TESIS DE LICENCIATURA

DEPARTAMENTO DE COMPUTACIÓN
FACULTAD DE CIENCIAS EXACTAS Y NATURALES
UNIVERSIDAD DE BUENOS AIRES

Tema: Reconocimiento de Dígitos Manuscritos con Redes Neuronales y Preprocesamiento basado en Características de Multirresolución

Alumno: Diego Javier Romero
Directora: Dra. Ana Ruedin
Codirectora: Lic. Leticia Seijas

Índice general

Resumen

El reconocimiento automático de dígitos manuscritos es sumamente difícil debido a la gran variedad de trazos e inclinaciones que tienen los dígitos. El problema tiene aplicaciones de interés, tales como el reconocimiento automático de códigos postales, el reconocimiento de importes en cheques bancarios y procesamiento automático de formularios. En este trabajo se presentan diferentes técnicas de preprocesamientos de dígitos manuscritos, cuyo desempeño para la clasificación de dígitos se evalúa con redes neuronales de tipo perceptrón multicapa. El preprocesamiento de los los dígitos manuscritos está basados en la transformada wavelet. Se han utilizado diferentes tipos de transformadas wavelet: la transformada wavelet contínua isotrópica, la transformada wavelet continua con orientación angular, la representación escala- ángulo de dicha transformada, y la transformada wavelet discreta. Con los coeficientes de aproximación de la transformada wavelet discreta, tanto como los coeficientes de la transformada wavelet continua isotrópica, se forma una imagen suavizada y más pequeña de los dígitos para mantener su correlación espacial. Luego, con las distintas transformadas wavelet se construye un vector de características que captura ciertas propiedades de los dígitos, tales como la orientación, el gradiente, la curvatura, medidos con la entropía y la energía de la transformada. El método propuesto da resultados altamente satisfactorios, en cuanto a la reducción de la dimensionalidad y las tasas de reconocimiento: del orden del 92.7 % sobre el conjunto de testeo para la base CENPARMI y del orden del 98.22 % para la base MNIST.

Abstract

Automatic recognition of handwritten numerals is difficult owing to the wide variety of strokes and orientations of these digits. The problem has interesting applications, such as automatic recognition of postal codes, recognition of amounts in banking checks an automatic processing of application forms. In this work we present different preprocessing techniques for handwritten digits, whose performance for classification is evaluated with neural networks of the multilayer perceptron type. The preprocessing techniques are based on the wavelet transform. Different wavelet transforms have been used: isotropic continuous wavelet transforms, continuous wavelet transforms with angular orientation, the scale-angle representation of the same transform, and the discrete wavelet transform. With the approximation coefficients of the discrete wavelet transform, as well as the isotropic continuous wavelet transform, a smooth and smaller version of the original sample digits is formed, to maintain the spatial correlation. Then, with different wavelet transforms a feature vector is constructed, that captures certain properties of the digits, such as orientation, gradient, curvature – measured with the entropy and the energy of the transform. The proposed method gives highly satisfactory results, regarding the dimensionality reduction as well as the recognition rates: 92.7 % on the testing set for CENPARMI database and 98.22 % for the MNIST database.

Capítulo 1

Introducción

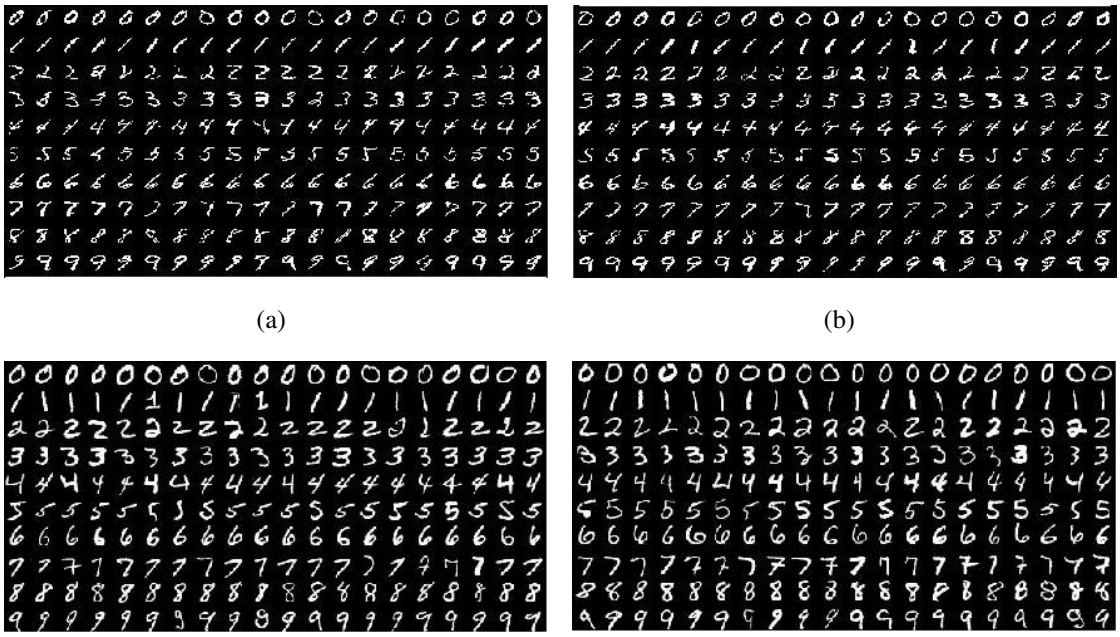
El problema del reconocimiento de dígitos manuscritos tiene aplicaciones tales como el reconocimiento automático de códigos postales, reconocimiento de importes en cheques bancarios y procesamiento automático de formularios entre otros. Estas aplicaciones son de gran interés ya que al automatizarlas se reduciría notablemente el esfuerzo manual que requieren para llevarse a cabo. Una de las principales dificultades que caracterizan a este problema es que la varianza entre los representantes de cada clase es alta, esto se debe a las diferentes formas que puede tener un mismo patrón debido al estilo particular de escritura de cada individuo. Así, por ejemplo, para dos dígitos de la misma clase podemos encontrar diferencias en el trazo, en la inclinación y en el tamaño. Hasta la actualidad no se ha presentado un modelo matemático capaz de cuantificar las variaciones entre los patrones [1]. Muchos modelos han sido presentados para lidiar con este problema pero ninguno se compara con el poder de clasificación que puede tener un ser humano. El uso de redes neuronales ha proporcionado buenos resultados en el reconocimiento de caracteres y dígitos manuscritos. Una gran cantidad de trabajos en la literatura existente aplica un método clásico para el reconocimiento de patrones: el uso de una red neuronal del tipo *feed-forward* (perceptrón multicapa) entrenada con el algoritmo de *back-propagation*. Esta arquitectura es bien conocida como una herramienta poderosa para resolver problemas de clasificación debido a su capacidad para discriminar, aprender y representar el conocimiento implícito.

El desempeño de un sistema de reconocimiento depende fuertemente de como se representa cada dígito, a través de características extraídas en la etapa de preprocesamiento. Las máscaras de Kirsch [2] han sido utilizadas por varios autores [3] [4] para extraer características direccionales. Por otro lado, en [5] [6] se utilizó el análisis de componentes principales para re-

ducir la dimensionalidad de los datos. La transformada wavelet ha probado ser una herramienta idónea para muchas aplicaciones relacionadas con el procesamiento de imágenes. Ha dado muy buenos resultados en el reconocimiento de bordes [7] y en la clasificación de texturas [8]. En [1] se utilizó, como método de preprocesamiento una transformada wavelet unidimensional, discreta y diádica aplicada al contorno previamente extraído de los dígitos. En [9] se utilizó como método de preprocesamiento, la aplicación de una transformada unidimensional multiwavelet discreta a los contornos de los dígitos previamente extraídos. La transformada wavelet discreta (*Discrete Wavelet Transform* o DWT) provee una descomposición de la imagen en detalles a diferentes resolución y orientación; esto es, una biyección entre el espacio de la imagen y el espacio de sus coeficientes [10][11]. La misma ha sido ampliamente utilizada en la compresión de imágenes. Para analizar imágenes presenta una desventaja: no es invariante frente a las traslaciones. Para el análisis de imágenes se utiliza preferentemente la transformada wavelet continua (*Continuous Wavelet Transform* o CWT), que es invariante a traslaciones, y provee una representación de la imagen que es redundante. La CWT en dos dimensiones ha sido extendida para construir lo que se llaman transformadas wavelets direccionales [12] dando así una dirección principal a la función wavelet alargándola sobre uno de sus ejes y agregando un ángulo de rotación como parámetro. La transformada resultante tiene 4 parámetros: escala, ángulo y posición (x, y) en la imagen. Esta CWT bidimensional ha sido aplicada para el reconocimiento de patrones en imágenes [13]. En [14] la misma ha sido utilizada para estimar el ángulo de pose en objetos dentro de imágenes de tipo SAR (*Syntethic Aperture Radar*). También la hemos utilizado en nuestros trabajos preliminares [15] [16], los cuales surgieron durante el desarrollo de este trabajo de tesis.

En esta tesis se emplearán métodos de preprocesamiento utilizando wavelets. Es decir, se implementaran métodos basados en la transformada wavelet (continua y discreta) y luego serán aplicados a los dígitos y con los dígitos preprocesados se entrenará a una red neuronal para tener una medida del desempeño del método de preprocesamiento. Se trabajará con dos bases de datos de dígitos manuscritos: la del CENPARMI (Centre for Pattern Recognition and Machine Intelligence at Concordia University, Canadá) cuyos dígitos fueron normalizados a un tamaño de tamaño de 16x16 píxeles y MNIST [17] con dígitos de tamaño 28x28 píxeles. Cada base de datos se dividirá en dos conjuntos, uno llamado de entrenamiento y el otro de testeo. Con el primero se entrenará a la red neuronal y con el segundo se evaluará su desempeño. En la figura 1.1 se pueden ver muestras del conjunto de entrenamiento y del de testeo para ambas bases de datos mencionadas; allí se puede observar la variabilidad existente en el trazo y en la inclinación.

Esta tesis está organizada de la siguiente manera: en el capítulo 2 se explicarán la transformada wavelet continua en dos dimensiones junto con algunos ejemplos de aplicación y la transformada wavelet discreta, en el capítulo 3 se explicarán los métodos de preprocesamiento basados en wavelets, en el capítulo 4 se darán las nociones básicas de redes neuronales y la definición del perceptrón multicapa, arquitectura que se utilizó en el sistema de reconocimiento. En el capítulo 5 se mostrarán los resultados obtenidos, sobre el conjunto de testeo y entrenamiento, luego de entrenar a la red neuronal y utilizando los métodos de preprocesamiento definidos en el capítulo 3, y por último, en el capítulo 6 se darán las conclusiones.



(a) (b)

(c) (d)

Figura 1.1: Algunas muestras de patrones de dígitos de los conjuntos (a) de entrenamiento y (b) de testeo CENPARMI. Algunas muestras de patrones de dígitos de los conjuntos (c) de entrenamiento y (d) de testeo de MNIST

Capítulo 2

Transformadas Wavelet

2.1. Transformada Wavelet Continua en 2 Dimensiones

La transformada wavelet ha dado excelentes resultados en distintas aplicaciones de procesamiento de imágenes. Su excelente localización espacial y su buena localización frecuencial hacen de ella una herramienta apta para el preprocesamiento de los dígitos. La transformada wavelet continua (CWT) ha sido utilizada para el análisis de imágenes y señales, en particular para la detección de características.

Sea s una función bidimensional (imagen) de energía finita, valores reales y de cuadrado integrable, es decir, $s \in L^2(\Re^2, d^2x), x = (x_1, x_2) \in \Re^2$:

$$\|s\|^2 = \int |s(x)| d^2x < \infty \tag{2.1}$$

en la práctica una imagen en blanco y negro se representa por una función acotada y positiva:

$$0 \leq s(x) < M, \forall x \in \Re^2, M < \infty \tag{2.2}$$

donde los valores discretos de $s(x)$ corresponden al nivel de gris de cada pixel. Definimos la transformada wavelet de s con respecto a una función wavelet $\psi : \Re^2 \rightarrow \Re$ de la siguiente forma [13]:

$$\begin{array}{rcl} S(b, a, \theta) & = & a^{-1} \int_{\Re^2} \psi(a^{-1}\, r_{-\theta}(b - x))\, s(x)\, dx, \\ & = & a \int_{\Re^2} e^{i.b.k} \widehat{\psi}(a r_{-\theta}(k))\, \widehat{s}(k)\, dk, \end{array} \tag{2.3}$$

donde $b = (b_x, b_y) \in \Re^2$ es un vector de traslación, $k = (k_x, k_y) \in \Re^2$ y $b.k = b_x k_x + b_y k_y$, $a \in \Re$ es la escala ($a > 0$), θ es un ángulo, $\widehat{\psi}$ y $\widehat{s}$ hacen referencia a la transformada Fourier de ψ y s respectivamente, por último $r_\theta(x)$ es el operador de rotación con ángulo θ, que actúa sobre un vector $x = (x_1, x_2) \in \Re^2$ de la manera siguiente:

$$r_\theta(x) = (x_1 \cos\theta - x_2 \sin\theta, x_1 \sin\theta + x_2 \cos\theta), 0 \leq \theta \leq 2\pi. \tag{2.4}$$

esto muestra que la transformada wavelet continua no es más que el producto escalar de la función s con una versión de la función ψ escalada, rotada y trasladada. Cada valor de la transformada se obtiene fijando un valor para la escala, uno para el ángulo de rotación y dos para las direcciones de traslación sobre el eje x e y. En este marco teórico no hay multiresolución, a diferencia de la transformada wavelet discreta. Los requerimientos que se imponen a la función ψ, para asegurar la recuperación de la función a partir de su transformada, son:

- $\psi(x)$ debe ser de cuadrado integrable: $\psi \in L^2(\Re^2)$
- ψ debe ser admisible, es decir, la siguiente integral debe ser finita:

$$c_\psi = \int_{\Re^2} |\widehat{\psi}(\omega)|^2 \frac{d\omega}{|\omega|} < \infty, \tag{2.5}$$

  donde $\widehat{\psi}(\omega)$ es la transformada Fourier de $\psi(x)$. Esa condición implica que

$$\widehat{\psi}(0) = 0, \tag{2.6}$$

  lo que es equivalente a la condición de *media cero*

$$\int_{\Re^2} \psi(x) dx = 0. \tag{2.7}$$

las propiedades mencionadas anteriormente implican que la función wavelet ψ está bien localizada en el dominio del tiempo (x) y en el dominio de la frecuencia (ω), luego también lo estará su versión transformada $\psi(a^{-1}\, r_{-\theta}(b - x))$ con el soporte trasladado en b, rotado en θ y escalado en a. Esto sumado a que la misma tiene media cero implican que la transformada $S(b, a, \theta)$ es apreciable sólo para los parámetros (b, a, θ) donde la señal s lo es. Dicho en otras palabras, la transformada wavelet continua actúa como un filtro local en las cuatro variables (b, a, θ) pues arroja valores significativos sobre la porción de la señal alrededor de (b, a, θ) y filtra el resto. Observar que dicha transformada realiza un análisis de la imagen en el dominio espacial a través del parámetro de traslación $b = (b_x, b_y)$ y en el dominio de la frecuencia espacial a través de los parámetros (a, θ) ya que como se puede ver en [13], $(a^{-1}, \theta) \equiv (\rho, \phi)$ donde

(ρ, ϕ) son las coordenadas polares de $\omega = (\omega_x, \omega_y)$ el cual describe el dominio de la frecuencia espacial, esto significa que los parámetros (a, θ) cumplen el rol de la frecuencia espacial en la transformada. Por lo tanto el espacio de la transformada wavelet continua en dos dimensiones es llamado un espacio *tiempo-frecuencia espacial* o lo que es lo mismo un *espacio de fase*.

2.1.1. Implementación: Las dos representaciones básicas

Al querer implementar la transformada wavelet continua bidimensional se presentan problemas debido a que la misma es una función de cuatro variables $b = (b_x, b_y), a, \theta$: por un lado está el problema de la visualización del resultado y por el otro está el problema del cómputo de la misma. Para subsanar éstos inconvenientes y obtener así una herramienta manejable, se fijan algunas de las variables dando lugar a una función unidimensional o bidimensional la cual se puede graficar sin problemas y se puede calcular en forma más eficiente. Para formar una función de dos variables hay seis posibilidades ya que en la fórmula de la transformada hay involucrada cuatro variables, pero entre ellas hay dos elecciones que se consideran las más naturales:

- Representación posicional

$$S_{a\theta}(b_x, b_y) = S(b_x, b_y, a, \theta) \quad a\ y\ \theta\ fijos \tag{2.8}$$

  observar que es una función de dos variables, es decir, tanto a como θ están fijos por lo tanto el resultado es una función en b_x y b_y. Con esta representación se analiza a la señal con una wavelet rotada en un ángulo θ y escalada en un factor a. Dicha wavelet se va desplazando en los ejes x e y acorde al valor de las variables b_x y b_y. Dicho de otra forma cada valor de $S_{a\theta}(b_x, b_y)$ será el cálculo de (2.3) en b_x, b_y, a y θ. Esta representación analiza a la imagen en el espacio para una escala (frecuencia) y ángulos determinados ((a, θ) fijos).

- Representación Escala - Ángulo

$$S_{b_x b_y}(a, \theta) = S(b_x, b_y, a, \theta) \quad b_x\ y\ b_y\ fijos \tag{2.9}$$

  observar que en este caso también tenemos una función de dos variables sólo que ahora están fijos b_x y b_y quedando la función en términos de a y θ. En contraste al caso anterior, en esta representación se analiza a la imagen con una wavelet situada en una posición

fija (b_x y b_y) pero la misma va sufriendo cambios en la rotación y en la escala acorde al valor de las variables θ y a respectivamente. Esta representación analiza a la imagen en el dominio de las frecuencias espaciales en una posición fija (b_x, b_y fijos).

