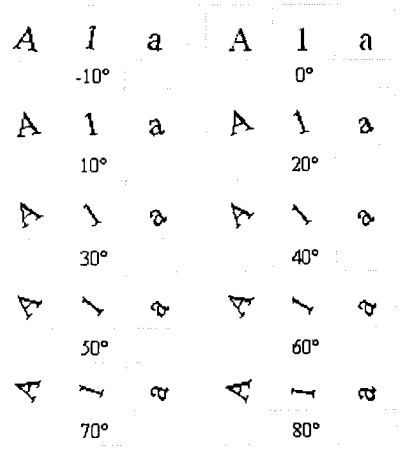


Figura 5. Muestra de caracteres girados de -10° a 80° con incrementos de 10°

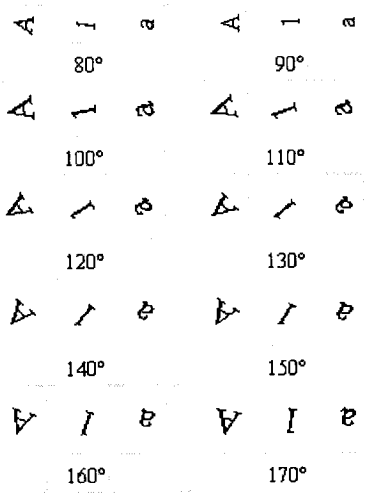


Figura 6. Los caracteres de la Figura 5, girados 90° en contra de las manecillas del reloj

Con un giro de 90°, en contra de las manecillas del reloj, se puede trasladar a los caracteres, al siguiente cuadrante, como es mostrado en la Figura 6. Con dos giros más, se pueden abarcar los cuatro cuadrantes, para detectar caracteres en cualquier ángulo de inclinación.

3.1.4 Formación de los Archivos en "Excel"

Una vez seleccionados los tipos de caracteres que se van a utilizar, junto con sus tamaños y ángulos de inclinación, se deben crear los archivos que los contemplen. Es conveniente agruparlos en archivos, relativamente pequeños, para que puedan ser manejados fácilmente en memoria. Por otro lado, esta organización será útil para el mantenimiento que será necesario, como se verá más adelante.

Primero se crean tres archivos, uno para las letras mayúsculas, otro para las letras minúsculas y un tercero para los números. En cada uno, se genera un renglón con cada uno de los caracteres correspondientes (10 para los números y 27 para las letras). Se copia cada renglón 17 veces, con lo que se tienen tres archivos con 18 renglones cada uno para cada tamaño.

En cada uno de los tres archivos, se selecciona cada renglón y al primero se le asigna un tamaño de 8 puntos, al segundo se le asigna un tamaño de 9 puntos, y así sucesivamente, hasta alcanzar el último renglón, al que se le asignan 25 puntos.

Hasta este momento, se tienen los tres archivos base para crear el resto. Estos archivos se pueden guardar como los caracteres "girados" a 0°. En este ejemplo se les asignaron los siguientes nombres: XRomanMa_0 para las letras mayúsculas, XRomanmi_0 para las letras minúsculas y XRomanNu_0 para los números, ya que el tipo base que se utilizó fue el "Times New Roman" y el prefijo "X" sirve para identificarlos como archivos de "Excel", independientemente de que "Excel" les agrega la extensión ".XLS".

En cada uno de estos archivos, se seleccionan todos los caracteres y la alineación se orienta a -10°, 10°, 20°, 30°, 40°, 50°, 60° y 70°. Cada giro se puede "guardar como" XRomanMa_-10, XRomanMa_10, ..., XRomanMa_70, para las letras mayúsculas, con sus equivalentes para las letras minúsculas y los números. Al terminar este paso, se tienen almacenados 9 archivos para cada una de las tres representaciones, es decir 27 archivos.

En cada uno de los tres archivos base, se seleccionan todos los caracteres y se cambian por el tipo siguiente, por ejemplo a "Abadi MT Condensed", este cambio se renombra y se puede "guardar como" XAbadiMa_0, ...mi_0 y ...Nu_0. Se giran como se indicó anteriormente y se guardan bajo la misma organización. Se siguen los mismos pasos para los demás tipos de caracteres.

