Reconocimiento de Números Manuscritos

José Luis Garbi, Paula Mercado
Facultad de Informática. Universidad Nacional de La Plata
La Plata, Argentina, 1900
{jlgarbi, mercado.paula}@gmail.com

Laura Lanzarini, Claudia Russo
III-LIDI (Instituto de Investigación en Informática LIDI)
Facultad de Informática. Universidad Nacional de La Plata
La Plata, Argentina, 1900
{laural, crusso}@lidi.info.unlp.edu.ar

Abstract

At present, handwritten text recognition still represents a wide source of research.

This paper presents a software tool which belongs to the area of ICR (Intelligent Character Recognition) for the recognition of handwritten integers. A classifier based on feedforward neural networks and a set of techniques belonging to digital image processing area are incorporated to this tool, which make the suitable adaptations over the input image. In this way, a handwritten integer made up by several digits is entered and, as a result, the recognition of each of its elements is obtained.

The results of applying this tool over a UCI repository number base have been successful. It is important to notice that, even though the results presented in this paper exclusively refer to handwritten number recognition, this tool can be applied to the complete set of characters.

Finally, some conclusions are presented together with some future lines of work.

Keywords: Handwritten Character Recognition, Preprocessing, Image Segmentation, Neural Networks

Resumen

En la actualidad, el reconocimiento de texto manuscrito sigue siendo una fuente de intensa investigación.

Este paper presenta una herramienta de software perteneciente al área de Reconocimiento Inteligente de Caracteres (ICR – Intelligent Character Recognition) para el reconocimiento de números enteros manuscritos. En ella se integra un clasificador basado en redes neuronales feedforward y un conjunto de técnicas pertenecientes al área de procesamiento de imágenes digitales que realiza las adaptaciones adecuadas sobre la imagen de entrada. De esta forma, se ingresa un número entero manuscrito formado por varios dígitos y se obtiene como resultado el reconocimiento de cada uno de los elementos que lo componen.

Los resultados de la aplicación de esta herramienta sobre una base de números del repositorio UCI han sido satisfactorios. Es importante destacar que, si bien los resultados expuestos en este artículo se refieren exclusivamente al reconocimiento de números manuscritos, esta herramienta puede ser aplicada al conjunto de caracteres completo.

Finalmente se incluyen algunas conclusiones así como algunas líneas de trabajo futuras.

Palabras claves: Reconocimiento de caracteres manuscritos, Preprocesamiento, Segmentación de imágenes, Redes Neuronales

1. Introducción

En la actualidad, el reconocimiento preciso en textos escritos a máquina se considera un problema resuelto. Sin embargo, no ocurre lo mismo con el reconocimiento de la impresión manual, es decir, aquella que proviene de la caligrafía humana, la cual sigue siendo una fuente de intensa investigación.

Cuando la información es ingresada on-line, es posible aprovechar la velocidad y la dirección de los segmentos trazados como información de entrada [12]. Los asistentes digitales personales o PDA (personal digital assistant) han obtenido excelentes resultados en el reconocimiento de caracteres escritos a mano alzada ya que su software aprovecha este tipo de información. Además, el usuario se puede entrenar y ayudar al dispositivo usando solamente formas específicas de letras. Estos mismos métodos no se pueden trasladar a los programas que se encargan de interpretar los caracteres de documentos escaneados.

Un problema aun más complejo de resolver es el reconocimiento de textos cursivos, es decir aquellos en el que todas las letras se encuentran conectadas formando una palabra [11][2]. En estos casos, la complejidad del problema puede reducirse recurriendo a información adicional ya sea gramatical o contextual. Por ejemplo, el reconocimiento de un conjunto de palabras previamente clasificadas es más fácil de resolver que tratar de analizar, de manera individual, los caracteres de la escritura; independientemente del tamaño del conjunto.