La representación posicional es la más estandard y la misma fue utilizada para varios propósitos en el procesamiento de imágenes: detección de posición, forma y contornos de objetos; reconocimiento de patrones; eliminación de ruido. Por otro lado la representación escala-ángulo se utilizó en aplicaciones donde lo más importante era el comportamiento escalar (fractales por ejemplo) o la detección de ángulos. Si consideramos la densidad de energía $|s(x)|^2$ (en el dominio del tiempo), $|\widehat{s}(\omega)|^2$ (en el dominio de la frecuencia), tenemos las siguientes igualdades de preservación de la energía:

$$\int_{\Re^2} |s(x)|^2 d^2x = \int_{\Re^2} |\widehat{s}(\omega)|^2 d^2\omega = \int\int\int |S(a,\theta,b)|^2 \frac{1}{a^3} d^2b d\theta \tag{2.10}$$

las igualdades anteriores nos llevan a la interpretación de $|S(a,\theta,b)|^2$ como una densidad de energía de la señal s en los parámetros de escala, ángulo y posición. Claramente alguna integración parcial de ésta densidad de energía en algún subconjunto de variables nos dará otra densidad de energía en las variables restantes. Existen cuatro densidades de energía de una dimensión, seis de dos dimensiones y cuatro de tres dimensiones. A continuación se mostrarán las fórmulas de las densidades de energías que se corresponden con las representaciones vistas anteriormente:

- Densidad Posicional:

$$E(b_x, b_y) = \int_0^\infty (\int_0^{2\pi} |S(a,\theta,b)|^2 d\theta) \frac{da}{a^3} \tag{2.11}$$

  notar que la densidad posicional es distinta a la representación posicional ya que la primera realiza, para cada posición en los ejes x e y, un promedio sobre todas las escalas y todos los ángulos justamente para eliminar la dependencia angular y escalar, mientras que la representación posicional, para cada posición en los ejes x e y, sólo realiza el cálculo utilizando un valor de escala a y un valor de ángulo θ.

- Densidad Escala - Ángulo:

$$E(a,\theta) = \int_{\Re^2} |S(a,\theta,b)|^2 d^2b \tag{2.12}$$

  por otro lado la densidad escala-ángulo se diferencia de la representación escala-ángulo en el hecho de que la primera realiza, para cada escala y ángulo, un promedio sobre todas

las posiciones para eliminar la dependencia del punto de observación, mientras que la representación escala-ángulo, para cada escala y ángulo, utiliza un valor de desplazamiento en el eje x y un valor de desplazamiento en el eje y.

Es evidente que las densidades aportan más información que sus respectivas representaciones, aunque a la hora de implementarlas son más costosas computacionalmente.

2.1.2. Tipos de Wavelets

Un paso fundamental a la hora de implementar la transformada wavelet continua es la elección de la función wavelet ψ, existen varias wavelets que cumplen con las condiciones de admisibilidad que se mencionaron anteriormente y la elección del tipo de wavelet a utilizar depende fuertemente del problema que se quiere resolver, ya que dependiendo de la aplicación alguna podría llegar a ser más eficiente que otra. Las wavelets se agrupan en las siguientes clases:

2.1.2.1. Wavelets isotrópicas

Cuando se quiere realizar un análisis puntual de la señal donde no interesa analizar las características orientadas de la misma, uno debería elegir una wavelet ψ que sea invariante frente a las rotaciones. Luego la dependencia sobre θ puede eliminarse en la fórmula de la transformada. El típico ejemplo de este tipo de wavelets es el siguiente:

- Mexican Hat, en su versión isotrópica esta wavelet es simplemente el Laplaciano de una Gaussiana:

$$\psi_{MH}(x, y) = (2 - (x^2 + y^2))e^{-\frac{1}{2}(x^2+y^2)} \tag{2.13}$$

  la misma es una función real y es invariante a rotaciones. Su transformada Fourier viene dada por:

$$\widehat{\psi}_{MH}(\omega_x, \omega_y) = (\omega_x^2 + \omega_y^2)e^{-\frac{1}{2}(\omega_x^2+\omega_y^2)} \tag{2.14}$$

2.1.2.2. Wavelets anisotrópicas

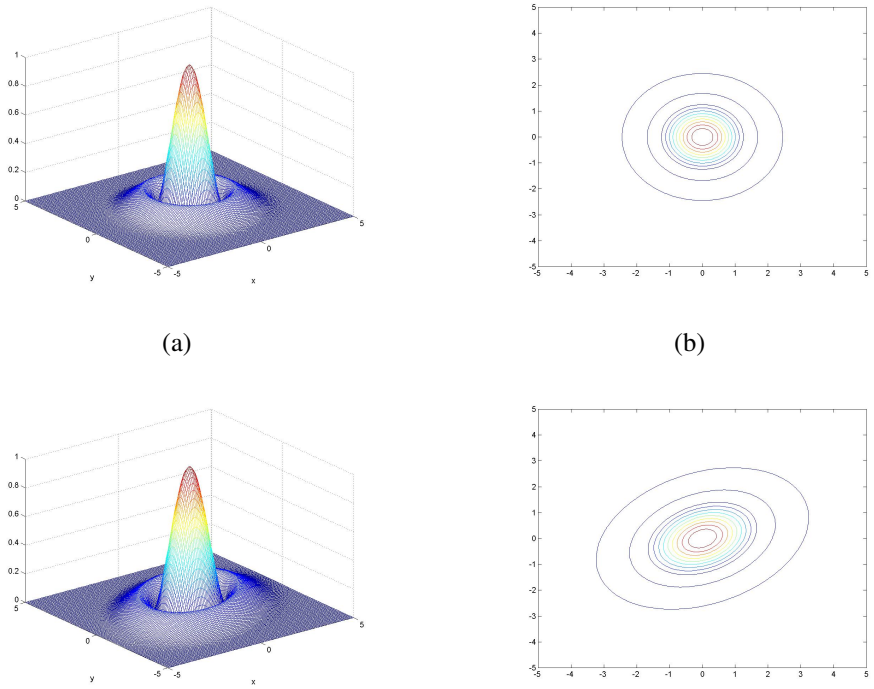
Este tipo de wavelets son aquellas que no son isotrópicas pero tampoco se ajustan a la definición de wavelets direccionales que se mostrará mas adelante. Se tratan de wavelets orientadas y un ejemplo de este tipo lo obtenemos al agregar un parámetro ϵ a la definición de la wavelet *Mexican Hat* y hacer que el mismo tome un valor mayor que 1:

$$\psi_{MH}(x,y) = (2 - (x^2 + \frac{y^2}{\epsilon}))e^{-\frac{1}{2}(x^2+\frac{y^2}{\epsilon})} \tag{2.15}$$

luego su transformada Fourier viene dada por:

$$\widehat{\psi}_{MH}(\omega_x,\omega_y) = \sqrt{\epsilon}(\omega_x^2 + \epsilon\omega_y^2)e^{-\frac{1}{2}(\omega_x^2+\epsilon\omega_y^2)} \tag{2.16}$$

cuando $\epsilon > 1$ la wavelet deja de ser invariante a rotaciones ya que éste parámetro hace que se produzca una dilatación de la función sobre el eje y. En la figura 2.1 se pueden ver los gráficos de las wavelets mencionadas, tanto isotrópicas como anisotrópicas, junto con sus curvas de nivel.



(a) (b)

(c) (d)

Figura 2.1: Imágenes de *Mexican Hat*. Versión isotrópica: (a) Gráfico en 3D con escala $a = 1$, (b) Curvas de nivel. Versión anisotrópica: (c) Gráfico en 3D con escala $a = 1, \epsilon = 2$, rotada con ángulo $\theta = 30°$ y (d) Curvas de nivel.

2.1.2.3. Wavelets direccionales

Cuando el objetivo es detectar características orientadas (segmentos, bordes, etc.) en una imagen, por ejemplo para realizar un filtrado direccional, uno tiene que elegir una wavelet ψ que no sea invariante a las rotaciones. La mejor selectividad angular será obtenida cuando la wavelet ψ es *direccional*. Esto significa que el soporte efectivo de su transformada Fourier $\widehat{\psi}$ está contenido en un cono convexo en el espacio de las frecuencias ω, con ápice en el origen o bien en la unión de finitos conos de éstas características (en ese caso, usualmente, uno dice que ψ es *multidireccional*). Notar que de acuerdo a ésta definición la wavelet *Mexican Hat* anisotrópica no es direccional ya que no importa que tan grande sea ϵ el soporte de $\widehat{\psi}_H$ está centrado en el origen. Esta definición está justificada en la fórmula 2.3 ya que como se puede ver en la segunda igualdad la wavelet actua como un filtro en el espacio de las frecuencias $\vec{\omega}$ (multiplicación por $\widehat{\psi}$). Por ejemplo, supongamos que la señal $s(\vec{x})$ está fuertemente orientada en la dirección del eje x, luego su transformada Fourier $\widehat{s}(\vec{\omega})$ estará fuertemente orientada en el eje y pero del dominio de las frecuencias. Entonces si queremos detectar tal señal, con una buena direccionalidad angular, necesitaremos que el soporte de $\widehat{s}(\vec{\omega})$ esté contenido en un cono en el dominio de las frecuencias. De esta forma la transformada arrojará valores significativos donde $\widehat{s}(\vec{\omega})$ esté alineada con $\widehat{s}(\vec{\omega})$ y valores no significativos en el resto. Observar que éste requerimiento también se podría haber impuesto a la wavelet ψ en el dominio temporal pero el mismo carece de sentido por el hecho de que a la hora de implementar la transformada la misma se calcula en base a la segunda igualdad de 2.3 (por razones de eficiencia) en la cual interviene la transformada Fourier. Es por eso que la definición de direccionalidad se impone sobre el soporte de $\widehat{\psi}(\vec{\omega})$ y no de $\psi(\vec{x})$. A continuación se muestrán algunos ejemplos de wavelets direccionales:

- Morlet

  Este es el prototipo de una wavelet orientada:

  $$\psi_M(x,y) = e^{ik_0 y} e^{-\frac{1}{2}(\frac{x^2}{\epsilon}+y^2)} + CT \tag{2.17}$$

  k_0 es el vector de onda, y $\epsilon \geq 1$ al igual que en la wavelet mostrada anteriormente es el parámetro anisotrópico. Para $\epsilon > 1$, el módulo de ésta wavelet es una Gaussiana elongada en la dirección x. CT es un término de corrección que asegura que la wavelet sea admisible, en particular que cumpla con la condición 2.6, pero la misma es insignificante cuando $|k_0| \geq 5{,}6$ y usualmente no se tiene en cuenta este término. Como se puede ver en [18] la mejor selectividad se logra tomando k_0 perpendicular al eje más largo del módulo de la wavelet, así $k_0 = (0, k'_0)$. Finalmente la fórmula de la wavelet queda:

$$\psi_M(x,y) = e^{ik_0 y} e^{-\frac{1}{2}(\frac{x^2}{\epsilon}+y^2)} \tag{2.18}$$

y su transformada Fourier es la siguiente:

$$\widehat{\psi}_M(k_x,k_y) = \sqrt{\epsilon}\, e^{-\frac{1}{2}(\epsilon k_x^2+(k_y-k_0)^2)} \tag{2.19}$$

dicha wavelet es compleja y la misma detecta, preferentemente, singularidades (ejes) en la dirección x y su eficiencia se incrementa a medida que ϵ crece.

- Cauchy

  Sea $C(\alpha,\beta)$ el cono convexo determinado por los vectores unitarios $\vec{e}_\alpha, \vec{e}_\beta$, donde $\alpha < \beta, \beta-\alpha < \pi$ y $e_\gamma \equiv (\cos\gamma, \sin\gamma)$. El eje del cono es $\vec{\zeta}_{\alpha\beta} = \vec{e}_{\frac{1}{2}(\alpha+\beta)}$. Luego tenemos

$$\begin{array}{rcl} C(\alpha,\beta) & = & \{\vec{k} \in \Re^2 : \alpha \leq arg(\vec{k}) \leq \beta\} \\ & = & \{\vec{k} \in \Re^2 : \vec{k}.\vec{\zeta}_{\alpha\beta} \geq \vec{e}_\alpha.\vec{\zeta}_{\alpha\beta} = \vec{e}_\beta.\vec{\zeta}_{\alpha\beta} > 0\} \end{array} \tag{2.20}$$

  además el cono dual, también convexo, es

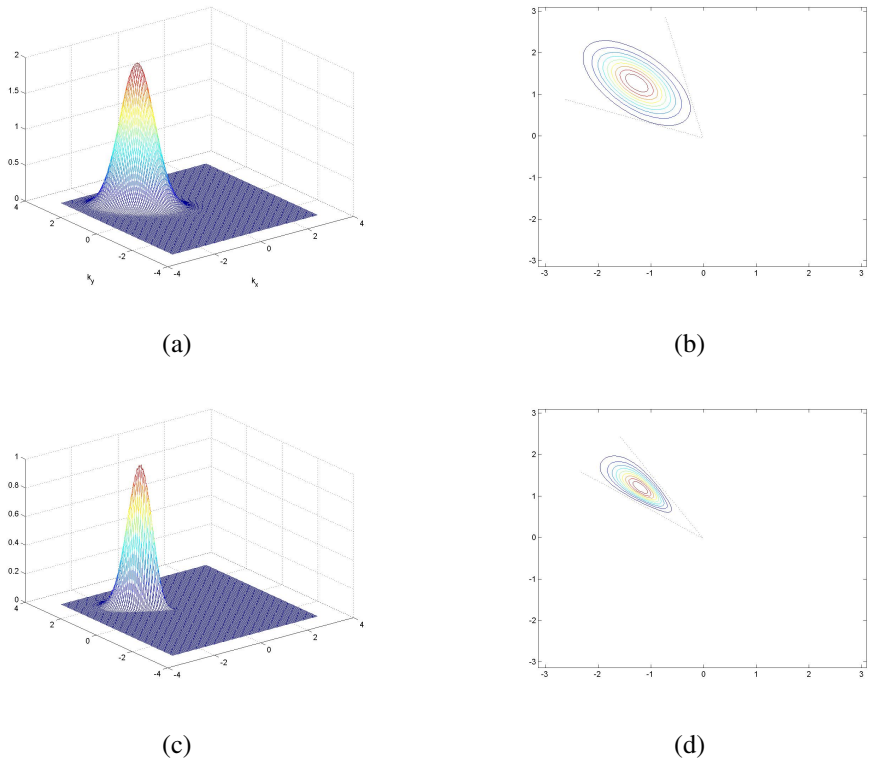
$$\widetilde{C} = C(\widetilde{\alpha},\widetilde{\beta}) = \{\vec{k} \in \Re^2, \vec{k}.\vec{k}' > 0, \forall \vec{k}' \in C(\alpha,\beta)\} \tag{2.21}$$

  donde $\widetilde{\beta} = \alpha + \frac{\pi}{2}, \widetilde{\alpha} = \beta + \frac{\pi}{2}$, y por lo tanto $\vec{e}_{\widetilde{\alpha}}.\vec{e}_\beta = \vec{e}_{\widetilde{\beta}}.\vec{e}_\alpha = 0$, mientras que $\vec{e}_{\widetilde{\alpha}}.\vec{e}_\alpha = \vec{e}_{\widetilde{\beta}}.\vec{e}_\alpha = \sin(\beta-\alpha)$. Así el eje de $\widetilde{C}$ es nuevamente $\vec{\zeta}_{\alpha\beta}$. Con ésta notación se define la wavelet *Cauchy* en dos dimensiones, con soporte en $C = C(\alpha,\beta)$, para algún $\vec{\eta} \in \widetilde{C}$ y $l, m \in N$:

$$\psi_C(\vec{x}) = \frac{i^{l+m+2}}{2\pi}\, l!m!\, \frac{[sin(\beta-\alpha)]^{l+m+1}}{[(\vec{x}+i\vec{\eta}).\vec{e}_\alpha]^{l+1}[(\vec{x}+i\vec{\eta}).\vec{e}_\beta]^{m+1}} \tag{2.22}$$

  luego su transformada Fourier viene dada por:

$$\widehat{\psi}_C(\vec{k}) = \begin{cases} (\vec{k}\,.\,\vec{e}_{\widetilde{\alpha}})^l\,(\vec{k}\,.\,\vec{e}_{-\widetilde{\alpha}})^m\, e^{-\vec{k}.\vec{\eta}}, & \vec{k} \in C(-\alpha,\alpha) \\ 0, & \text{caso contrario} \end{cases} \tag{2.23}$$



(a) (b)

(c) (d)

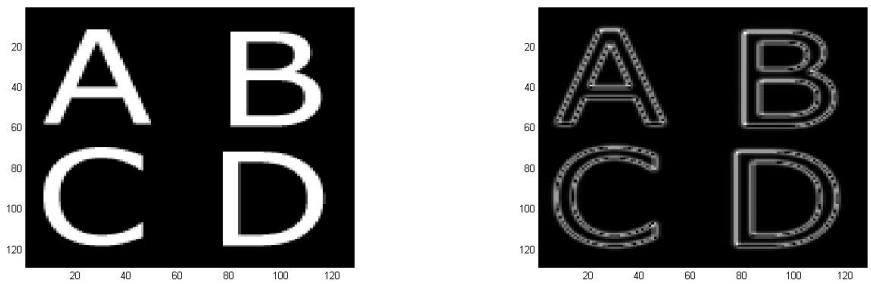
Figura 2.2: Comparación de las transformadas Fourier de una wavelet orientada y de una wavelet direccional: (a) y (b) Gráfico en 3D de la transformada Fourier de la wavelet Morlet y sus curvas de nivel con escala $a = 0{,}8$ y ángulo $\theta = 45°$. (c) y (d) Gráfico en 3D de la transformada Fourier de la wavelet Cauchy y sus curvas de nivel con escala $a = 0{,}6$ y ángulo $\theta = 45°$ en el cono $C(-15°, 15°)$.

2.1.3. Aplicaciones

A continuación se ejemplificaran algunos problemas que fueron tratados utilizando la transformada wavelet continua en dos dimensiones.

2.1.3.1. Detección de Contorno

En éste primer ejemplo veremos una aplicación utilizando una wavelet isótropica. Dado que la transformada wavelet continua es sensible a las discontinuidades la misma es eficiente en la detección del contorno. Esta característica también fue aprovechada para el reconocimiento de dígitos manuscritos ([9]). Para identificar el contorno se calcula la representación posicional (2.8) con una escala a_0 pequeña y una wavelet isótropica. Observar que el requerimiento de utilizar éste tipo de wavelets es porque no interesa analizar segmentos o líneas orientadas si no que se analizará toda la imagen en forma global. La escala tiene que ser suficientemente pequeña para detectar los detalles más finos. En la figura 2.3 se puede ver un ejemplo. La wavelet utilizada fue *Mexican Hat* y la escala $a_0 = 0{,}5$.