En total se generan 9 x 3 x 9 = 243 archivos en "Excel" partiendo de nueve fuentes, 9 tamaños y 3 tipos, con los siguientes nombres como base:

XAbadiMa_-10 ... Ma_70
XAbadimi_-10 ... mi_70
XAbadiNu_-10 ... Nu_70
XArialMa_-10 ... Ma_70
XArialmi_-10 ... mi_70
XArialNu_-10 ... Nu_70
XAvaGaMa_-10 ... Ma_70
XAvaGami_-10 ... mi_70
XAvaGaNu_-10 ... Nu_70
XCGtimMa_-10 ... Ma_70
XCGtimmi_-10 ... mi_70
XCGtimNu_-10 ... Nu_70
XCouriMa_-10 ... Ma_70
XCourimi_-10 ... mi_70
XCouriNu_-10 ... Nu_70
XRomanMa_-10 ... Ma_70
XRomanmi_-10 ... mi_70
XRomanNu_-10 ... Nu_70
XUniveMa_-10 ... Ma_70

XUnivemi_-10 ... mi_70
XUniveNu_-10 ... Nu_70
XUnivcMa_-10 ... Ma_70
XUnivcmi_-10 ... mi_70
XUnivcNu_-10 ... Nu_70
XVerdaMa_-10 ... Ma_70
XVerdami_-10 ... mi_70
XVerdaNu_-10 ... Nu_70

3.1.5 Conversión de los Archivos en "Paint"

La siguiente etapa consiste en convertir, los archivos generados en la hoja de cálculo de "Excel", a un formato gráfico, que sea fácil de leer para poder convertirlo a formato binario.

Una forma de hacerlo, consiste en utilizar una de las herramientas gráficas que proporciona "Windows", por ejemplo el programa "Paint". En cada uno de los 243 archivos con extensión ".XLS" se seleccionan todos los caracteres y se "copian" de "Excel", para "pegarlos" en el "Mapa de bits" de "Paint".

"Paint" puede generar archivos BMP (Microsoft® Windows Bitmap), y estos archivos, para un mejor control, pueden conservar el nombre y, en este caso, se puede cambiar el prefijo "X", por el prefijo "P". Un ejemplo de nombre es el de PArialMa_30.BMP, para el tipo "Arial", girado a 30°. Al final, se tendrán 243 archivos en formato ".BMP" provenientes de "Paint".

3.1.6 Formación de las Máscaras Binarias de los Caracteres

El último paso, para la formación del conjunto de aprendizaje, es convertir los archivos gráficos de formato ".BMP" a formatos binarios (en este sistema se utiliza la extensión ".BYT" de "BYTE"), para que puedan ser alimentados a las Redes Neuronales, en forma de plantillas binarias, para este ejemplo se seleccionaron mallas cuadradas.

Un análisis realizado previamente, permitió determinar la medida adecuada para cada tamaño y grupo de carácter. Se seleccionó la malla cuadrada para que no afecte en las rotaciones de 90° que sean aplicadas a los caracteres.

Los tamaños de la malla para los caracteres de 8 a 25 puntos, son:

- Letras mayúsculas: 14, 14, 17, 17, 19, 19, 22, 22, 26, 26, 28, 28, 31, 31, 34, 34, 36 y 36.
- Letras minúsculas: 12, 12, 15, 15, 17, 17, 20, 20, 23, 23, 25, 25, 28, 28, 31, 31, 32 y 32.
- Números: 11, 11, 12, 12, 15, 15, 18, 18, 20, 20, 22, 22, 24, 24, 27, 27, 28 y 28.

Se puede desarrollar un programa, escrito en cualquier lenguaje de programación, para leer los archivos ".BMP" y convertirlos al binario correspondiente, organizándolos en caracteres. Para desarrollar este ejemplo se utilizó "MATLAB", basado en el siguiente algoritmo:

```
para i=1 a 3   % grupos
 para j=1 a 9   % tipos
  para k=0 a 8   % ángulos
   abrir archivo de i, j y k ".BMP" 'r'
   leer archivo ".BMP"
   abrir archivo de i, j y k ".BYT" 'w'
   para l=8 a 25   % tamaños
    para m=1 a 27 o 10   % caracteres
     localizar marco de Excel
     extraer el carácter contenido
     enmarcar el carácter en su malla
     grabar la malla en archivo ".BYT"
    fin de para m
   fin de para l
   cerrar archivo de i, j y k ".BMP" 'r'
   cerrar archivo de i, j y k ".BYT" 'w'
  fin de para k
 fin de para j
fin de para i
```

La Figura 7 muestra parte de un archivo de "Excel" correspondiente al fuente "Univers Condensed" que contiene los diez números con inclinación 30° y tamaños de 8 a 15 puntos, utilizado en el algoritmo descrito.

