Arquitectura Neuronal Apoyada en Señales Reconstruidas con Wavelets para Predicción de Series de Tiempo Caóticas

por

Ángel Mario García Pedrero

Tesis sometida como requisito parcial para obtener el grado de

MAESTRO EN CIENCIAS EN EL ÁREA DE CIENCIAS COMPUTACIONALES

en el

Instituto Nacional de Astrofísica, Óptica y Electrónica
Noviembre 2009
Tonantzintla, Puebla

Supervisada por:

Dra. María del Pilar Gómez Gil, INAOE

©INAOE 2009
El autor otorga al INAOE el permiso de reproducir y distribuir copias en su totalidad o en partes de esta tesis

Agradecimientos

A todas aquellas personas que durante los últimos dos años me han apoyado, ya sea con la realización de esta tesis o en un apreciar de forma más optimista los momentos de la vida.

A los profesores de la Coordinación de Computación del instituto por haber depositado su confianza en mí persona para realizar este posgrado. A mi asesora, la Dra. María del Pilar Gómez Gil, por sus consejos y apoyo incondicional a lo largo de la tesis. A mis sinodales por sus valiosos comentarios a fin de mejorar este escrito. Al CONACyT por su apoyo a través de la beca 212427.

A todas las personas que me han escuchado en mis cortos días y largas noches. A todos los amigos que durante mi estancia han compartido sus conocimientos académicos y experiencias de vida. A mi madre por todo el apoyo y cariño que me ha dado durante mi vida.

A todos ellos muchas gracias.

Resumen

La predicción de series de tiempo que presentan características caóticas es un problema con un amplio rango de aplicaciones. Aunque se han desarrollado gran cantidad de métodos para tratar de obtener una predicción precisa, éste continúa siendo un problema abierto al día de hoy. La naturaleza caótica del sistema generador de la serie dificulta la predicción al ser altamente sensible a las condiciones iniciales; una pequeña perturbación puede provocar que los valores subsecuentes diverjan exponencialmente.

Los métodos tradicionales de predicción basados en sistemas de aproximación lineales fracasan fuertemente cuando se aplican para la predicción en este tipo de series. Algunos métodos no lineales como las redes neuronales artificiales muestran un mejor desempeño en estos casos; sin embargo, estos métodos aún no consiguen una predicción suficientemente satisfactoria, en especial cuando estiman a la vez varios valores del futuro.

En este trabajo se propone una nueva arquitectura inspirada en la red recurrente conocida como *red neuronal híbrida compleja* (HCNN, del inglés *Hybrid-Complex Neural Network*) para predicción a corto plazo de series de tiempo caóticas y se analiza su desempeño comparándola con otros modelos neuronales. Para lograr la predicción, el modelo propuesto aprovecha la información resultante de la descomposición de frecuencias de la transformada *wavelet* y de las capacidades oscilatorias de las redes neuronales recurrentes. El modelo propuesto mostró un mejor desempeño que una red totalmente recurrente y una red *de alimentación*

progresiva, también conocida como red de alimentación hacia adelante (del inglés, *feedforward* network), en la predicción de 56 valores de la series Sumsin (Misiti y colaboradores, 2008) y 11 series de un conjunto de 111 series temporales llamado NN5 (Crone, 2008) el cual es utilizado en varios trabajos de predicción de series de tiempo a largo plazo y en la *"Artificial Neural Network & Computational Intelligence Forecasting Competition"*. El nuevo modelo obtuvo un error porcentual absoluto medio simétrico (sMAPE) promedio de 27.217 % en la NN5, comparado con 60.755 % obtenido por la red neuronal totalmente recurrente y 49.276 % por la red *de alimentación progresiva*.

Abstract

Chaotic time series forecasting is a problem with a wide range of applications. Although several methods have been developed to obtain an accurate prediction, nowadays, forecasting is still an open problem. The chaotic nature of a time series generator system makes difficult a prediction since it is highly sensitive to its initial conditions; a small change may cause future estimations to diverges exponentially.

Traditional prediction techniques based on linear approximation methods fail drastically when they are used to forecast this kind of time series. Some nonlinear methods such as neural networks show a better performance in this cases, but an accurate prediction is not possible yet, especially when they estimate many future values.

In this work, a new architecture inspired on a recurrent neural network known as *Hybrid-Complex Neural Network* for short time series forecasting is proposed and its performance compared with another neural models. To achieve a prediction, the proposed model takes advantage of information obtained by frequency decomposition from wavelet transforms and of the oscillatory abilities of recurrent neural networks. The proposed model shows a better performance than a fully recurrent neural network and a feedforward neural network in forecasting 56 values of the Sumsin (Misiti y colaboradores, 2008) time series and 11 of 111 series from the NN5 dataset (Crone, 2008), the latter used as benchmark in several works and in the *"Artificial Neural Network & Computational Intelligence Forecasting Competition"*. The new model obtained an symmetric mean absolute percentage error

(sMAPE) average of 27.217 %, compared to 60.755 % obtained by fully recurrent neural network and 49.276 % obtained by a feedforward neuronal network.

Índice general

Agradecimientos i

Resumen iii

Abstract v

Lista de Acrónimos xvii

Lista de Símbolos xix

1. Introducción 1

1.1. Problemática 1

1.2. Objetivos 2

1.2.1. Objetivo general 2

1.2.2. Objetivos específicos 3

1.3. Metodología 3

1.4. Aportación principal 4

1.5. Estructura 4

2. Marco Teórico y Trabajo Relacionado 5

2.1. Series de Tiempo 5

2.1.1. Características de las series de tiempo 5

2.2. Fundamentos de predicción 8

2.3. Redes neuronales para predicción 12

2.3.1. Redes neuronales *de alimentación progresiva* 14
2.3.2. Redes neuronales recurrentes 17
2.4. Red neuronal híbrida compleja 18
2.5. Algoritmos de aprendizaje 21
2.5.1. Algoritmo de entrenamiento RTRL-EKF 23
2.6. Teoría de *Wavelets* 26
2.6.1. Definición de *Wavelet* 27
2.6.2. Transformada *Wavelet* Continua 28
2.6.3. Transformada *Wavelet* Discreta 29
2.7. Trabajo relacionado 32
2.7.1. Trabajos que emplean redes neuronales perceptrón multicapa 32
2.7.2. Trabajos que emplean redes neuronales *wavelets* 34
2.7.3. Trabajos que emplean *máquinas de vectores de soporte* . . 36
2.7.4. Trabajos que emplean redes neuronales recurrentes 37
2.8. Análisis de trabajos relacionados 39
2.9. Resumen del capítulo 43

3. Predictor neuronal propuesto 45
3.1. Descripción de la arquitectura 45
3.2. Proceso de construcción del predictor 47
3.2.1. Escalamiento 47
3.2.2. Descomposición de la señal y reconstrucción de coeficientes *wavelet* 49
3.2.3. Entrenamiento de las subredes 51
3.2.4. Integración de la arquitectura 52
3.2.5. Entrenamiento de la arquitectura propuesta 52
3.2.6. Predicción de valores futuros 52
3.3. Resumen del capítulo 54

4. Experimentos y resultados **57**

4.1. Evaluación de la Precisión 58

4.1.1. Error Cuadrático Medio 58

4.1.2. Error Porcentual Absoluto Medio Simétrico 58

4.1.3. Error Escalado Absoluto Medio 59

4.2. Descripción de las series utilizadas 59

4.2.1. Serie de Tiempo *Sumsin* 60

4.2.2. Series de Tiempo *NN5* 61

4.3. Experimentos realizados 61

4.3.1. Predicción de la serie *Sumsin* 62

4.3.2. Predicción de las series NN5 63

4.4. Análisis de resultados 79

4.5. Resumen del capítulo 87

5. Conclusiones y trabajo futuro **89**

5.1. Conclusiones 89

5.2. Trabajo futuro 90

Bibliografía **91**

Apéndices **96**

A. Tablas detalladas de resultados **97**

B. Gráficas del mejor y peor caso **117**

Índice de figuras

2.1. Red perceptrón multicapa. 15
2.2. Arquitectura de una SVM para regresión. 17
2.3. Configuración de la red de Elman. 19
2.4. Configuración de la red neuronal totalmente recurrente. 20
2.5. Red Híbrida Compleja. 22
2.6. Predicción hecha por la HCNN. 22
2.7. Banco de filtros de análisis 31
2.8. Descomposición *wavelet* de 3 niveles 31
2.9. Banco de filtros de síntesis 32
2.10. Red Neuronal de $1 + \frac{1}{2}$ capas. 35

3.1. Arquitectura neuronal propuesta. 46
3.2. Proceso para construir la arquitectura propuesta. 48
3.3. Señales reconstruidas correspondientes a la serie NN5-001. 50
3.4. Entrenamiento de la arquitectura propuesta. 53
3.5. Predicción empleando la arquitectura propuesta. 54

4.1. Serie de Tiempo *Sumsin*. 60
4.2. Serie de tiempo *NN5-001*. 66
4.3. Señales reconstruidas correspondientes a la serie *Sumsin* empleando la *wavelet bior2.4(.)*. 68
4.4. Conjuntos de entrenamiento, prueba y validación. 70

4.5. Serie de tiempo *NN5-109*. 70
4.6. Señales reconstruidas correspondientes a la serie *NN5-109* empleando la *wavelet db10(.)*. 71
4.7. Serie de tiempo *NN5-107*. 74
4.8. Señales reconstruidas correspondientes a la serie *NN5-107* empleando la *wavelet db10(.)*. 76

B.1. Predicciones del conjunto de prueba de la serie *NN5-109*. 118
B.2. Predicciones del conjunto de validación de la serie *NN5-109*. . . . 119
B.3. Predicciones del conjunto de prueba de la serie *NN5-107*. 120
B.4. Predicciones del conjunto de validación de la serie *NN5-107*. . . . 121

Índice de tablas

2.1. Comparación de los predictores descritos en los trabajos previos y relacionados. 41

4.1. Espectro de Lyapunov de las series *NN5* 67
4.2. Precisión de las predicciones de serie de tiempo *Sumsin* 69
4.3. Precisión de las predicciones del conjunto de prueba de la serie *NN5-109* 72
4.4. Resultados de la predicción del conjunto de validación de la serie *NN5-109*. 73
4.5. Precisión de las predicciones del conjunto de prueba de la serie *NN5-107* 77
4.6. Resultados de la predicción del conjunto de validación de la serie *NN5-107*. 78
4.7. Comparación de los promedio errores de predicción del conjunto de prueba. 81
4.8. Tiempos promedios de entrenamiento de los modelos. 82
4.9. Tiempos promedios de predicción de los modelos. 83
4.10. Errores de predicción del conjunto de prueba de la series NN5 con la arquitectura propuesta. 84
4.12. Resultados del concurso de predicción *NN5* 84
4.12. Resultados de la predicción de 56 valores del conjunto de 11 series del conjunto *NN5* (Crone, 2009) (*continuación página previa*). . . 85

4.11. Errores de predicción del conjunto de validación de la series NN5. 86

A.1. Precisión de las predicciones del conjunto de prueba de la serie *NN5-101* 98

A.2. Precisión de las predicciones del conjunto de prueba de la serie *NN5-102* 99

A.3. Precisión de las predicciones del conjunto de prueba de la serie *NN5-103* 100

A.4. Precisión de las predicciones del conjunto de prueba de la serie *NN5-104* 101

A.5. Precisión de las predicciones del conjunto de prueba de la serie *NN5-105* 102

A.6. Precisión de las predicciones de serie de tiempo *NN5-106* 103

A.7. Precisión de las predicciones del conjunto de prueba de la serie *NN5-108* 104

A.8. Precisión de las predicciones del conjunto de prueba de la serie *NN5-110* 105

A.9. Precisión de las predicciones del conjunto de prueba de la serie *NN5-111* 106

A.10. Resultados de la predicción del conjunto de validación de la serie *NN5-101.* 107

A.11. Resultados de la predicción del conjunto de validación de la serie *NN5-102.* 108

A.12. Resultados de la predicción del conjunto de validación de la serie *NN5-103.* 109

A.13. Resultados de la predicción del conjunto de validación de la serie *NN5-104.* 110

A.14. Resultados de la predicción del conjunto de validación de la serie *NN5-105.* 111

A.15.Resultados de la predicción del conjunto de validación de la serie *NN5-106*. 112
A.16.Resultados de la predicción del conjunto de validación de la serie *NN5-108*. 113
A.17.Resultados de la predicción del conjunto de validación de la serie *NN5-110*. 114
A.18.Resultados de la predicción del conjunto de validación de la serie *NN5-111*. 115

Lista de Acrónimos

AR modelo de auto-regresión.

ARIMA modelo de auto-regresión integrado de medias móviles.

ARMA modelo de auto-regresión de medias móviles.

BPN red de retropropagación.

BPTT algoritmo de retropropagación a través del tiempo.

CWT transformada wavelet continua.

DWT transformada wavelet discreta.

EKF algoritmo de filtro de Kalman extendido.

HCNN Red Neuronal Híbrida Compleja.

iCWT transformada inversa wavelet continua.

iDWT transformada inversa wavelet discreta.

IPE error de predicción iterativo.

kNN k vecinos más cercanos.

MA modelo de medias móviles.

MASE error escalado absoluto medio.

MLP red neuronal percetrón multicapa.

MSE error cuadrático medio.

NN5 conjunto de series de tiempo del concurso de predicción para redes neuronales artificiales y métodos de inteligencia computacional (*NN5 Artificial Neural Network & Computational Intelligence Forecasting Competition*).

PE porcentaje de error.

PSO-EA combinación de un algoritmo optimización de enjambre de partículas con un algoritmo evolutivo.

RBF red neuronal con función de activación de base radial.

RTRL algoritmo de aprendizaje recurrente en tiempo real.

RTRL-EKF algoritmo de filtro de Kalman Extendido basado en aprendizaje recurrente en tiempo real para el cálculo del gradiente.

sMAPE error porcentual absoluto medio simétrico.

SVM máquina de vectores de soporte.

Lista de Símbolos

$\varphi(.)$ función de activación.

ϕ parámetros de auto-regresión.

$Cov(.,.)$ covarianza.

$DE(.)$ desviación estándar.

$\mathbb{Z}$ conjunto de números enteros.

u entradas de un sistema.

$f(.)$ función arbitraria.

$max(.)$ función máximo.

$E\{\}$ esperanza matemática.

θ parámetros de medias móviles.

$min(.)$ función mínimo.

$m(.)$ modelo de predicción.

ε ruido blanco.

w_{ji} conexión de la i-ésima a la j-ésima neurona.

$< \cdot >$ producto punto.

$\mathbb{R}$ conjunto de números reales.

y_i salida del sistema.

: tal que.

b umbral.

$Var(.)$ varianza de un conjunto de observaciones.

$\psi(.)$ función *wavelet*.

Capítulo 1

Introducción

1.1. Problemática

La predicción de series de tiempo que presentan características caóticas es un problema con un amplio rango de aplicaciones en áreas como economía, medicina, meteorología, procesamiento de señales y control, entre otras. Se dice que los procesos de la naturaleza son caóticos, por lo cual el estudio de las series generadas por éstos despierta cierto interés. En muchos casos, no existe información exacta sobre los parámetros del proceso generador de la serie que pueda ayudar a la predicción, por lo que sólo se utiliza la serie de tiempo conocida para modelar estos procesos e inferir su comportamiento.

Las series generadas por sistemas caóticos son altamente sensibles a las condiciones iniciales, por lo que una pequeña perturbación durante la predicción puede provocar que los valores predichos subsecuentes diverjan exponencialmente de los reales.

A pesar de que se han propuesto un sinfín de métodos para lograr una predicción más o menos precisa de valores futuros, al día de hoy, la predicción de series de tiempo caóticas es un problema que continúa abierto y sobre el que se desarrollan varias vertientes de investigación, debido a la importancia que tienen

estas señales en áreas económicas, sociales y médicas.

Los métodos de predicción *tradicionales*[1] fracasan fuertemente cuando se aplican para la predicción consecutiva de varios valores en este tipo de series, es decir, el horizonte de predicción que pueden alcanzar es muy pequeño. Por esta razón, es necesario buscar métodos no lineales que permitan un mejor desempeño. Dentro de los métodos no lineales, se encuentran los basados en inteligencia computacional como los vecinos más cercanos (kNN), inferencia *Bayesiana*, algoritmos genéticos y redes neuronales artificiales.

Debido a su capacidad de abstracción de información esencial a través de aprendizaje basado en ejemplos, las redes neuronales han mostrado ser de los mejores métodos para modelar sistemas caóticos. En especial, los mejores desempeños en algunas aplicaciones se han obtenido con redes recurrentes, ya que pueden desarrollar capacidades de oscilación caótica acotada, lo que permite su utilización en este caso. Es por ello que esta investigación gira entorno de las redes neuronales recurrentes.

1.2. Objetivos

1.2.1. Objetivo general

Esta investigación tiene por objetivo diseñar e implementar una nueva arquitectura basada en componentes neuronales recurrentes inspirados en la red neuronal híbrida compleja (Gómez-Gil, 1998) capaz de predecir 56 valores consecutivos de series caóticas, comparando su desempeño con otros modelos neuronales.

[1]Entiéndase por tradicionales a los métodos ARMA y ARIMA (Makridakis y colaboradores, 1983).

1.2.2. Objetivos específicos

Para alcanzar el objetivo propuesto se identifican los siguientes objetivos particulares:

- Definir una arquitectura neuronal y proceso de entrenamiento para la predicción de una serie de tiempo basada en la Red Neuronal Híbrida Compleja.
- Realizar un análisis comparativo entre el predictor implementado, y los modelos de redes neuronales convencionales sobre la serie de tiempo *Sumsin* (Misiti y colaboradores, 2008) y el conjunto de series *NN5* (Crone, 2008). La primera es una serie de tiempo empleada como base en la literatura para probar las capacidades del análisis *wavelet* de identificar diferentes frecuencias en el tiempo. El segundo es un conjunto de series de tiempo económicas proveniente del concurso internacional de predicción de series de tiempo *"NN5 Forecasting Competition for Artificial Neural Networks and Computational Intelligence"* y es usado para probar la eficacia de las predicciones del modelo propuesto.

1.3. Metodología

En el presente trabajo se desarrolló una arquitectura neuronal para predicción de series de tiempo caóticas. El modelo emplea pequeñas subredes neuronales totalmente recurrentes capaces de aprender las relaciones de tiempo y frecuencia de la serie de tiempo. Dichas relaciones son obtenidas de señales reconstruidas a través de los coeficientes resultantes de la descomposición *wavelet*.

La precisión de las predicciones son evaluadas usando las series *Sumsin* y 11 series del conjunto de datos NN5. Los resultados obtenidos por la arquitectura propuesta son comparados con dos modelos de redes neuronales convencionales: una red *de alimentación progresiva* y una red totalmente recurrente. Para cada serie y cada modelo de red neuronal, se ejecutaron 12 experimentos, cada uno con

la misma configuración del modelo neuronal correspondiente, pero con diferentes pesos iniciales aleatorios. Los resultados promedio obtenidos por la arquitectura propuesta en la serie *Sumsin* señalan un MSE de 0.068 ± 0.039, el cual es menor al MSE de 0.094 ± 0.038 obtenidos por las redes *de alimentación progresiva* y el MSE de 0.131 ± 0.140 de las redes totalmente recurrentes. Con respecto a las 11 series de tiempo de *NN5* empleadas, la arquitectura propuesta obtuvo un MSE promedio de 34.053 ± 20.116, mientras que las redes *de alimentación progresiva* obtuvieron un MSE promedio de 250.122 ± 226.046 y la redes totalmente recurrentes un MSE promedio de 198.693 ± 131.121.