Este artículo presenta una herramienta de software para el reconocimiento de números enteros manuscritos perteneciente al área de Reconocimiento Inteligente de Caracteres o ICR (Intelligent Carácter Recognition). Dicha herramienta integra un clasificador basado en redes neuronales feedforward y un conjunto de técnicas pertenecientes al área de procesamiento de imágenes digitales que realiza las adaptaciones adecuadas sobre la imagen de entrada.

La redes neuronales feedforward, entrenadas mediante una técnica de gradiente, han demostrado ser excelentes clasificadores con una gran tolerancia al ruido naturalmente presente en la entrada de datos. Esto último es una característica importante a la hora de procesar imágenes digitales. Sin embargo, cuando el patrón a reconocer es directamente una imagen, la respuesta de la red es muy sensible al tamaño y posicionamiento del sector a reconocer. Por tal motivo se incluyó en esta herramienta un sistema de preprocesamiento y segmentación de los distintos elementos de la imagen que, junto con un escalado adecuado, ayudarán al clasificador a brindar una respuesta correcta.

Este artículo está organizado de la siguiente forma: en la sección 2 se da una breve descripción del sistema propuesto en este artículo y en las secciones 3,4 y 5 se detallan cada uno de sus módulos y se incluyen los resultados obtenidos. Finalmente la sección 6 contiene las conclusiones así como algunas líneas de trabajo futuras.

2. Descripción de la herramienta de software propuesta

La herramienta propuesta en este artículo trabaja sobre imágenes binarias de números enteros manuscritos. El primer paso es reconocer, dentro de la imagen, los componentes correspondientes a dígitos. Esto se lleva a cabo tomando los componentes conexos de la misma. Dado que habitualmente la imagen de entrada presenta cierto ruido, ya sea porque se encuentra presente en la señal original o porque se produce durante el paso de adquisición de la misma, se utiliza un módulo de procesamiento digital de imágenes para normalizar el dígito y así prepararlo para que sea tomado como entrada del clasificador basado en redes neuronales. Si el clasificador puede determinar de qué dígito se trata, se da por terminada la tarea para ese posible dígito y se continúa con el siguiente. Si el reconocimiento falla, se asume que se trata de dígitos que se encuentran

superpuestos de algún modo y se intenta separarlos mediante un algoritmo de segmentación. Luego de segmentar un dígito, nuevamente se envía al clasificador, previa normalización. La segmentación continúa hasta reconocer un dígito apropiadamente o hasta que se han intentado todas las posibles segmentaciones; en cuyo caso se determina ese posible dígito como no reconocido. La figura 1 representa el algoritmo descripto.

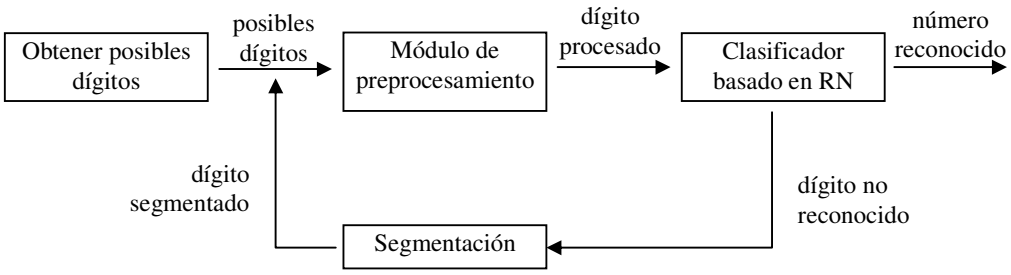


Figura 1: Esquema general de la herramienta de software propuesta

A continuación se describe con más detalle cada uno de los módulos antes mencionados.

3. Módulo de Segmentación

Si bien existen diversos mecanismos de segmentación para dígitos [3][2][11], se seleccionó el método desarrollado por Punnoose denominado EDF (Extended Drop Fall) [9] ya que ha sido utilizado en aplicaciones similares con muy buenos resultados [8].