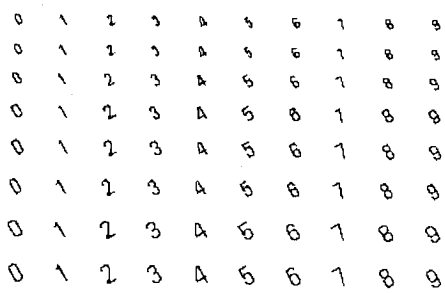


Figura 7. Marco de Excel que contiene los 10 números de "Univers Condensed" de 8 a 15 puntos con 30° de inclinación

Una vez separados los caracteres, cada uno forma una malla cuadrada que será alimentada a la RNA correspondiente. En la Figura 8 se puede apreciar la máscara de 28x28 pixeles del fuente "Univers Condensed", correspondiente a 25 puntos y con 30° de inclinación.

Este último paso deja disponibles los 243 archivos en formato ".BYT", donde el prefijo "P" se cambia por el prefijo "M" (de MATLAB), quedando nombres como el de MVerdami_-10, que corresponde a las letras minúsculas del tipo "Verdana" con rotación de -10°.

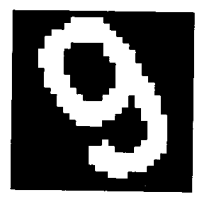


Figura 8. Máscara del número 9 de 25 puntos y con 30° de inclinación, que es alimentada a las RNA

3.1.6 Ruido

El ruido puede ser originalmente formado por caracteres especiales, y posteriormente ser complementado con los "falsos positivos". Se genera para todos los tamaños y se puede girar tres veces 90°. Estos caracteres no alfanuméricos, pasan por el mismo proceso que pasaron los caracteres alfanuméricos, y se le puede agregar más "ruido" en la etapa de conversión a un archivo tipo "Paint".

La Figura 9 muestra un ejemplo de algunos caracteres que fueron seleccionados como "ruido", que contiene solamente un conjunto de 8 tamaños que van de 8 a 15 puntos.

3.2 Estructura de las RNA

Si tomamos en cuenta que la persona promedio cuenta con 10^{11} neuronas en su cerebro y, si se considera que las computadoras son dotadas cada vez con mayor memoria, podemos entonces, diseñar RNA de cualquier tamaño, de acuerdo a la magnitud del problema.

Para el diseño de las RNA, de los tres tipos de caracteres (mayúsculas, minúsculas y números), se consideraron los siguientes aspectos:

- Cada Red contempla dos tamaños.

- Cada Red contempla tres ángulos de inclinación.

Así que, se formarán un total de 81 RNA provenientes de los 9 grupos de tamaños [(8,9),(10,11),...,(24,25)], de los 3 grupos de inclinaciones [(-10,0,10),(20,30,40) ,(50,60,70)] y de los 3 grupos de caracteres [mayúsculas, minúsculas y números], dando un total de 81 RNA.

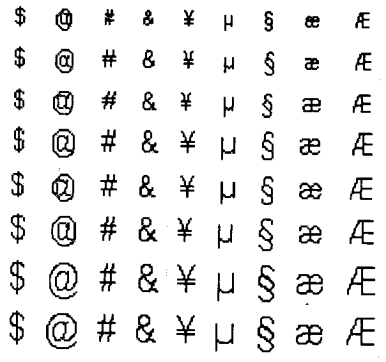


Figura 9. Ejemplo de caracteres no alfanuméricos que fueron seleccionados como "ruido"

El número de las neuronas en las capas de entrada de cada una de las RNA, sobre la base del tamaño del carácter, son las siguientes:

Para las letras mayúsculas: 196, 289, 361, 484, 676, 784, 961, 1156 y 1296.

Para las letras minúsculas: 144, 225, 289, 400, 529, 625, 784, 961 y 1024.

Para los números: 121, 144, 225, 324, 400, 484, 576, 729 y 784.

El número de las neuronas en las capas de salida de cada una de las RNA son las siguientes: para las letras mayúsculas 28, para las letras minúsculas 28 y para los números 11. Se contempló una neurona adicional, en cada una, para indicar la salida del "símbolo desconocido".

Para las capas ocultas, se eligieron tres niveles de capas con los siguientes números de neuronas: para las letras mayúsculas, primera capa 60, segunda capa 80 y tercer capa 70; para las letras minúsculas, primera capa 45, segunda capa 65 y tercer capa 55; y para los números, primera capa 30, segunda capa 50 y tercer capa 40.

A estos números se llegó después de una experimentación exhaustiva, donde se inició con una capa y unas pocas neuronas, observándose que las redes simplemente no convergían. Después se probó con dos capas, variando el número de neuronas, observándose de nuevo una muy lenta convergencia.