1.4. Aportación principal

La aportación principal de esta investigación ha sido la creación de un modelo de predicción cuya estructura toma ventaja de la información de descomposición de frecuencias de la transformada *wavelet* y de las capacidades oscilatorias de la red neuronal híbrida compleja. Asimismo, como aporte secundario, se obtuvo una revisión actualizada de la bibliografía de las técnicas de predicción de series de tiempo no lineales, sus avances y retos actuales.

1.5. Estructura

Este documento consta de 5 capítulos. En el capítulo 2 se presentan algunos conceptos básicos necesarios para el resto de la tesis y se aborda el trabajo relacionado con predicción de series de tiempo. En el capítulo 3 se describe la arquitectura neuronal propuesta y el proceso de construcción de la misma. En el capítulo 4 se presentan los resultados obtenidos de los experimentos realizados con la arquitectura propuesta y en el capítulo 5 se dan las conclusiones de la investigación, así como sugerencias sobre el trabajo futuro en este ámbito.

Capítulo 2

Marco Teórico y Trabajo Relacionado

En este capítulo se definen algunos conceptos básicos necesarios para el entendimiento de esta investigación y se describen los principales trabajos relacionados. Las primeras cinco secciones presentan los conceptos y por último se presenta una descripción de trabajos relacionados.

2.1. Series de Tiempo

Una serie de tiempo se define como una secuencia ordenada de valores observados de un fenómeno físico o financiero. Dichas observaciones son realizadas en intervalos de tiempo Δt igualmente espaciados y son representadas como un conjunto de valores discretos $x_1, x_2, x_3, ..., x_n$ (Palit y Popovic, 2005).

2.1.1. Características de las series de tiempo

De acuerdo a Zemke (2003), los principales rasgos característicos de una serie de tiempo son: comportamiento estacionario, estacionalidad, linealidad (o no linealidad), determinismo (o aleatoriedad), explicadas a continuación.

Comportamiento estacionario

Una serie de tiempo es estacionaria cuando su media y desviación estándar permanecen constantes a través del tiempo. Además, para este tipo de series, la covarianza entre observaciones x_t y x_{t-d} sólo depende de la distancia, d, entre las dos observaciones, permaneciendo constante en el tiempo. Entonces las siguientes relaciones deben mantenerse (Palit y Popovic, 2005):

$$E\{x_t\} = \mu \tag{2.1}$$

$$Var(x_t) = E\{(x_t - \mu)^2\} = \kappa_0 \tag{2.2}$$

$$Cov(x_t, x_{t-d}) = E\{(x_t - \mu)(x_{t-d} - \mu)\} = \kappa_d \tag{2.3}$$

con $t = 1, 2, ...,$ $d = ..., -2, -1, 1, 2...$ y donde μ, κ_0 y κ_d son algún valor constante finito; $E\{x_t\}$ y $Var(x_t)$ representan respectivamente el valor esperado y la varianza de la serie, mientras que $Cov(x_t, x_{t-d})$ denota la covarianza entre dos observaciones.

Estacionalidad

La estacionalidad se refiere a un patrón de fluctuación periódica de la serie de tiempo. Por ejemplo, cada año durante el periodo navideño las tiendas departamentales registran un aumento de sus ventas, las cuales, decrecen después de las vacaciones. Este comportamiento es muy común en series de tiempo de economía, donde el patrón puede repetirse en días, semanas, meses o años. La estacionalidad puede ser descrita mediante la siguiente ecuación:

$$f(x) = f(x + d) + \varepsilon \tag{2.4}$$

donde $d \in \mathbb{R}$ representa el periodo de repetición del patrón, $f(.)$ representa una función arbitraria y ε representa al ruido blanco[1].

[1]El ruido blanco es una sucesión de variables aleatorias (proceso estocástico) con esperanza nula, varianza constante, y covarianzas nulas para distintos valores de t (Gujarati, 2002).

Linealidad y no linealidad

Ambas nociones son amplias y dependen del contexto en el que se encuentren. Usualmente, la linealidad significa que una entidad puede ser descompuesta en sub-entidades, que permiten analizar el todo de forma aditiva. Entonces, las siguientes relaciones deben cumplirse:

$$f(x+y) = f(x)+f(y) \tag{2.5}$$

$$f(\lambda x) = \lambda f(x) \tag{2.6}$$

donde $f(.)$ es una función y λ es un valor constante. En los sistemas no lineales, las relaciones son más complejas y por tanto, las relaciones de linealidad no se cumplen. En el modelado de sistemas no lineales, se requiere que el método de predicción sea no lineal, dado que los métodos lineales no son apropiados para desvelar las complejas dependencias de la serie (Zemke, 2003).

Determinismo y aleatoriedad

Una serie de tiempo generada por un sistema dinámico tiene una magnitud que varía en el tiempo según una ley determinista o estocástica. Por determinista se entiende que dadas las mismas circunstancias, la transición de un estado a otro siempre es la misma. Por otro lado, en un sistema estocástico, aún dadas las mismas circunstancias las transiciones entre estados son aleatorias.

Un sistema determinista es considerado caótico cuando su *trayectoria*[2] en el *espacio de fases*[3] es sensiblemente dependiente a las condiciones iniciales, por lo cual, pequeñas diferencias son magnificadas exponencialmente con el tiempo. Entonces, a pesar de ser determinista una serie de tiempo caótica es impredecible a

[2]La trayectoria u órbita de un sistema, dado el estado inicial, es la sucesión de puntos correspondientes a los estados en cada instante de tiempo (Catsigeras, 2001).

[3]El espacio de fases es el conjunto de todos los estados posibles, no importa si van a ser alcanzados o no por el sistema en algún instante tiempo (Catsigeras, 2001).

largo plazo (Kantz y Schreiber, 2003). Por esta razón, a menudo las series caóticas son confundidas con series estocásticas.

El método estándar para identificar la sensibilidad de las condiciones iniciales es el cálculo de los exponentes de *Lyapunov*. Los exponentes de *Lyapunov* cuantifican el promedio de separación exponencial entre trayectorias cercanas en el espacio de fases (Akay, 2000). Un exponente positivo es considerado evidencia de caos, uno negativo indica que la serie es convergente y uno nulo representa un comportamiento cíclico (Edmonds, 1996).

2.2. Fundamentos de predicción

La predicción de series de tiempo consiste en la estimación de valores futuros de una serie $x_{t+1}, x_{t+2}, \ldots$, basándose en las observaciones de su pasado, $x_1, x_2, \ldots, x_t$ (Gómez-Gil, 2007). Dentro del problema de predicción de series de tiempo se encuentran la predicción de un paso y la predicción a largo plazo.

La predicción de un paso, también llamada predicción a corto plazo, se lleva a cabo cuando múltiples observaciones de los valores pasados de la serie son empleados para calcular el siguiente punto. Formalmente la predicción de un paso es definida como (Lendasse y colaboradores, 2004)

$$\hat{x}_{t+1} = m(x_t, x_{t-1}, \ldots, x_{t-p}, u_t, u_{t-1}, \ldots, u_{t-q}, \rho) \tag{2.7}$$

donde $m(.)$ es el modelo empleado para predecir la serie de tiempo, x_t son entradas del modelo (valores de la serie), u_t son algunas *variables exógenas*[4], ρ el conjunto de parámetros del modelo y $\hat{x}_{t+1}$ es la salida en el instante $t+1$ (valor predicho). p representa el total de entradas del pasado usadas en el modelo y q es el número de variables exógenas del pasado empleadas en el modelo.

[4]Una variable exógena es una variable independiente, que es predeterminada y dada fuera del modelo. Esta variable es usada para establecer condiciones externas arbitrarias al modelo que se define (Ito, 1987).

La predicción a largo plazo se deriva de la predicción de un paso, por tanto puede definirse matemáticamente como una extensión directa de la ecuación 2.7 (Lendasse y colaboradores, 2004):

$$(\hat{x}_{t+h}, \ldots, \hat{x}_{t+2}, \hat{x}_{t+1}) = m(x_t, x_{t-1}, \ldots, x_{t-p}, u_t, u_{t-1}, \ldots, u_{t-q}, \rho) \tag{2.8}$$

donde h denota el *horizonte de tiempo* final de la predicción a largo plazo. La meta de la predicción a largo plazo, por tanto, es obtener el conjunto completo de h valores futuros al tiempo t. Dentro de las estrategias empleadas para obtener predicciones a largo plazo se encuentran la estrategia recursiva y la estrategia directa. La estrategia de predicción recursiva emplea recursivamente el modelo de predicción elegido, tanto como sea necesario para cubrir el horizonte de predicción h. Durante este tipo de predicción los valores predichos son empleados como valores conocidos. La estrategia de predicción directa emplea un modelo con múltiples salidas similar al descrito en la ecuación 2.8. En dicho modelo el número de salidas es el mismo que el horizonte de tiempo h.

De acuerdo a Lillekjendlie y colaboradores (1994) la predicción de series de tiempo con características caóticas, de particular interés en esta investigación, es sólo posible a corto plazo. Lo anterior debido a la extrema sensibilidad de sus inciertas condiciones iniciales, cuyo efecto es que el error de predicción crezca exponencialmente mientras se avanza en el tiempo.

Los primeros trabajos relacionados a la predicción de series de tiempo con características caóticas datan de 1987 (Lillekjendlie y colaboradores, 1994). Al día de hoy existen un sinfín de métodos de predicción que proceden de áreas de investigación muy diversas, tales como matemáticas, estadística e inteligencia computacional.

De acuerdo a Kantz y Schreiber (2003) los métodos de predicción pueden ser divididos en lineales y no lineales. La mayoría de los modelos lineales están basados en métodos de auto-regresión utilizados comúnmente en economía. Los modelos ARMA, ARIMA y el filtro de Kalman (Makridakis y colaboradores, 1983;

Särkkä y colaboradores, 2007), explicados a continuación, son ejemplos de métodos de predicción lineal.

Los modelos de auto-regresión de medias móviles (ARMA, del inglés *autoregressive moving average model*) resultan de la combinación de los modelos AR (auto-regresión) y MA (medias móviles). Los modelos AR expresan el valor actual de una serie de tiempo como una suma lineal finita de los valores pasados y de ruido blanco. Por su parte, los modelos MA son una regresión lineal del valor actual de la serie contra el ruido blanco o el *choque aleatorio*[5] de uno o varios valores previos de la serie. Los modelos son representados mediante la siguiente ecuación:

$$x_t = \varepsilon_t + \sum_{i=1}^{p} \phi_i x_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{2.9}$$

donde x_t es la serie de tiempo, ε_t representa al ruido blanco, ϕ_i son los parámetros de auto-regresión, θ_i son los parámetros de medias móviles, p y q son los órdenes del modelo de auto-regresión y de medias móviles, respectivamente.

Los modelos de auto-regresión integrados de medias móviles (ARIMA, del inglés *autoregressive integrated moving average*) se caracterizan por convertir una serie de tiempo no estacionaria en estacionaria mediante la operación de diferencia. Dicha operación se define como:

$$\nabla^d x_t = \sum_{k=0}^{d} \binom{d}{k} (-1)^{d-k} x_{t-k} \tag{2.10}$$

donde ∇ es el símbolo empleado para denotar la operación de diferencia, x_t es la serie de tiempo no estacionaria, $\binom{\cdot}{\cdot}$ es el coeficiente binomial y d es un número natural que controla el nivel de diferenciación, usualmente con valor 1. Si $d = 1$ entonces $\nabla x_t = x_t - x_{t-1}$. El modelo general de ARIMA es representado por la siguiente ecuación:

$$W_t = \varepsilon_t + \sum_{i=1}^{p} \phi_i W_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}, \tag{2.11}$$

[5] A cada momento t de un proceso de ruido blanco se le denomina choque aleatorio.

donde W_t representa la serie de tiempo convertida en estacionaria mediante la ecuación 2.10.

El filtro de *Kalman* (Welch y Bishop, 1995; Gao, 2008) es un conjunto de ecuaciones matemáticas que estima recursivamente el valor del estado de un proceso, de tal forma que se minimice el error cuadrático medio (MSE, del inglés *mean squared error*). El proceso está gobernado por el sistema lineal discreto:

$$w_{t+1} = Aw_t + Bu_t + \omega_t \tag{2.12}$$

$$d_t = Hw_t + \nu_t \tag{2.13}$$

donde w_t representa el estado del proceso en el tiempo $t+1$, u_t es la entrada del sistema, d_t es la observación del estado, A, B y H son matrices constantes relacionadas con los parámetros estáticos del sistema, ω_t y ν_t representan el ruido en el proceso y el ruido en las mediciones la cuales respectivamente satisfacen las distribuciones normales con media cero y matrices de covarianza Q_t y R_t:

$$Q_t = \omega_t \omega_t^T \tag{2.14}$$

$$R_t = \nu_t \nu_t^T \tag{2.15}$$

Para resolver el sistema formado por las ecuaciones 2.12 y 2.13 el filtro de Kalman requiere de dos etapas conocidas como: *predicción* y *corrección*. En la etapa de *predicción* el valor del estado w_t y la matriz de covarianza P_t son actualizados de acuerdo al sistema de ecuaciones:

$$P_{\bar{t}} = AP_{t-1}A^T + Q_t \tag{2.16}$$

$$\hat{w}_{\bar{t}} = A\hat{w}_{t-1} + Bu_{t-1} \tag{2.17}$$

donde $\hat{w}_t$ son valores temporales para el estado estimado y $P_{\bar{t}}$ su covarianza. Cuando la observación d_t está disponible, el valor de su estado y su covarianza son modificados (etapa de *corrección*) de acuerdo a las siguientes ecuaciones

$$P_t = ((P_{\bar{t}})^{-1} + H^T R_t^{-1} H)^{-1} \tag{2.18}$$

$$\hat{w}_t = \hat{w}_{\bar{t}} + P_t H^T R_t^{-1}(d_t - H\hat{w}_{\bar{t}}) \tag{2.19}$$

La expresión $(d_t - H\hat{w}_{\bar{t}})$ se denomina innovación de la medida y refleja la diferencia entre la medida real y la predicha.

Los métodos lineales tienen la característica de ser relativamente fáciles de entender y de implementar. Sin embargo, resultan inadecuados para predecir el comportamiento de un fenómeno no lineal, como es el caos. Los métodos no lineales, al ser capaces de percibir dependencias no lineales de las series de tiempo, han sido utilizados en múltiples áreas para realizar predicciones de diversos fenómenos físicos y biológicos.

Algunos de los métodos no lineales aplicados a la predicción de series de tiempo provenientes del área de inteligencia computacional son vecinos más cercanos (kNN), algoritmos genéticos, y redes neuronales, entre otros. De ellos, las redes neuronales artificiales son de los métodos que mejores resultados han tenido durante la predicción de series de tiempo. A continuación se describen los principales modelos de redes neuronales empleados para la predicción de series de tiempo.

2.3. Redes neuronales para predicción

Una red neuronal es un modelo simplificado del sistema de procesamiento de información de un ser vivo (Haykin, 1998). Una red neuronal está formada por un conjunto de unidades de procesamiento llamadas *neuronas*. El modelo básico de una red con una neurona se conoce como perceptrón.

Un perceptrón está constituido básicamente por los siguientes elementos (Haykin, 1998):

- Un conjunto de n entradas, u_i, $i = 1, 2, \ldots, n$ que suministran la información del entorno. Estos datos pueden ser externos, o bien, pertenecer a la salidas de otras neuronas.
- Un conjunto de sinapsis, caracterizadas cada una por tener un peso propio w_{ji}. El peso w_{ji} está asociado a la sinapsis que conecta la i-ésima unidad con j-ésima neurona. Adicionalmente una red puede poseer un umbral, b_i,

que tiene el efecto de incrementar o disminuir la entrada de la neurona. Comúnmente, los umbrales se consideran un peso más de la red, por lo cual no se hacen distinciones con respecto a los otros pesos sinápticos.

- Un sumador que integra las entradas, ponderadas con respectivos pesos y umbrales.
- Una función de activación, $\varphi(.)$, que limita la amplitud de la salida de la neurona.

La salida de la neurona está definida como:

$$y_j = \varphi(\sum_{i=1}^{n} w_{ji}u_i + b_j) \tag{2.20}$$

La función de activación es la que determina la salida de la red neuronal. La funciones de activación más comunes son las siguientes:

- *Lineal.* Tiene la forma $\varphi(x) = x$ y se emplea cuando no se desea acotar la salida de la red.
- *Logística.* Las funciones sigmoidales son un conjunto de funciones no lineales, crecientes, monótonas y acotadas. La función sigmoidal más común es la función logística definida como:

$$\varphi(x) = \frac{1}{1 + e^{-x}} \tag{2.21}$$

  La función logística está acotada entre 0 y 1.
- *Tangente hiperbólica.* Pertenece a la familia de las sigmoidales y su valor está acotado entre -1 y 1. La tangente hiperbólica se define como:

$$\varphi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.22}$$

Debido a sus características, las redes neuronales han sido de los métodos más empleados para la predicción de series de tiempo. De acuerdo a Palit y Popovic (2005) las principales características de las redes neuronales que ayudan a la predicción son:

- Mapeo no lineal entre un subconjunto de valores pasados y valores futuros de la serie de tiempo.
- Capacidad de detectar relaciones entre los datos, principalmente cuando no hay conocimiento *a priori*, cuando es muy difícil describirlas matemáticamente o cuando los datos no son fiables.
- Capacidad de comportarse como aproximadores universales de funciones, lo que permite el modelado de funciones no lineales arbitrarias a cualquier nivel de exactitud.
- Capacidad de aprendizaje y generalización a partir de ejemplos empleando un enfoque auto-adaptativo dirigido por datos.

La forma en que se interconectan los neurones de una red neuronal, puede ser vista como un grafo dirigido. Si este grafo es acíclico, la red se denomina red neuronal *de alimentación progresiva* (en inglés *feedforward network*) y en el caso de que posea ciclos, se denomina red recurrente. Dentro del primer grupo se encuentran las redes perceptrón multicapa y las máquinas de vectores de soporte, entre otras (Haykin, 1998); en el segundo grupo se encuentran la red de Elman y la totalmente recurrente. A continuación se describen los principales modelos de redes neuronales alimentadas hacia adelante y recurrentes empleados para la predicción de series de tiempo.