(a) (b)

Figura 2.3: Aplicación de Representación Posicional para extraer el contorno. (a) Imagen con letras y (b) imagen con el contorno de esas letras extraído a partir de la aplicación de Representación Posicional con la wavelet *Mexican Hat* con escala $a_0 = 0{,}5$

2.1.3.2. Filtrado Direccional

En este ejemplo se ilustrará una aplicación llamada filtrado direccional. Para llevarla a cabo se necesita una wavelet direccional (en el ejemplo se utilizó *Cauchy*) y el cálculo de la representación posicional definido en 2.8. La idea es que se tiene una imagen con segmentos o líneas en varias direcciones y a partir de ella se quiere obtener otra en la cual se visualicen sólo los segmentos o líneas de una dirección en particular. El procedimiento es el siguiente:

1. Se calcula la representación posicional definida en 2.8 con una escala a_0 y un ángulo θ de acuerdo a la orientación de los segmentos o líneas a preservar.
2. En este punto tenemos una imagen donde los segmentos o líneas en la dirección θ son más notorios que los otros. Luego se aplica un umbral para remover los menos notorios ya que se los considera ruido.
3. Finalmente obtenemos una imagen que contiene solamente los elementos cuya dirección es θ.

En las imágenes mostradas en la figura 2.4 se puede ver un ejemplo de un filtrado direccional.

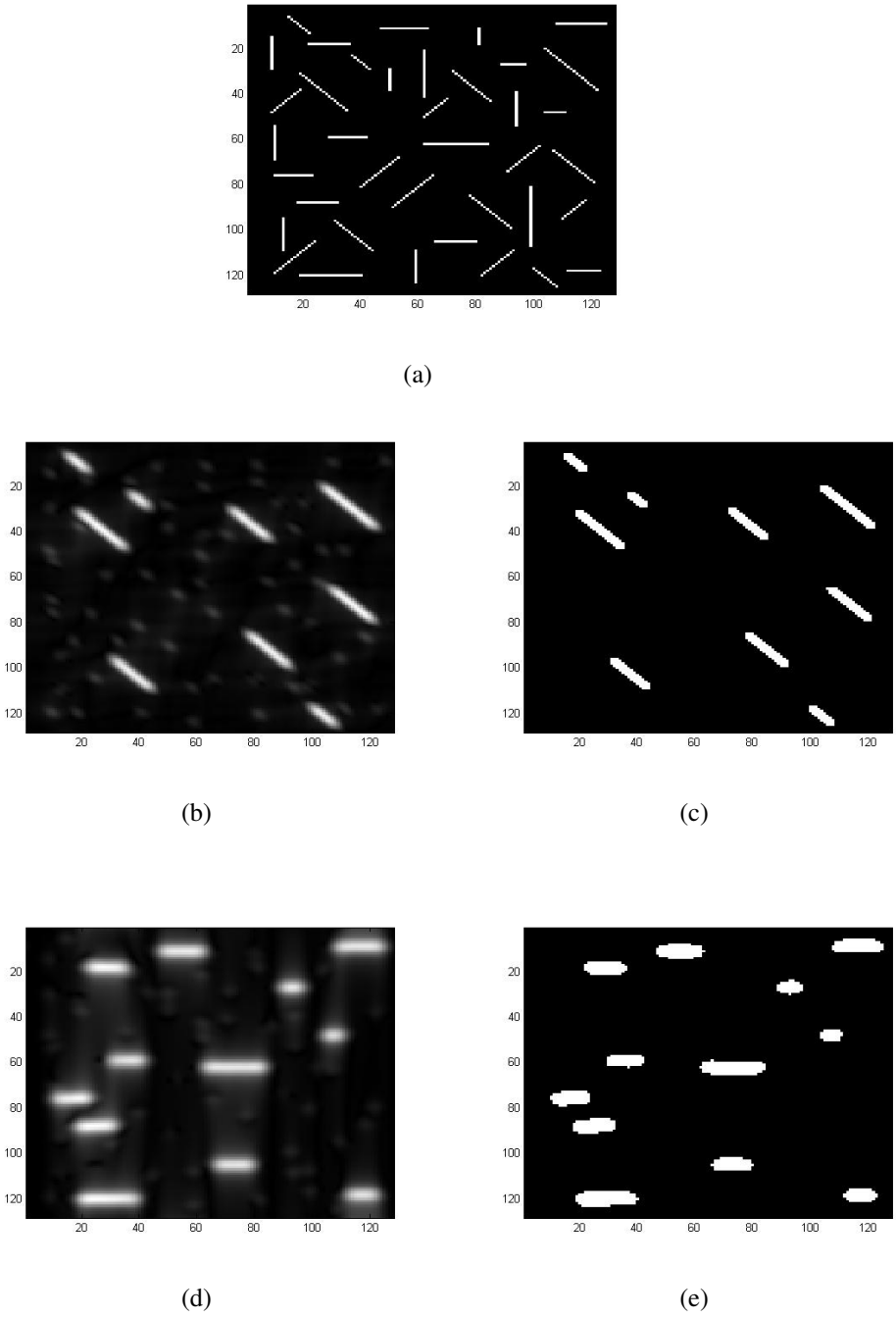


Figura 2.4: Filtrado Direccional utilizando Representación Posicional. (a) Imagen con segmentos orientados a $0°, 45°, 90°$ y $135°$. (b) Imagen resultado después de aplicar Representación Posicional con la wavelet *Cauchy* en $C(-10°, 10)$, $a_0 = 0{,}8$, $\theta = 45°$ y (c) Misma imagen después de aplicar un umbral del $40\,\%$. En (d) y (e) se repite el procemiento pero con $\theta = 0°$

2.1.3.3. Estimación del ángulo de orientación

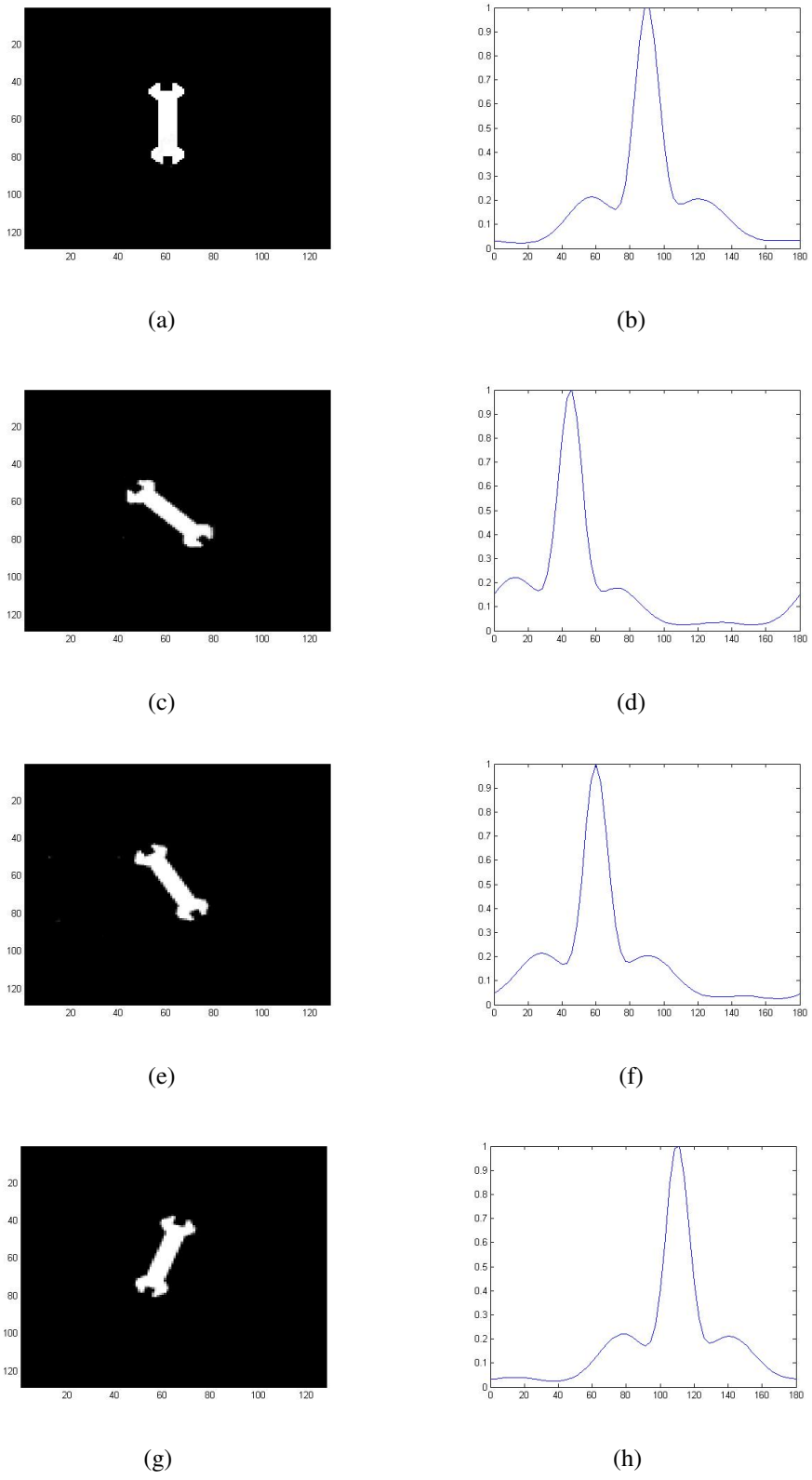
En éste ejemplo de aplicación se mostrará cómo se puede estimar el ángulo de orientación global de un objeto utilizando la *densidad escala-ángulo* definida en 2.12. Para ello se procede a calcular la misma teniendo en cuenta lo siguiente:

- Se debe utilizar una wavelet orientada: *Morlet* o *Cauchy* por ejemplo.
- Se fija la escala a en un valor determinado empíricamente. Llamemos a ese valor a_0. Observar que éste hecho determina que la densidad angular sea unidimensional:

$$E_{a_0}(\theta) = \int_{\Re^2} |S(a_0, \theta, b)|^2 d^2b \tag{2.24}$$

- Se evalúa en un rango de ángulos entre 0^o y 180^o.

una vez calculada la densidad se toma como ángulo de orientación del objeto áquel que maximiza a la función $E_{a_0}(\theta)$. Para los ejemplos mostrados a continuación se utilizó la wavelet *Cauchy*, la escala fijada a_0 fue de $2{,}5$ y para los ángulos se tomaron 64 valores entre 0^o y 180^o. En la figura (a) se puede ver un objeto con orientación de 90^o y en la (b) la densidad aplicada a ese objeto la cual tiene un máximo en ese ángulo. En las figuras (c)-(d), (e)-(f) y (g)-(h) se repite el experimento pero para el objeto con orientado a 45^o, 60^o y 110^o respectivamente. La orientación en todos los casos debe considerarse en el sentido horario.



(a) (b) (c) (d) (e) (f) (g) (h)

Figura 2.5: Estimación del ángulo de orientación de un objeto utilizando la wavelet *Cauchy* con escala $a_0 = 2{,}5$ y 64 ángulos entre 0^o y 180^o. (a) Objeto con una orientación de 90^o y (b) su densidad angular maximizada en 90^o. En (c)-(d), (e)-(f) y (g)-(h) se repite el experimento pero con un ángulo de orientación de 45^o, 60^o y 110^o respectivamente.

2.2. Transformada Wavelet Discreta

2.2.1. Análisis de multirresolución en una dimensión

La transformada wavelet discreta (DWT) tiene parámetros de escala y traslación que son discretos. La teoría de la DWT se enriquece con el análisis de multirresolución y los espacios de aproximación y detalle. Un análisis de multirresolución consiste en una secuencia de subespacios anidados

$$.. \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset ..$$

cuya unión es densa en $L^2(\Re)$ y cuya intersección es la función nula.

El subespacio V_0 es generado por las traslaciones enteras de una función $\Phi(x)$, llamada *función de escala,* y los demás subespacios V_j son generados por las traslaciones enteras de una versión dilatada o contraída de la misma $\Phi(x)$.

$$V_j = \overline{gen\{\Phi(2^j\ x - k)\}_k}$$

de la inclusión $V_0 \subset V_1$ se deduce que $\Phi(x)$, que es base de V_0, debe poder expresarse como combinación de las bases de V_1. Entonces deben existir constantes h_k tal que

$$\Phi(x) = \sum_{k=0}^{N} h_k\ \ \Phi(\ 2\ x - k\ ). \tag{2.25}$$

la ecuación 2.25 es llamada ecuación de dilatación o de refinamiento.

Consideramos los casos en que el conjunto

$$\{\Phi(x - k)\}_k$$

es ortonormal. Para cada entero j , W_j es el complemento ortogonal de V_j en V_{j+1}:

$$V_{j+1} = V_j \oplus W_j.$$

los subespacios W_j son generados por las traslaciones enteras de versiones dilatadas de una función $\Psi(x)$ llamada $wavelet$ (ondelette, ondita, ondícula):

$$W_j = \overline{gen\{\Psi(2^j\ x - k)\}_k}.$$

de la inclusión $W_0 \subset V_1$ se deduce que $\Psi(x)$, que es base de W_0, debe poder expresarse como combinación de las bases de V_1. Entonces deben existir constantes g_k tal que

$$\Psi(x) = \sum_k g_k \, \Phi(\, 2\, x - k\,). \tag{2.26}$$

conocida como ecuación de la wavelet. Una vez conocidos los coeficientes h_k de la función de escala $\Phi(x)$, la wavelet asociada $\Psi(x)$ se puede hallar directamente, es decir se pueden calcular los coeficientes g_k de la ecuación de la wavelet como

$$g_k = (-1)^k \, h_{1-k}. \tag{2.27}$$

en la figura 2.7 se pueden ver ejemplos de funciones de escalas junto con sus funciones wavelets asociadas.

2.2.1.1. Procesamiento en una dimensión

En el procesamiento de señales unidimensionales el espacio V_0 representa a la función $f_0(x)$ asociada a la señal original en su nivel más alto de resolución. Las sucesivas proyecciones $f_{-1}(x)$, $f_{-2}(x)$, ... de $f_0(x)$ sobre los subespacios V_{-1}, V_{-2}, son representaciones de $f_0(x)$ en una resolución cada vez menor, mientras que los detalles o diferencias desde una proyección a la otra quedan capturados en los subespacios W_j (componente $r_j(x)$). Así la función original se descompone en una aproximación burda de la misma y la suma de todos los componentes de detalle en distintas escalas. Llamamos $c^{(0)}$ a la señal original y sea $f_0(x)$ su función asociada, donde $f_0(x) \in V_0$, si se descompone a $f_0(x)$ en la suma de sus proyecciones sobre V_{-1} y W_{-1}, se tiene

$$\begin{aligned} f_0(x) &= \sum_{k \in Z} c_k^{(0)} \Phi(x-k) \\ &= f_{-1}(x) + r_{-1}(x) \\ &= \sum_{k \in Z} c_k^{(-1)} \tfrac{1}{\sqrt{2}} \Phi(2^{-1}x - k) + \sum_{k \in Z} d_k^{(-1)} \tfrac{1}{\sqrt{2}} \Psi(2^{-1}x - k) \\ &= \sum_{k \in Z} c_k^{(-1)} \Phi_{-1,k}(x) + \sum_{k \in Z} d_k^{(-1)} \Psi_{-1,k}(x) \end{aligned}$$

donde

$$\Phi_{j,k} = 2^{j/2} \Phi(2^j x - k); \Psi_{j,k} = 2^{j/2} \Psi(2^j x - k)$$

observar que la función $f_{-1}(x) \in V_{-1}$, es una versión más suave de la función original ya que la misma tiene menor resolución, en consecuencia, sus coeficientes $c_k^{(-1)}$ son llamados

coeficientes de aproximación. Por otro lado, la función $r_{-1}(x) \in W_{-1}$ contiene los detalles de $f_0(x)$, y sus coeficientes $d_k^{(-1)}$, son llamados *coeficientes de detalle*.

La obtención de los coeficientes de aproximación $\{c_k^{(-1)}\}$ y de detalle $\{d_k^{(-1)}\}$ a partir de los coeficientes de la señal original $\{c_k^{(0)}\}$ constituye un paso de la transformada wavelet. Una de las ventajas más sobresalientes de esta transformada es que es una transformada rápida, y se puede demostrar que

$$c_k^{(-1)} = \frac{1}{\sqrt{2}} \sum_j h_{j-2k}\, c_j^{(0)} \tag{2.28}$$

$$d_k^{(-1)} = \frac{1}{\sqrt{2}} \sum_j g_{j-2k}\, c_j^{(0)} \tag{2.29}$$

de la fórmulas de análisis (2.28) y (2.29) se deduce que los coeficientes de aproximación $c^{(-1)}$ se pueden expresar como una convolución de la señal original $c^{(0)}$ con el filtro

$$\frac{1}{\sqrt{2}} h' = \frac{1}{\sqrt{2}} [\, \ldots\ h_3\ h_2\ h_1\ h_0]$$

seguida de un submuestreo, mientras que los coeficientes de detalle $d^{(-1)}$ se obtienen de una convolución de $c^{(0)}$ con el filtro

$$\frac{1}{\sqrt{2}} g' = \frac{1}{\sqrt{2}} [\, \ldots\ g_{-2}\ g_{-1}\ g_1\ g_0]$$

seguido de un submuestreo:

$$c^{(-1)} = \left(c^{(0)} * \frac{h'}{\sqrt{2}} \right) \downarrow 2 \tag{2.30}$$

$$d^{(-1)} = \left(c^{(0)} * \frac{g'}{\sqrt{2}} \right) \downarrow 2 \tag{2.31}$$

esto se indica en el lado izquierdo del esquema (2.6). De la misma forma si uno quisiera calcular $c^{(-2)}$ y $d^{(-2)}$ tendría que aplicar el mismo procedimiento pero partiendo desde $c^{(-1)}$, eso equilvadría a aplicar dos pasos de la transformada.