EDF traza un camino vertical dentro de la imagen desde un punto de partida (x_0,y_0), ubicado en la primera fila, hasta el punto final (x_n,y_n) ubicado en la última fila. Este camino segmenta la imagen en dos porciones; la que se encuentra a su derecha y la que se encuentra a su izquierda.
Existen un conjunto de reglas que determinan el camino trazado. Las reglas son evaluadas en orden. Si una regla falla, se intenta con la siguiente, así hasta la última. Si el camino se encuentra en la posición (x_i, y_i), se evalúan las siguientes reglas en este orden:

REGLA 1:	Si $(x_i + 1, y_i) = 0 \rightarrow$ moverse a $(x_i + 1, y_i)$. Si abajo es negro (el color del fondo), moverse hacia abajo.
REGLA 2:	Si $(x_i + 1, y_i + 1) = 0 \rightarrow$ moverse a $(x_i + 1, y_i + 1)$. Si abajo a la derecha es negro, moverse hacia abajo a la derecha.
REGLA 3:	Si $(x_i + 1, y_i - 1) = 0 \rightarrow$ moverse a $(x_i + 1, y_i - 1)$. Si abajo a la izquierda es negro, moverse hacia abajo a la izquierda.
REGLA 4:	Si $(x_i, y_i + 1) = 0 \rightarrow$ moverse a $(x_i, y_i + 1)$. Si a la derecha es negro, moverse hacia la derecha.
REGLA 5:	Si $(x_i, y_i - 1) = 0$ ^ dirección = derecha$\rightarrow$ moverse a $(x_i, y_i - 1)$. Si a la izquierda es negro y se viene desde la derecha, moverse hacia la izquierda.
REGLA 6:	Moverse a $(x_i + 1, y_i)$. Si fallan todas las anteriores, mover hacia abajo.

En la REGLA 6 es donde se produce el corte, pues se intentaron todos los caminos previos y no hubo posibilidad de seguir por el color de fondo. El algoritmo EDF, intenta moverse por el color de fondo lo más que puede. Una vez que ingresa a píxeles pertenecientes al color del dígito, aplica el mismo conjunto de reglas tratando de mantenerse dentro de este color, hasta que no tiene otra

alternativa que volver al color de fondo o hasta que llegó a la última fila; en donde se termina la segmentación. La figura 2 muestra el resultado de aplicar EDF entre dos dígitos cero unidos, situación que se encuentra comúnmente en los números manuscritos.

Original con trazo de segmentación | Porción izquierda | Porción derecha

Figura 2: Segmentación EDF implementada

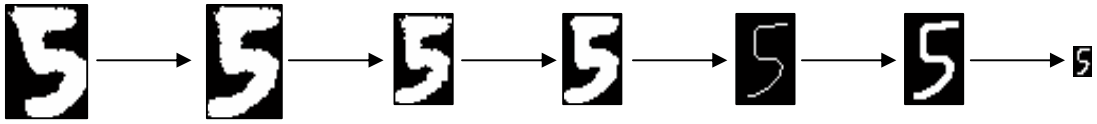
4. Módulo de preprocesamiento

El objetivo del módulo de preprocesamiento es adaptar o normalizar apropiadamente los dígitos del número entero para que sean entradas adecuadas de las redes neuronales del clasificador.

La escritura manuscrita presenta varios desafíos a la hora de ser considerada como entrada en un sistema de reconocimiento. En sistemas de reconocimiento óptico de caracteres (OCR), cada caracter puede variar en tamaño y fuente. En la escritura manuscrita, además de estas complicaciones, se encuentran las relacionadas al tipo de lápiz, lapicera o marcador utilizados, pues cada uno produce un grosor y una terminación diferente que influye en proceso de escaneo. Además se deben considerar los problemas de la inclinación de la letra manuscrita y, como ya se ha dicho antes, la unión de números que ocurre comúnmente durante la escritura; que deben ser segmentados.