2.3.1. Redes neuronales *de alimentación progresiva*

Red percetrón multicapa

Múltiples son las estructuras neuronales propuestas para predicción, de ellas las redes perceptrón multicapa (MLP, del inglés *multilayer perceptron*) son las más empleadas. Particularmente, la configuración más común es la de tres capas, en la cual la capa de entrada y de salida están directamente interconectadas mediante una capa oculta. En la figura 2.1 se muestra una red MLP de 3 capas. Debido a su estructura en forma de cascada, la red MLP es capaz de hacer un mapeo no lineal

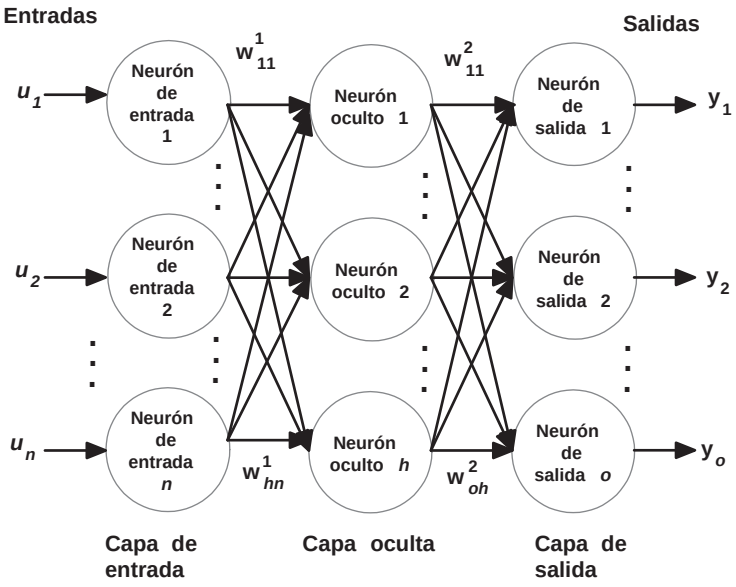


Figura 2.1: Arquitectura de una red perceptrón multicapa. El modelo consta de 3 capas: de entrada, oculta y de salida; con n, h y o neuronas en cada capa, respectivamente. Las entradas de la red son denotadas por u y las salidas por y. Las conexiones entre neuronas son representadas por w^l_{ji}, donde los subíndices indican una conexión del nodo j al i, el superíndice l indica el número de nivel entre las capas conectadas.

entre las entradas y salidas. Para una red MLP de una capa oculta este mapeo generalmente se representa como (Palit y Popovic, 2005):

$$y = \varphi_o(\sum w_h \varphi_h(\sum w_i^T u)) \tag{2.23}$$

donde los subíndices i, h y o indican respectivamente las capas de entrada, escondida y salida.

Máquinas de vectores de soporte

Las máquinas de vectores de soporte (SVM, del inglés *support vector machine*) se basan en el principio de minimización del riesgo estructural formulado por Vapnik en 1992 (Burges, 1998). Originalmente diseñadas para resolver problemas de reconocimiento de patrones, su aplicación se ha extendido a la estimación por regresión no lineal y a la predicción de series de tiempo.

En la SVM para regresión no lineal, la idea básica consiste en realizar un mapeo no lineal de los datos de entrenamiento a un espacio de dimensión mayor donde se pueda realizar una regresión lineal. La meta es encontrar una función $f(x)$ que tenga a lo más una desviación ε de la salida para todos los datos de entrenamiento. El caso en el que la función f es lineal es descrito por Smola y Schölkopf (2004) y se muestra a continuación:

$$\begin{aligned} f(x) &= < w \cdot x > + b \\ &= x^T w + b \\ &= \sum_{i=1}^{n} w_i x_i + b \end{aligned} \tag{2.24}$$

$$\text{minimizar} \quad \tfrac{1}{2}||w||^2 \tag{2.25}$$

sujeto a las restricciones:

$$\begin{aligned} y_i - (< w \cdot x_i > + b) &\leq \varepsilon \\ < w \cdot x_i > + b - y_i &\leq \varepsilon \end{aligned}$$

donde w representa los pesos del modelo, x el vector de entradas, b los umbrales y y las salidas. Los puntos que satisfacen alguna de las condiciones de la ecuación 2.25 reciben el nombre de vectores de soporte. En la figura 2.2 se muestra la estructura de una SVM para regresión, la salida y depende de los datos x_i con mayor relevancia (vectores de soporte SV).

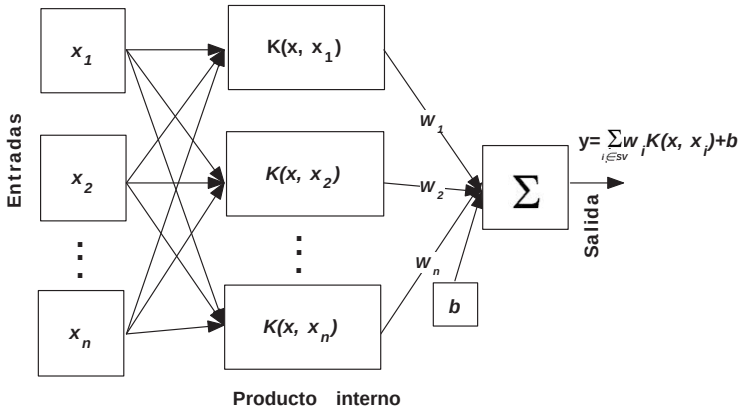


Figura 2.2: Arquitectura de una SVM para regresión. x representa el vector de ejemplos de entrada, $K(x, x_n)$ denota el producto interno, b los umbrales y y la salida. La salida depende de los datos x_i de mayor relevancia llamados vectores de soporte.

2.3.2. Redes neuronales recurrentes

Debido a su capacidad de abstracción de información temporal, las redes recurrentes son un modelo computacional más poderoso que las redes alimentadas hacia adelante. A continuación, se describe un modelo general de red recurrente con n nodos y m entradas (Doya, 2002):

$$y_j(t+1) = \varphi\left(\sum_{i=1}^{n+m} w_{ji} z_i(t)\right) \tag{2.26}$$

$$z_i(t) = \begin{cases} y_i(t), & i \leq n \\ u_{i-n}, & i > n \end{cases}$$

donde y_j denota la salida de la j-ésima neurona y w_{ji} las conexiones del j-ésimo al i-ésimo neurón. Las entradas externas u_i y las entradas recurrentes y_i son representadas como z_i por conveniencia.

Red neuronal recurrente de Elman

La red de Elman es una red neuronal de 4 capas que incluyen una capa de entrada, una capa oculta, una capa de salida y una *capa de contexto*. Los nodos de la capa de contexto tiene una conexión local hacia la capa oculta, la cual tiene un retraso temporal de un paso. La figura 2.3 muestra la configuración de la red de Elman. Entre las aplicaciones que ha tenido este modelo destacan: el procesamiento de lenguaje, la identificación y control de sistemas y la predicción de series de tiempo (Cai y colaboradores, 2007).

Red neuronal totalmente recurrente

La red neuronal totalmente recurrente consiste de n neuronas con m entradas externas. Esta red tiene dos capas distintas: una capa en la que se encuentran concatenadas las entradas externas y de retroalimentación y la capa de procesamiento. De acuerdo a Mandic y Chambers (2001), esta red tiene una representación del pasado muy rica. La figura 2.4 muestra la configuración de una red totalmente recurrente.

2.4. Red neuronal híbrida compleja

El modelo propuesto en esta investigación parte de la *Red Neuronal Híbrida Compleja* (HCNN, del inglés *Hybrid-Complex Neural Network*) diseñada por Gómez-Gil (1998). La HCNN hace uso del análisis de Fourier para entrenar múltiples *generadores armónicos* y con ello ayudar a la predicción. Un *generador armónico* es una red neuronal recurrente de 3 nodos totalmente conexa (Oldham y Gómez-Gil, 1997), la cual es capaz de oscilar autónomamente generando series

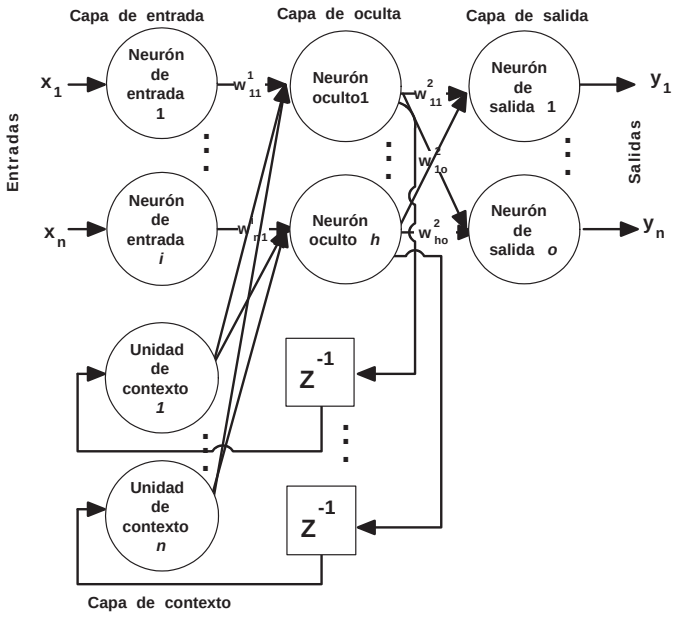


Figura 2.3: Configuración de la red de Elman. Este modelo presenta una recurrencia local en la capa oculta. El bloque etiquetado con Z^{-1} indica un retardo temporal, x representa el vector de entradas, y las salidas. Las conexiones entre nodos es indicada mediante w^l_{ji} donde los subíndices indican una conexión del nodo j al i, el superíndice l indica el número de nivel entre las capas conectadas.

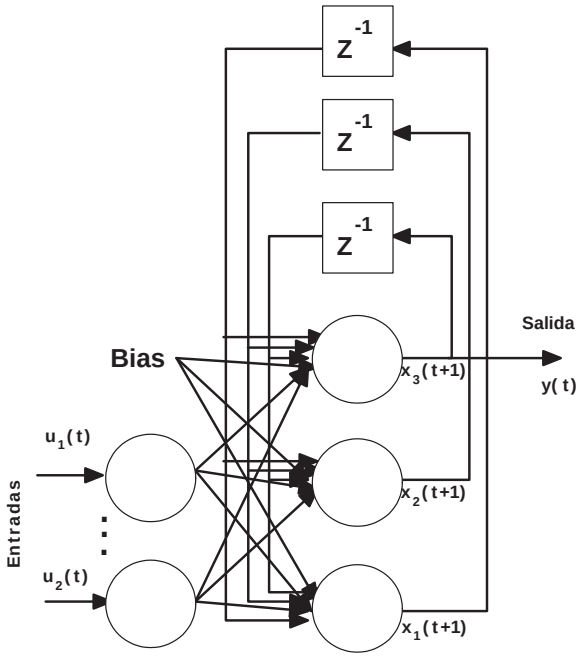


Figura 2.4: Configuración de la red neuronal totalmente recurrente. $u(t)$ representa las entradas externas de la red en el tiempo t, x_{t+1} las entradas recurrentes, Z^{-1} indica un retardo temporal de un tiempo y $y(t)$ denota la salida de la red en el tiempo t.

sinusoidales. En la HCNN, cada generador armónico se conecta a otras neuronas recurrentes mediante conexiones de alimentación hacia adelante. En la figura 2.5 se muestra la arquitectura de la HCNN con una capa oculta. Esta red utiliza como algoritmo de entrenamiento retropropagación a través del tiempo (Werbos, 1994). La dinámica de cada neurona de la red está dada por la siguiente ecuación (Pearlmutter, 1989):

$$\frac{dy_i}{dt} = -y_i + \varphi(x_i) + u_i \tag{2.27}$$

donde

$$x_i = \sum_j w_{ji} y_j \tag{2.28}$$

x_i representa la entrada total a la i-ésima neurona proveniente de otros neurones, u_i es una entrada externa a la i-ésima neurona, w_{ji} es el peso que conecta la neurona i a la neurona j y $\varphi(.)$ es una función diferenciable arbitraria, comúnmente una sigmoide. La HCNN es probada en la predicción del comportamiento de electrocardiogramas empleando 512 puntos para entrenar. Los resultados mostrados señalan que el MSE de la serie original contra la predicción es de $2.5e^{-3}$. En la figura 2.6 se muestra la predicción de 2048 puntos del comportamiento de un electrocardiograma hecha por la HCNN. La estrategia de predicción empleada en este trabajo es la recurrente.

2.5. Algoritmos de aprendizaje

Los algoritmos de aprendizaje[6] de redes neuronales modifican los parámetros configurables de la red tratando de minimizar una medida de error E. Los principales algoritmos de entrenamiento se basan en el cálculo del gradiente de la función de error, es decir, calculan la derivada del error con respecto a los parámetros ajustables[7] de la red. El objetivo es encontrar el mínimo de la función de error

[6] A lo largo de la tesis los términos algoritmo de aprendizaje y algoritmo de entrenamiento son empleados indistintamente.

[7] Entiéndase por parámetros ajustables a los pesos y bias de la red neuronal.

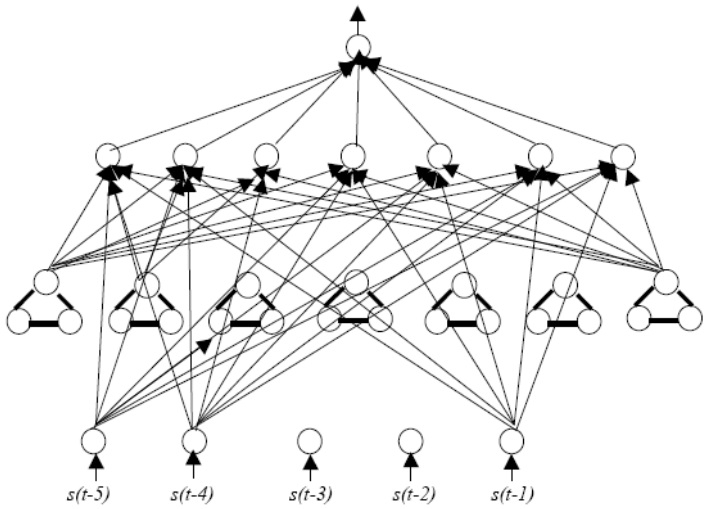


Figura 2.5: Red Neuronal Híbrida Compleja con 7 generadores armónicos, 5 entradas externas, una capa oculta con 7 nodos y 1 nodo de salida. Las entradas de la red son los valores de la serie temporal, $s(t)$, en el tiempo t. Imagen tomada de Gómez-Gil (1998).

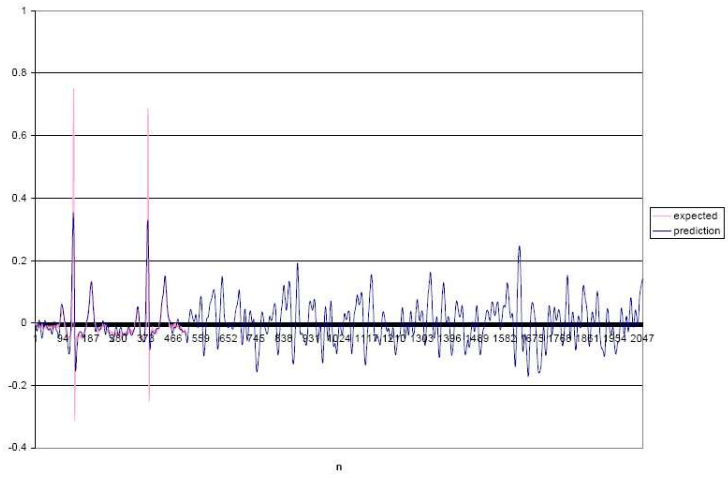


Figura 2.6: Predicción de 2048 puntos del comportamiento de un electrocardiograma hecha por la HCNN. Imagen tomada de Gómez-Gil (1998).

mediante la búsqueda de un punto en el cual el gradiente sea nulo. Aunque esta condición es necesaria, no implica que sea suficiente debido a la existencia de mínimos locales; por lo cual, son necesarias técnicas heurísticas.

Una de las formas más comunes de emplear el gradiente para modificar los parámetros de la red es el descenso del gradiente. En él, el ajuste de los parámetros se realiza individualmente en sentido opuesto al vector gradiente $\partial E(t)/\partial W_i(t)$:

$$W_i(t+1) = W_i(t) - \alpha \frac{\partial E(t)}{\partial W_i(t)} \tag{2.29}$$

donde W_i representa los parámetros ajustables, t la iteración en la que se encuentra el algoritmo y α es un parámetro conocido como factor de aprendizaje, cuyo valor por lo general es relativamente pequeño.

Dentro de los algoritmos basados en el cálculo del gradiente para entrenar redes neuronales recurrentes se encuentran: retropropagación a través del tiempo (BPTT, del inglés *backpropagation through time*), aprendizaje recurrente en tiempo real (RTRL) y filtro de Kalman extendido (EKF, del inglés *extended Kalman filter*). Los dos primeros, emplean el descenso de gradiente por lo que son considerados como distintas formas de calcular el valor de la derivada correspondiente. El EKF emplea las derivadas calculadas por cualquiera de los métodos anteriores para modificar los parámetros ajustables de manera más eficiente que el descenso del gradiente (Haykin, 1998). A continuación se describe el algoritmo de filtro de Kalman extendido basado en aprendizaje recurrente en tiempo real para el cálculo del gradiente (RTRL-EKF por sus siglas en inglés) empleado en esta investigación.

2.5.1. Algoritmo de entrenamiento RTRL-EKF

El algoritmo de aprendizaje RTRL-EKF (Haykin, 1998) comprende dos fases: el cálculo del gradiente y el ajuste de los parámetros modificables. El algoritmo empleado para el cálculo de las derivadas pertinentes es el RTRL, propuesto por Williams y Zipser (1989), mientras que el ajuste de los pesos se realiza mediante el filtro de Kalman extendido.

El algoritmo RTRL-EKF tiene una complejidad computacional en tiempo, dada por el algoritmo RTRL, de $O(n^4)$, siendo n el número de neurones de la red (Williams y Zipser, 1995). A continuación se detalla el algoritmo RTRL-EKF en dos fases, primeramente la modificación de los pesos con el EKF y posteriormente el cálculo del gradiente con el RTRL.

Modificación de los pesos con el EKF

Para poder aplicar el filtro de Kalman al entrenamiento de las redes neuronales, el aprendizaje se considera como un problema de filtrado en el que los parámetros óptimos de la red se estiman de forma recursiva de la ecuaciones del filtro de Kalman.

Se considerará que el estado de la red, denotado por w_t, viene dado por el valor de sus pesos. La ecuación que describe el sistema es lineal y se sigue de la ecuación 2.12 con $A = I$ (matriz identidad), $B = 0$ y $\omega = 0$ (Welch y Bishop, 1995):

$$w_{t+1} = w_t \tag{2.30}$$

La observación del estado del sistema es la salida deseada de la red neuronal y se describe como:

$$d_t = y_t + \nu_t \tag{2.31}$$

donde y_t es la salida de la red cuando se aplica a sus entradas u_t.

Debido a que la ecuación del estado es lineal, el filtro de Kalman empleará las ecuaciones 2.12 y 2.16 con $A = I$, $B = 0$ y $Q_t = 0$:

$$w_{t+1} = w_t \tag{2.32}$$

$$P_t^- = P_{t-1} \tag{2.33}$$

Dado la no linealidad de la ecuación 2.31, establecida por la salida de la red,

es necesario añadir las siguientes ecuaciones del filtro:

$$K_t = P_{\bar{t}} H_t^T (H_t P_{\bar{t}} H_t^T + R_t)^{-1} \tag{2.34}$$

$$\hat{w}_t = \hat{w}_{\bar{t}} + K_t(d_t - y_t) \tag{2.35}$$

$$P_t = (I - K_t H_t) P_{\bar{t}} \tag{2.36}$$

donde H_t es el jacobiano de las derivadas parciales respecto al estado:

$$H_t = \begin{bmatrix} \frac{\partial y_1}{\partial w_1} & \frac{\partial y_1}{\partial w_2} & \cdots & \frac{\partial y_1}{\partial w_n} \\ \frac{\partial y_2}{\partial w_1} & \frac{\partial y_2}{\partial w_2} & \cdots & \frac{\partial y_1}{\partial w_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_p}{\partial w_1} & \frac{\partial y_p}{\partial w_2} & \cdots & \frac{\partial y_p}{\partial w_n} \end{bmatrix}, \tag{2.37}$$

con p y n denotando el número de las neuronas de salida y pesos, respectivamente y y_i representa la salida del i-ésimo neuron.