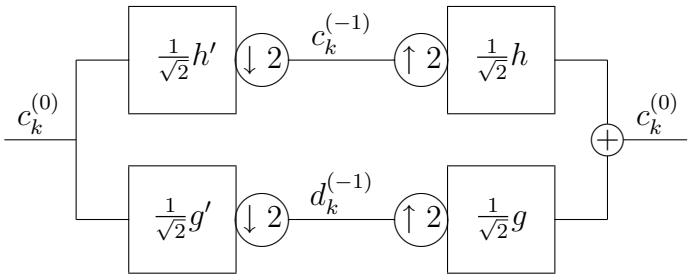


Figura 2.6: Esquema de análisis-síntesis en dimensión 1

2.2.1.2. Procesamiento en dos dimensiones (imágenes)

A la hora de realizar el procesamiento con imágenes lo que se suele hacer es aplicar un paso de la transformada wavelet en una dimensión a las filas y luego a las columnas. Esto da origen a las funciones de escala y a las wavelets *separables*, es decir, una función de escala $\Phi(x)\Phi(y)$ y 3 wavelets asociadas: $\Psi(x)\Phi(y)$, $\Phi(x)\Psi(y)$ y $\Psi(x)\Psi(y)$. Las dos primeras wavelets capturan los detalles de una imagen en el sentido vertical y horizontal, respectivamente, para cada escala. Mientras que la última de ellas hace lo mismo con los detalles diagonales. En términos de los subespacios lo anterior equivale a tomar el producto tensorial de los espacios generados por un análisis de resolución en una dimensión:

$$\begin{aligned} V_0 &= V_0^{\{x\}} \otimes V_0^{\{y\}} = gen\{\Phi(x-i)\Phi(y-j)\} \\ &= \left[V_{-1}^{\{x\}} \oplus W_{-1}^{\{x\}}\right] \otimes \left[V_{-1}^{\{y\}} \oplus W_{-1}^{\{y\}}\right] \\ &= V_{-1} \oplus W_{-1} \end{aligned}$$

donde $N \times N$ es la dimensión de la imagen y además:

$$\begin{aligned} i &= 0 \ldots \tfrac{N}{2} \\ j &= 0 \ldots \tfrac{N}{2} \\ V_{-1} &= \left[V_{-1}^{\{x\}} \otimes V_{-1}^{\{y\}}\right] \\ W_{-1} &= \left[\left\{V_{-1}^{\{x\}} \otimes W_{-1}^{\{y\}}\right\} \oplus \left\{W_{-1}^{\{x\}} \otimes V_{-1}^{\{y\}}\right\} \oplus \left\{W_{-1}^{\{x\}} \otimes W_{-1}^{\{y\}}\right\}\right] \end{aligned}$$

luego para una imagen X de dimensión $N \times N$ siendo $f(x,y) \in V_0$ su función asociada, se puede demostrar que las fórmulas, para un paso de la transformada, son las siguientes:

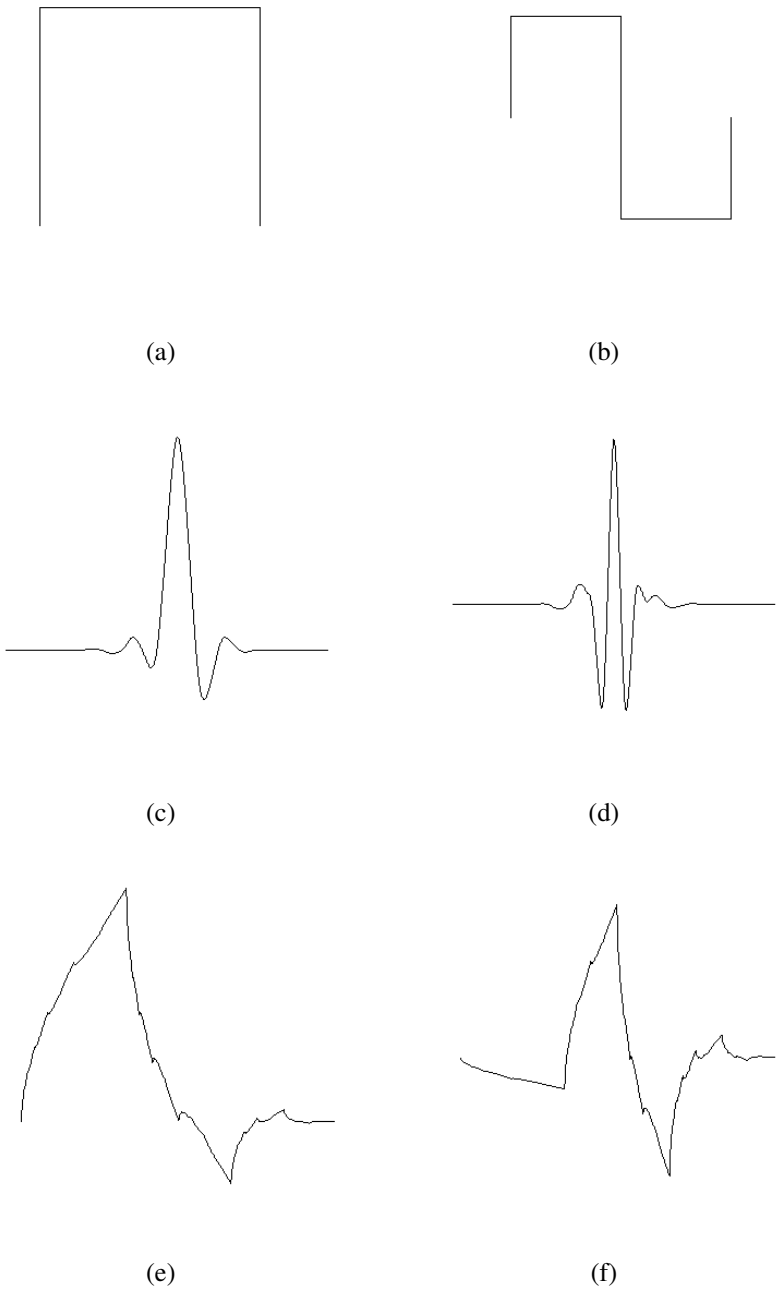


Figura 2.7: Ejemplos de funciones de escalas y wavelets. Haar: (a) y (b), Symmlet 8: (c) y (d) y Daubechies 2: (e) y (f)

$$
\begin{aligned}
f(x,y) &= \sum_{i=0}^{N/2-1}\sum_{k=0}^{N/2-1} LL_{i,k}\Phi_{-1,i}(x)\Phi_{-1,k}(y) + \sum_{i=0}^{N/2-1}\sum_{k=0}^{N/2-1} LH_{i,k}\Psi_{-1,i}(x)\Phi_{-1,k}(y) \\
&= \sum_{i=0}^{N/2-1}\sum_{k=0}^{N/2-1} HL_{i,k}\Phi_{-1,i}(x)\Psi_{-1,k}(y) + \sum_{i=0}^{N/2-1}\sum_{k=0}^{N/2-1} HH_{i,k}\Psi_{-1,i}(x)\Psi_{-1,k}(y)
\end{aligned}
$$

donde LL corresponde a la submatriz que contiene los coeficientes de aproximación de la imagen, la misma es una versión de menor resolución de la original y sus coeficientes están asociados a la base $\Phi(x)\Phi(y)$ y a sus traslaciones enteras. la sigla LL (*low-low*) es debido a que sus coeficientes han sido filtrados por un filtro pasa bajos en dirección horizontal y vertical. También en la fórmula aparece la submatriz LH la cual resalta los detalles horizontales de la imagen y sus coeficientes están asociados a la base $\Psi(x)\Phi(y)$. De forma análoga la submatriz LH resalta los detalles verticales y sus coeficientes están asociados a la base $\Phi(x)\Psi(y)$. Finalmente, la submatriz HH resalta los detalles diagonales y sus coeficientes están asociados a la base $\Psi(x)\Psi(y)$.

En la figura 2.8 se puede ver un ejemplo de la aplicación de la transformada a un dígito manuscrito utilizando la base *Haar* en uno (a) y dos pasos (b). En la primera la submatriz LL está situada arriba a la izquierda, la submatriz LH abajo a la izquierda, la submatriz HL arriba a la derecha y la submatriz HH abajo a la derecha. Observar que en la segunda imagen (b) el segundo paso se realiza sobre la submatriz LL del paso anterior, obteniéndose el mismo patrón de submatrices.

(a)

(b)

(c)

Figura 2.8: Ejemplo de la transformada wavelet discreta en dos dimensiones utilizando la función Haar. (a) Imagen original de un dígito 5, (b) 1 paso de la transformada y (c) 2 pasos de la transformada.

Capítulo 3

Métodos de Preprocesamiento Basados en Wavelets

En este capítulo se describirán los métodos de preprocesamiento basados en wavelets. Se definieron tres métodos, de los cuales dos preservan la forma y estructura del patrón original. El tercero consiste en el cálculo de un vector cuyas componentes fueron obtenidas a través de funciones (características) basadas principalmente en la transformada wavelet continua en dos dimensiones. Posteriormente estos métodos fueron aplicados a los dígitos y evaluados en forma aislada y combinada en el sistema de reconocimiento.

3.1. CWT: representación posicional con submuestreo

Este método consiste principalmente en la aplicación de representación posicional denotada por la ecuación (2.8), la cual preserva la forma, estructura y dimensión del patrón original. Con la intención de disminuir lo máximo posible la dimensión de la entrada a la red, el resultado fue submuestreado a un factor de 2, de esa forma la dimensión del resultado queda reducida a la cuarta parte de la dimensión del patrón original. Observar que el hecho de tener una entrada pequeña para la red neuronal beneficia su rendimiento y además permite la combinación de este método con otros que veremos más adelante. La wavelet utilizada fue *Mexican Hat* en su versión isotrópica (i.e. $\epsilon = 1$) con una escala $a_0 = 2{,}2$. Notar que se prefirió usar la versión isotrópica de *Mexican Hat* para analizar a los dígitos en todas las direcciones y no en una en particular y de esta forma preservar la estructura y forma del patrón original. Como consecuencia, podemos

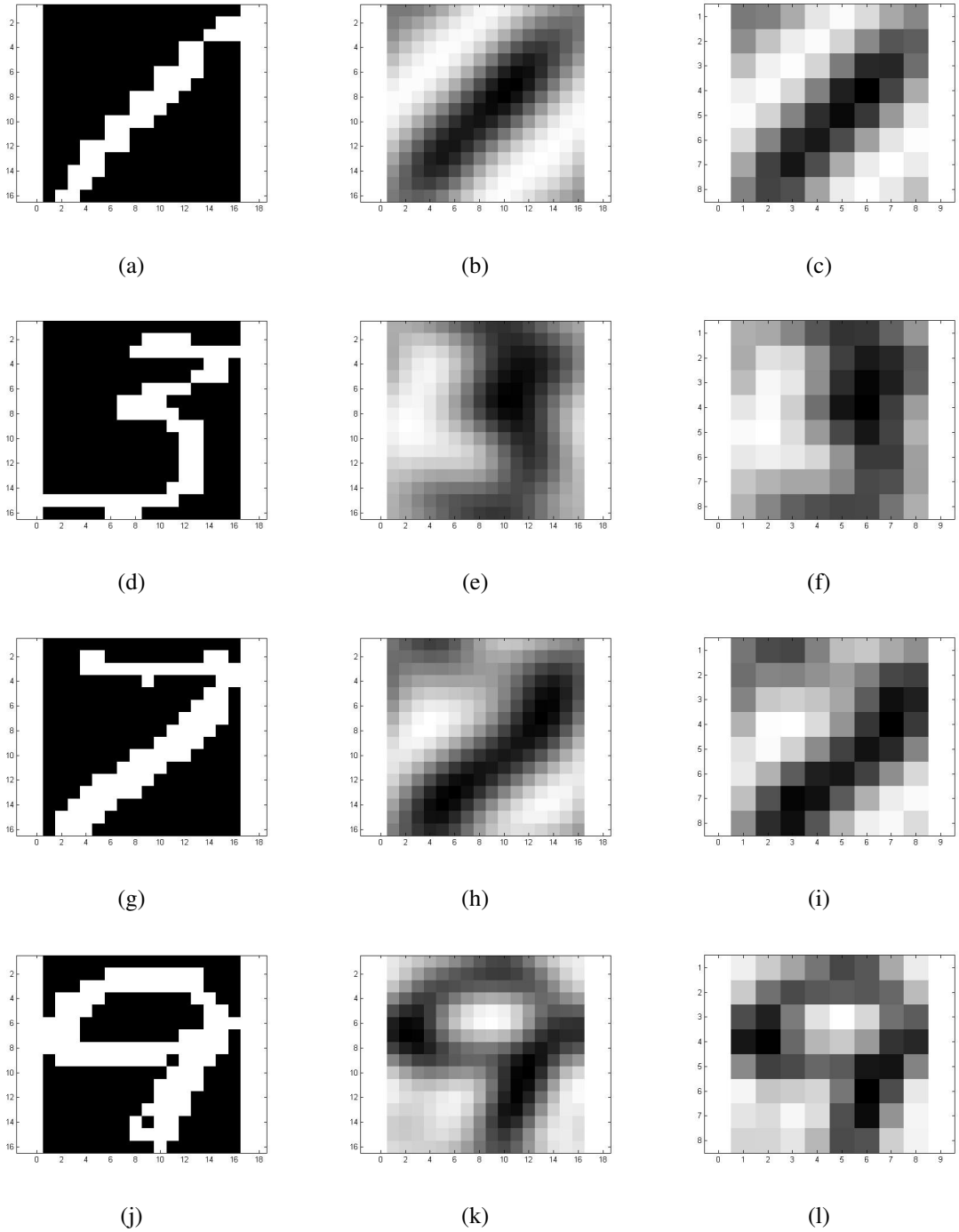
obviar el ángulo θ de la ecuación (2.8). En términos formales, sea $s(x)$ la imagen de un dígito de dimensión $n \times n$, donde, $x = (i, j) \in \Re^2$ y además $1 \leq i, j \leq n$, entonces tenemos:

$$W(i,j) = S_{a_0}(i,j) = a_0^{-1} \sum_{i'=1}^{n} \sum_{j'=1}^{n} \psi(a_0^{-1}(i - i', j - j'))s(i', j') \tag{3.1}$$

donde ψ es la wavelet *Mexican Hat* isotrópica definida en la ecuación (2.13). Luego a partir de W, calculamos R de la siguiente forma:

$$R = W(2i - 1, 2j - 1) \quad \forall i, j / \;\; 1 \leq i, j \leq \frac{n}{2} \tag{3.2}$$

donde R de dimensión $\frac{n}{2} \times \frac{n}{2}$ es lo que realmente se utilizó como entrada de la red neuronal. A continuación se muestran ejemplos de su aplicación tanto para dígitos de la base de datos del CENPARMI como para los de MNIST. Para cada caso se muestra el dígito original, el mismo luego de la aplicación de la representación posicional (i.e. W) y luego del submuestreo (i.e. R).



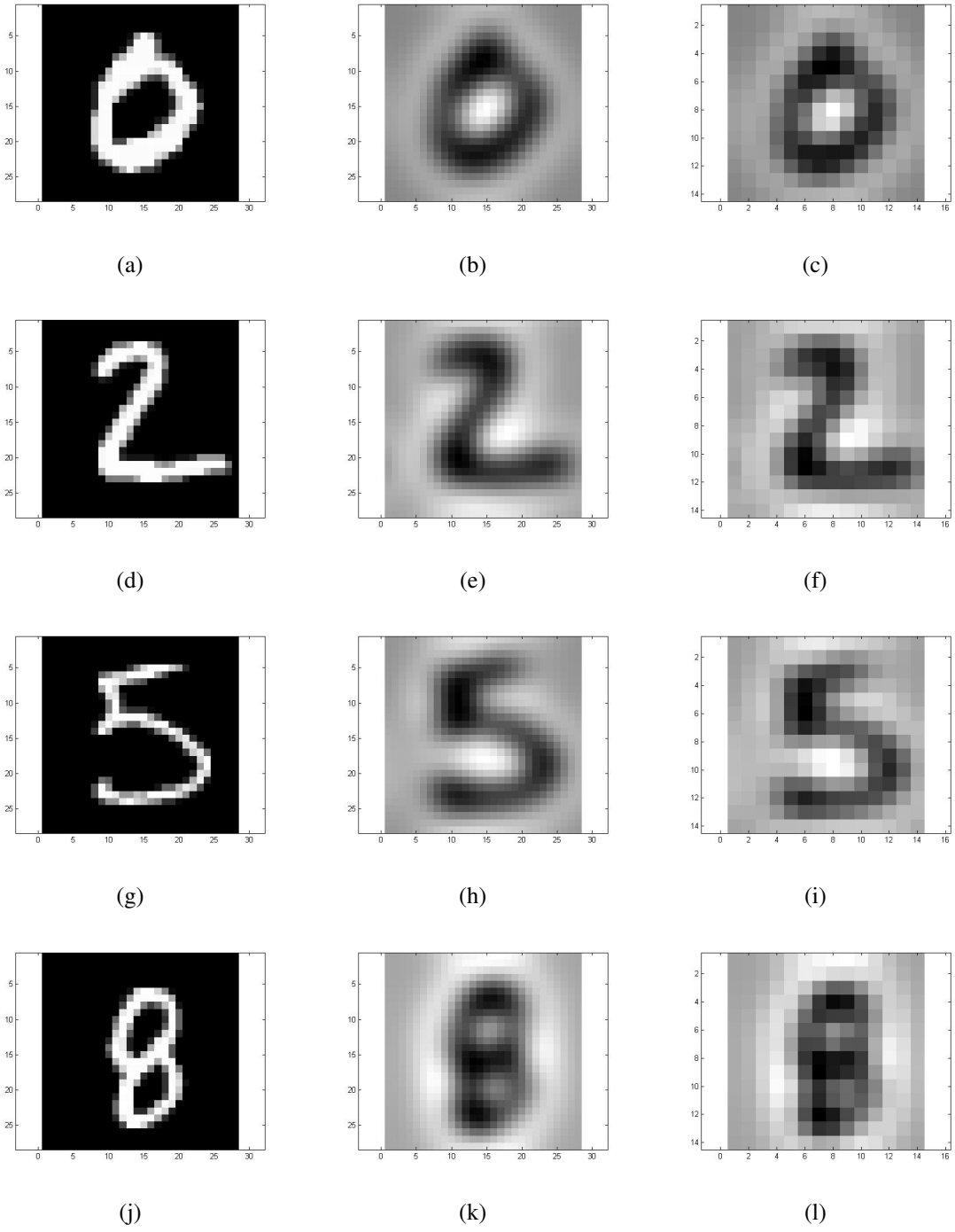
(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Figura 3.1: (a)Imagen original de un uno desde la base de datos de CENPARMI. (b) el mismo uno luego de aplicar representación posicional con *Mexican Hat* ($\epsilon = 1$ y $a_0 = 2{,}5$), y (c) luego del submuestreo. Lo mismo para un tres, un siete y un nueve.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