Adicionalmente, el reconocedor debe tener en cuenta las diferentes formas de escribir un mismo número. Por ejemplo, existen diferentes formas de escribir los dígitos 1, 4 o 7. Estas diferencias son comunes dentro de una misma población y se acentúan más entre poblaciones con culturas diferentes. De todas formas, esta consideración forma parte del entrenamiento del sistema con un conjunto de dígitos apropiados a la población en consideración.

El preprocesamiento a realizar consiste de los siguientes pasos: a) Corrección de la inclinación, b) Redimensionamiento, c) Filtrado, d) Adelgazamiento, e) Ensanchamiento, f) Reducción del tamaño



Original a) b) c) d) e) f)

Figura 3: Pasos realizados por el módulo de preprocesamiento

A continuación se explicará cada uno de estos pasos.

4.1. Corrección de la inclinación

Comúnmente, cualquier texto escrito a mano presenta cierta inclinación. La inclinación de un dígito influye en su reconocimiento. Para solucionar este problema, debe calcularse la inclinación de la imagen y corregirla en la dirección apropiada.

Para determinar el ángulo de inclinación θ se utiliza la siguiente fórmula, derivada a partir del cálculo de los momentos generales invariantes [7].

$$\theta = \frac{\tan^{-1}\left(\frac{2\,\mu_{11}}{\mu_{20}-\mu_{02}}\right)}{2}$$

donde se define

$\mu_{11} = m_{11} - m_{10}m_{01}/m_{00}; \mu_{20} = m_{20} - m_{10}^2/m_{00}; \mu_{02} = m_{02} - m_{01}^2/m_{00}$

siendo $m_{pq} = \sum_{x=0}^{N-1}\sum_{y=0}^{M-1} x^p y^q I(x,y); p,q = 0,1,...\infty;$ el momento discreto p q de la imagen binaria discreta I, de tamaño NxM.

Una vez determinado el ángulo θ es necesario corregir la inclinación para lo cual se consideraron dos alternativas: rotar la imagen según el ángulo calculado o realizar un corrimiento horizontal de los píxels utilizando el método descripto en [5]. La segunda alternativa mantiene intactos los píxeles de la fila inferior y luego cada píxel de las filas superiores se desplaza un determinado número de posiciones según su altura logrando un dígito no inclinado. Para determinar la nueva posición de un píxel se utiliza la siguiente fórmula.

$\forall I(x,y) = 1,\ I'(x',y') = 1, siendo\ x' = x - y\tan(\theta),\ y' = y,$ siendo I la imagen original e I´ la imagen resultado.

En definitiva, lo que se logra es un corrimiento horizontal de filas. A medida que se incrementa la altura de una fila el desplazamiento será mayor. Los resultados de ambos métodos de corrección de la inclinación se pueden observar en la figura 4.

Original | Corrección con rotación | Corrección implementada

Figura 4: Mecanismos de corrección de la inclinación analizados para el desarrollo de la herramienta

Para la implementación de la herramienta propuesta en este artículo se seleccionó la segunda opción, pues genera una imagen de mejor calidad.

4.2. Redimensionamiento

El cambio de tamaño de la imagen original independiza a las redes neuronales del tamaño de escritura y de la resolución del escáner. Además, los algoritmos posteriores de procesamiento digital tienen un mejor rendimiento sobre imágenes más pequeñas.

La estandarización de la relación de aspecto es un paso previo necesario para la modificación de tamaño que sufrirá la imagen. El tamaño final de la imagen preparada para ingresarse a la red neuronal es de 13x9 y su relación de aspecto es 1.44 (13/9 = 1.44). El ancho de la imagen es aproximadamente el 69% de su altura. Se seleccionaron estos valores teniendo en cuenta la relación de aspecto media del conjunto de dígitos de la base de datos con la que trabajamos. Pero antes de llegar a este tamaño, la imagen original se modifica a una de 39x27 píxeles. La relación de aspecto se mantiene y el tamaño es adecuado para el procesamiento restante, aunque es demasiado grande para ser la entrada de la red neuronal.