Los programas generan los nombres de los archivos correspondientes, bajo el siguiente formato:

$$\mathbf{P}g_i n_e\_n_{c1}\_n_{c2}\_n_{c3}\_n_s\mathbf{.byt}$$

donde:

P	prefijo de Peso.
g_i	grados de inclinación. 0 con -10°, 0° y 10°. 1 con 20°, 30° y 40°. 2 con 50°, 60° y 70°.
n_e	neuronas de entrada.
n_{c1}	neuronas de la capa oculta 1.
n_{c2}	neuronas de la capa oculta 2.
n_{c3}	neuronas de la capa oculta 3.
n_s	neuronas de salida.
.byt	extensión del archivo.

Basados en este formato, la cadena para generar el nombre del archivo, que contemple a las letras minúsculas, inclinadas unos 25°, y de un tamaño de 17 puntos, quedaría de la siguiente manera:

P1529_45_65_55_28.byt

Los programas se deberán diseñar para que generen los nombres de los archivos, con cualquier número de capas y de neuronas, con el objeto de poder seleccionar la RNA apropiada.

3.3 Aprendizaje de las RNA

Los valores esperados se definen como, '1' para el valor correcto de la neurona de salida correspondiente y, '0' para el resto de las neuronas, por lo que se puede formar una matriz diagonal del orden del total de los caracteres (incluyendo el del "símbolo desconocido"), así, cada renglón representa los valores esperados para cada carácter.

El programa fue diseñado para entrenar las RNA, en el número de etapas que sean necesarias, por lo que "salva" el archivo de los pesos cada vez que termina de entrenar, con los 9 tipos de caracteres, a cada red neuronal (una época).

La primera vez que se evoca un archivo de pesos, el programa detecta que no existe éste, y crea un archivo nuevo, con números aleatorios, comprendidos entre -0.5 y +0.5, como lo sugiere la bibliografía. A partir de la segunda vez, lee el archivo del disco.

La grabación, de los pesos que se van calculando, se realiza con suficiente frecuencia, evitando con ello la pérdida de una gran cantidad de proceso en caso de que se presente alguna contingencia, simulando un "check point".

El aprendizaje se puede hacer con todas las RNA, o se puede enfocar a un grupo en particular, e inclusive a una sola red neuronal.

El proceso aquí descrito, para el aprendizaje de todas las RNA contempladas, se basa en el siguiente algoritmo:

$Umbral_p$=0.49
while $Umbral_p$>$Umbral_g$
para i=1 a 3 % letras y números
valores esperados
para j=1 a 9 % ángulos de inclinación
para k=8 a 25 % tamaños de caracteres
para l=1 a 9 % tipos de caracteres
si l==1
lectura de pesos
fin de si l==1
lectura de caracteres
Pesos=aprendizaje($Umbral_p$)
fin de para l
graba los pesos
fin de para k
fin de para j
fin de para i
*$Umbral_p$*factor*
fin de while $Umbral_p$

$Umbral_g$, es el error máximo permitido para las RNA y $Umbral_p$ es un umbral intermedio que permite irlas "educando" poco a poco. Como error promedio se permite un máximo de $Umbral_p$, y como error máximo en cualquiera de sus salidas se permite hasta un valor de 2*$Umbral_p$. El $Umbral_p$ es útil para rescatar a la red cuando cae en un mínimo local, multiplicado por un *factor* menor a la unidad. Si no se hiciera esto, la RNA, simplemente, tardaría muchas más épocas en converger.

Se observó que las redes con menor número de neuronas en las capas ocultas, no convergen en un tiempo razonable, como en el caso de una red para letras mayúsculas con 9, 14 y 12 neuronas, que después de 500 épocas sus valores máximos de error se encuentran arriba del 90%, no obstante que su error promedio se sitúa en el 2%. Se recomienda pues construir redes más robustas con más de 15 neuronas por capa.