El valor de la medida real de la ecuación 2.35 es el valor de la salida de la red. Las derivadas parciales de la matrix H_t se calculan mediante el RTRL, explicado a continuación.

Cálculo del gradiente con el RTRL

Empleando la notación de la ecuación 2.26, el cálculo de las derivadas parciales mediante el algoritmo RTRL está dada por la siguiente ecuación (Doya, 2002):

$$\frac{\partial y_i(t+1)}{\partial w_{kl}} = \sigma'(x_i(t))[\sum_{j=1}^{n} w_{ij} \frac{\partial y_j(t)}{\partial w_{kl}} + \delta_{ik} z_l(t)] \tag{2.38}$$

donde $\sigma'(.)$ es la derivada de la función de activación[8], $x_i(t) = \sum_{j=1}^{n+m} w_{ij} z_j(t)$ es la entrada de la red al neuron, δ_{ik} es la delta de Kronecker ($\delta_{ik} = 1$ si $i = k$ ó de lo contrario es 0). El término $\delta_{ik} z_l(t)$ representa el efecto de los pesos sobre la neurona k y el término $\sum_{j=1}^{n} w_{ij} \frac{\partial y_j(t)}{\partial w_{kl}}$ denota el efecto sobre todas las neuronas debido a la salida de la red.

[8]La derivada de la función logística es $\sigma(x)(1 - \sigma(x))$. La derivada de la función tangente hiperbólica es $1 - \sigma(x)^2$.

Debido a que el estado inicial de la red $y_i(0)$ es independiente de las conexiones sinápticas, la condición inicial apropiada de la ecuación 2.38 es

$$\frac{\partial y_j(0)}{\partial w_{kl}} = 0 \tag{2.39}$$

Las derivadas parciales $\frac{\partial y_j(t)}{\partial w_{kl}}$ son calculadas progresivamente en el tiempo al iterar la ecuación 2.38 simultáneamente con las salidas de la red representada por la ecuación 2.26.

2.6. Teoría de *Wavelets*

El análisis *wavelet* es una técnica matemática que se originó a finales de los años 80. Sin embargo, sus bases se remontan al trabajo propuesto por Fourier en 1807, donde establece un método para representar una señal periódica como una serie de coeficientes basados en una función de análisis. El primero en emplear el término *wavelet* fue Alfred Haar en 1909 en su tesis doctoral (Merry, 2005). En 1930, Paul Levy encontró a las funciones base Haar superiores a las funciones base de Fourier para estudiar pequeños y complicados detalles del movimiento Browniano (Graps, 1995). En 1989, Stephane Mallat desarrolló el análisis multiresolución basado en bancos de filtros (Mallat, 1989). Alrededor de 1998, Ingrid Daubechies, basándose en el trabajo de Mallat, creó su propia familia de *wavelets*, consideradas como la piedra angular de la aplicaciones *wavelet* actuales (Merry, 2005).

El análisis *wavelet* calcula la correlación entre la señal a analizar y la función *wavelet* $\psi(.)$. La similitud entre la señal y la función *wavelet* es calculada separadamente para diferentes intervalos de tiempo, dando como resultado una representación en dos dimensiones (tiempo y frecuencia). La función $\psi(.)$ generalmente es referida como *wavelet madre*. El análisis *wavelet* ha sido empleado particularmente para el análisis de señales aperiódicas, ruidosas, intermitentes y trascendentes (Addison, 2002).

A diferencia del análisis de Fourier, el análisis *wavelet* se basa en una representación local de frecuencias. La diferencias entre las funciones sinusoidales empleadas en el análisis de Fourier y las *wavelets* radican en sus propiedades de localización. La función seno se localiza en el dominio de frecuencias, pero no en el dominio del tiempo, mientras las *wavelets* se localizan en ambos dominios. Por lo anterior, el análisis *wavelet* es capaz de revelar aspectos locales de las señales analizadas tales como tendencias locales, discontinuidades y puntos de ruptura, entre otros.

2.6.1. Definición de *Wavelet*

Una función *wavelet* $\psi(.)$ es una onda pequeña, es decir, presenta oscilaciones en un periodo limitado de tiempo. A diferencia de las ondas sinusoidales, las *wavelets* son irregulares y pueden ser asimétricas. Una función $\psi(.)$ es considerada una *wavelet* si cumple los siguientes criterios matemáticos (Addison, 2002):

1. La energía E, que equivale a la integral del cuadrado de la magnitud de la función, debe ser finita, esto es:

$$E = \int_{-\infty}^{\infty} |\psi(t)|^2 dt < \infty \tag{2.40}$$

2. Si $\Psi(f)$ es la transformada de Fourier de $\psi(t)$:

$$\Psi(f) \equiv \int_{-\infty}^{\infty} \psi(t) e^{-i2\pi f} dt \tag{2.41}$$

entonces la siguiente condición debe mantenerse:

$$C_\psi = \int_0^{\infty} \frac{|\Psi(f)|^2}{f} df < \infty \tag{2.42}$$

Esta condición implica que la *wavelet* $\psi(.)$ debe tener una media igual a cero. La ecuación 2.42 es conocida como la condición de admisibilidad, mientras que C_ψ es llamada constante de admisibilidad.

2.6.2. Transformada *Wavelet* Continua

La Transformada *Wavelet* Continua (CWT, por sus siglas en inglés) de una señal $x(t)$ se define matemáticamente como (Merry, 2005):

$$X_{CWT}(\tau, s) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t)\psi^*(\frac{t-\tau}{s})dt : s, \tau \in \mathbb{R} \; y \; s \neq 0 \tag{2.43}$$

donde la señal transformada $X_{CWT}(\tau, s)$ es una función de escala s y traslación τ. La función *wavelet* es denotada por ψ; $*$ indica que el complejo conjugado es usado en el caso de una *wavelet* compleja. Los elementos en X_{CWT} son llamados coeficientes *wavelet*, cada coeficiente *wavelet* está asociado a una frecuencia (escala) y a un punto en el dominio del tiempo. La energía de la señal es normalizada en cada escala dividiendo los coeficientes *wavelet* por $1\sqrt{|s|}$. La definición 2.43 permite la representación de la señal $x(t)$ en el plano tiempo-escala.

La contracción y dilatación de la función *wavelet* es determinada por el parámetro de escala s. La variación en dicho parámetro modifica la frecuencia central f_c de la *wavelet* y la longitud de la ventana de análisis. La *wavelet* tiene su propia frecuencia central en cada escala, la escala s es inversamente proporcional a esa frecuencia. Por tanto, una escala grande corresponde a una frecuencia baja que proporciona información global de la señal analizada. Las escalas pequeñas correspondientes a las altas frecuencias, proveen la información del detalle de la señal.

El parámetro de traslación τ contiene la información en el tiempo e indica la posición en la que se encuentra la wavelet con respecto a la señal analizada.

A la par de la transformada de Fourier, la CWT también tiene una transformación inversa. La inversa de la Transformada *Wavelet* Continua (iCWT por sus siglas en inglés) se define como (Merry, 2005):

$$x(t) = \frac{1}{C_\psi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} X_{CWT}(\tau, s) \frac{1}{s^2} \psi(\frac{t-\tau}{s}) d\tau ds : s, \tau \in \mathbb{R} \; y \; s \neq 0 \tag{2.44}$$

donde C_ψ debe cumplir la condición de admisibilidad (ecuación 2.42). Al proceso de reconstrucción de la señal se le conoce como síntesis y se realiza sólo después

de la descomposición de la señal.

El cálculo de la CWT es comúnmente efectuado tomando valores discretos para los parámetros de escala s y traslación τ. Para *wavelets* discretas los parámetros de escala y traslación se eligen de tal forma que en el nivel j, la *wavelet* $s_0^j \psi(s_0^{-j} t)$ es s_0^j veces el ancho de $\psi(t)$. Por tanto, el parámetro de escala es $s = s_0^j : j \in \mathbb{Z}$, mientras que el parámetro de traslación es $\tau = k\tau_0 s_0^j : k, j \in \mathbb{Z}$.

La familia de *wavelets* con parámetros discretos está dada por la siguiente ecuación:

$$\psi_{j,k}(t) = s_0^{-\frac{j}{2}} \psi(s_0^{-j} t - k\tau_0) : j, k \in \mathbb{Z} \tag{2.45}$$

con lo cual la transformada puede ser calculada de la siguiente forma:

$$CWT_{j,k} = s_0^{-\frac{j}{2}} \int_{-\infty}^{\infty} x(t) \psi_{j,k}(t) dt \tag{2.46}$$

Generalmente los valores empleados para los parámetros de escala y traslación son $s_0 = 2$ y $\tau_0 = 1$.

La reconstrucción de la señal está dada por la siguiente ecuación (Addison, 2002):

$$x(t) \approx \sum_j \sum_k \psi_{j,k} CWT_{j,k} : j, k \in \mathbb{Z} \tag{2.47}$$

La CWT es empleada usualmente para señales continuas. A continuación se presenta la Transformada *Wavelet* Discreta empleada para analizar señales discretas, tales como series de tiempo.

2.6.3. Transformada *Wavelet* Discreta

Para realizar la Transformada *Wavelet* Discreta (DWT) se emplea comúnmente un algoritmo desarrollado por Mallat (1989), el cual se basa en un *banco de filtros*[9] para llevar a cabo el análisis de una señal discreta.

[9]Un banco de filtros consiste de filtros que descomponen una señal en bandas de frecuencias (Merry, 2005).

Se entiende por filtro un sistema formado por una ecuación de diferencia $y(n)$ y una respuesta al impulso $q(n)$ donde $n = 2^j$ y $j \in \mathbb{Z}$ que denota el nivel de descomposición de la señal analizada (Rosas-Orea, 2006). El filtrado de una señal $x(n)$ equivale a la operación matemática de convolución definida por:

$$y(n) = x(n) * q(n) = \sum_{m=-\infty}^{\infty} x(m)q(n-m) \tag{2.48}$$

En el algoritmo DWT la señal discreta $x(t)$ pasa por el banco de filtros donde es filtrada por filtros llamados $L(z)$ y $H(z)$, los cuales separan el contenido de frecuencias de la señal de entrada en bandas de frecuencia de la misma longitud. Los filtros $L(z)$ y $H(z)$ son filtros pasa bajas y pasa altas respectivamente. Este banco de filtros es conocido como banco de filtros de análisis. Cada una de las salidas de los filtros contienen la mitad del contenido de frecuencias y la misma cantidad de puntos que la señal original. Lo anterior se traduce en que al unir las salidas de los filtros se tendrá un contenido de frecuencias igual al de la señal de entrada, pero la longitud de los datos será del doble. Por tanto, un decimado (*downsampling)* por un factor de 2, denotado por $\downarrow 2$, es aplicado a las salidas de los filtros. La operación de decimado consiste en descartar los elementos pares de la salida del filtro y es una operación no invertible. En la figura 2.7 se muestra un banco de filtros de análisis. Las salidas de los filtros L y H son los coeficientes *wavelet* de la trasformada, también conocidos como coeficientes de aproximación y de detalle, respectivamente.

Para señales de mayor complejidad no basta con sólo dos filtros, por lo que debe hacerse una descomposición de más niveles para separar las características relevantes y analizarlas independientemente. El procedimiento consiste en iterar el proceso de filtrado, es decir, aplicándolo a las señales de salida de la primera etapa y así sucesivamente hasta el nivel de precisión deseado. Lo anterior da origen a una descomposición multinivel conocida como árbol de descomposición *wavelet*, ejemplificado en la figura 2.8, donde cA_i representa los coeficientes de aproximación obtenidos de un filtro pasa bajas y cD_i los coeficientes de detalle

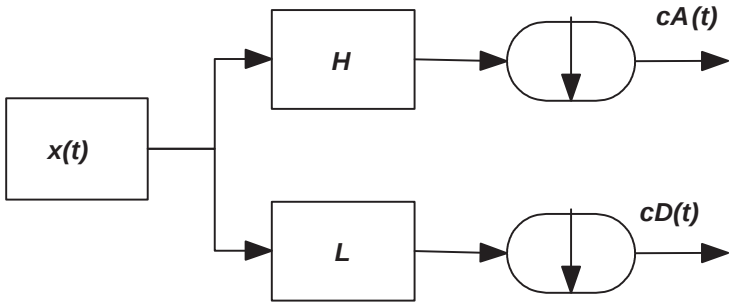


Figura 2.7: Banco de filtros de análisis. La salidas de los filtros L (pasa bajas) y H (pasa altas) son conocidas como coeficientes de aproximación y detalle, respectivamente

producto de un filtro pasa bajas a nivel i.

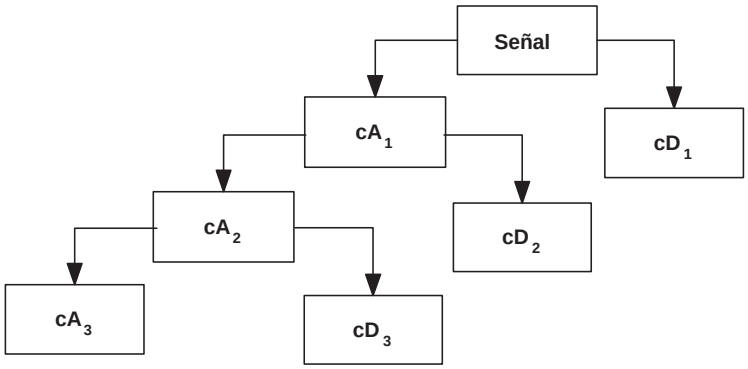


Figura 2.8: Descomposición *wavelet* de 3 niveles. El proceso de descomposición puede realizarse iterativamente hasta que sólo quede un ejemplo de la serie.

La reconstrucción de la señal original o Transformada Inversa Wavelet Discreta (iDWT) es realizada empleando un banco de filtros de síntesis. El procedimiento consiste en undecimar[10] por un factor de 2 la señal antes de pasar por los filtros de síntesis $L'(z)$ y $H'(z)$. Los filtros de síntesis están basados en los filtros del

[10]La operación de undecimado (*upsampling*) consiste en insertar ceros entre los valores de la señal de entrada y es denotado por el símbolo $\uparrow$.

banco de análisis (Merry, 2005). La señal reconstruida es obtenida sumando las salidas de los filtros de síntesis. En la figura 2.9 se muestra un banco de filtros de síntesis, las entradas de los filtros L' y H' son los coeficientes de aproximación y de detalle, respectivamente.

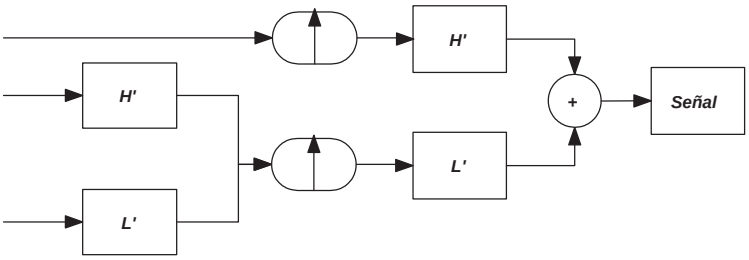


Figura 2.9: Banco de filtros de síntesis. Los filtros de síntesis $L'(z)$ y $H'(z)$ están basados en los filtros del banco de análisis y sirven para reconstruir una señal a partir de sus coeficientes *wavelet*. Las entradas de los filtros L' y H' son los coeficientes de aproximación y de detalle, respectivamente.

2.7. Trabajo relacionado

En esta sección se describen los trabajos relacionados con la predicción de series de tiempo empleando algunas de las redes neuronales descritas en las sección 2.3. La tabla 2.1 localizada al final de este capítulo, contiene un concentrado con los métodos descritos en la literatura científica.

2.7.1. Trabajos que emplean redes neuronales perceptrón multicapa

La mayoría de los trabajos encontrados de redes *feedforward* para predicción de series de tiempo utilizan la topología *MLP* (*Perceptrón Multicapa*) para aproximar una función que pase por los puntos dados en la serie de tiempo. Algunos trabajos de este tipo se describen a continuación:

Wong y colaboradores (2000) compararon experimentalmente el uso de MLP para predicción con diferentes algoritmos de entrenamiento y diferentes algoritmos para la generación de los pesos. La serie de tiempo empleada en sus experimentos fue *"Shanghai Stock Exchange"*, que consiste en los datos de comercio diario de 11 compañías entre 1994 y 1996. Los primeros 500 datos de la serie fueron usados como conjunto de entrenamiento, mientras que los 150 restantes conformaron el conjunto de prueba. Los algoritmos de entrenamiento empleados fueron gradiente descendente y gradiente conjugado, para la inicialización de los pesos emplearon inicialización aleatoria e inicialización de regresión lineal múltiple. Los algoritmos de entrenamiento y de inicialización de pesos fueron combinados entre sí, formando 4 diferentes redes. Los resultados de los experimentos señalan que en promedio el MSE de las 4 redes es menor a 0.00285. Sin embargo, la red entrenada con gradiente conjugado e inicialización de regresión lineal múltiple requiere de menos iteraciones que la otras combinaciones probadas. La estrategia de predicción utilizada no es definida por los autores.

Sarmiento y Villa (2008) presentan una aplicación de las redes neuronales en la tarea de pronóstico de demanda eléctrica en Colombia. Emplean redes MLP con el algoritmo de entrenamiento de retropropagación y redes de base radial (RBF). La serie de tiempo corresponde a la demanda de electricidad con resolución horaria de junio 2004 a junio 2007 suministrada por el Centro Nacional de Despacho (CND). Su propuesta consiste en un ensamble de 5 redes (MLP o RBF) entrenadas con diferentes conjuntos de entrenamiento, que corresponden a semanas con días festivos, o semanas comprendidas dentro de un periodo vacacional, entre otros. El conjunto de validación del método comprende el periodo de enero de 2006 a junio de 2007, donde el horizonte de predicción del método es de una semana. El tipo de estrategia empleado por los autores no es especificado. Los resultados obtenidos por el ensamble para una semana sin días festivos señalan un error porcentual (PE, del inglés percentage error) por debajo del 0.08 %, mientras que para una semana del periodo vacacional el error porcentual se ubica por debajo del 0.14 %.

De acuerdo a los autores las RBF presentaron un desempeño aceptable, pero no igualaron ni superaron a las MLP.

2.7.2. Trabajos que emplean redes neuronales *wavelets*

Utilizando la teoría de descomposición *wavelet*[11] se han desarrollado algunas representaciones de redes neuronales basadas en *wavelets*. Algunos de los trabajos encontrados emplean *wavelets* como función de activación y otros como preprocesamiento.

Entre las representaciones de redes *wavelets* tenemos el trabajo de Zhang y Benveniste (1992) en el que proponen una red *wavelet*. La estructura de la red tiene 2 capas y es llamada red neuronal de $1 + \frac{1}{2}$ capas. La figura 2.10 muestra este tipo de red, donde x es la entrada, g es la salida, b_i son los umbrales de cada neurona, σ son la funciones de activación (que en este caso es una función *wavelet*), a y w son los coeficientes de la primera y segunda capa, respectivamente. Aunque originalmente esta red fue diseñada como aproximador de funciones, ha sido empleada por otros autores para predecir series de tiempo. Algunos trabajos publicados sobre la aplicación de redes neuronales *wavelet* para la predicción de series de tiempo se detallan a continuación.