Para la modificación de tamaño se requiere, en primer lugar, centrar el dígito teniendo en cuenta la relación de aspecto 1.44, en caso de ser necesario. Luego se lleva la imagen al tamaño de 39x27 píxeles. Existen diferentes algoritmos para modificar el tamaño de una imagen. Entre los más conocidos podemos mencionar a la interpolación del vecino más próximo, lineal, bilineal y bicúbica, entre otras [5]. Como se están utilizando imágenes binarias, se seleccionó la interpolación del vecino más próximo ya que sus resultados no generan nuevos valores de píxeles.

4.3. Filtrado

La imagen filtrada se diferencia de la original en que se ha logrado detallar y resaltar sus píxeles en los lugares donde más lo necesita. Muchas veces, al escanear un dígito se producen imperfecciones debido al ruido y características del trazo con el que se hizo el número. Estas imperfecciones merman la capacidad de detección de las redes neuronales, lo que hace imperioso intentar reducirlas. Con este objetivo en mente, se filtró la imagen de 39x27 píxeles con el filtro de la mediana de tamaño 3x3 [5].

4.4. Adelgazamiento

El adelgazamiento es un paso fundamental en las etapas del procesamiento digital. Esta fase independiza a la red neuronal del grosor con el que fue escrito el número. Su objetivo es encontrar una imagen binaria del grosor de un píxel que respete de la mejor manera posible las características morfológicas del dígito original. La misma debe estar contenida en la imagen y conservar la conectividad.

Existen diferentes algoritmos para obtener este resultado. Una manera es obtener el esqueleto de la imagen que está formado por aquellos píxeles que se encuentran equidistantes al menos de dos píxeles del borde. Otra forma de obtener una imagen del grosor de un píxel es utilizando un algoritmo llamado adelgazamiento. Desde el punto de vista de la herramienta propuesta, este algoritmo, que consiste en una eliminación progresiva de los píxeles del borde hasta llegar al grosor deseado, aporta mejores resultados que la obtención del esqueleto. Esto puede observarse en la figura 5.

De todas formas, el algoritmo utilizado para encontrar la representación de grosor mínimo del dígito es el propuesto en [6]. Este algoritmo no siempre produce exactamente un resultado de grosor uno en todos los tramos del dígito, pero es más efectivo en la preservación de la morfología del mismo.

Figura 5: Diferentes resultados de adelgazamiento

4.5. Ensanchamiento

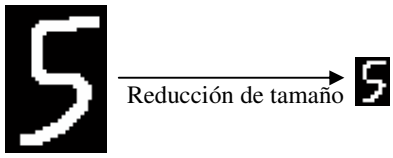
El siguiente paso del procesamiento de la imagen es ensanchar el resultado obtenido en la etapa anterior. Esto logra un grosor uniforme para todos los dígitos que se vayan a reconocer. Además, con este procedimiento se evita perder definición cuando la imagen sea finalmente redimensionada a su tamaño definitivo de 13x9 píxeles; es decir, la tercera parte del tamaño con el que se viene

trabajando. El algoritmo de ensanchamiento es realmente simple. Por cada píxel de la imagen original que pertenece al dígito, sus 8 vecinos son puestos en 1, es decir, que aproximadamente se triplica el grosor de cada línea que conforma al dígito.

4.6. Reducción del tamaño

Como ya se ha dicho anteriormente, el tamaño final de la imagen es de 13x9 píxeles. Este tamaño es adecuado para las redes neuronales y aún preserva mayormente las características del dígito original gracias al procesamiento realizado.

Para llevar la imagen de 39x27 píxeles a una de 13x9, se subdivide la imagen original en regiones de 3x3 píxeles, dando lugar a una matriz de 13x9 regiones. Cada región corresponde a un píxel de la imagen final. El píxel de la imagen reducida será 1 si más de la mitad de los píxeles de la región original correspondiente es 1. En la figura 6 se puede observar el resultado del algoritmo de reducción de tamaño.