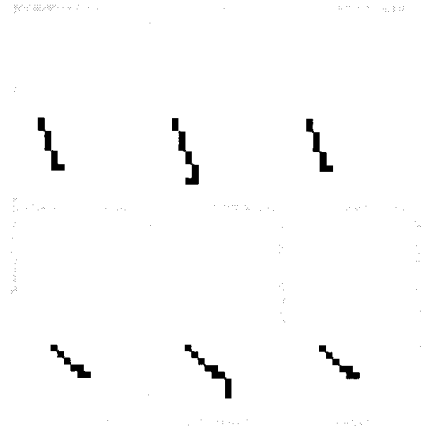


Figura 10. Caracteres i (izquierda), j (centro) y l (derecha), girados a 20° en la primer fila y a 50° en la segunda fila del tipo Arial de 8 puntos (magnificados)

Al girar los caracteres, "Excel" puede formar caracteres idénticos, provenientes de dos distintos. Como en el caso de las letras 'i' y 'l' del tipo Arial, que al girarlas 20° o 50°, las genera idénticas punto por punto, o puede transformar un carácter en otro, por ejemplo una 'c' en 'o'. En la Figura 10, se muestran las letras 'i', 'j' y 'l' del tipo Arial, con las dos inclinaciones que producen dos caracteres idénticos.

Cuando se alimentan a la red para su aprendizaje, al ser idénticos los caracteres, la red se "confunde" y queda oscilando alrededor de algún valor de error. En "Paint" se pueden "retocar" los caracteres, para que sean diferentes y más representativos del carácter al que pertenecen. En la figura 11 se muestra este cambio.

Figura 11. Los caracteres de la Figura 10 después de haber sido modificados en "Paint"

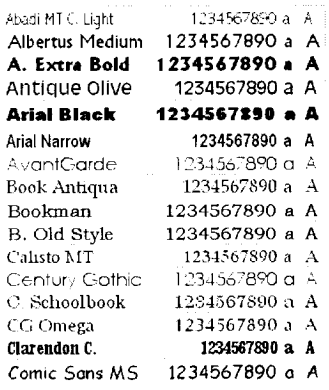


Figura 12. 16 tipos seleccionados para probar el desempeño de las RNA previamente entrenadas

3.4 Prueba de Generalización de las RNA

3.4.1 Selección del Conjunto de Muestras de Prueba

Para verificar el funcionamiento de las redes, después de haber sido entrenadas, se seleccionaron varios (diferentes a los usados en el aprendizaje) de los tipos con que cuenta "Excel", contemplando los tamaños de 8 a 25 puntos, propuestos al inicio de este capítulo, y ángulos de -10° a 79°, de grado en grado, para cubrir los 90°. Los 16 tipos seleccionados para la prueba, se muestran en la Figura 12.

La Figura 13 muestra un ejemplo de la orientación seudoaleatoria que se les dio a los caracteres de cada tipo, para poder contemplar los 90° entre todos los tamaños. Se presenta, como ejemplo, el tipo Abadi MT Condensed Light, con los caracteres correspondientes a los tamaños de 13 a 17 puntos, que muestran varias inclinaciones.

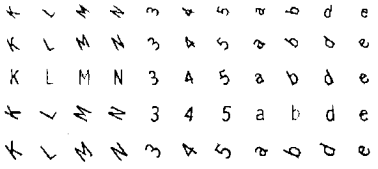


Figura 13. Ejemplo de inclinación de los caracteres del tipo Abadi MT Condensed Light en los tamaños de 13 a 17 puntos.

El total de caracteres para cada tipo de cada uno de los 3 conjuntos fue el siguiente: para los números fue de 180 (18x10), para las letras mayúsculas y minúsculas fue de 486 (18x27), dando un gran total de 18,432 (16x18x(10+27+27)) caracteres para prueba.

Se utilizan las tres redes que contemplan todas las inclinaciones. Los caracteres que presentan una inclinación mayor a 70° y que no son reconocidos por la red de 50° a 70°, son girados 90° para someterse a la red de -10° a 10°.

Como inicio el carácter se centra en el marco, pero como frecuentemente contiene píxeles que hacen que se desplace del centro, éste se mueve una posición hacia arriba, abajo, derecha e izquierda, para compensar este desplazamiento.

Si no se identifica al carácter como llega, se hace otro intento con su "esqueleto" obtenido mediante técnicas morfológicas. Si después de estos intentos no ha sido identificado, un último intento consiste en "dilatar" el "esqueleto" y someterlo a las redes. Si tampoco es identificado se marca como error.

El objetivo es identificar todos los caracteres, independientemente de que sean identificados falsos positivos, ya que, mediante otras técnicas, pueden ser eliminados estos últimos dejando a los caracteres verdaderos.

4 Resultados

El desempeño las RNA entrenadas por medio de la técnica descrita en la sección 3 es verificado en esta sección en varios escenarios. En un primer escenario, el conjunto de RNA es verificado con el conjunto de muestras diseñado en la sección 3.4.1. En un segundo escenario, el mismo conjunto de RNA es verificado con varias muestras extraídas directamente de varias imágenes de mapas ráster escaneados en color.