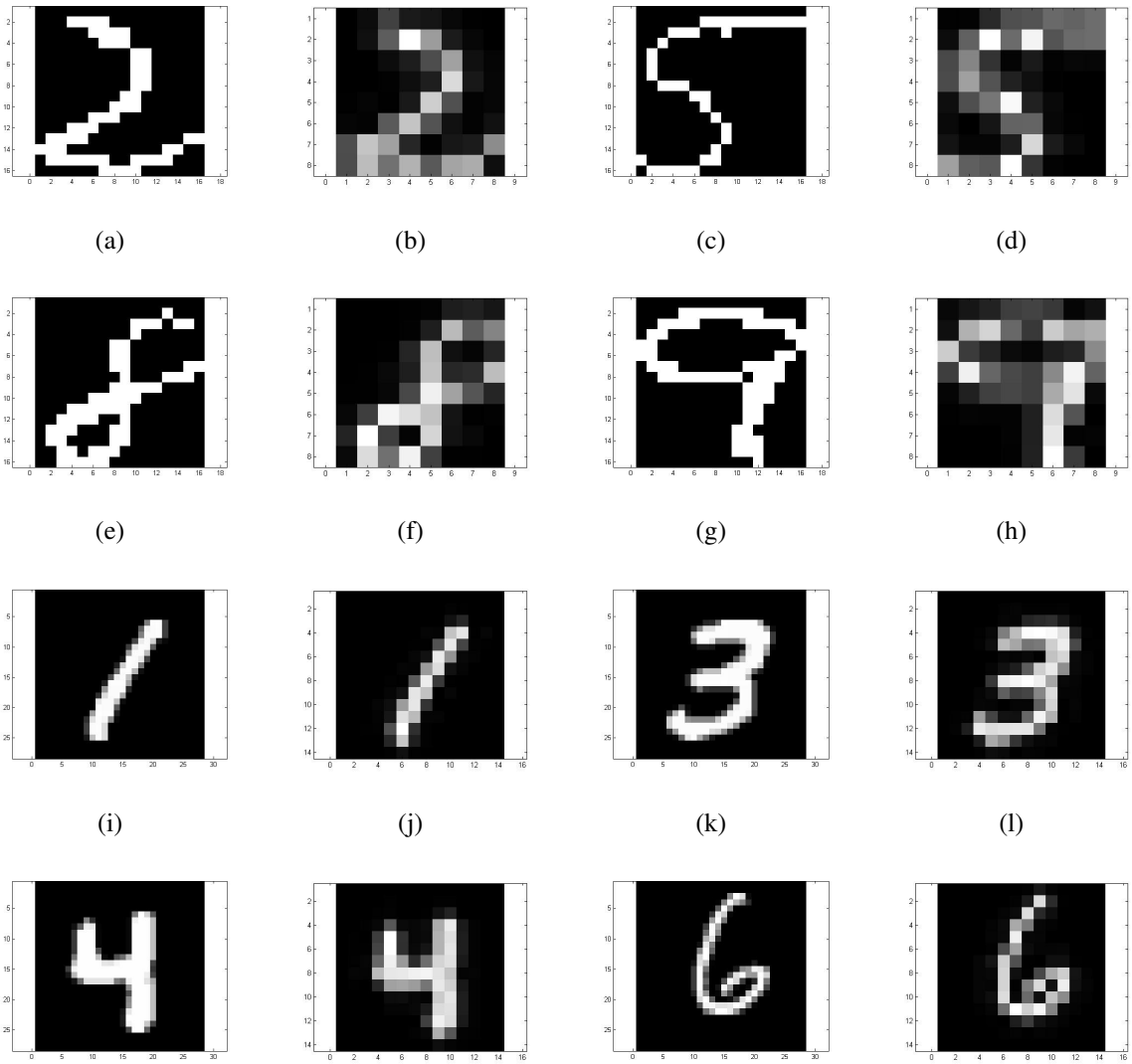
Figura 3.2: (a)Imagen original de un cero desde la base de datos de MNIST. (b) el mismo cero luego de aplicar representación posicional con *Mexican Hat* ($\epsilon = 1$ y $a_0 = 2{,}5$), y (c) luego del submuestreo. Lo mismo para un dos, un cinco y un ocho.

3.2. DWT: coeficientes de aproximación

Como segundo método se utilizó la submatriz LL del primer paso de la transformada wavelet discreta bidimensional con una wavelet biortogonal CDF 9/7. Esto se realizó con los mismos objetivos que se mencionaron en el método anteriormente descripto: Por un lado el hecho de contar con una imagen que preserve la estructura del patrón original. De las cuatros submatrices LL, LH, HL y HH, la primera de ellas es la que contiene mayor información del patrón original ya que se trata de una aproximación burda del mismo. Y por otro lado, la dimensión de cada submatriz luego de un paso de la transformada es la mitad de la imagen original, y esto resulta beneficioso para poder combinar este método con otro sin sobrepasar la dimensión del patrón original. Con respecto a las wavelets biortogonales, las mismas tienen dos conjuntos de funciones de escalas y wavelets: $\Phi(x)$ y $\Psi(x)$ son llamadas funciones de escala y wavelet de análisis, mientras que $\widetilde{\Phi}(x)$ y $\widetilde{\Psi}(x)$ son llamadas funciones de escala y wavelet de síntesis. Las primeras se utilizan para la descomposición de la señal y las segundas para la reconstrucción de la misma. Estas wavelets, en contraste a las ortogonales, tienen la propiedad de ser simétricas y además con ellas se pueden reconstruir perfectamente a la señal original. El cuadro siguiente muestra los coeficientes asociados a las funciones de escala $\Phi(x)$ (coeficientes h_k) y $\widetilde{\Phi}(x)$ (coeficientes h'_k) para el caso de la wavelet CDF 9/7 (*Cohen - Daubechies - Feauveau*):

h_k	h'_k
0,0267	0
−0,0169	−0,0456
−0,0782	−0,0288
0,2669	0,2956
0,6029	0,5575
0,2669	0,2956
−0,0782	−0,0288
−0,0169	−0,0456
0,0267	0

Cuadro 3.1: Coeficientes asociados a las funciones de escala de la wavelet CDF 9/7



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (ñ) (o)

Figura 3.3: (a)Imagen original de un dos de CENPARMI, (b) Coeficientes de aproximación del dos luego de aplicar la DWT con CDF 9/7. Lo mismo para un cinco, un ocho y un nueve. Luego en (i)-(j), (k)-(l), (m)-(n) y (ñ)-(o) se muestra lo mismo pero para un uno, un tres, un cuatro y un seis tomados desde la base de datos del MNIST.

3.3. Vector de características basadas en wavelets

Como tercer método se pensó en la conformación de un vector de características formado por valores calculados a partir de funciones o variantes de la transformada wavelet continua en dos dimensiones. Este vector si bien no preserva la forma y estructura del patrón original fue utilizado con la idea de "ayudar" a clasificar los patrones junto a otro preprocesamiento que sí la respeta. A continuación se describirán las funciones utilizadas.

3.3.1. Definición de características

Las primeras funciones que se describirán son derivadas a partir de lo que se denomina el gradiente wavelet ([19]), el mismo se obtiene calculando la primera derivada de una función gaussiana en ambas variables x, y. Es decir, sea $g(x, y)$ una función gaussiana bidimensional, definimos las siguientes wavelets:

$$\psi_1(x, y) = \frac{\partial g(x, y)}{\partial x} \tag{3.3}$$

y

$$\psi_2(x, y) = \frac{\partial g(x, y)}{\partial y} \tag{3.4}$$

donde $\psi_1(x, y)$ es la primer derivada de $g(x, y)$ con respecto a x y $\psi_2(x, y)$ es la primer derivada con respecto a y. Con esas definiciones el gradiente wavelet es:

$$T_\psi[f](b, a) = [S_{\psi_1}[f](b, a), S_{\psi_2}[f](b, a)] \tag{3.5}$$

donde $S_{\psi_1}[f](b, a)$ es la transformada wavelet continua en dos dimensiones aplicada con la wavelet ψ_1 (Idem $S_{\psi_2}[f](b, a)$ con ψ_2) y $f = f(x, y)$ una señal bidimensional. El valor de T_ψ para cada par (b, a) es un vector cuyas componentes son los respectivos coeficientes de la transformada wavelet usando $\psi_1(x, y)$ y $\psi_2(x, y)$ como wavelets. Observar que el ángulo θ, no está presente en la fórmula ya que el mismo no es relevante cuando la wavelet es la primer derivada de una gaussiana ya que la misma es isotrópica. Dado que $b \in \Re^2$, la función $T_\psi[f](b, a)$ tiene tres variables, luego se fija $a = a_0$ y se obtiene el gradiente wavelet en función

del parámetro de desplazamiento de la misma forma que la representación posicional definida en (2.8):

$$T_{a_0\psi}[f](b) = [S_{a_0\psi_1}[f](b), S_{a_0\psi_2}[f](b)] \tag{3.6}$$

entonces cada elemento de $T_{a_0\psi}[f](b)$ es un vector $v = (S_{a_0\psi_1}[f](b), S_{a_0\psi_2}[f](b))$, el cual se denomina vector gradiente wavelet. En la figura 3.4 se muestra un ejemplo, en el mismo se puede ver el módulo, las componentes sobre los ejes x, y y los ángulos (fase) de los vectores gradiente wavelets, o sea, el ángulo que cada vector del gradiente wavelet forma con el eje horizontal.

A partir del gradiente wavelet se calcularon y se utilizaron las siguientes funciones (características):

- Suma de cuadrados de los módulos del gradiente wavelet:
  Definimos el módulo del gradiente wavelet como

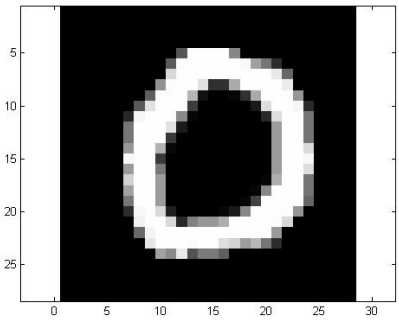
  $$M_{a_0\psi}[f](b) = |T_{a_0\psi}[f](b)| = [(S_{a_0\psi_1}[f](b))^2 + (S_{a_0\psi_2}[f](b))^2]^{1/2} \tag{3.7}$$

  Luego la suma de cuadrados es un número que se calcula como

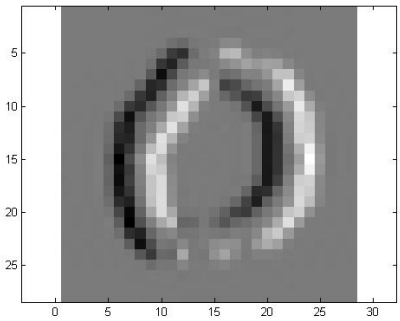
  $$sc = \sum (M_{a_0\psi}[f](b))^2 \tag{3.8}$$

- Entropía del histograma de los ángulos del gradiente wavelet:
  Como se mencionó anteriormente, cada elemento de $T_{a_0\psi}[f](b)$ es un vector $v = (S_{a_0\psi_1}[f](b), S_{a_0\psi_2}[f](b))$, definimos $A_{a_0\psi}[f](b)$ como la función cuyos coeficientes son los ángulos que forman cada vector v con el eje horizontal. Esa función, por ejemplo, es la que se muestra en la figura 3.4 (e) y como se puede ver es una imagen en dos dimensiones. A dicha imagen se le calcula el histograma y por último para cuantificar la dispersión del histograma se calcula la entropía. Ese valor resultante es el que se utiliza como característica.

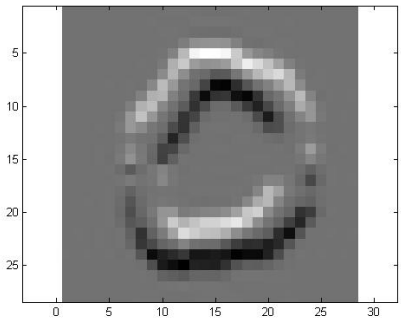
- Entropía del histograma de los módulos del gradiente wavelet:
  Esta característica es similar a la anterior, sólo que el histograma se calcula a partir del módulo del gradiente wavelet. Es decir, primero se obtiene $M_{a_0\psi}[f](b)$, luego se calcula el histograma y por último se obtiene la entropía de ese histograma.



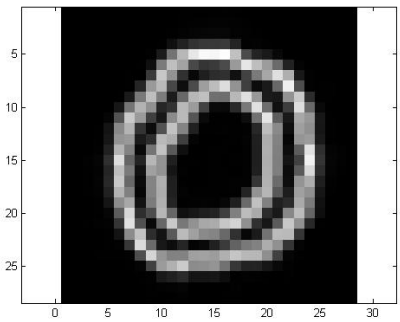
(a)



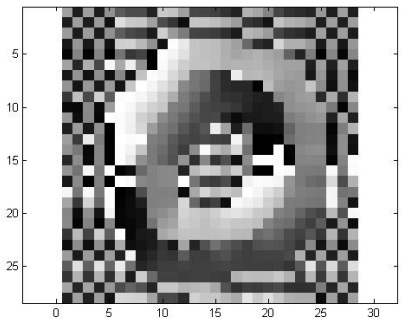
(b)



(c)



(d)



(e)

Figura 3.4: Ejemplo del cálculo del vector gradiente para un cero de MNIST con $a_0 = 1$. (a) Imagen Original (b) Componente x del vector gradiente o el resultado de la transformada $S_{a_0\psi_1}[f](b)$, (c) Componente y del vector gradiente o el resultado de la transformada $S_{a_0\psi_2}[f](b)$, (d) Módulo de cada vector gradiente wavelet y (e) Ángulo (fase) de cada vector gradiente wavelet

- Curvatura 2D:
  La curvatura 2D es una medida que indica cómo varian localmente los vectores gradiente. Por ejemplo, si consideramos un rectángulo, los vectores gradiente no varian, es decir, su curvatura es cero ya que los vectores irán siempre en dirección perpendicular a algún lado del mismo. Se define la curvatura 2D, para una señal f como,

$$k = \nabla . \frac{\nabla f}{\|\nabla\|} = \frac{f_{xx} f_y^2 - 2 f_x f_y f_{xy} + f_{yy} f_x^2}{(f_x^2 + f_y^2)^{3/2}} \tag{3.9}$$

  donde f_x, f_y, f_{xx}, f_{yy} y f_{xy} hacen referencia a las primeras derivadas parciales de f con respecto a x y a y, a las se segundas derivadas parciales de f con respecto a x y a y, y a las derivadas parciales con respecto a x y a y, respectivamente. Luego tomando que $f_x = S_{a_0\psi_1}[f](b)$ y $f_y = S_{a_0\psi_2}[f](b)$, las demás derivadas parciales se pueden obtener en base a éstas y por ende tener una curvatura 2D del gradiente wavelet. Además de la curvatura 2D del gradiente también se utilizó la curvatura de la imagen original del dígito, la misma se calcula aproximando las derivadas parciales directamente desde la imagen original del dígito sin utilizar la transformada.

Hasta aquí se han descripto las características derivadas desde el gradiente wavelet. A continuación se mostrarán las características derivadas a partir de la utilización de representación posicional vista en (2.8) y la densidad escala - ángulo ya mencionada en (2.24). En ambas se utilizó como wavelet la *Mexican Hat* en su versión anisotrópica definida en la ecuación (2.15). Observar que el objetivo del empleo de dicha wavelet es contar con la posibilidad de utilizar el ángulo θ de la fórmula de la transformada y el parámetro ϵ de dicha wavelet, de esta forma al tenerlos como variables nos da la posibilidad de obtener más componentes para el vector de características combinadas.

- Suma de cuadrados de representación posicional:
  Sea

$$S_{a_0\theta_0}(bx, by) = a_0^{-1} \int_{\Re^2} \psi_{\epsilon_0}(a_0^{-1}\, r_{-\theta_0}(b - x))\, s(x)\, dx \tag{3.10}$$

  la representación posicional calculada para la imagen $s(x)$ con la wavelet *Mexican Hat* anisotrópica ψ_{ϵ_0} con $\epsilon = \epsilon_0$, con un ángulo fijo θ_0 y una escala fija a_0, luego la suma de cuadrados se obtiene como:

$$sc = \sum (S_{a_0\theta_0}(bx, by))^2 \tag{3.11}$$

- Entropía del histograma de módulos de representación posicional:
  Para esta característica calculamos el módulo de la representación posicional (i.e. $|S_{a_0\theta_0}(bx, by)|$), luego calculamos el histograma y por último la entropía.

- Entropía de la densidad escala - ángulo :
  Para la obtención de esta característica se calcula la densidad escala ángulo definida en (2.12) para un número finito de escalas y un número finitos de ángulos, es decir,

$$E(a_i, \theta_j) = \int_{\Re^2} |S(a_i, \theta_j, b)|^2 d^2b \tag{3.12}$$

  con $i \leq 1 \leq n$ y $j \leq 1 \leq m$. Notar que el resultado es una imagen (o una matriz). Luego de esta imagen se toma la fila de mayor energía, o sea, la fila cuya suma de sus coeficientes es máxima. Por último se calcula la entropía de esa fila, y ese número es el utilizado como característica.

En la figura 3.5 se pueden ver gráficos del tipo x, y donde se comparan algunas características entre sí. Es decir, cada punto del gráfico queda determinado por el cómputo de las características enfrentadas, esto es,

$$(x, y) = (C_i(A), C_j(A)) \tag{3.13}$$

donde C_i, C_j son algunas de las características definidas anteriormente y A representa a la imagen de un dígito. Las mismas se computan para 100 representantes de una clase de dígito y también para 100 de otra clase de dígito distinta a la primera. Si bien las características observadas no separan totalmente los patrones, se observa que hay cierto grado de separación entre los dígitos: se forman nubes de puntos concentradas en distintos lugares del plano.

3.3.2. Conformación del vector de características

En base a las características definidas anteriormente, se procede a mostrar la conformación del vector de características utilizado. El mismo tiene 85 valores y su estructura es la siguiente:

- Componente 1: Se utilizó la curvatura 2D sobre la imagen original.

- Componentes 2 a 5: Se utilizó la suma de cuadrados de los módulos del gradiente con escalas $a_0 = \{1; 1{,}5\}$ y por cada a_0 se calculó $M_{a_0\psi}[f](b)$ sin umbral ($u = 0$) y con un

umbral $u = 0{,}3$. El umbral se aplicó como un porcentaje del valor máximo de $M_{a_0\psi}[f](b)$, es decir, sea $m = \max\{M_{a_0\psi}[f](b)\}$, los valores que se tuvieron en cuenta fueron aquellos que eran mayores o iguales que el producto $u.m$

- Componentes 6 a 13: Aquí se utilizó la entropía del histograma de los ángulos del gradiente. Se calcularon 8 variantes de $A_{a_0\psi}[f](b)$ acorde a la terna (a_0, u, q) donde, $a_0 = \{1; 1{,}5\}$, $u = \{0; 0{,}3\}$ y $q = \{10; 15\}$. La escala y el umbral fueron aplicados al cálculo de $T_{a_0\psi}[f](b)$ y luego el valor de q fue aplicado a $A_{a_0\psi}[f](b)$, el mismo es un paso de cuantización aplicado de la siguiente forma:

$$A_{a_0\psi}[f](b) = \left[\frac{A_{a_0\psi}[f](b)}{q}\right].q \tag{3.14}$$

- Componentes 14 a 17: Se utilizó la entropía del histograma de los módulos del gradiente. Se calcularon cuatro variantes de $M_{a_0\psi}[f](b)$ acorde a la dupla (a_0, u) con $a_0 = \{1; 1{,}5\}$, $u = \{0; 0{,}3\}$.