39x27 13x9

Figura 6: Reducción de tamaño de la imagen ensanchada

5. Clasificador basado en Redes Neuronales

El clasificador de números manuscritos recibe la imagen preprocesada según lo indicado en la sección anterior y utiliza redes neuronales feedforward para realizar el reconocimiento. Este tipo de redes, adaptadas mediante la técnica del gradiente descendente de manera supervisada, son conocidas como MLP (multilayer perceptron) y proporcionan muy buenos resultados cuando los datos utilizados en el entrenamiento cubren adecuadamente el espacio de entrada. Su convergencia depende de la consistencia de la información que se les suministra.

Por todo lo anterior, y con el objetivo de acotar el tiempo de entrenamiento, se ha utilizado un clasificador basado en un sistema de votación formado por tres redes neuronales feedforward, con idéntica arquitectura, que reconocen en paralelo una misma entrada. Posteriormente, el resultado de cada red es supervisado por una unidad parametrizable que permite especificar el grado de precisión mínimo de la red para que su respuesta sea aceptada. Luego, se utiliza un sistema de votación, que en base al resultado de cada red toma una decisión respecto de qué digito se trata, o si no hay suficiente confiabilidad, determina al dígito como “no reconocido”.

En las secciones siguientes se explicará más en detalle cada parte del módulo de reconocimiento.

5.1. Arquitectura de cada red neuronal

Las tres redes neuronales utilizadas presentan la misma arquitectura; son feedfroward completamente conectadas formadas por: una capa de entrada, una oculta y una de salida.

La entrada de la red neuronal está conformada por 117 entradas independientes correspondientes a cada píxel de la imagen de entrada (de tamaño 13x9).

La capa oculta de cada red está conformada por 50 neuronas. Este número se ha seleccionado en base a estudios empíricos teniendo en cuenta el rendimiento del sistema. Para elegir la cantidad de neuronas de la capa oculta se realizaron distintas pruebas sobre el conjunto de dígitos con 30, 40, 50, 60 y 70 neuronas; para cada una de ellas se utilizaron los siguientes valores para la velocidad de

aprendizaje: 0.005 y 0.0005. Las mediciones realizadas muestran que 50 neuronas en la capa oculta permiten obtener la mejor clasificación en un tiempo de entrenamiento aceptable. El valor elegido para la velocidad de aprendizaje es 0.005.

La capa de salida está conformada por 10 neuronas, una para cada dígito. Cada neurona de salida puede tomar un valor en el intervalo [0,1]. Dada una entrada, se espera que la red genere una salida formada por 10 números reales entre [0,1] donde uno de ellos es notablemente superior al resto y cercano a uno.

5.2. Entrenamiento de las redes neuronales

El entrenamiento de cada red del clasificador se realiza de manera independiente comenzado desde distintos valores aleatorios para sus matrices de pesos.

Los datos utilizados provienen de una combinación de los dígitos de [1] y [4]. La base de datos está conformada por 7620 dígitos en total. A partir de ella, se crearon tres conjuntos, un conjunto de entrenamiento, un conjunto de validación y un conjunto de testeo. El primer conjunto está conformado por el 50% de los dígitos (3809 dígitos). El segundo conjunto con el 15% de los dígitos (1291 dígitos). Finalmente, el conjunto de testeo se conforma con el 35% restante (2520 dígitos).

Para entrenar cada red neuronal utilizada se practica un esquema de "parada temprana". Mediante este enfoque se pretende evitar que la red pierda generalidad. Tradicionalmente una red MLP se entrena tratando de minimizar su error hasta que el mismo sea inferior a cierta cota. Se practica un enfoque de parada temprana para evitar que la red se especialice demasiado en los dígitos contenidos en el conjunto de entrenamiento y así pierda generalidad sobre dígitos que se puedan presentar cuando la red esté en funcionamiento en una aplicación de reconocimiento real.