4.1 Resultados con el Conjunto de Muestras de Prueba

En este primer experimento, al numerar los tipos utilizados del 1 al 16 podemos ver, en la Tabla 1, los resultados para cada uno de ellos. De los 18,432 caracteres analizados, se tiene un total de 23 caracteres no reconocidos, lo que arroja un porcentaje de error total de solamente un 0.125%, y por lo tanto un porcentaje de aciertos del 99.875%. Este porcentaje es bastante alto tomando en cuenta que las muestras usadas son completamente diferentes a las usadas para el aprendizaje.

	Números		Minúsculas		Mayúsculas	
No.	Fallos	%	Fallos	%	Fallos	%
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	1	0.56	1	0.21	1	0.21
4	0	0	3	0.62	0	0
5	0	0	2	0.41	0	0
6	0	0	0	0	0	0
7	0	0	0	0	0	0
8	0	0	0	0	0	0
9	1	0.56	1	0.21	0	0
10	1	0.56	1	0.21	0	0
11	0	0	0	0	1	0.21
12	0	0	0	0	0	0
13	0	0	0	0	1	0.21
14	0	0	1	0.21	0	0
15	0	0	1	0.21	6	1.23
16	0	0	0	0	1	0.21
	3		3		10	

Tabla 1. Resultados obtenidos al aplicar las RNA al conjunto de caracteres de prueba

Ejemplos de caracteres que no pudo reconocer el sistema, se muestran en la Figura 14. De izquierda a derecha y de arriba hacia abajo las letras 'U' de 8 puntos y 11°, 'U' de 16 puntos y 73°, 'H' de 9 puntos y 61°, y 'o' de 12 puntos y 28° del tipo *Clarendon Condensed*; 'o' de 10 puntos y -10° del tipo *Albertus Extra Bold*, y la 's' de 10 puntos y -6° de *Arial Narrow*.

4.2 Resultados con Muestras de Mapas Ráster

Para verificar realmente la utilidad de las RNA entrenadas, se usaron 1025 muestras de caracteres numéricos, minúsculas y mayúsculas extraídas directamente de imágenes de mapas ráster.

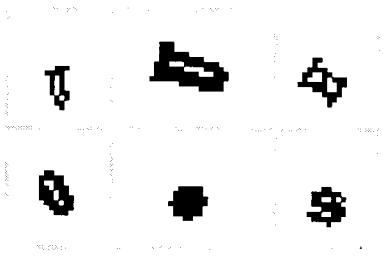


Figura 14. Caracteres que no pudo reconocer el sistema de Redes Neuronales. 'U', 'U', 'H', 'o', 'o' y 's'

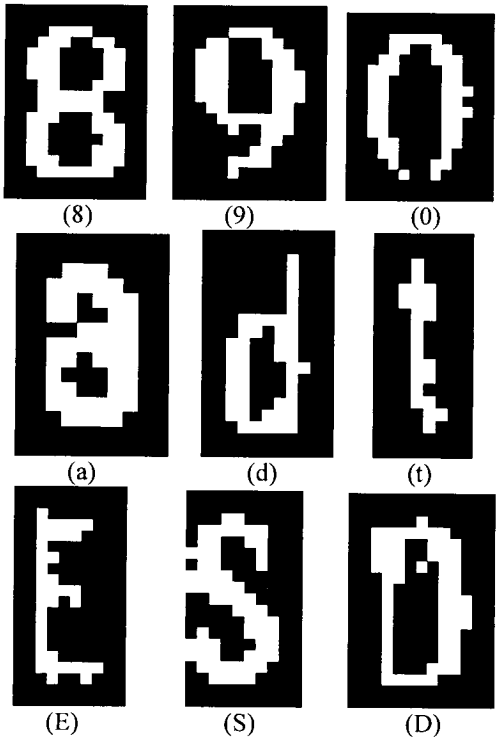


Figura 15. Muestras extraídas de un mapa para probar el desempeño de las RNA.

En la Figura 15 se muestra algunos de los caracteres extraídos de un mapa. Éstos fueron obtenidas por medio de la técnica de aislamiento descrita en [Velázquez, 2001 y Velázquez, 2002]

El porcentaje de reconocimiento del sistema fue en este caso del 95.512%. Se trata de un porcentaje bastante alto a pesar de que las RNA no fueron entrenadas con muestras reales. Seguramente, el desempeño de dichas RNA mejoraría si se usaran también muestras de caracteres extraídas de los mapas ráster para el aprendizaje.