Alarcon-Aquino y colaboradores (2005) hacen uso de la red propuesta en (Zhang y Benveniste, 1992) empleando un algoritmo de entrenamiento no iterativo, semejante al entrenamiento de una red de base radial, para aproximar un conjunto de series de tiempo caóticas. Las experimentos se hicieron empleando una configuración 1-6-1 (una capa de entrada, seis nodos ocultos y un nodo de salida) en las redes de retropropagación (BPN) y *wavelet*. Se emplearon dos diferentes funciones de activación: la *wavelet* Battle-Lemarie y la *wavelet* Haar, en dos redes *wavelet* llamadas por los autores wavenet(1) y wavenet(2) respectivamente. Se emplearon 300 datos para entrenar la red y 200 para pruebas. Los resultados

[11] Ver sección 2.6.

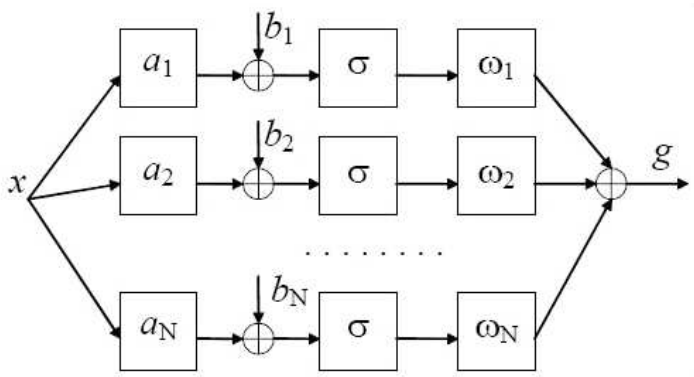


Figura 2.10: Red Neuronal llamada $1 + \frac{1}{2}$ capas. Este tipo de red *wavelet* emplea una wavelet como función de activación que es denotada por σ. x y g son las entradas y salidas de la red, respectivamente. b_i son los umbrales de cada neurona, a y w son los coeficientes de la primera y segunda capa, respectivamente. Imagen tomada de Garcia-Treviño y Alarcon-Aquino (2006).

mostrados en este trabajo señalan que las redes de tipo *wavelet* usadas tienen un mejor desempeño que las redes de tipo retropropagación. Lo anterior debido a que con la serie Lorentz en un horizonte de predicción de 200 puntos los errores de predicción (MSE) fueron 0.058, 0.060 y 0.062 obtenidos por wavenet(1), wavenet(2) y BPN respectivamente. El tipo de estrategia de predicción empleado no es especificado.

Garcia-Treviño y Alarcon-Aquino (2006) hacen uso de la red de Zhang; pero a diferencia de (Alarcon-Aquino y colaboradores, 2005) el algoritmo de entrenamiento es el gradiente estocástico, una modificación del algoritmo de retroprogación del error. Las series de tiempo empleadas en este trabajo son Mackey-Glass y Lorentz. Las configuraciones de la red de retroprogación (BPN) y la red wavelet son: 1-10-1 (una capa de entrada, diez nodos ocultos y un nodo de salida) y 1-15-1 (una capa de entrada, quince nodos ocultos y un nodo de salida). En el caso de la Mackey-Glass se usaron 500 puntos para el entrenamiento. El MSE de la BPN

y de la red *wavelet* fue de 0.645 y 0.0148 respectivamente empleando la configuración (1-10-1), mientras que para la configuración (1-15-1) el MSE fue de 0.0564 y 0.0041 empleando la BPN y la red wavelet respectivamente. Los resultados con la serie Lorentz son muy similares a los anteriores. Por tanto, Garcia-Treviño concluye que la redes de tipo *wavelet* son superiores a las redes *de alimentación progresiva* con funciones de activación de tipo sigmoidal para este tipo de señales. El tipo de estrategia de predicción empleado y el horizonte de predicción no son especificados por los autores.

2.7.3. Trabajos que emplean *máquinas de vectores de soporte*

Entre los trabajos para la predicción de series de tiempo empleando máquinas de vectores de soporte se encuentran:

Mukherjee y colaboradores (1997) investigaron experimentalmente el desempeño de una máquina de vectores de soporte en la estimación por regresión no lineal de un conjunto de series de tiempo caóticas, comparando los resultados obtenidos con otros métodos, como aproximación polinomial, funciones de base radial y redes neuronales. Según lo reportado, la máquina de vectores de soporte supera a las otras técnicas empleadas para comparar. El error durante las pruebas fue medido mediante la ecuación 2.51. De las pruebas realizadas podemos señalar la predicción a largo plazo de la serie de tiempo *Mackey-Glass*, de la que se emplearon 500 y 1,000 puntos para el entrenamiento y las pruebas, respectivamente. Lo resultados reportados señalan un error de 0.0044 de la SVM, mientras que los otros métodos (aproximación polinomial, función de base radial y red neuronal) obtuvieron un error superior al 0.0100.

Otros trabajos realizan modificaciones a la máquina de vectores de soporte para mejorar su desempeño. Espinoza y colaboradores (2005) tomando como información *a priori* la simetría de la serie de tiempo *K.U. Leuven* (Suykens, 2000),

modifican el *kernel* de la máquina de vectores de soporte de mínimos cuadrados añadiendo restricciones de simetría. Dicha restricción consiste en añadir una constante a de valor -1 o 1 al *kernel*

$$K(x_l, x_k) = \frac{1}{2}[K(x_l, x_k) + aK(-x_l, x_k)] \tag{2.49}$$

donde $K(x_l, x_k)$ es la función *kernel*, generalmente polinomial de grado n o una función de base radial. La serie de tiempo *K.U. Leuven* es generada por un circuito de Chua generalizado y consta de 2,000 datos, los cuales fueron empleados como conjunto de entrenamiento. La predicción es a largo plazo empleando una estrategia recursiva. Este trabajo obtuvo predicciones con un MSE de 0.24 en un horizonte de predicción de 200 puntos.

2.7.4. Trabajos que emplean redes neuronales recurrentes

Entre los trabajos que emplean redes neuronales recurrentes para la predicción de series de tiempo se encuentran los siguientes:

Cai y colaboradores (2007) proponen un algoritmo de aprendizaje, que combina optimización de enjambre (*swarm*) de partículas y algoritmos evolutivos (PSO y EA por sus siglas en inglés respectivamente). El algoritmo es empleado en el entrenamiento de una red neuronal recurrente de tipo *Elman* utilizada para la predicción de series de tiempo. Los autores afirman que su algoritmo muestra ventajas sobre el algoritmo de retropropagación a través del tiempo (BPTT) en cuanto a que evita quedar atrapado en mínimos locales. El predictor es probado con la serie *CATS Benchmark* (Lendasse y colaboradores, 2007), una serie de tiempo artificial de 5,000 datos, con 100 valores faltantes distribuidos en 5 bloques. La predicción de los valores se realiza mediante 5 redes, cada una de ellas empleando 980 valores de la serie. El horizonte de predicción de cada red es de 20 valores empleando una estrategia recursiva. El resultado de la predicción de los 100 valores faltantes tiene un MSE de 351.

Beliaev y Kozma (2007) implementan una red neuronal de tipo *caótica* basada en un modelo biológico llamado conjunto K_{III}. Este conjunto de redes es capaz de oscilar en un régimen aperiódico complejo, que puede ser referido como un régimen caótico. La red es entrenada empleando la regla de aprendizaje *hebbiano* modificada:

$$\Delta w_{ij} = \alpha(\sigma_i - \sigma)(\sigma_j - \sigma) \tag{2.50}$$

donde α es el coeficiente de aprendizaje, σ_i y σ_j son la desviación estándar de dos nodos i y j, Δw_{ij} es el cambio en el peso que conecta esos dos nodos. Los resultados de la red caótica se combinan con el método de vecinos más cercanos (kNN) para la predicción de los valores de la serie de tiempo *CATS Benchmark*. La predicción es a largo plazo empleando una estrategia directa. El conjunto de entrenamiento está formado por alrededor de 45 ejemplos de entrada para la red, donde cada ejemplo consiste en 10 puntos de la serie. El resultado de la predicción obtenida sobre los 100 valores de la serie tiene un MSE de 73.

Gao y colaboradores (2007) implementan una red recurrente de tipo espiral, con una topología de forma circular donde las recurrencias se encuentran entre los n vecinos más cercanos. Esta red emplea como algoritmo de entrenamiento el filtro de Kalman extendido basado en aprendizaje recurrente en tiempo real para el cálculo del gradiente. La predicción es a largo plazo empleando una estrategia recursiva. Las pruebas sobre la eficacia de la predicción se realizan empleando la serie de tiempo Lorentz en un horizonte de 15 puntos hacia el futuro, obteniendo un error de predicción iterativo (IPE, por sus siglas en inglés) promedio de 0.33. El error de predicción iterativo es definido como:

$$E_{IPE} = \frac{1}{n} \sum_{\tau=1}^{n} \frac{(x_\tau - \hat{x}_\tau)^2}{\sigma^2} \tag{2.51}$$

donde n es el número total de puntos predichos, x_τ es el valor real de la serie en el tiempo τ, $\hat{x}_\tau$ es el valor predicho y σ^2 es la varianza de la serie de tiempo.

2.8. Análisis de trabajos relacionados

Las líneas de investigación en el área de redes neuronales para predicción de series de tiempo son:

- Preprocesamiento de los datos. Consiste en la aplicación de estrategias para tratar de obtener información relevante de las series de tiempo, tales como tendencias y periodicidad, entre otros. Además, contempla la eliminación de valores atípicos a través de operaciones de filtrado. Esta investigación contempla el uso de la transformación lineal lineal para facilitar el aprendizaje del modelo neuronal, así como el análisis *wavelet* para encontrar patrones ocultos en la serie de tiempo. La eliminación de valores atípicos no fue considera debido a la hipótesis que establece que dichos valores pueden aportar información relevante para la predicción (Paggi, 2003).
- Algoritmos de aprendizaje. Las investigaciones en esta área están enfocadas al diseño/modificación de algoritmos de entrenamiento que sean eficientes y eficaces. Los algoritmos más comúnmente empleados son los basados en descenso de gradiente y filtros bayesianos. En esta investigación se utilizó el filtro de Kalman extendido basado en aprendizaje recurrente en tiempo real para el cálculo de gradiente debido a su facilidad de aprender relaciones temporales de series no estacionarias (Haykin, 1998).
- Diseño/modificación de modelos neuronales. El diseño de nuevos modelos neuronales se basa particularmente en información a priori del problema a resolver mediante la adición de variables que contemplen dicha información o bien inspirándose en sistemas biológicos conocidos. De los modelos de redes neuronales encontrados en la literatura los recurrentes son los que mejores resultados han obtenido en predicción de series de tiempo, debido a su capacidad de abstracción temporal (Mandic y Chambers, 2001). En este trabajo se aborda la modificación del modelo HCNN, propuesto por Gómez-Gil (2007) y descrito en la sección 2.4. Se empleó el modelo HCNN debido a

que esta basado en el análisis de Fourier. El análisis de Fourier nos permite obtener un conjunto de series sinusoidales, que facilita la aproximación de la serie analizada y por tanto, la predicción. El modelo HCNN se apoya en una amplia oscilación y periodicidad de la serie de tiempo (electrocardiograma). Al basarse en el análisis de Fourier para obtener información relevante que ayude a la predicción, la HCNN considera solamente información en el dominio de frecuencias sin considerar el dominio del tiempo. Mediante el modelo propuesto en esta investigación se pretende obtener una predicción más precisa, sin considerar la periodicidad de la serie de tiempo como imperativa. Por lo tanto, se considera que con el análisis *wavelet*, que proporciona información tanto de frecuencias como de tiempo (ver sección 2.6), se pueden obtener mejores resultados.

Tabla 2.1: Comparación de los predictores descritos en los trabajos previos y relacionados.

Nombre del método	**Tipo de predicción**	**Serie de tiempo utilizada**	**Horizonte de predicción**	**Criterio de evaluación**	**Resultado**
MLP, diferentes algoritmos de aprendizaje y de generación de pesos (Wong y colaboradores, 2000)	no definida	*"Shanghai Stock Exchange"*	150 datos	MSE	0.285
Ensamble de redes MLP y RBF (Sarmiento y Villa, 2008)	no definida	Demanda eléctrica del Centro Nacional de Despacho de Colombia	1 semana (7 datos)	PE	0.08 %
Red de Elman entrenada con PSO-EA (Cai y colaboradores, 2007)	recursiva	*CATS Benchmark*	100 valores en intervalos de 20 datos	MSE	351
Red neuronal Caótica (Beliaev y Kozma, 2007)	directa	*CATS Benchmark*	100 valores en intervalos de 20 datos	MSE	73
Red recurrente en espiral (Gao y colaboradores, 2007)	recursiva	*Lorentz*	15 valores	IPE	0.33
Red *wavelet* con algoritmo no iterativo (Alarcon-Aquino y colaboradores, 2005)	no especificada	*Lorentz*	200 puntos	MSE	0.058

continúa en la siguiente página

continuación de la página previa

Nombre del método	**Tipo de predicción**	**Serie de tiempo utilizada**	**Horizonte de predicción**	**Criterio de evaluación**	**Resultado**
Red *wavelet* con algoritmo de gradiente estocástico (Garcia-Treviño y Alarcon-Aquino, 2006)	no definida	*Mackey-Glass*	no definida	MSE	0.0564
SVM de regresión no lineal (Mukherjee y colaboradores, 1997)	no definida	*Mackey-Glass*	1000 puntos	IPE	0.0044
SVM de mínimos cuadrados con restricción de simetría (Espinoza y colaboradores, 2005)	no definida	*K.U. Leuven*	200 valores	MSE	0.24
Método propuesto	recursiva	11 series de NN5	56 valores	MSE sMAPE MASE	34.053 27.217 % 194.992

2.9. Resumen del capítulo

En este capítulo se han descrito las características principales de las series de tiempo, al igual que los dos tipos de predicción más comunes: la predicción de un paso y la predicción a largo plazo. Se habló de las diversas arquitecturas neuronales empleadas para la predicción de series de tiempo, incluyendo la *Red Neuronal Híbrida Compleja* propuesta por (Gómez-Gil, 1998) y en la cual se basa esta investigación. Se describen además, los algoritmos de aprendizaje y la teoría de *wavelet* empleados en el capítulo siguiente para realizar el entrenamiento de la arquitectura neuronal propuesta. Se presentaron algunos de los trabajos que emplean redes neuronales para la predicción de diversas series de tiempo.

En el siguiente capítulo se describe la arquitectura propuesta para predecir series de tiempo caóticas, así como el proceso de construcción de la misma.

Capítulo 3

Predictor neuronal propuesto

En este capítulo se describe el diseño de la arquitectura propuesta. Así como, la descripción de los pasos necesarios para su implementación. Dichos pasos van desde el preprocesamiento de la serie de tiempo hasta la predicción de valores futuros de la misma.

3.1. Descripción de la arquitectura

En la *HCNN*, descrita en la sección 2.4, pequeñas redes neuronales recurrentes son entrenadas para aprender información de la frecuencia de la serie de tiempo, correspondiente a los 7 primeras funciones seno de los armónicos de la señal. Estas pequeñas redes se conocen como generadores armónicos. Los generadores armónicos se conectan a un nivel escondido recurrente, a su vez conectado a un nivel de salida (ver figura 2.5). En el modelo propuesto en esta investigación, se sustituyen los generadores armónicos por pequeñas redes totalmente recurrentes capaces de generar las señales reconstruidas de los coeficientes *wavelets* (véase sección 2.6.3). El objetivo de las subredes es aprender las relaciones tiempo-frecuencia que componen la señal original, y de esta manera contribuir al aprendizaje y a la predicción de la señal analizada.

La arquitectura propuesta posee una capa de entrada, dos capas ocultas y una

capa de salida. La capa de entrada se conecta directamente a ambas capas ocultas. Las salidas de la primera capa oculta, correspondiente al conjunto de subredes recurrentes, alimentan a un segundo nivel escondido de neurones totalmente recurrentes, que a su vez se conectan a la capa de salida. En la figura 3.1 se puede apreciar el diseño de la arquitectura propuesta.

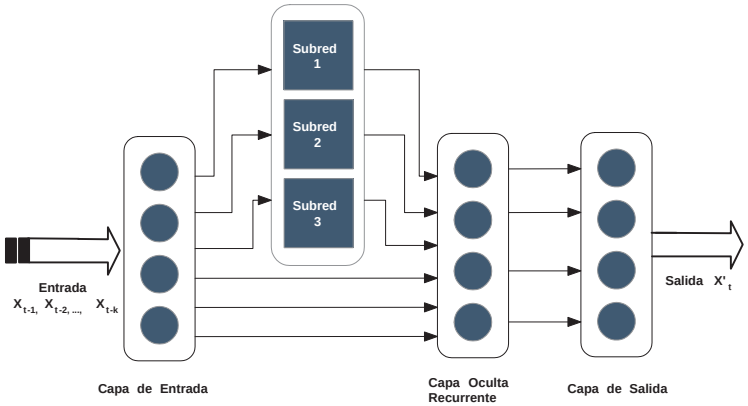


Figura 3.1: Arquitectura neuronal propuesta. La arquitectura consta de 4 capas: 1 de entrada, 2 ocultas y una de salida. La primer capa oculta está conformada por 3 subredes totalmente recurrentes. La segunda capa oculta presenta neurones totalmente recurrentes. La entradas de la red son k valores de la serie $x_{t-1}, x_{t-2}, \ldots, x_{t-k}$, la salida esperada es el siguiente valor de la serie x_t.

El número de neurones de la capa de entrada es 5, mientras que el de la capa de salida es igual a 1. El segundo nivel oculto posee 10 nodos. El número de nodos de los niveles fue determinado experimentalmente, mediante evaluación del comportamiento de aprendizaje. La cantidad de unidades establecida es la mínima suficiente para que las predicciones tengan un comportamiento oscilatorio. La función de activación es la tangente hiperbólica para todos los nodos, excepto para el nodo de salida, donde se emplea una función lineal.

3.2. Proceso de construcción del predictor

Para construir el predictor apropiado al tipo de señal, se siguen los siguientes pasos:

1. Escalamiento de la señal original.
2. Descomposición y reconstrucción de los coeficientes *wavelet* de la señal.
3. Entrenamiento de las subredes empleando como algoritmo de aprendizaje el *RTRL-EKF* (Haykin, 1998).
4. Integración y entrenamiento de la arquitectura completa con el algoritmo *RTRL-EKF*.
5. Predicción de los valores futuros de la serie.

En la figura 3.2 se muestra el esquema del proceso de construcción del predictor. A continuación se presenta una explicación detallada de cada uno de los pasos antes mencionados.

3.2.1. Escalamiento

Debido a que las funciones de activación de una red neuronal generalmente están definidas dentro del intervalo $(0, 1)$ en el caso de la función logística, o $(-1, 1)$ para la tangente hiperbólica, los datos de entrada deben ser escalados a estos rangos para facilitar el aprendizaje. Por tanto, estrategias de preprocesamiento como el escalamiento lineal o la normalización[1] son requeridas.

El escalamiento lineal se define como (Crone y colaboradores, 2006):

$$z_t = lb + \frac{x_t - \text{mín}(X)}{\text{máx}(X) - \text{mín}(X)}(ub - lb) \tag{3.1}$$

donde lb y up son respectivamente los límites inferior y superior del intervalo al que se desea escalar los datos, x_t es el punto de la serie de tiempo X que se desea escalar.

[1]También conocida como normalización de media cero, debido a que los valores de la serie después de la transformación siguen una distribución con media cero.