Teniendo esto en mente, lo que se hace es utilizar el conjunto de validación. El entrenamiento entonces minimiza el error de la red sobre el conjunto de entrenamiento, pero en cada paso verifica que también descienda el error del conjunto de validación. Cuando el error del conjunto de validación en el paso j + 1 sea mayor que en el paso anterior j, quiere decir que la red se está especializando demasiado sobre el conjunto de entrenamiento para reconocer cada patrón y puede estar perdiendo generalidad. En este momento se detiene el entrenamiento. En la herramienta propuesta la medida de error utilizada corresponde al error cuadrático medio del conjunto correspondiente.

5.3. Precisión de la solución

Una vez obtenida la respuesta de una red neuronal en base a un dígito de entrada, las 10 neuronas de la capa de salida son analizadas. Una unidad parametrizable se encarga de garantizar cierto grado de confiabilidad a la respuesta producida por la red neuronal. Cada red MLP tiene su propia unidad de precisión y utiliza dos parámetros. El primero, denominado *UMBRAL*, se aplica al valor más alto entre las respuestas de las neuronas de la capa de salida y determina una cota inferior, para este valor, a fin de que sea considerado válido. De no ser así, el dígito se clasifica como no reconocido.

El segundo parámetro, llamado *DIF*, permite cuantificar la diferencia entre los dos máximos valores de las 10 neuronas de salida. Esto evita que una entrada sea reconocida como dos o más dígitos con un grado similar de confiabilidad. Si esto sucede, nuevamente el dígito se considera no reconocido. En resumen, las reglas utilizadas son las siguientes:

REGLA 1: SI max(salida) < *UMBRAL* ENTONCES no reconocido

REGLA 2: SI max(salida) * *DIF* < segundo_max(salida) ENTONCES no reconocido

5.4. Unidad de votación

La unidad de votación es el módulo encargado de recoger todos los resultados de las redes neuronales y en base a ellos tomar la decisión de definir el dígito en cuestión o clasificarlo como no reconocido.

En la herramienta propuesta se encuentran implementadas dos unidades, llamadas "votacion23" y "votacion100". En la primera de ellas, se toma la decisión de clasificar al dígito de entrada si dos de las tres redes llegan al mismo resultado. "Votacion100" escoge el resultado si las tres redes coinciden respecto a una entrada, o sea, existe un acuerdo del 100%.

Cada módulo de votación determina un comportamiento diferente para el clasificador. Con "votacion23", el sistema es más tolerante, produciendo una mayor cantidad de aciertos en cuanto al número de dígitos reconocidos, pero también un mayor número de falsos positivos, es decir, dígitos reconocidos incorrectamente. Con "votacion100" la cantidad de dígitos reconocidos disminuye, pero también descienden los errores cometidos, incrementándose la cantidad de dígitos no reconocidos. Dependiendo del uso del sistema en casos particulares, los cuales definen la precisión deseada, podría ser preferible tener una mayor cantidad de dígitos no reconocidos, a que estos sean identificados incorrectamente.

5.5. Resultados de la clasificación

Para demostrar el comportamiento del módulo de reconocimiento se presenta la siguiente tabla, en la cual se muestran los resultados de las tres redes neuronales trabajando de manera independiente sobre la base de datos de dígitos, como así también el escrutinio de ambas unidades de votación; siendo UMBRAL = 0.55 y DIF = 0.85.