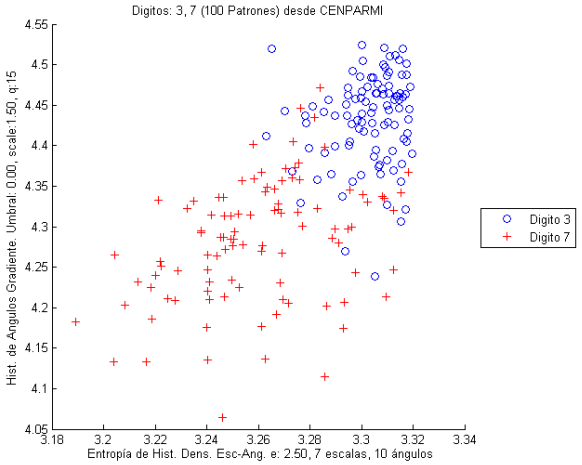
- Componentes 18 a 21: Curvatura 2D calculada en base al gradiente wavelet. Acá también se obtuvieron cuatro variantes de la curvatura con la dupla (a_0, u) con $a_0 = \{1; 1{,}5\}$, $u = \{0; 0{,}3\}$.

- Componentes 22 a 51: Aquí se utilizó la suma de cuadrados de la representación posicional. El cálculo de la representación posicional $S_{a_0\theta_i}(bx, by)$ se realizó en base a la dupla (a_0, θ_i) con $a_0 = \{0{,}8; 1; 1{,}8\}$ y $\theta_i = i\frac{\pi}{9}$ con $0 \leq i \leq 9$. Además el parámetro de excentricidad de la wavelet *Mexican Hat* fue de 2.5 (i.e. $\epsilon = 2{,}5$).

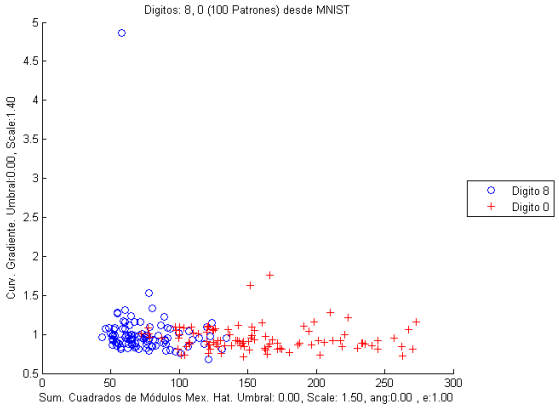
- Componentes 52 a 81: Se utilizó la entropía del histograma de módulos de representación posicional y el cálculo de $S_{a_0\theta_i}(bx, by)$ se realizó con los mismos valores de (a_0, θ_i) mencionados en el caso anterior.

- Componentes 82 a 85: Por último, se utilizó la entropía de la densidad escala - ángulo. La densidad escala - ángulo se calculó en cuatro oportunidades:

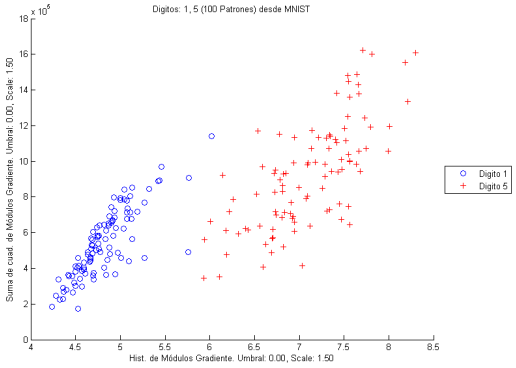
  - Con un vector de escalas $a_i = \{0{,}8; 1{,}2; 1{,}6; 2; 2{,}5; 2{,}8; 3{,}2\}$, un vector de ángulos $\theta_i = i\frac{\pi}{9}$ con $0 \leq i \leq 9$, y cada vez calculada con un valor de excentricidad tomado de $\epsilon = \{2{,}5; 3{,}5\}$
  - Con el mismo vector de escalaes a_i que en el caso anterior, un vector de ángulos $\theta_i = i\frac{\pi}{12}$ con $0 \leq i < 12$ y, también, calculada cada vez con un valor de excentricidad tomado de $\epsilon = \{2{,}5; 3{,}5\}$



(a)



(b)



(c)

Figura 3.5: Gráficos comparativos de características. (a) *Entropía de la densidad escala - ángulo* versus *entropía del histograma de ángulos del gradiente* wavelet para 100 representantes del dígito 3 y 7 tomados desde CENPARMI. (b) *Suma de cuadrados de representación posicional* versus *Curvatura 2D del gradiente wavelet* para 100 representantes del dígito 8 y 0 tomados desde MNIST. (c) *Suma de cuadrados de los módulos del gradiente wavelet* versus *entropía el histograma de los módulos del gradiente* para 100 representantes del dígito 1 y 5 tomados desde MNIST

Capítulo 4

Sistema de Reconocimiento Basado en Redes Neuronales

4.1. Motivación

Las redes neuronales artificiales (RNA) ofrecen una alternativa frente al software tradicional para la resolución de ciertas problemáticas, donde el desarrollo de programas es reemplazado por el desarrollo de arquitecturas de red apropiadas y algoritmos de entrenamiento que permiten la adaptación del rendimiento de la red a un problema específico.

El reconocimiento de dígitos manuscritos ha sido tratado amplia y exitosamente utilizando RNA, en especial las del tipo *feed forward* multicapa (también denominadas Perceptrón multicapa) entrenadas con el algoritmo de *backpropagation* (retropropagación). Esta arquitectura ha sido reconocida como una herramienta poderosa para la resolución del problema de la clasificación de patrones, dado que su potencialidad radica en su poder discriminativo, y en su capacidad de aprender y representar conocimiento implícito. Como antecedentes, podemos mencionar los siguientes trabajos relacionados: en [20] utilizan como métodos de preprocesamiento la aplicación de un paso de la DWT con las bases CDF 2/2, CDF 2/4, CDF 3/3 y CDF 3/7. Como clasficador emplean un cluster de 4 redes neuronales con una capa oculta cada una, a su vez cada red neuronal toma como entrada una submatriz (LL, LH, HL y HH) producto de la descomposición de la transformada wavelet. Los dígitos son provenientes de la base de datos de CENPARMI y también los normalizan a 16×16. De esta forma, el tamaño de la entrada total a la red es de 16×16. El mejor resultado que obtienen es del orden de los 94.7 %. En [21]

combinan 3 redes neuronales de una capa oculta cada una donde la primera tiene como entrada el patrón original del dígito normalizado a 32×32, la segunda la submatriz LL del primer paso de la DWT utilizando la wavelet *Daubechies-4* y la tercera la submatriz LL del segundo paso de la DWT con la misma wavelet ambas aplicadas al dígito normalizado de 32×32. Con este esquema, la clasificación de un dígito la obtienen combinando las clasificaciones individuales de cada red neuronal. Para realizar dicha combinación aplican los siguientes métodos: regla de la suma (*sum rule*), regla del producto (*product rule*), voto por mayoría (*majority voting*) y voto por mayoría ponderado (*weighted majority voting*). Luego evaluan cuál es la mejor estrategia resultando ser la del voto por mayoría ponderado. Una de las bases de datos con la que trabajan es MNIST y obtienen un resultado del orden de los 98.04 % sobre ella.
En general, las redes neuronales han demostrado ser eficientes en tareas de clasificación: su uso ha demostrado ser una opción viable en términos de confiabilidad y tiempo de ejecución ya que una vez entrenada la misma es capaz de reconocer patrones malformados o con ruido. En este capítulo se darán las definiciones básicas para comprender la arquitectura de un perceptrón multicapa entrenada con el algoritmo de *backpropagation* ya que dicha arquitectura fue la que se eligió para implementar el sistema de reconocimiento.

4.2. Nociones básicas

4.2.1. Modelo de una neurona

La definición de una neurona será la base para definir una red neuronal en términos matemáticos. El modelo de una neurona artificial es una imitación simplificada de una neurona biológica, la misma puede verse como una unidad de procesamiento de información la cual es fundamental para el diseño y funcionamiento de una red neuronal. En la figura 4.1 se puede ver el modelo de una neurona. Allí se pueden identificar los tres elementos básicos ([22]):

- Un conjunto de sinapsis o conexiones, cada uno de los cuales tiene asociado un peso del conjunto $w_{k1}, ..., w_{km}$, a éstos valores se los denomina pesos sinápticos. Es decir, cada valor de la señal de entrada x_j será multiplicado, en la neurona k, por el peso sináptico w_{kj}. Los pesos sinápticos podrán ser valores positivos como negativos.

- Un sumador, que suma el producto de cada valor de entrada con su correspondiente peso sináptico.

- Una función de activación la cual limita la amplitud o el rango del valor de salida de la neurona. Típicamente el rango de salida de una neurona es de $[0, 1]$ ó $[-1, 1]$.

El modelo también incluye un umbral o bias denotado por b_k. En términos matemáticos se puede definir a una neurona k a través de las siguientes ecuaciones:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{4.1}$$

y

$$y_k = \varphi(u_k + b_k) \tag{4.2}$$

donde $x_1, ..., x_m$ son los valores de la señal de entrada; $w_{k1}, ..., w_{km}$ son los pesos sinápticos de la neurona k; u_k es la suma del producto de los pesos sinápticos y los valores de la señal de entrada; b_k es el bias; $\varphi(\cdot)$ es la función de activación; e y_k es el valor de salida de la neurona.

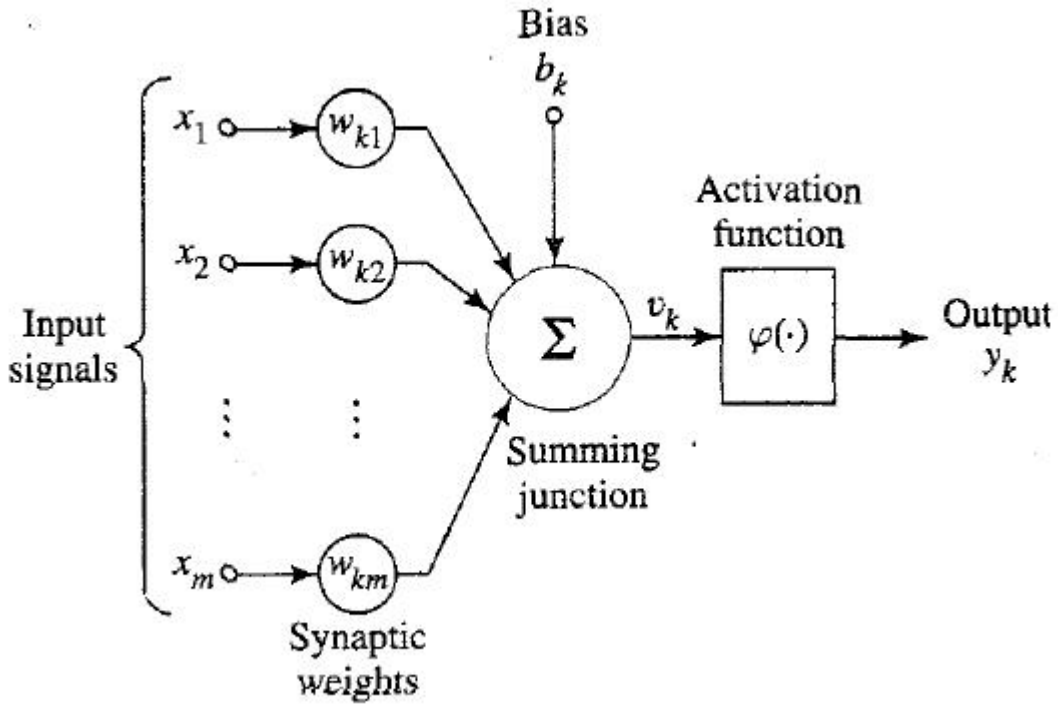


Figura 4.1: Modelo de una neurona, donde *Input signals:* señales de entrada, *Synaptics weights:* pesos sinápticos, *Activation function:* función de activación y *Output*: salida.

4.2.1.1. Tipos de funciones de activación

La función de activación es la que determina que una neurona sea activa o pasiva de acuerdo a su valor de salida. A continuación se muestran dos ejemplos:

- Umbral (*Threshold function*), el gráfico de este tipo de funciones se puede ver en 4.2, la misma viene dada por

$$\varphi(v) = \begin{cases} 1, & \text{si } v \geq 0 \\ 0, & \text{si } v < 0 \end{cases} \tag{4.3}$$

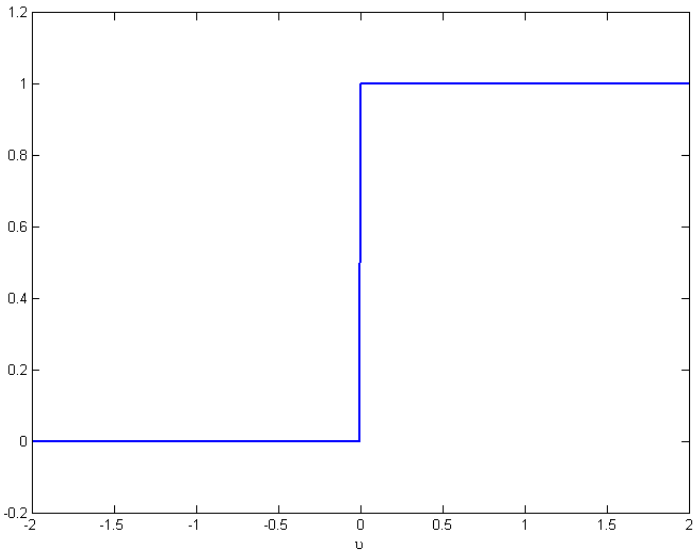


Figura 4.2: Gráfico de la función de activación de tipo umbral

  o sea que si la salida de la neurona es mayor o igual que cero entonces el resultado será de 1 caso contrario valdrá 0. Una neurona con tal función de activación es conocida en la literatura como el modelo de neurona de *McCulloc-Pitts*, en el cual el comportamiento de *todo o nada* se hace evidente a partir de la definición de la función.

- Logística sigmoidea (*Logistic function*), su gráfico puede verse en la figura 4.3 y se define de la siguiente forma:

$$\varphi(v) = \frac{1}{1 + e^{-\alpha v}} \tag{4.4}$$

mientras que la función de tipo umbral devuelve 0 ó 1, el rango de ésta es todo el intervalo continuo $[0, 1]$, además la misma es diferenciable. El parámetro α es un parámetro de inclinación también llamado $slope$, observar que cuando el mismo tiende a infinito la función se convierte en la función umbral.

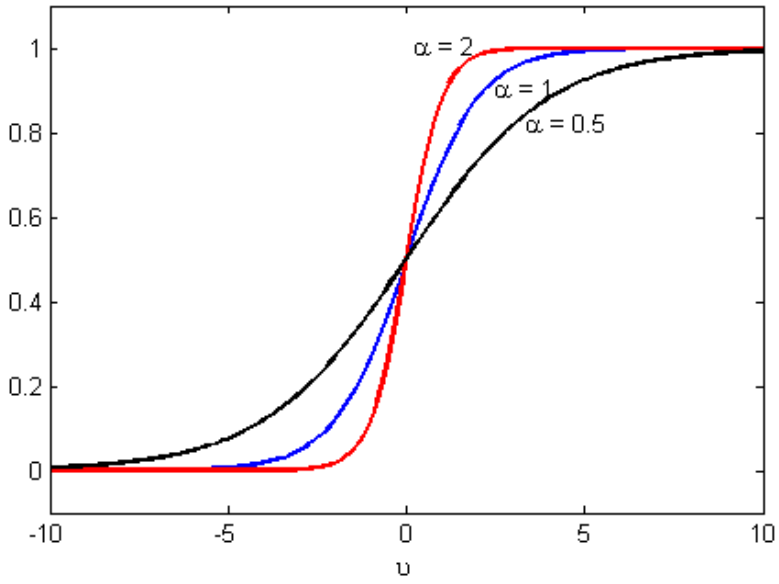


Figura 4.3: Gráfico de la función de activación de tipo Logística sigmoidea para $\alpha \in \{0{,}5; 1; 2\}$

4.2.2. Perceptrón

El primer modelo exitoso de redes neuronales fue el del perceptrón. Dicho modelo está basado fuertemente en el modelo de neurona visto anteriormente utilizando como función de activación la del tipo umbral (modelo de neurona de *McCulloc-Pitts*). El perceptrón tiene una capa de entrada (los valores de entrada) y una capa de salida (el valor de la función de activación). Como se puede ver en la ecuación 4.1, el producto de los pesos sinápticos y los valores de entrada se suman y a ese resultado se le suma el bias, a continuación se aplica la función de activación umbral, si el resultado es positivo obtenemos un valor de 1, caso contrario obtenemos un valor de 0. Este flujo se puede ver en la figura 4.4, donde $w_1, ..., w_m$ son los pesos sinápticos del perceptrón, $x_1, ..., x_m$ son sus correspondientes valores de entrada y el bias es denotado por b. La entrada neta a la neurona, denotada como v, se obtiene a partir de:

$$v = \sum_{i=1}^{m} w_i x_i + b \tag{4.5}$$

y por último se aplica a v la función umbral definida en 4.3 (denotada por $\varphi(\cdot)$). El objetivo del perceptrón es clasificar $x_1, ..., x_m$ correctamente en la clase $\mathcal{C}_1$ cuando $\varphi(v)$ es positivo o en la clase $\mathcal{C}_2$ en caso contrario. Una técnica útil para para analizar el comportamiento de este tipo de redes es la dibujar un mapa de las regiones de decisión creadas en el espacio bidimensional. Estas regiones de decisión especifican cuáles son los valores de entrada que resultan en la clase $\mathcal{C}_1$ y cuales en la clase $\mathcal{C}_2$. El perceptrón forma dos regiones separadas por un hiperplano definido por:

$$\sum_{i=1}^{m} w_i x_i + b = 0 \tag{4.6}$$

Como se muestra en la figura 4.5, existen problemas que no son linealmente separables lo que implica que no se pueden resolver con el perceptrón de una capa con función de activación umbral. Según el tipo de problema podrá resolverse con un perceptrón con función de activación logística, por ejemplo (en este caso los patrones a clasificar deberán ser linealmente independientes), o utilizando un perceptrón multicapa, modelo orientado a problemas de clasificación complejos, el cual se detallará más adelante.