Ejemplos de caracteres que no pudieron ser identificados por el sistema se muestran en la Figura 16.

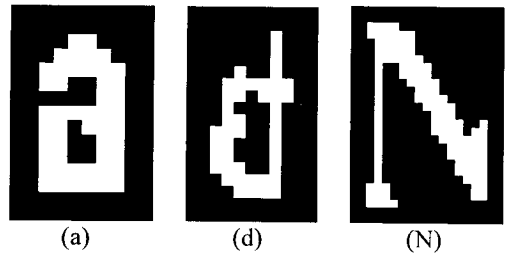


Figura 16. Muestras extraídas de un mapa que no pudieron ser reconocidas por las RNA

5 Conclusiones y Trabajo Futuro

A través de los experimentos realizados se puede apreciar un buen rendimiento del conjunto de RNA entrenado, que puede ser utilizado para identificar casi cualquier tipo de caracteres.

[Frischknecht *et al* 1997] sólo logra porcentajes de 88.5% y 96.4% y [Deseilligny *et al* 1997] de 95% mientras que con el conjunto propuesto de RNA se consigue un 99.875%. [Wenyin *et al* 1997] sólo permite texto horizontal o vertical, siendo que los planos temáticos contemplan todo tipo de inclinaciones, que si pueden ser reconocidos por nuestras redes. Por lo anterior se recomienda el uso de la técnica propuesta para el entrenamiento de RNA para el reconocimiento de caracteres alfanuméricos.

Tipos como "Courier" o "Lucida Handwriting", no son fácilmente identificados por el conjunto de redes entrenado según la técnica propuesta, pero estos no son utilizados en los mapas cartográficos. Otros conjuntos de RNA pueden ser entrenados para reconocer éstos y otros estilos, que son utilizados en textos normales. Se observaron también algunos problemas con muestras provenientes de los mapas ráster, los cuales pueden ser superados con un aprendizaje de las RNA con muestras no solo sintéticas sino también con las obtenidas de estos mapas.

Actualmente, se está trabajando en el diseño de un sistema experto que permita seleccionar el grupo de RNA más apropiada para cada candidato a carácter alfanumérico, en función de su tamaño y posible orientación.

El mismo sistema deberá ser capaz de eliminar todos los candidatos negativos, y determinar el tamaño y la inclinación de cada carácter para poder formar palabras o números completos. La metodología completa será aplicada para separar eficientemente la capa de caracteres alfanuméricos de un mapa cartográfico ráster escaneado en color.

Agradecimientos

Los autores desean expresar su agradecimiento al CIC-IPN, el IMP y al CONACYT por el apoyo económico brindado para el desarrollo de este trabajo.

Referencias

N. Ansari and K. Li. Landmark-based shape recognition by a modified Hopfield neural network. Pattern Recognition, 26(4):531-542, 1993.

N. Ansari and X. Liu. Recognizing partially occluded objects by a bidirectional associative neural network. Optical Engineering, 32(7):1539-1548, 1993.

A. K. Chhabra. Graphic Symbol Recognition: an Overview, IAPR Lecture Notes in Computer Science 1389. Graphics Recognition Algorithms and Systems, pp. 68-79, 1997.

DARPA. 1988. "Neural Network Study", MA: M. I. T. Lincoln Laboratory.

J. R. Hilera y V. J. Martínez. Redes Neuronales Artificiales. Alfaomega ra-ma. pp. 131-180, 2000.

P. Deseilligny, R. Mariani, J. Labiche and R. Mullot, Topographic Maps Automatic Interpretation: Some Proposed Strategies, IAPR Lecture Notes in Computer Science 1389 Graphics Recognition Algorithms and Systems, pp. 175-193, 1997.

S. Frischknecht and E. Kanani. Automatic Interpretation of Scanned Topographic Maps: a Raster-Based Approach, IAPR Lecture Notes in Computer Science 1389 Graphics Recognition Algorithms and Systems, pp. 207-220. 1997.

J. N. Hwang and H. Li. A traslation / rotation / scalling / occlusion invariant neural network for 2D/3D object classification. In Proceedings of the IEEE Int. Conf. on Accoustics, Spech and Signal Proc., vol. 2, pages 397-400, 1992.

J. H. Kim, S. H. Yoon, Y. H. Kim, E. H. PArk, and C. Nituen. An efficient matching algorithm by a hybrid Hopfield network for object recognition. In Proceedings of the Int. Symposium in Circuits and Systems, Vol. 6, pages 2888-2892, 1992.