	Correctas	Rechazadas	Incorrectas
MLP 1	80.60 %	17.78 %	1.63 %
MLP 2	80.99 %	17.62 %	1.39 %
MLP 3	80.36 %	18.10 %	1.55 %
Votacion23	81.71 %	17.14 %	1.15 %
Votacion100	72.74 %	26.71 %	0.56 %

Tabla 1: Comportamiento del módulo de reconocimiento (umbral = 0.55 dif = 0.85)

En la tabla puede observarse, además del comportamiento de cada red individual, las características que antes se explicaron acerca de las unidades de votación. El módulo "votacion23" permite obtener un valor más bajo de incorrectos, teniendo en cuenta el comportamiento de cada red en particular, sin perder efectividad en el porcentaje de dígitos reconocidos correctamente. Con "votacion100" se reduce la cantidad de dígitos correctos, pero lo mismo sucede con los dígitos reconocidos incorrectamente.

6. Conclusiones y líneas de trabajo futuras

Se ha presentado una herramienta de software para el reconocimiento de números enteros manuscritos que integra un clasificador basado en redes neuronales feedforward y un conjunto de técnicas pertenecientes al área de procesamiento de imágenes digitales que realiza las adaptaciones adecuadas sobre la imagen de entrada. Los resultados de la aplicación de esta herramienta sobre una base de números del repositorio UCI han sido satisfactorios.

Es importante destacar, que el uso de clasificadores basados en redes neuronales permite obtener una respuesta en un tiempo computacional breve ya que la etapa de entrenamiento se realiza off-line. Además, el preprocesamiento de la imagen de entrada ha permitido obtener un vector de características adecuado para cada una de las redes neuronales resolviendo los problemas habituales

de posicionamiento y escalado. Finalmente,.el sistema de votación ha demostrado ser efectivo para resolver este tipo de problemas y su respuesta puede ajustarse en función de una cota de error prefijada.

Actualmente se está trabajando en el diseño de un clasificador basado en una red neuronal competitiva que permita incorporar nuevo conocimiento sin necesidad de rehacer el entrenamiento completo. Resulta de interés analizar el impacto que esta modificación pude provocar sobre el tiempo de respuesta.

7. Referencias

[1] Alpaydin E., Kayna C. Optical Recognition of Handwritten Digits. *UCI Machine Learning Repository*. ftp://ftp.ics.uci.edu/pub/machine-learning-databases/optdigits. Último acceso julio de 2007.

[2] Arica, N. Yarman-Vural, F.T. Optical character recognition for cursive handwriting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol 24, nro. 6, pp. 801-813.

[3] R. G. Casey y E. Lecolinet, "A Survey of Methods and Strategies in Character Segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18(7), 690{706 (1996).

[4] Duin R. Multiple Features Database. *UCI Machine Learning Repository*. ftp://ftp.ics.uci.edu/pub/machine-learning-databases/mfeat. Último acceso julio de 2007.

[5] Gonzalez-Woods. "Digital Image Processing". Prentice Hall. 2002 2da edición.

[6] A. Gupta, M.V. Nagendraprasad, A. Liu, P. Wang, S. Ayyadurai. "An Integrated Architecture for Recognition of Totally Unconstrained Handwritten Numerals". Massachusetts Institute of Technology 1993.

[7] D. Maravall Gomez-Allende. "Reconocimiento de formas y visión artificial". Addison-Wesley Iberoamericana. 1993.

[8] Palacios R y Gupta A. A system for processing handwritten bank checks automatically. Image and Vision Computing, 2002.

[9] Punnoose J. An Improved Segmentation Module for Identification of Handwritten Numerals. *Department of Electrical Engineering and Computer Science at the MIT.* 1999

[10] Tyan J., Neubauer C. Character segmentation method for vehicle license plate recognition. US Patent 6,473,517, 2002

[11] Yamada H., Nakano Y. Cursive Handwritten Word Recognition Using Multiple Segmentation Determined by Contour Analysis. *IEICE Transactions on Information and Systems*. Vol.E79-D, No.5, pp.464-470. 1996

[12] Yanivello D., Lanzarini L. Reconocimiento de Comandos Gestuales utilizando GesRN. *X Congreso Argentino de Ciencias de la Computación. CACIC 2004*. Universidad Nacional de La Matanza. Bs.As. Argentina. ISBN 987-9495-58-6