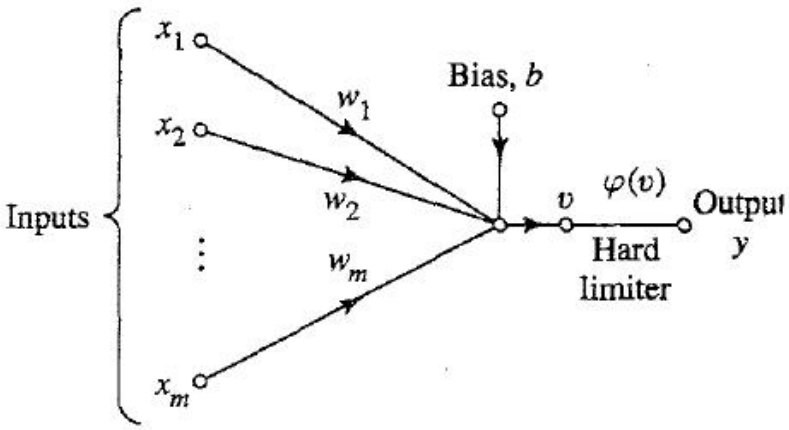


Figura 4.4: Flujo del perceptrón

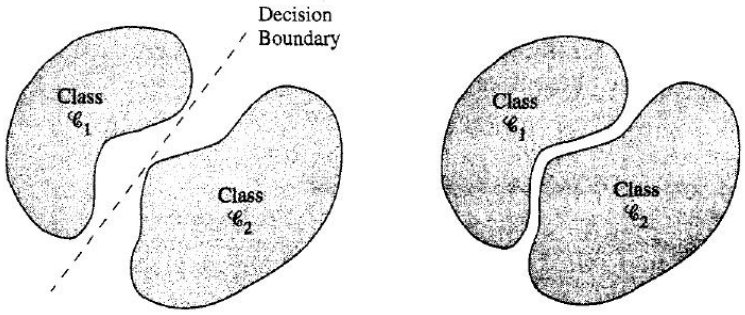


Figura 4.5: Separabilidad lineal

4.2.2.1. Regla de aprendizaje

Para poder encontrar el hiperplano que separa las dos regiones en que serán clasificados los patrones en el caso que estos sean linealmente separables, es necesario utilizar un algoritmo de aprendizaje que ajuste los pesos sinápticos hasta encontrar los valores apropiados. Para esto, se inicializan los w_j y los bias b_j en forma aleatoria. Llamemos ς a la salida esperada (o clase a la que pertenece el patrón) de la red para el patrón $\mathbf{x} = (x_1, ..., x_m)$ y $\mathbf{w} = (w_1, ..., w_m)$ al vector de pesos. Entonces para cada patrón presentado a la red se calcula el valor de salida y y se lo compara con ς, si son distintos entonces se aplica la siguiente regla:

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \eta\mathbf{x}(y - \varsigma) \tag{4.7}$$

donde $\mathbf{w}(t)$ hace referencia al vector de pesos en la iteración t, η es el coeficiente de velocidad de aprendizaje y pertenece al intervalo $(0, 1]$. Dicha regla es conocida como la regla de aprendizaje del Perceptrón, y converge a la solución (si existiera) en un número finito de pasos.

4.2.3. Perceptrón multicapa

Como se mencionó anteriormente, el perceptrón multicapa (MLP - *Multilayer Perceptron*) permite el tratamiento de problemas de clasificación complejos como lo es el reconocimiento de dígitos manuscritos. Típicamente la arquitectura de este tipo de redes consiste en una capa de entrada, una o más capas ocultas y una capa de salida, como lo muestra la figura 4.6. De esta forma, los valores ingresados en la capa de entrada se van propagando hacia adelante, capa por capa, hasta llegar a la capa de salida y producir una respuesta.

La arquitectura de un perceptrón multicapa presenta las siguientes características:

- Cada modelo de neurona utiliza una función de activación diferenciable (Logística sigmoidea por ejemplo).
- La red contiene una o más capas ocultas.
- La red presenta un alto grado de conectividad, determinado por los sinapsis.

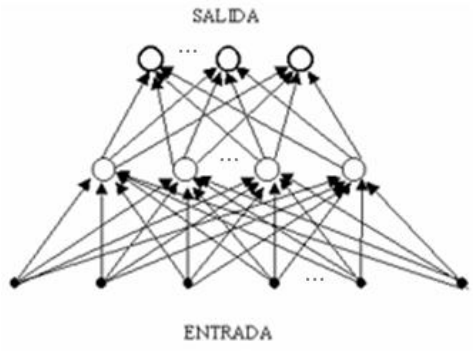


Figura 4.6: Ejemplo de una arquitectura de un perceptrón multicapa con una capa oculta

4.2.3.1. Backpropagation

Considerado una generalización del perceptrón simple, el MLP es entrenado con el algoritmo de retropropagación del error ampliamente conocido como *backpropagation* ([22] y [23]), dentro del paradigma de aprendizaje supervisado el cual implica la utilización de la salida esperada como parte del conjunto de entrenamiento.
La idea es la siguiente: se presentan a la red el patrón de entrada y el patrón de salida deseado, luego en base a la salida real se calcula la diferencia contra la salida esperada y se van ajustando

los pesos de la red de tal manera que esa diferencia vaya disminuyendo cada vez. Formalmente lo que se tiene es una función de costo definida en base a los pesos de la red y lo que se intenta hacer es minimizarla de forma iterativa. Para definir la función de costo adecuadamente tenemos lo siguiente para una iteración t:

- w_{ij} sinapsis que conecta a la neurona i de la capa m con la neurona j de la capa $m-1$
- $\upsilon_i = \sum_{j \in J} w_{ij} y_j$ entrada neta a la neurona i donde J incluye a todas las neuronas de la capa anterior a la neurona i
- $y_i = \varphi(\upsilon_i)$ salida de la neurona i, donde φ es la función de activación. Si y_i está en la capa de entrada de la red entonces toma los valores del patrón ingresado.
- ζ_i salida esperada en la neurona i (forma parte de la clase a la que pertenece un patrón)
- O_i salida real para la neurona i
- $E(t) = \frac{1}{2} \sum_{i \in C} (\zeta_i - O_i)^2$ donde C incluye a todas las neuronas de la capa de salida de la red, es el error cuadrático medio en la iteración t

En base a esto la la función de costo queda determinada por:

$$E(w) = \frac{1}{N} \sum_{\mu \in P, i \in C} (\zeta_i^\mu - O_i^\mu)^2 \tag{4.8}$$

donde C incluye a todas las neuronas de la capa de salida de la red y P incluye a todos los patrones que conforman el conjunto de entrenamiento. Notar que O_i es de la forma $\sum_{j \in J} w_{ij} y_j$ luego es correcto que E esté definida en función del vector de pesos w. La misma es una función diferenciable y tiene un mínimo absoluto. Para minimizarla se utiliza la técnica del descenso del gradiente, la cual consiste en alterar los pesos en la dirección que produce el máximo descenso en la superficie de error. La dirección de cambio se obtiene mediante el gradiente (derivadas parciales de la función de costo con respecto a los pesos) ya que el mismo especifica la dirección que produce el máximo incremento, por lo que el mayor descenso es el negativo de esa dirección.

Luego, los pesos se actualizan de acuerdo a la siguiente regla de aprendizaje, denominada regla delta:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t) \tag{4.9}$$

$$\Delta w_{ij}(t) = -\eta \frac{\partial E(t)}{\partial w_{ij}(t)} \tag{4.10}$$

donde η es el coeficiente de velocidad de aprendizaje.

La regla delta de aprendizaje también se aplica en el entrenamiento del perceptrón simple con unidades asociadas a valores continuos y a funciones de activación diferenciables y utiliza la técnica de descenso por gradiente para encontrar el mínimo de la función de costo. En función de este objetivo, el valor de la velocidad de aprendizaje debe ser elegido en forma adecuada ya que la convergencia de la red es muy susceptible a su valor: si η es un valor muy alto se pueden producir oscilaciones en la convergencia de la red pero por otro lado si el mismo es un valor pequeño la convergencia puede tornarse muy lenta.

En el algoritmo de *backpropagation*, la regla de aprendizaje puede aplicarse teniendo en cuenta dos alternativas:

- *batch*, el ajuste de los pesos se realiza luego que se hayan presentado todos los patrones del conjunto de entrenamiento.
- *estocástico*, el ajuste de los pesos se realiza incrementalmente, es decir, por cada patrón presentado a la red y elegido en un orden aleatorio.

La alternativa estocástica utiliza la estimación del gradiente basada en el gradiente local, y permite una amplia exploración de la superficie de costo, lo que es beneficioso a la hora de evitar los mínimos locales. Esta aproximación es la que utilizamos para implementar las pruebas descriptas en el siguiente capítulo, y es la que en general reporta mejores resultados.

4.2.3.2. Regla delta generalizada

Como ya hemos mencionado, la elección de un valor para la velocidad de aprendizaje es una tarea delicada, ya que la convergencia del método depende en cierta forma de este parámetro. Un valor muy pequeño para η ocasionará un ajuste menor en los pesos sinápticos resultando en una convergencia que puede tornarse muy lenta. El incremento de la velocidad podría causar grandes cambios en los pesos sinápticos resultando en una red inestable donde la trayectoria buscando el mínimo fuese oscilatoria.

Un método sencillo que permite incrementar la velocidad de aprendizaje evitando el peligro de la inestabilidad, consiste en considerar el parámetro de inercia o *momentum* α, agregando un término a la regla delta mencionada en (4.10)

$$\Delta w_{ij}(t) = -\eta \frac{\partial E(t)}{\partial w_{ij}(t)} + \alpha \Delta w_{ij}(t-1) \tag{4.11}$$

esta regla se denomina *regla delta generalizada*, para $0 \leq \alpha < 1$. Notar que la regla delta es un caso particular para $\alpha = 0$. La idea de agregar el término de momento, se basa en hacer que el cambio realizado en cada peso tienda a ser en la dirección de descenso promedio: tiene un efecto estabilizador cuando las direcciones en iteraciones consecutivas oscilan en signo y acelera la velocidad efectiva cuando la superficie de costo es plana. Otra ventaja de utilizar el coeficiente de momento es la de saltear mínimos locales durante el proceso de aprendizaje ya que el método de descenso por gradiente garantiza alcanzar un mínimo de la función de error, pero no necesariamente el absoluto.

Por otro lado, durante el entrenamiento el parámetro η puede permanecer constante o no: la incorporación del concepto de velocidad de aprendizaje adaptativa, también contribuye a una convergencia más rápida evitando mínimos locales. La idea general consiste en mantener un valor de η tan alto como sea posible mientras que el aprendizaje sea estable (i.e. no oscilante). Si el error se va decrementando en iteraciones consecutivas entonces se aumenta el valor de η en un factor η_{inc} y se actualizan los pesos, pero si el error se incrementa entonces el valor de η se decrementa en un factor de η_{dec} y se descarta la actualización de pesos en esa iteración.

Finalmente, el modelo de MLP entrenado con el algoritmo de *backpropagation* estocástico con parámetro de momento y velocidad de aprendizaje adaptativa es el que hemos utilizado para la realización de las distintas pruebas en el contexto de este trabajo.

Capítulo 5

Resultados

En este capítulo se mostrarán los resultados obtenidos luego de clasificar imágenes de dígitos de las bases de CENPARMI y MNIST preprocesados con los métodos descriptos anteriormente y utilizando una red neuronal entrenada para tal fin. Cada método de preprocesamiento fue evaluado por separado y luego se realizaron combinaciones que también fueron evaluadas a través del entrenamiento y testeo de la red neuronal utilizada como clasificador. Para cada método (y para cada combinación) se trabajó con una determinada arquitectura de red ajustada y optimizada experimentalmente. El tipo de red utilizada en todos los casos fue el del perceptrón multicapa con las siguientes características:

- 1 sola capa oculta, es decir, con n neuronas en la capa de entrada, m neuronas en la capa oculta y 10 neuronas en la capa de salida (i.e. $n \times m \times 10$). Notar que n es la dimensión de la entrada a la red y coincide con la dimensión del patrón preprocesado por cada método (o combinación) utilizado. A su vez, 10 es la cantidad de clases de dígitos distintas a reconocer. La cantidad m fue determinada empíricamente para cada método de preprocesamiento seleccionando áquel que arrojó el mejor resultado.

- La red fue entrenada con el algoritmo de *backpropagation* estocástico con momento y tasa de aprendizaje adaptativa [23] [22] ya descripto en el capítulo 4.

- La función de activación utilizada tanto en las neuronas de la capa oculta como en las de la capa de salida fue *Logística sigmoidea* definida en (4.4)

Los parámetros de tasa de aprendizaje adaptativa, momento y cantidad de épocas de entrenamiento también fueron seleccionados experimentalmente. Así, por ejemplo, los valores de

parámetros que han permitido mejorar los resultados para el preprocesamiento *MH-4* fueron, tasa de aprendizaje de 0.01, momento de 0.95 y cantidad de épocas máximas de 3500. En cuanto a la cantidad de neuronas en la capa oculta m, para el caso de la base de datos de CENPARMI fue de 150 y para el caso de MNIST fue de 110, resultando en una arquitectura $64 \times 150 \times 10$ y $196 \times 110 \times 10$ para cada una de las bases de datos respectivamente.

A continuación se muestra una tabla con abreviaciones de los métodos de preprocesamiento descriptos en el capítulo 3:

Abreviación del Método	Descripción
MH-4	Rep. posicional con Mexican Hat, con $a_0 = 2{,}2$, $\epsilon = 1$ y submuestreo
CDF 9/7	DWT biortogonal Cohen-Daubechies-Feauveau
FVW	Vector de características basadas en wavelets
SP	Sin preprocesamiento

Cuadro 5.1: Tabla de abreviaciones de los métodos de preprocesamiento

Las combinaciones de métodos de preprocesamiento se indicaran concatenando las abreviaciones presentadas en el cuadro 5.1 de la siguiente manera: se escribirá *MH-4 + FVW* para indicar, por ejemplo, que los métodos de "Representación posicional con submuestreo" junto con el cálculo del "Vector de características basadas en wavelets" fueron combinados.

Notar que con el objetivo de comparar el rendimiento de los métodos, se entrenó a la red neuronal sin aplicar ningún preprocesamiento alguno, ese caso se abreviará *SP* que denotará "Sin Preprocesamiento". La experimentación de este trabajo de tesis, que presentamos en el presente capítulo, fue realizada utilizando las bases de datos antes mencionadas: CENPARMI y MNIST. Para cada una de ellas, los resultados se muestran en una tabla donde cada entrada contiene el método de preprocesamiento utilizado, la arquitectura de red utilizada, el porcentaje de reconocimiento obtenido sobre el conjunto de entrenamiento y el porcentaje de reconocimiento obtenido sobre el conjunto de testeo. Además esta misma información se muestra en un gráfico de barras con los porcentajes de reconocimiento sobre el conjunto de testeo por cada método de preprocesamiento. También se muestra información más detallada acerca de la *performance* de la red neuronal para el método de preprocesamiento *MH-4*: ejemplos de dígitos bien y mal clasificados, matriz de confusión y gráfico con la evolución del error durante la etapa de aprendizaje.

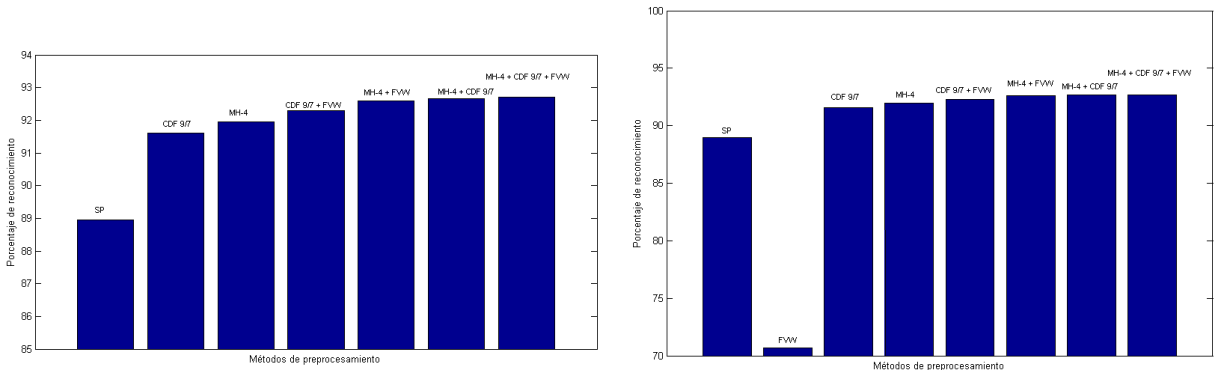
5.1. CENPARMI

La base de datos del CENPARMI que se utilizó está compuesta por 6000 dígitos escritos a mano sin restricción alguna. Los mismos fueron provistos por el Servicio Postal de los Estados Unidos, y extraidos de códigos postales manuscritos en los sobres de la correspondencia. De los 6000 dígitos, 4000 (400 por dígito) fueron utilizados para el conjunto de entramiento y el resto de los 2000 (200 por dígito) se utilizó como conjunto de testeo. Observar que esta base de datos se encuentra balanceada ya que encontramos la misma cantidad de representantes por dígito. Se utilizaron imágenes de los dígitos normalizadas en tamaño a 16x16 ya que los patrones de la base tienen diferente tamaño. En el cuadro 5.2 se pueden ver los resultados obtenidos, no sólo para cada método de preprocesamiento por separado sino también para combinaciones entre los mismos. En la figura 5.1 se muestra la misma información en un gráfico de barras.