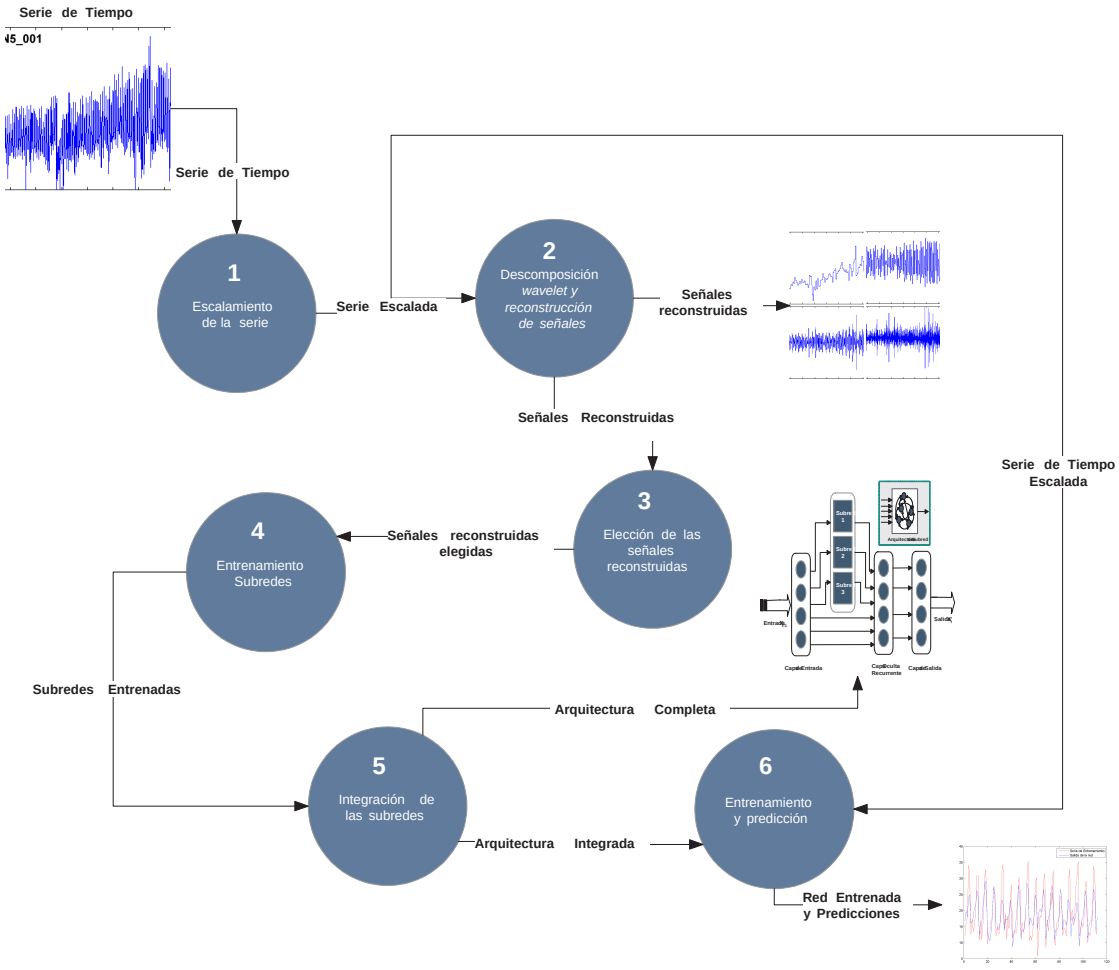


Figura 3.2: Proceso para construir la arquitectura propuesta. 1) El proceso comienza con el escalamiento lineal de la serie de tiempo. 2) La serie transformada es posteriormente descompuesta en coeficientes wavelet, con los coeficientes se reconstruyen 4 señales. 3) Se eligen 3 de las 4 señales reconstruidas. 4) Con cada señal se entrena una red totalmente recurrente. 5) Una vez entrenadas las subredes, se integran al modelo completo. 6) La arquitectura completa es entrenada y usada posteriormente para predecir valores futuros.

Por otra parte, la normalización emplea la media y la desviación estándar (DE) de la serie a escalar. La transformación de la serie mediante la normalización se define como:

$$z_t = \frac{x_t - E\{X\}}{DE(X)} \tag{3.2}$$

donde $DE(X)$ es la desviación estándar de la serie de tiempo X y $E\{.\}$ la media de la serie.

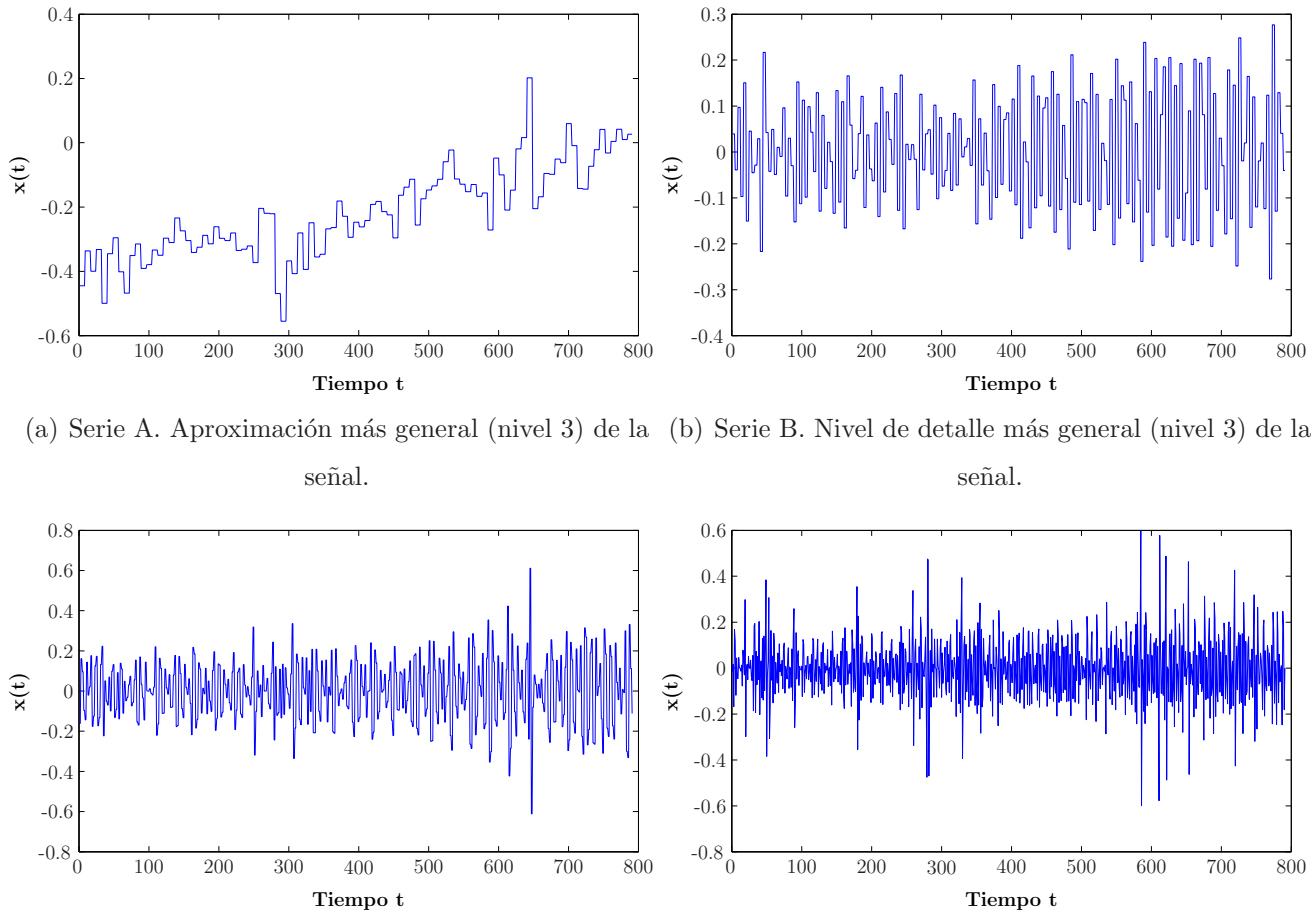
Crone y colaboradores (2006) determinaron experimentalmente que el escalamiento lineal en el intervalo $(-0.5, 0.5)$ es superior a la normalización, particularmente para series económicas. Basados en estos hallazgos, el escalamiento lineal en dicho intervalo es empleado en este trabajo.

3.2.2. Descomposición de la señal y reconstrucción de coeficientes *wavelet*

La descomposición *wavelet* permite separar una señal en varios componentes de menor resolución, lo cual es de utilidad si consideramos la complejidad de algunas de las series empleadas en esta investigación. Para este proyecto la serie es separada mediante un árbol de descomposición *wavelet* de 3 niveles (ver capítulo 2.6.3 para una explicación detallada).

El resultado de la descomposición es un conjunto de coeficientes a partir de los cuales se pueden obtener señales reconstruidas tanto de aproximación como de detalle. Tomando como referencia el árbol de descomposición *wavelet* mostrado en la figura 2.8, se obtuvieron las señales reconstruidas correspondientes a los coeficientes cA_3, cD_3, cD_2 y cD_1, dichas señales se identifican en este proyecto como A, B, C y D respectivamente. En la figura 3.3 se muestra un ejemplo de las señales reconstruidas a partir de la serie *NN5-001* (ver sección 4.2) obtenidas empleando la *wavelet Daubechies 1 (db1(.))*.

Para determinar la función *wavelet* a emplear en cada serie se realiza la descomposición y reconstrucción de la serie de tiempo empleando cada una de las



(a) Serie A. Aproximación más general (nivel 3) de la señal.

(b) Serie B. Nivel de detalle más general (nivel 3) de la señal.

(c) Serie C. Nivel 2 de detalle de la señal.

(d) Serie D. Nivel de detalle máximo (nivel 1).

Figura 3.3: Señales reconstruidas correspondientes a la serie NN5-001 empleando la *wavelet Daubechies 1*.

funciones *wavelet* disponibles en *Matlab* (Misiti y colaboradores, 2008) en el caso de la serie *Sumsin*. En el caso de las 11 series NN5 sólo se emplearon las funciones *wavelet* pertenecientes a la familia *Daubechies*[2] con el fin de disminuir el espacio de búsqueda y con ello reducir el tiempo de ejecución. Las señales reconstruidas de aproximación (A) son empleadas para entrenar una red neuronal totalmente recurrente de 11 neurones ocultos; con el objetivo de evaluar su capacidad de generalización prediciendo 56 valores de la serie reconstruida. La predicción con mejor ajuste a la señal de acuerdo a la medida de error MSE (ecuación 4.1) determina la *wavelet* a usar. El motivo de emplear la señal reconstruida de aproximación (A) para tomar esta decisión, es que posee la información de bajas frecuencias de la señal y por tanto, mayor información de cambios a largo plazo.

3.2.3. Entrenamiento de las subredes

Las subredes que integran la arquitectura propuesta son redes totalmente recurrente (véase figura 2.4). El número de nodos de cada subred es determinado experimentalmente evaluando su capacidad de generalización al predecir los siguientes 56 valores de las series reconstruidas.

Después de seleccionar el número de nodos de cada subred, se crean 15 redes. Cada subred es entrenada alrededor de 500 épocas para aprender las señales reconstruidas. El entrenamiento es realizado mediante el algoritmo RTRL-EKF, descrito en la sección 2.5.1. Posteriormente, se realiza una predicción de 56 valores con cada una de las subredes siendo la de menor MSE elegida para integrar la arquitectura.

Para disminuir el número total de neurones de la red integrada y con ello el

[2]Dicha familia de funciones se caracteriza por ser ortonormal y de soporte compacto. La función *wavelet* Daubechies de orden 1 es conocida como *wavelet* Haar y es una función escalón. Las funciones *wavelet* Daubechies de órdenes superiores no son fáciles de describir mediante una expresión analítica. El orden de las funciones Daubechies denota el número de momentos de desvanecimiento o momentos nulos de la función *wavelet*.

tiempo en el proceso de entrenamiento, sólo 3 señales reconstruidas son empleadas para entrenar. Se crean combinaciones de la señal de aproximación con 2 de las 3 señales de detalle, las señales de la combinación son sumadas. El resultado de la suma es restado a la señal original y la diferencia empleada para calcular el MSE de la combinación. La combinación con el menor MSE es entonces elegida.

3.2.4. Integración de la arquitectura

En este paso, las subredes entrenadas son incluidas en la arquitectura propuesta, lo cual implica que los pesos entrenados de las subredes son insertados en el primer nivel oculto de la arquitectura. Los pesos integrados pueden ser establecidos como fijos o modificables. En esta investigación, los pesos se declararon constantes.

3.2.5. Entrenamiento de la arquitectura propuesta

Durante la fase de entrenamiento, los pesos de las subredes no son modificados, con la idea de que el aprendizaje previo contribuya en el entrenamiento de la serie original. El entrenamiento de la red se realiza mediante el algoritmo de filtro de Kalman extendido basado en aprendizaje recurrente en tiempo real para el cálculo del gradiente. La figura 3.4 ilustra la fase de entrenamiento de la arquitectura propuesta.

3.2.6. Predicción de valores futuros

En la fase de predicción, las entradas de la arquitectura en un tiempo actual son las salidas de la red en un tiempo previo. Durante este procedimiento no son necesarias entradas externas, excepto por los valores de las condiciones iniciales de la red, las cuales son establecidas durante el entrenamiento. En este caso, el modelo de predicción es recursivo. Cabe mencionar que en esta fase, las subredes funcionan como generadores autónomos siendo alimentadas mediante sus propias salidas. La

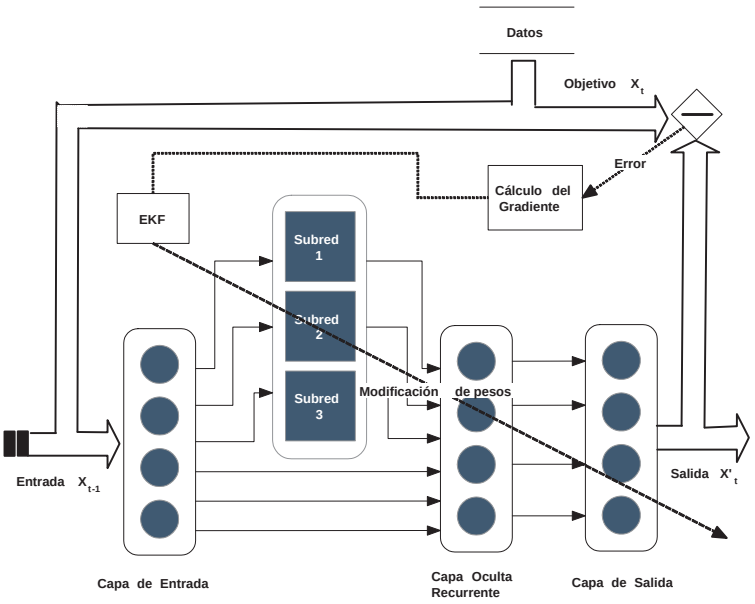


Figura 3.4: Entrenamiento de la arquitectura propuesta. El signo (−) indica la diferencia entre la salida de la red y el valor deseado. La diferencia es vista como el error a disminuir, para lo cual se calcula el gradiente. Después con el EKF se establecen los nuevos valores de los parámetros modificables de la red para reducir el error. La línea punteada indica la modificación de dichos parámetros.

fase de predicción se ilustra en la figura 3.5. Después de que las predicciones son generadas por la red, éstas son re-escaladas para situarlas en el intervalo original de la serie. La función de re-escalado se deriva de la ecuación 3.1 y se define como:

$$x_t = min(X) + \frac{z_t - lb}{ub - lb}(max(X) - min(X)) \tag{3.3}$$

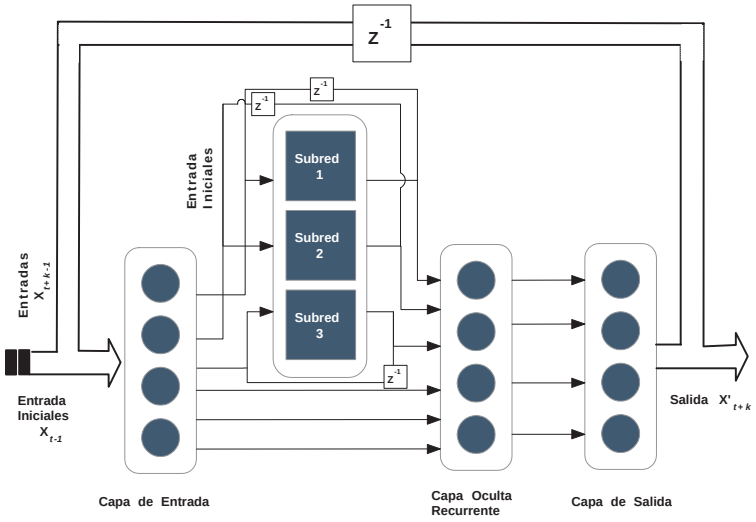


Figura 3.5: Predicción empleando la arquitectura propuesta. Durante la etapa de predicción la arquitectura es alimentada con los valores obtenidos por las últimas predicciones de las red. Las subredes funcionan como generadores autónomos siendo alimentadas por sus propias salidas. El modelo completo emplea el valor predicho en un tiempo pasado para calcular el valor en el tiempo actual.

3.3. Resumen del capítulo

En este capítulo se describió la arquitectura neuronal desarrollada, así como el proceso para la implementación. El proceso contempla cinco pasos: el escalamiento de la serie de tiempo, descomposición de la serie de tiempo y reconstrucción de señales a partir de los coeficientes *wavelet*, entrenamiento de las subredes, la

integración de las subredes al modelo completo, así como su entrenamiento y por último, la predicción de valores futuros. En el siguiente capítulo se hablará de los experimento realizados y de la evaluación los resultados obtenidos con el modelo de predicción desarrollado.

Capítulo 4

Experimentos y resultados

Este capítulo describe los experimentos y resultados obtenidos durante el desarrollo de la investigación, así como las medidas de error y series de tiempo empleadas para evaluar la eficacia del predictor.

La evaluación del modelo propuesto se lleva a cabo utilizando tres medidas de error frecuentemente empleadas en la literatura (Hyndman, 2006; Makridakis y colaboradores, 1983) para evaluar las predicciones hechas por diversos modelos de predicción. Las medidas de error son las siguientes: error cuadrático medio, error porcentual absoluto medio simétrico y error escalado absoluto medio. Dichas medidas evalúan el comportamiento global de las predicciones, sin considerar los errores máximos que usualmente son encontrados en valores extremos. Dicha consideración es tomada en concursos de predicción (Lendasse y colaboradores, 2007; Crone, 2008) debido a que la exactitud en dichos valores es difícilmente alcanzada y por tanto, el principal interés radica en encontrar modelos que sigan las tendencias de las series de tiempo analizadas en horizontes cada vez mayores[1].

Las predicciones hechas por tres modelos de redes neuronales son evaluadas de acuerdo a las medidas de error antes mencionadas. Los modelos neuronales empleados en esta investigación son: red *de alimentación progresiva*, red totalmente

[1]La longitud del horizonte varía dependiendo de los requerimientos y del problema (serie de tiempo) analizado.

recurrente y la arquitectura propuesta. Las series de tiempo a predecir son las series *Sumsin* y NN5 descritas en el apartado 4.2. Para cada serie y cada modelo de red neuronal, se ejecutaron 12 experimentos, cada uno con la misma configuración del modelo neuronal correspondiente, pero con diferentes pesos iniciales aleatorios.