S. P. Levachkine and E. A. Polchkov. Integrated Technique forAutomated Digitization of Raster Maps, Revista Digital Universitaria, Vol. 1, No. 1, 30 de junio 2000, http://www.revista.unam.mx/.

H. Luo and K. Rangachar. Improved Directional Morphological Operations for Separatión of Characters from Maps/Graphics, IAPR Lecture Notes in Computer Science 1389 Graphics Recognition Algorithms and Systems, pp. 35-47, 1997.

D. A. Mitzias and B. G. Mertzios. Shape recognition with a neural classifier based on a fast polygon approximation technique. Pattern Recognition, 27(5):627-636, 1994.

G. K. Myers, G. Mulgaonkar, C. H. Chen, J. L. DeCurtins and E. Chen. Verification-Based Approach for Automated Text and Feature Extraction from Raster-Scanned Maps, IAPR Lectures Notes in Computer Science Graphics Recognition, pp. 190-203, 1995.

N. R. Pal, P. Pal, and A. K. Basu. A new shape representation scheme and its application to shape discrimination using a neural network. Pattern Recognition, 26(4):543-551, 1993.

S. P. Raman and U. B. Desai. 2-D object recognition using Fourier Mellin transform and a MLP network. In Proceedings of the IEEE Int. Conf. on Neural Networks, pages 2154-2156, 1995.

] **D. M. Tsai and R. Y. Tsai**. Use neural networks to determine matching order for recognizing overlaping objects. Pattern Recognition Letters, 17:1077-1088, 1996.

A. Velázquez, S. Levachkine and V. Alexandrov. Color Image Segmentation using False Colors and its Applications to Geo-Images Treatment: Alphanumeric Character Recognition. International Geoscience and Remote Sensing Symposium, 2001.

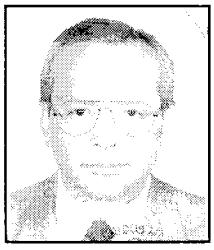
S. S. Wang and W. G. Lin. A new self- organizing neural model for invariant pattern recognition. Pattern Recognition, 29(4):677-687, 1996.

L. Wenyin and D. Dov. Genericity in Graphics Recognition Algorithms, IAPR Lecture Notes in Computer Science 1389 Graphics Recognition Algorithms and Systems, pp. 9-20. 1997.

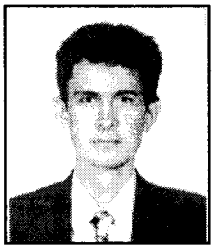
M. Mezard and J. P. Nadal. Learning in feedforward layered networks: the tiling algorithm. Journal of Physics A, 22:2191-2203. 1989

J. M. Torres M., M. B. Gordon. Efficient Adaptive Learning for Classification Tasks with Binary Units. Neural Computation, 10(4):1007-1030, 1998.

A. Velázquez. Localización, Recuperación e Identificación de la Capa de Caracteres, contenida en los Planos Cartográficos. Tesis doctoral. Centro de Investigación en Computación del IPN.



Aurelio Velázquez, *(1948) recibió el grado de Maestro en Ciencias con Especialidad en Computación Electrónica del Centro Nacional de Cálculo del IPN (CENAC-IPN), en 1980 y el grado de Doctor en Ciencias de la Computación del Centro de Investigación en Computación del IPN (CIC-IPN) en 2001. Es investigador del Instituto Mexicano del Petróleo. Sus áreas de interés incluyen el tratamiento de imágenes de percepción remota, las redes neuronales y la digitalización de documentos en particular mapas.*



Juan Humberto Sossa Azuela, *(1956) recibió el grado de Maestro en Ciencias en Ingeniería Eléctrica del Centro de Investigación y Estudios Avanzados del IPN de México (CINVESTAV-IPN), en 1987 y el grado de Doctor en Informática del Instituto Nacional Politécnico de Grenoble, Francia en 1992. Es profesor Titular en el Centro de Investigación en Computación del IPN. Sus áreas de interés incluyen la visión por computadora, el procesamiento de imágenes y el control de sistemas electromecánicos usando información visual.*

Serguei Levachkine, *(1962) recibió el grado de Maestro en Ciencias en 1983 y el Grado de Doctor en 1986 en Física y Matemáticas de la Universidad Estatal de Moscú (M.V. Lomonosov), Rusia. Es profesor investigador en el Centro de Investigación en Computación del IPN, México. Sus intereses de investigación son la visión por computadora, la digitalización de documentos en particular mapas y el reconocimiento de patrones y optimización en sistemas usando información visual.*