Método de Preprocesamiento	Arquitectura de Red	% Reconocimiento Conjunto de Entrenamiento	% Reconocimiento Conjunto de Testeo
SP	$256 \times 220 \times 10$	99,13	88,95
FVW	$85 \times 170 \times 10$	78,35	70,70
CDF 9/7	$64 \times 150 \times 10$	99,22	91,60
MH-4	$64 \times 150 \times 10$	99,22	91,95
CDF 9/7 + FVW	$149 \times 170 \times 10$	99,20	92,30
MH-4 + FVW	$149 \times 200 \times 10$	99,22	92,60
MH-4 + CDF 9/7	$128 \times 210 \times 10$	99,28	92,65
MH-4 + CDF 9/7 + FVW	$213 \times 180 \times 10$	99,28	92,70

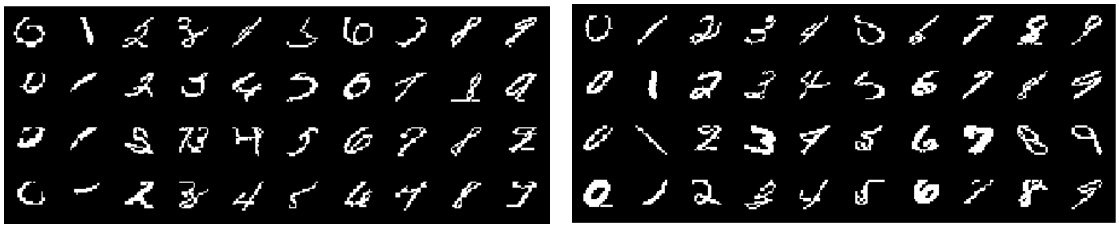
Cuadro 5.2: Resultados obtenidos sobre la base de datos CENPARMI

En la figura 5.2 se pueden observar muestras de dígitos del conjunto de testeo de la base de datos de CENPARMI que han sido bien y mal clasificados, cuando se los procesó con el método *MH-4* y clasificó con un MLP con arquitectura $64 \times 150 \times 10$. El entrenamiento se realizó con *backpropagation* estocástico con *momentum* (valor utilizado de 0.95) y velocidad de aprendizaje adaptativa (con valor inicial de 0.01 y factores de incrementeo y decremento de 1.05 y 0.7 respectivamente). En la figura 5.3 se puede observar la evolución del error durante el proceso de aprendizaje de este red, llegando a un valor menor a 0.001 en 942 épocas. En el cuadro 5.3, la matriz de confusión permite observar los resultados del testeo.



(a) (b)

Figura 5.1: Gráfico con los resultados sobre el conjunto de testeo de CENPARMI. (a) Sin el método FVW y (b) incluyendo el método FVW



(a) (b)

Figura 5.2: (a) Muestras de dígitos de testeo de CENPARMI, mal clasificados al ser preprocesados con el método $MH-4$. En (b) se muestran otras muestras de testeo pero bien clasificadas con el mismo método de preprocesamiento

Dígito	0	1	2	3	4	5	6	7	8	9
0	192	0	2	6	1	3	5	1	0	1
1	1	196	0	0	1	0	2	4	7	0
2	1	0	180	5	0	7	0	5	0	0
3	0	1	7	175	0	5	0	1	6	1
4	3	1	3	2	195	0	7	3	6	2
5	0	0	1	4	0	178	0	1	6	0
6	1	0	4	0	2	2	184	0	0	0
7	0	1	0	2	0	0	0	178	0	5
8	0	0	3	5	0	4	1	2	173	3
9	2	1	0	1	1	1	1	5	2	188

Cuadro 5.3: Matriz de confusión denotando los resultados obtenidos sobre el conjunto de testeo de CENPARMI luego de entrenar una red neuronal, con 150 neuronas en la capa oculta, y preproceando a los dígitos con el método *MH-4*. En el mismo se puede observar que se alcanza la meta (error menor a 0.001) en la época 942 siendo 3500 la cantidad de épocas máxima.

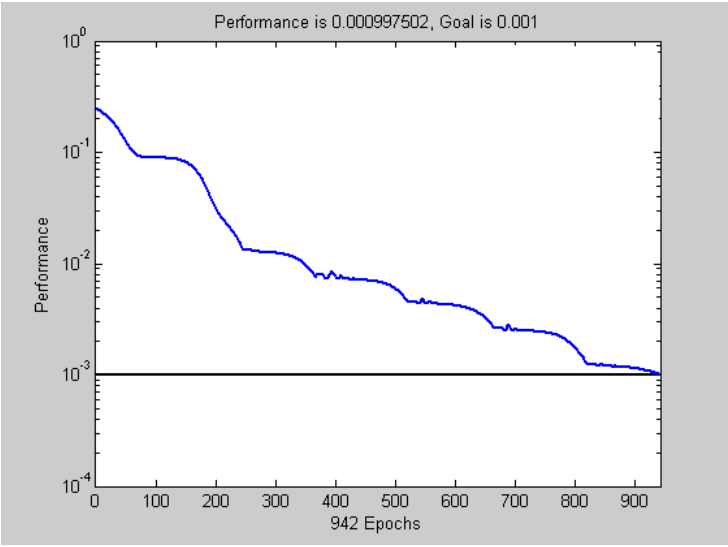


Figura 5.3: Evolución del error durante el entrenamiento de una red neuronal, con 150 neuronas en la capa oculta, utilizando *MH-4* como método de preprocesamiento sobre la base de datos del CENPARMI

5.2. MNIST

Por otro lado, la base de datos MNIST está compuesta por 70.000 dígitos manuscritos de tamaño 28x28 píxeles, los mismos fueron descargados desde [17]. Del total de dígitos, 60.000 fueron utilizados para el conjunto de entrenamiento y el resto de los 10.000 para el conjunto de testeo. Dicha separación ya estaba realizada de antemano y es la misma con la que ya han trabajado otros autores. Una de las principales difrencias con CENPARMI, además de la dimensión de los datos, es que la misma se encuentra desbalanceada, en el cuadro 5.5 se puede ver la distribución de los dígitos sobre ambos conjuntos. En el cuadro 5.4 se pueden ver los resultados obtenidos sobre MNIST y en la figura 5.4 se muestra la misma información en gráficos de barra.

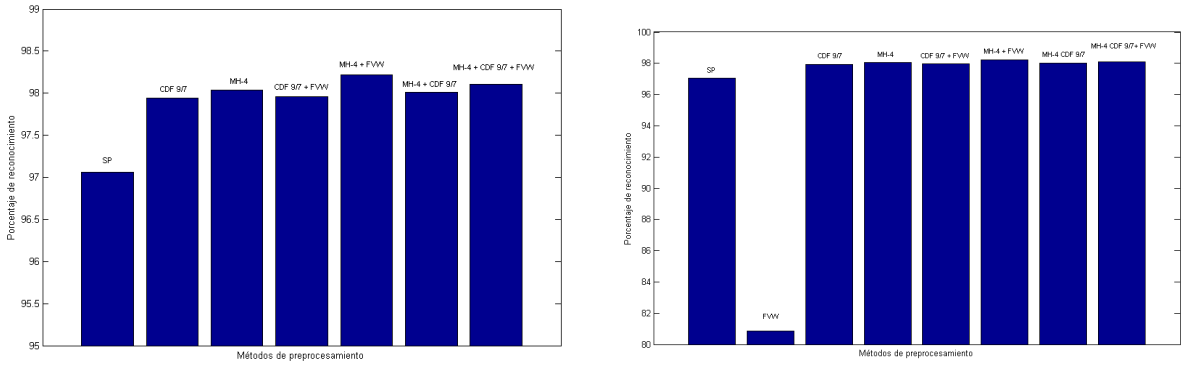
Método de Preprocesamiento	Arquitectura de Red	% Reconocimiento Conjunto de Entrenamiento	% Reconocimiento Conjunto de Testeo
SP	$784 \times 110 \times 10$	99,29	97,06
FVW	$85 \times 110 \times 10$	79,88	80,85
CDF 9/7	$196 \times 110 \times 10$	99,44	97,94
MH-4	$196 \times 110 \times 10$	99,46	98,04
CDF 9/7 + FVW	$281 \times 110 \times 10$	99,01	97,96
MH-4 + FVW	$281 \times 130 \times 10$	99,46	98,22
MH-4 + CDF 9/7	$392 \times 110 \times 10$	99,49	98,01
MH-4 + CDF 9/7 + FVW	$477 \times 110 \times 10$	99,50	98,11

Cuadro 5.4: Resultados obtenidos sobre la base de datos MNIST

En la figura 5.5 se pueden observar muestras de dígitos del conjunto de testeo de la base de datos MNIST que han sido bien y mal clasificados, cuando se los procesó con el método $MH - 4$ y clasificó con un MLP con arquitectura $196 \times 110 \times 10$. El entrenamiento se realizó con *backpropagation* estocástico con *momemtum* (valor utilizado de 0.95) y velocidad de aprendizaje (con valor inicial de 0.01 y factores de incremento y decremento 1.05 y 0.7 respectivamente). En la figura 5.6 se puede observar la evolución del error durante el proceso de aprendizaje de esta red, llegando a un valor menor a 0.001 en 2831 épocas. En el cuadro 5.6, la matriz de confusión permite observar los resultados del testeo.

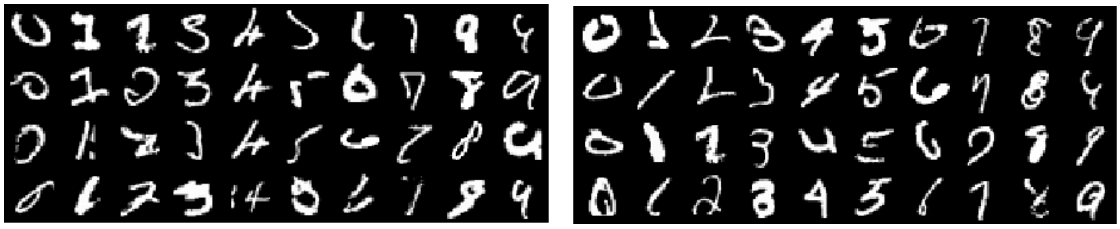
Dígito	Cantidad en el conjunto de entrenamiento	Cantidad en el conjunto de testeo
0	5923	980
1	6742	1135
2	5958	1032
3	6131	1010
4	5842	982
5	5421	892
6	5918	958
7	6265	1028
8	5851	974
9	5949	1009
Total	60000	10000

Cuadro 5.5: Distribución de los dígitos de la base de datos MNIST



(a) (b)

Figura 5.4: Gráfico con los resultados sobre el conjunto de testeo de MNIST. (a) Sin el método FVW y (b) incluyendo el método FVW



(a) (b)

Figura 5.5: (a) Muestras de dígitos de testeo de MNIST, mal clasificados al ser preprocesados con el método $MH-4$. En (b) se muestran otras muestras de testeo pero bien clasificadas con el mismo método de preprocesamiento

Dígito	0	1	2	3	4	5	6	7	8	9
0	971	0	4	0	0	2	6	1	2	3
1	1	1128	1	0	0	0	1	2	1	4
2	0	3	1011	4	4	4	1	7	3	0
3	0	0	3	987	0	10	1	2	4	3
4	0	0	1	1	964	2	3	2	4	7
5	0	0	0	4	0	864	1	0	4	3
6	2	2	0	0	4	1	941	0	2	1
7	1	0	6	5	0	2	1	1008	2	5
8	3	2	4	4	2	5	3	2	948	1
9	2	0	2	5	8	2	0	4	4	982

Cuadro 5.6: Matriz de confusión denotando los resultados obtenidos sobre el conjunto de testeo de MNIST luego de entrenar una red neuronal, con 110 neuronas en la capa oculta, y preprocesando a los dígitos con el método *MH-4*

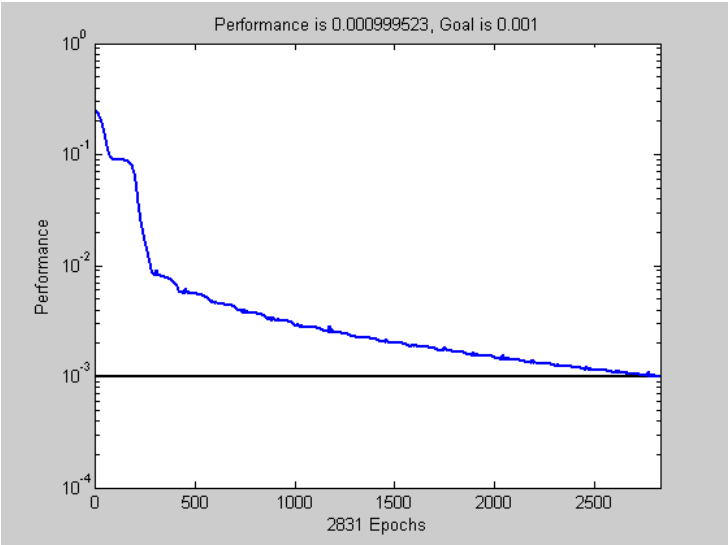


Figura 5.6: Evolución del error durante la etapa de entrenamiento de una red neuronal, con 110 neuronas en la capa oculta, utilizando *MH-4* como método de preprocesamiento sobre la base de datos de MNIST. En el mismo se puede observar que se alcanza la meta (error menor a 0.001) en la época 2831 siendo 3500 la cantidad de épocas máxima.

Capítulo 6

Conclusiones

En este trabajo se presentan diferentes preprocesamientos de los dígitos manuscritos, basados en la transformada wavelet, y el entrenamiento de una red neuronal perceptrón multicapa para su reconocimiento. En base a los resultados se ha mostrado que nuestro enfoque presenta varias ventajas.

En primer lugar, es evidente la conveniencia de utilizar el preprocesamiento de los dígitos antes de ingresarlos a la red neuronal, lo cual mejora notablemente el porcentaje de reconocimiento, frente al uso de los dígitos sin preprocesar.

En segundo lugar, el preprocesamiento que hemos utilizado, basado en la transformada wavelet, permite reducir considerablemente la dimensión de los patrones, lo cual implica un costo computacional menor, un requerimiento menor de memoria, y un tiempo menor de entrenamiento de la red.

En tercer lugar, la construcción de un vector de características que captura ciertas propiedades de los dígitos, tales como la orientación, el gradiente, la curvatura, la entropía y la energía, si bien no da buenos resultados por sí solo, da excelentes resultados cuando se lo combina con alguna versión suavizada y pequeña de la imagen.

Por otro lado, los resultados obtenidos fueron similares para las dos bases de datos a pesar que las mismas tienen diferencias significativas: cantidad de patrones, dimensión de patrones, etc. Esto es un buen indicador acerca de la utilización de wavelets en el reconocimiento de dígitos manuscritos. Nuestro sistema logró un porcentaje de reconocimiento del orden del 92.70 % sobre el conjunto de testeo para la base CENPARMI y del orden del 98.22 % para la base MNIST.

Los mejores resultados se obtuvieron combinando diferentes transformadas wavelets con el vector de características basadas en wavelets.

Podemos observar que la combinación de características apropiadas junto con la reducción de la dimensión del descriptor del patrón con el que se entrenó y testeó la red neuronal, permitieron mejorar el rendimiento y la capacidad de generalización del clasificador.

También es de destacar que nuestros resultados son comparables con los reportados para otras técnicas que tratan esta problemática [21][9][1].

Bibliografía

[1] P. Wunsch and A. Laine, "Wavelet descriptors for multiresolution recognition of handprinted characters," Pattern Recognition **28,** 12371249 (1995).

[2] W. Pratt, *Digital Image Processing* (1978).

[3] D. Gorgevik and D. Cakmakov, "An Efficient Three-Stage Classifier for Handwritten Digit Recognition," Proceedings of the 17th International Conference on Pattern Recognition (ICPR04) **4,** 507–510 (2004).

[4] L. Seijas and E. C. Segura, "Detection of ambiguous patterns in SOM based recognition system: application to handwritten numeral classification," Proceedings of the 6th International Workshop on Self-Organizing Maps (WSOM 2007), Bielefeld, Alemania (2007).

[5] B. Zhang, M. Fu, and H. Yan, "A nonlinear neural network model of mixture of local principal component analysis: application to handwritten digits recognition," Pattern Recognition 34(2) **2,** 203–214 (2001).

[6] S. B. Cho, "Self-Organizing Map with Dynamical Node Splitting: Application to Handwritten Digit Recognition," Neural Computation **9,** 1345–1355 (1997).

[7] A. Ruedin, "A Nonseparable multiwavelet for edge detection," Wavelet Appl. Signal Image Proc. X, Proc. SPIE **5207,** 700–709 (2003).

[8] S. Liapis and G. Tziritas, "Color and Texture Image Retrieval Using Chromaticity Histograms and Wavelet Frames," IEEE Transactions on Multimedia **6,** 676–686 (2004).

[9] G. Chen, T. Bui, and A. Krzyzak, "Contour-based handwritten numeral recognition using multiwavelets and neural networks," Pattern Recognition **36,** 1597–1604 (2003).

[10] I. Daubechies, *Ten lectures on wavelets, Society for Industrial and Applied Mathematics* (1992).

[11] S. Mallat, in *A Wavelet Tour of Signal Processing*, A. Press, ed., (1999).

[12] J. Antoine, P. Vandergheynst, K. Bouyoucef, and R. Murenzi, "Target detection and recognition using two-dimensional isotropic and anisotropic wavelets," Automatic Object Recognition V, SPIE Proc **2485,** 20–31 (1995).

[13] J.-P. Antoine and R. Murenzi, "Two-dimensional directional wavelets and the scale-angle representation," Signal Processing **52,** 256–281 (1996).

[14] L. Kaplan and R. Murenzi, "Pose estimation of SAR imagery using the two dimensional continuous wavelet transform," Pattern Recognition Letters **24,** 2269 – 2280 (2003).

[15] J. Romero, L.Seijas, and A. Ruedin, "Reconocimiento de Dígitos Manuscritos Usando La Transformada Wavelet Continua en 2 Dimensiones y Redes Neuronales," XII Congreso Argentino de Ciencias de la Computacin CACIC 2006 (2006).

[16] J. Romero, L.Seijas, and A. Ruedin, "Directional Continuous Wavelet Transform Applied to Handwritten Numerlas Recognition Using Neural Networks," Journal of Computer Science And Technology **7,** 67–71 (2007).

[17] http://yann.lecun.com/exdb/mnist/.

[18] J.-P. Antoine, R. Murenzi, and P. Vandergheynst, "Two-dimensional directional wavelets in image processing," Int. J. Imaging System And Techn. **7,** 152–165 (1996).

[19] H. F. Jelinek, R. M. C. Jr., and J. J. G. Leandro, "Exploring Wavelet Transforms for Morphological Differentiation Between Functionally Different Cat Retinal Ganglion Cells," Brain and Mind **4,** 6790 (2003).

[20] S. Correia and J. M. Carvalho, "Optimizing the Recognition Rates of Unconstrained Handwritten Numerals Using Biorthogonal Spline Wavelets," ICPR 2000, Barcelona - España, Septiembre 2000 .

[21] U. Bhattacharya, S.Vajda, A. Mallick, B.B.Chaudhuri, and A. Belaid, "On the Choice of Training Set, Architecture and Combination Rule of Multiple MLP Classifiers for Multiresolution Recognition of Handwritten Characters," Proc. Of the 9th Int. Workshop on Frontiers in Handwritten Recognition (IWFHR-9 2004), IEEE Computer Society 2004 .

[22] S. Haykin, in *Neural Networks A Comprenhesive Foundation*, P. Hall, ed., (1999).

[23] J. Hertz, A. Krogh, and R. Palmer, in *Introduction to the Theory of Neural Computation*, S. F. I. E. Board, ed., (1990).