4.1. Evaluación de la Precisión

La eficacia del modelo de predicción es calculada empleando las siguientes medidas estadísticas del error: error cuadrático medio (MSE, por sus siglas en inglés), error porcentual absoluto medio simétrico (sMAPE, del inglés *symmetric mean absolute percentage error*) y error escalado absoluto medio (MASE, del inglés *mean absolute scaled error*). A continuación se definen matemáticamente las medidas del error empleadas en este trabajo. En todas las ecuaciones, x_t representa al valor real de la serie en el tiempo t y $\hat{x}_t$ el valor predicho en el mismo tiempo.

4.1.1. Error Cuadrático Medio

El error cuadrático medio es una de las medidas de precisión más populares en modelado estadístico. Sin embargo, esta medida es dependiente de la escala en la que se encuentren los datos, siendo solamente útil para comprobar la precisión de predicción entre modelos en una misma serie. Cuando se trata de evaluar un modelo con diferentes conjuntos de datos suelen emplearse otras medidas como sMAPE o MASE. La ecuación 4.1 muestra el cálculo del error cuadrático medio.

$$E_{MSE} = \frac{1}{n}\sum_{t=1}^{n}(x_t - \hat{x}_t)^2 \tag{4.1}$$

4.1.2. Error Porcentual Absoluto Medio Simétrico

El error porcentual absoluto medio simétrico tiene la ventaja de ser independiente de la escala, por tanto es empleado frecuentemente para comparar el

desempeño de predicción entre diferentes series de tiempo (Crone, 2008; Hyndman, 2006). Esta medida de precisión es usada en la competencia de predicción de las series de tiempo *NN5* para determinar al ganador (Crone, 2008). El sMAPE se define como:

$$E_{SMAPE} = \frac{1}{n}\sum_{t=1}^{n}(\frac{|x_t - \hat{x}_t|}{(x_t + \hat{x}_t)/2})(100\,\%) \tag{4.2}$$

Esta medida tiene el inconveniente de que si el valor real x_t es cero, el valor predicho $\hat{x}_t$ probablemente será también cercano a cero, lo cual implicaría una división por un valor cercano cero. Además, si la serie original o la predicción contiene valores negativos, el valor del sMAPE puede ser negativo, lo cual resulta en una interpretación ambigua al tratarse de un porcentaje.

4.1.3. Error Escalado Absoluto Medio

El error escalado absoluto medio, propuesto por Hyndman (2006), es generalmente empleado para medir la precisión de la predicción de series de tiempo. Esta medida es independiente de la escala de los datos, por lo cual es empleada para evaluar modelos con diferentes series de tiempo. A diferencia del sMAPE, la única circunstancia en la cual el *MASE* podría ser infinito o indefinido es cuando todas las observaciones pasadas sean iguales. Además, el resultado del *MASE* es siempre positivo, lo cual evita ambigüedades. El error escalado absoluto medio se define como:

$$E_{MASE} = \frac{1}{n}\sum_{t=1}^{n}|\frac{x_t - \hat{x}_t}{\frac{1}{n-1}\sum_{i=2}^{n}|x_i - x_{i-1}|}| \tag{4.3}$$

4.2. Descripción de las series utilizadas

Las series de tiempo empleadas para esta investigación fueron las series *Sumsin* y *NN5*. La primera es una serie de tiempo empleada en la literatura para probar las capacidades del análisis *wavelet* para identificar diferentes frecuencias en el tiempo. La segunda se usa como caso de prueba de la capacidad de la arquitectura

propuesta para la predicción de series de tiempo caóticas y para comparar con otros modelos de predicción del concurso del cual provienen las series. En todas las series se emplean los primeros 679 valores como conjunto de entrenamiento, mientras que el conjunto de prueba está formado por los siguientes 56 datos. A continuación se describen las series de tiempo utilizadas.

4.2.1. Serie de Tiempo *Sumsin*

Para probar la validez del método de predicción se empleó como caso base la serie de tiempo *Sumsin* (Misiti y colaboradores, 2008). La serie está disponible como *benchmark* en la versión 7.4 de *Matlab* y es producida por la siguiente ecuación:

$$s(t) = \sin(3t) + \sin(0.3t) + \sin(0.03t) \tag{4.4}$$

donde la variable t indica el tiempo. En la figura 4.1 se muestra la evaluación de la serie *Sumsin* para $t = 1, 2, \ldots, 735$.

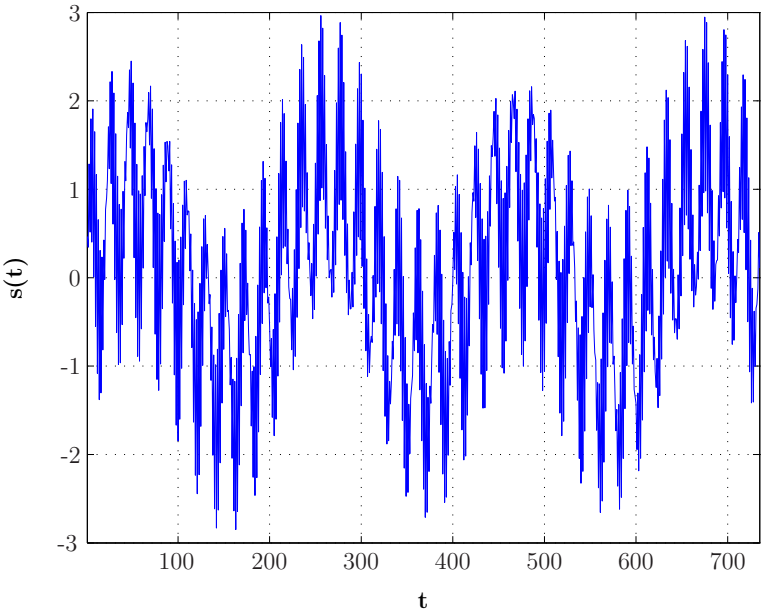


Figura 4.1: 735 puntos de la serie de tiempo *Sumsin*.

4.2.2. Series de Tiempo *NN5*

El conjunto de series de tiempo *NN5* proviene de una competencia de predicción del mismo nombre (Crone, 2008). Estas series contienen la información sobre el retiro de efectivo realizado diariamente en diversos cajeros automáticos de distintos lugares de Inglaterra. El conjunto consta de 111 series, cada una con una longitud de 735 valores correspondientes al periodo comprendido entre el 18 marzo de 1996 y el 22 de marzo de 1998. Las series pueden contener múltiples componentes estadísticos tales como estacionalidad, tendencias locales, valores atípicos, ceros y/o valores faltantes. La figura 4.2 muestra la gráfica de la primera serie del conjunto de datos *NN5*.

Para determinar la caoticidad de las series de tiempo empleadas, se procedió a calcular el máximo exponente de *Lyapunov* de cada una de ellas. La rutina empleada fue *lyap_spec* que implementa un método para el cálculo del espectro de *Lyapunov*, basado en el trabajo de Sano y Sawada (Kantz y Schreiber, 2003). La tabla 4.1 muestra los 10 primeros exponentes de *Lyapunov* de cada serie. La existencia de al menos un exponente positivo es un indicativo de la presencia de caos. Nótese que en las 11 series analizadas el exponente de *Lyapunov* mayor es positivo.

4.3. Experimentos realizados

El predictor se probó con series de tiempo de diferente complejidad a fin de evaluar su desempeño en circunstancias distintas. Estas series han sido descritas en la sección 4.2 de este documento.

Los experimentos se realizaron empleando dos computadoras con la versión 7.4 de *Matlab*. Las características de las computadoras son las siguientes:

1. Procesador de núcleo cuádruple a 2.33GHz con 2GB de memoria RAM.
2. Procesador de doble núcleo a 1.83GHz con 2GB de RAM.

Aproximadamente el 80 % de los experimentos realizados se llevaron a cabo en la computadora número 1. El software empleado en esta investigación se implementó en Matlab, empleando las librerías del algoritmo de aprendizaje disponibles al público (Cernansky, 2008).

4.3.1. Predicción de la serie *Sumsin*

Partiendo de la metodología de predicción propuesta en la sección 3.2, la señal *Sumsin* es descompuesta y reconstruida empleando la función *wavelet bior2.4(.)*. Las señales elegidas para entrenar las subredes son la A, B y D. En la figura 4.3 se muestran las señales reconstruidas. La cantidad de nodos ocultos de las subredes A, B y D son 11, 6 y 4 respectivamente; dichas cantidades fueron determinadas experimentalmente. El número de nodos de la segunda capa oculta de la arquitectura propuesta y a la que se conectan las subredes es 10.

Se realizaron un total de 12 experimentos empleando la arquitectura propuesta con diferentes pesos iniciales para predecir los 56 valores de prueba de la serie *Sumsin*. Las redes fueron entrenadas durante 300 épocas siendo el tiempo de ejecución de cada experimento en promedio de 1.2 horas. La tabla 4.2 contiene los resultados de cada experimento. Al finalizar los 12 experimentos, se obtuvieron los siguientes promedios y desviaciones estándar: MSE= 0.068 ± 0.039 y MASE= 54.243 ± 15.551.

Para comparar los resultados predichos por la arquitectura propuesta se entrenaron 2 tipos de redes neuronales: una red *de alimentación progresiva* y una totalmente recurrente. La red *de alimentación progresiva* consta de 5 nodos en la capa de entrada, 1 capa oculta de 26 nodos y un nodo de salida. La red totalmente recurrente está compuesta por 5 nodos de entrada, 26 nodos ocultos y una salida. La función de activación de los neurones en ambas redes es la tangente hiperbólica. El número de nodos en la capa oculta de ambas redes se definió siguiendo la heurística que recomienda no tener más pesos en una red que el número de datos

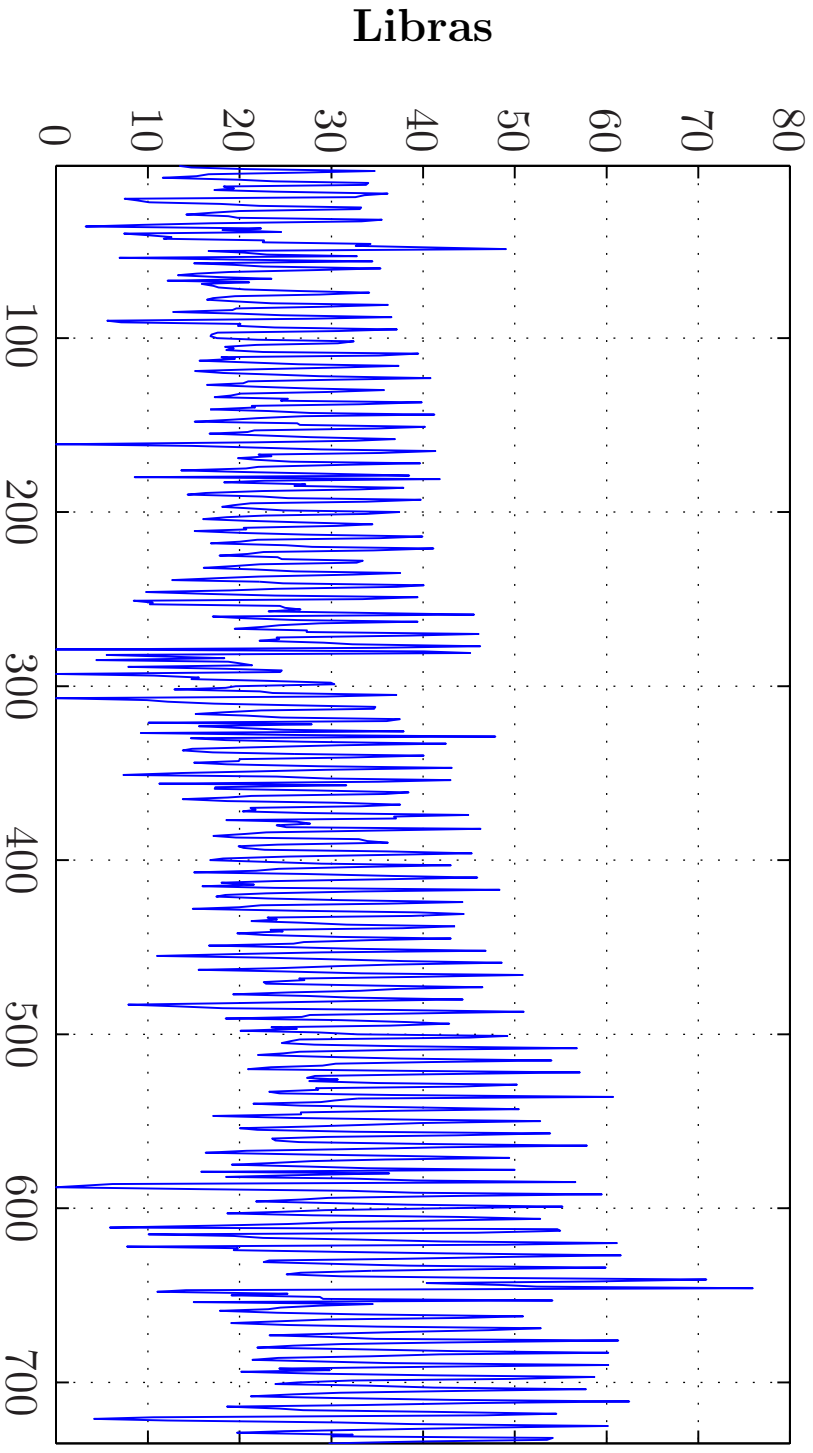
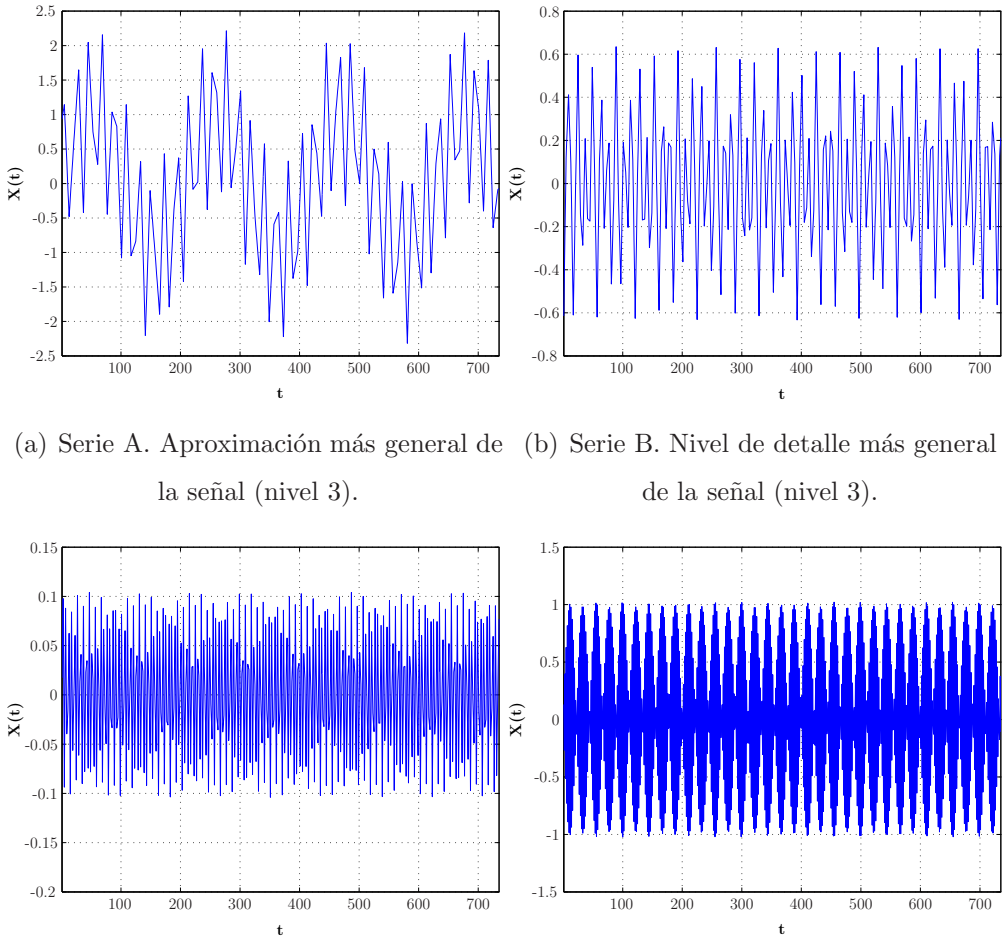


Figura 4.2: Gráfica de la serie de tiempo *NN5-001* (Datos tomados de Crone, 2008).

Tabla 4.1: Espectro de *Lyapunov* de las series *NN5*.

Serie de Tiempo	*Espectro de Lyapunov*									
	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9	λ_{10}
NN5-101	**0.0267**	0.0166	-0.0001	-0.0017	-0.0144	-0.0297	-0.0356	-0.0482	-0.0581	-0.0756
NN5-102	**0.6007**	0.2589	0.1666	0.1177	0.0862	0.0650	0.0588	0.0448	0.0318	0.0278
NN5-103	**0.0378**	0.0202	0.0069	-0.0053	-0.0079	-0.0218	-0.0347	-0.0462	-0.0558	-0.0695
NN5-104	**0.0565**	0.0401	0.0260	0.0295	0.0067	0.0026	-0.0016	-0.0119	-0.0248	-0.0309
NN5-105	**0.0486**	0.0251	0.0049	-0.0063	-0.0113	-0.0351	-0.0391	-0.0492	-0.0774	-0.0882
NN5-106	**0.0612**	0.0504	0.0373	0.0289	0.0190	0.0139	0.0028	-0.0053	-0.0091	-0.0148
NN5-107	**0.0678**	0.0532	0.0323	0.0276	0.0216	0.0155	0.0041	-0.0054	-0.0126	-0.0251
NN5-108	**0.0384**	0.0128	-0.0157	-0.0545	-0.0888	-0.1106	-0.1700	-0.2118	-0.3319	-0.7372
NN5-109	**0.8405**	0.3177	0.2004	0.1239	0.0992	0.0913	0.0682	0.0593	0.0506	0.0444
NN5-110	**0.0621**	0.0431	0.0243	0.0115	0.0040	-0.0155	-0.0198	-0.0249	-0.0481	-0.0498
NN5-111	**0.0220**	0.0220	-0.0093	-0.0151	-0.0409	-0.0440	-0.0640	-0.0726	-0.0958	-0.1065



(a) Serie A. Aproximación más general de la señal (nivel 3).

(b) Serie B. Nivel de detalle más general de la señal (nivel 3).

(c) Serie C. Nivel 2 de detalle de la señal.

(d) Serie D. Nivel 1 de detalle máximo.

Figura 4.3: Señales reconstruidas correspondientes a la serie *Sumsin* empleando la *wavelet bior2.4(.)*. Debido a su escala, la señal C es la que menos información aporta al reconstruir la señal original y por tanto, no es considerada para entrenar alguna subred.

Tabla 4.2: Resultados de la predicción de 56 valores de la serie *Sumsin*.

Número de Experimento	***Redes Feedforward***		***Redes Recurrentes***		***Arquitectura Propuesta***	
	MSE	**MASE**	**MSE**	**MASE**	**MSE**	**MASE**
1	0.112	60.827	0.236	100.004	0.11	73.184
2	0.089	49.823	0.083	56.638	0.092	66.314
3	0.07	47.931	0.036	40.038	0.037	41.687
4	0.099	58.184	0.191	86.501	0.034	41.807
5	0.061	48.613	0.023	34.78	0.029	37.863
6	0.165	80.531	0.09	53.566	0.04	43.689
7	0.096	59.478	0.003	12.399	0.132	78.45
8	0.049	49.816	0.521	107.336	0.052	49.518
9	0.104	74.631	0.146	78.909	0.054	49.677
10	0.063	55.017	0.064	53.904	0.116	67.361
11	0.063	50.018	0.087	73.61	0.099	68.531
12	0.16	84.994	0.086	61.378	0.021	32.832
Promedio	0.094	59.989	0.131	63.255	**0.068**	**54.243**
Desv. Est.	0.038	13.044	0.140	27.553	0.039	15.551

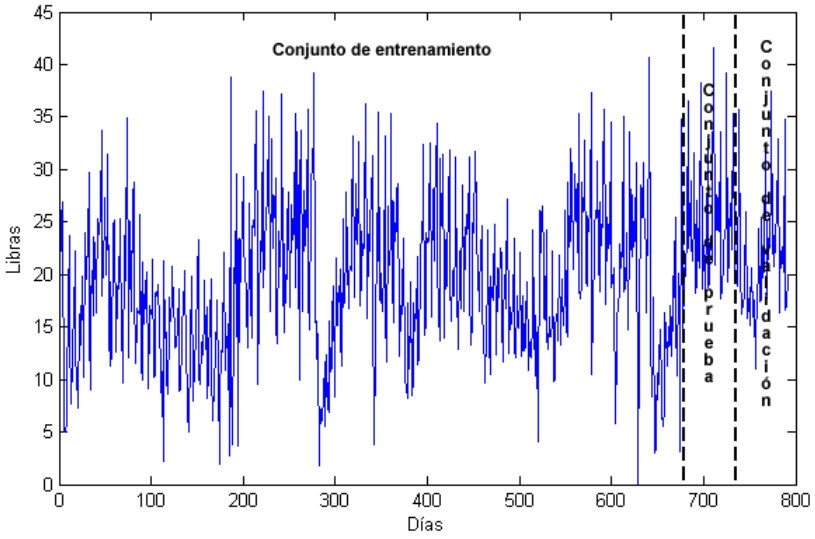


Figura 4.4: Conjuntos de entrenamiento, prueba y validación de la serie *NN5-101*.

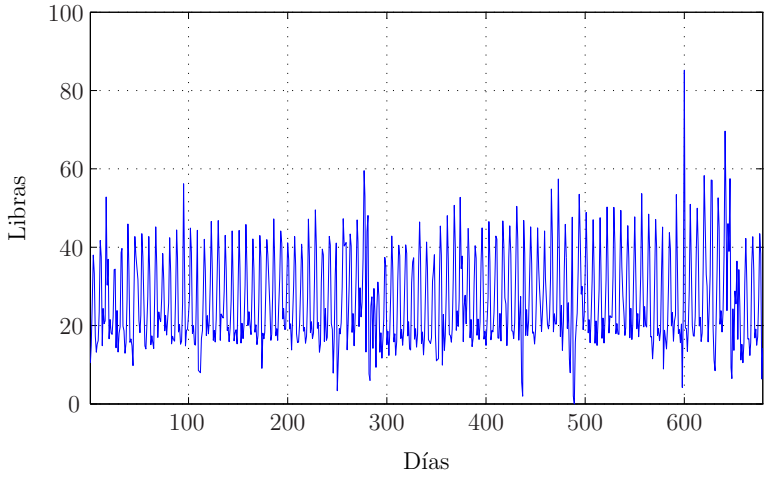
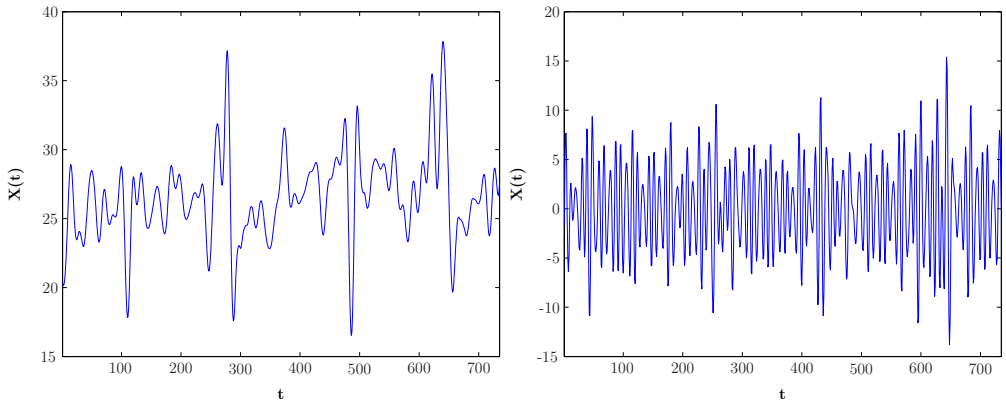
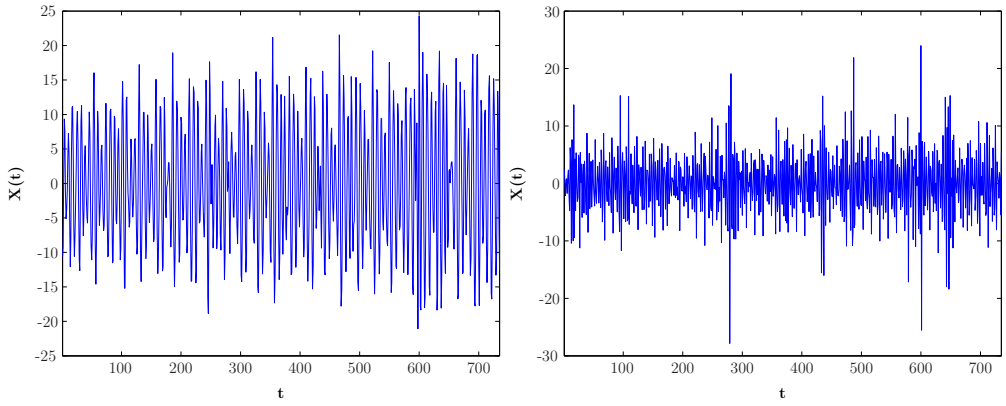


Figura 4.5: Gráfica de la serie de tiempo *NN5-109* (Datos tomados de Crone, 2008).



(a) Serie A. Aproximación más general de la señal (nivel 3).

(b) Serie B. Nivel de detalle más general de la señal (nivel 3).



(c) Serie C. Nivel 2 de detalle de la señal.

(d) Serie D. Nivel 1 de detalle máximo.

Figura 4.6: Señales reconstruidas correspondientes a la serie *NN5-109* empleando la *wavelet db10(.)*.

Tabla 4.3: Precisión de las predicciones del conjunto de prueba de la serie *NN5-109*. Los 56 valores pertenecen al conjunto de prueba.

Número de Experimento	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
	MSE	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	167.477	33.977	74.769	383.245	61.794	123.251	38.610	22.622	42.144
2	397.571	62.945	141.690	28.421	16.801	32.447	34.497	22.276	39.808
3	1,768.143	97.693	253.339	246.102	53.157	105.380	30.479	19.774	36.709
4	23.106	16.243	30.048	504.710	81.727	149.485	26.993	18.680	33.946
5	20.613	15.650	28.986	119.927	40.492	77.627	34.547	21.163	38.533
6	87.471	27.835	60.240	427.146	80.383	146.850	33.974	21.084	38.461
7	33.843	17.873	36.343	60.221	25.594	50.544	29.204	19.925	36.211
8	20.538	15.244	29.124	62.279	29.508	51.911	29.893	20.178	36.748
9	26.477	15.642	30.108	25.195	18.294	34.555	30.972	19.721	36.478
10	170.007	35.105	79.004	578.675	78.271	156.367	25.891	18.878	33.880
11	154.798	38.238	85.486	130.665	36.904	72.195	31.847	20.383	37.794
12	18.685	14.720	27.420	127.646	36.774	75.592	31.745	19.697	37.069
Promedio	240.727	32.597	73.046	224.519	46.642	89.684	31.554	20.365	37.315
Desv. Est.	493.700	25.047	66.446	198.009	23.937	45.315	3.517	1.221	2.304

Tabla 4.4: Resultados de la predicción del conjunto de validación de la serie *NN5-109* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	52.059	19.791	39.251
2	53.268	20.800	38.783
3	45.363	17.817	35.552
4	43.343	16.865	33.216
5	46.637	17.754	35.584
6	48.114	19.058	37.349
7	44.679	17.530	35.158
8	41.196	16.765	33.547
9	54.468	21.660	41.556
10	44.418	17.544	35.279
11	46.237	19.112	36.969
12	44.083	17.381	34.927
Promedio	46.989	18.507	36.431
Desv. Est.	4.188	1.579	2.454

Experimentos con la serie NN5-107

El peor de los casos, de acuerdo al promedio de la evaluación del error (%sMAPE) de las predicciones del conjunto de validación, se presenta en la serie NN5-107. La serie NN5-107 (ver figura 4.7) fue descompuesta y reconstruida empleando la función *wavelet db10*. Las señales reconstruidas elegidas para entrenar las subredes que forman la arquitectura propuesta fueron la A, C y D. La figura 4.8 muestra las señales reconstruidas empleadas. El número de nodos ocultos de las subredes A, C y D son 11, 8 y 4, respectivamente.

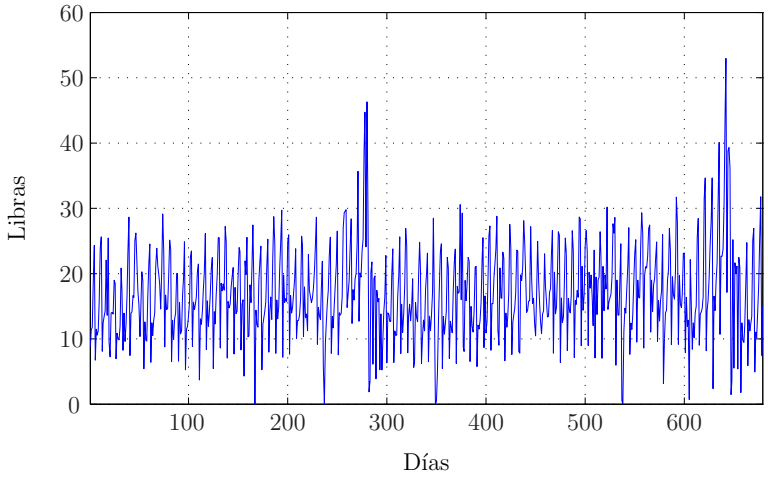
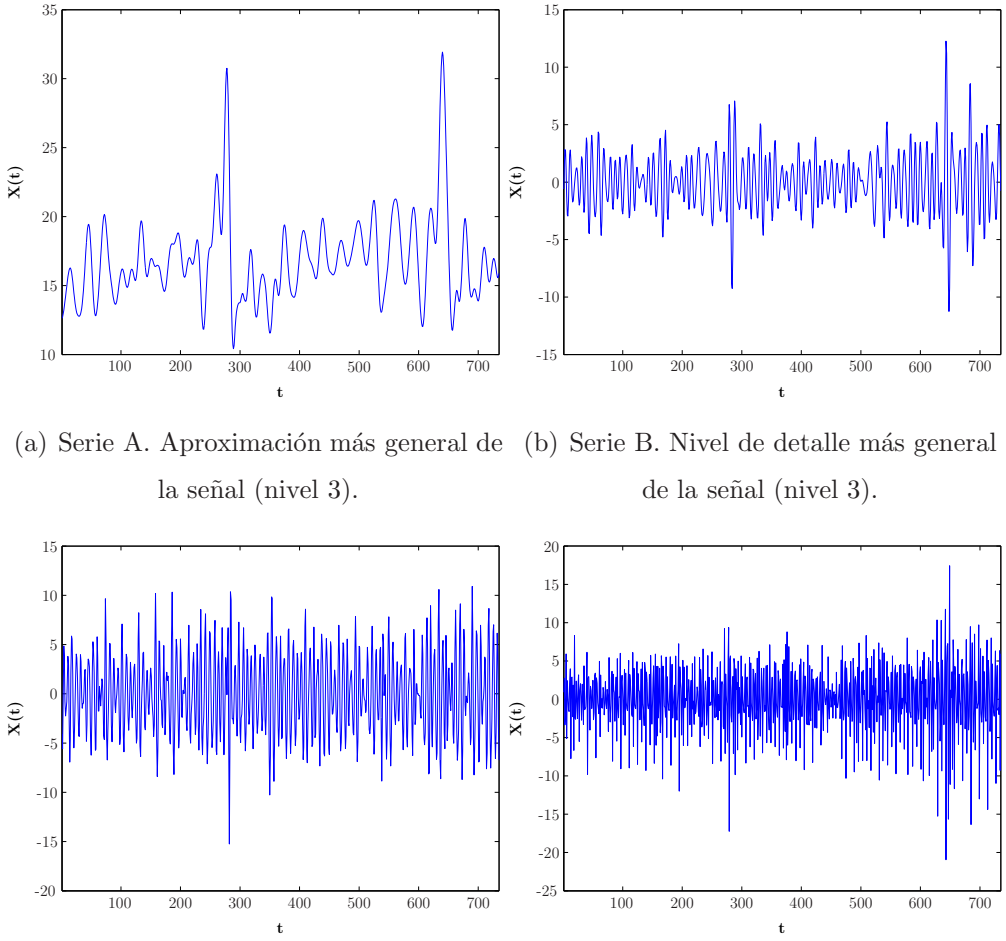


Figura 4.7: Gráfica de 679 valores de la serie de tiempo *NN5-107* (Datos tomados de Crone, 2008).

Después de integrar las subredes a la arquitectura propuesta, ésta es entrenada durante 50 épocas. Se realizaron 12 experimentos empleando la arquitectura propuesta, redes alimentadas hacia adelante y redes totalmente recurrentes. Los resultados obtenidos durante la predicción del conjunto de prueba empleando el predictor propuesto señalan errores promedio y desviaciones estándar de: MSE= 60.352 ± 91.831, %sMAPE= 40.543 ± 25.482 y MASE= 159.467 ± 109.365. Mientras que los errores promedios de las redes *de alimentación progresiva* son MSE= 107.87 ± 64.352, %sMAPE= 53.937 ± 14.100 y MASE= 240.842 ± 59.082;

de las redes totalmente recurrentes se obtuvieron los siguientes errores promedios: MSE= 187.560±114.126, %sMAPE= 62.201±21.731 y MASE= 294.838±108.063. Puede observarse que en el peor de los casos las predicciones del predictor propuesto son superiores en promedio a los modelos de redes conectadas hacia adelante y redes totalmente recurrentes. Los resultados de predicción con el conjunto de prueba se reportan en la tabla 4.5. La figura B.3 del apéndice B muestra la mejor predicción de cada arquitectura respecto a los valores reales del conjunto de prueba para la serie NN5-107.

Los errores promedios obtenidos por la arquitectura propuesta al predecir el conjunto de validación son los siguientes: MSE= 133.190 ± 216.226, %sMAPE= 53.168 ± 44.841 y MASE= 1488.116 ± 1339.254. En la tabla 4.6 se muestran los resultados obtenidos por el predictor propuesto al predecir el conjunto de validación de esta serie. La figura B.4 del apéndice B muestra la mejor predicción de la arquitectura propuesta para el conjunto de validación de la serie NN5-107.



(a) Serie A. Aproximación más general de la señal (nivel 3).

(b) Serie B. Nivel de detalle más general de la señal (nivel 3).

(c) Serie C. Nivel 2 de detalle de la señal.

(d) Serie D. Nivel 1 de detalle máximo.

Figura 4.8: Señales reconstruidas correspondientes a la serie *NN5-107* empleando la *wavelet db10(.)*.

Tabla 4.5: Resultados de la predicción de 56 valores de la serie *NN5-107*. Los 56 valores pertenecen al conjunto de prueba.

Número de	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	92.219	49.635	233.587	26.980	31.284	123.142	18.446	28.292	105.846
2	66.474	45.069	197.033	154.588	NaN	295.570	21.172	30.212	113.932
3	100.663	55.586	246.355	237.632	85.615	383.542	255.782	99.477	410.217
4	45.916	37.621	159.753	353.404	53.956	354.523	25.414	32.733	126.118
5	132.265	64.735	279.279	271.675	80.508	379.624	18.696	29.275	108.194
6	97.567	51.801	243.764	95.135	43.778	200.707	20.379	28.376	108.662
7	61.530	41.323	181.792	328.561	78.906	425.251	23.240	31.344	118.313
8	76.946	49.771	232.243	276.078	70.798	368.106	258.034	90.228	375.262
9	276.731	89.690	384.075	44.118	35.503	147.645	21.174	29.366	114.102
10	62.725	43.568	196.030	43.242	39.361	152.299	21.003	30.499	116.528
11	100.290	52.833	248.545	227.993	88.745	363.226	20.245	27.861	106.563
12	180.601	65.608	287.646	191.316	75.761	344.423	20.634	28.849	109.869
Promedio[a]	107.827	53.937	240.842	187.560	62.201	294.838	60.352	40.543	159.467
Desv. Est.	64.352	14.100	59.082	114.126	21.731	108.063	91.831	25.482	109.365

[a]Los promedios y desviaciones estándar de columnas con valores NaN fueron calculados excluyendo dichos valores

Tabla 4.6: Resultados de la predicción del conjunto de validación de la serie *NN5-107* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	35.314	31.761	844.628
2	36.551	32.641	864.543
3	566.012	148.066	4,273.663
4	35.729	34.196	900.704
5	33.872	31.573	818.062
6	35.166	31.251	842.244
7	36.742	32.671	859.143
8	623.323	149.228	4,404.146
9	77.614	48.499	1,347.522
10	42.159	33.228	947.131
11	37.772	32.080	875.664
12	38.031	32.818	879.946
Promedio	133.190	53.168	1488.116
Desv. Est.	216.226	44.841	1339.254

4.4. Análisis de resultados

La tabla 4.7 resume los resultados obtenidos por el predictor propuesto, redes *de alimentación progresiva* y recurrentes al evaluar la predicción de los 56 valores del conjunto de prueba para las series *Sumsin* y *NN5 reducido*. Los promedios de error MASE señalan que la arquitectura propuesta tiene un mejor desempeño al predecir el conjunto de prueba comparado con los modelos de redes *de alimentación progresiva* y redes totalmente recurrentes (véase tabla 4.7). Los tiempos promedio de entrenamiento (ver tabla 4.8) de cada modelo son: 1.9 minutos para las redes *de alimentación progresiva*, 3.9 horas para las redes totalmente recurrentes y 1.2 horas para la arquitectura propuesta. Los tiempos promedios de predicción (veáse tabla 4.9) son: 1.520, 3.801 y 2.397 segundos de los modelos de redes *de alimentación progresiva*, redes totalmente recurrentes y la arquitectura propuesta, respectivamente.

Los resultados de las predicciones del conjunto de prueba de las series *NN5 reducido* (ver tabla 4.10) señalan que las desviaciones estándar de los errores (%sMAPE) son mínimos, excepto para la serie NN5-107, lo que es un indicio de que las predicciones del modelo propuesto no varían en demasía. El promedio de los resultados de los errores obtenidos (%sMAPE) al predecir el conjunto de validación es 30.926 ± 10.386 el cual se sitúa entre los lugares 19 y 20 del concurso de predicción NN5 (véase tabla 4.12). Los promedios de error de las predicciones del conjunto de validación hechas por el predictor propuesto para cada serie de *NN5 reducido* se muestran en la tabla 4.11. En dicha tabla se puede observar que los errores más grandes se presentan en las series NN5-102, NN5-107 y NN5-110, siendo la NN5-107 la que muestra el mayor error. Por otra parte, en las series restantes el error (%sMAPE) se sitúa entre 18 y 35 %, siendo las mejores predicciones las de la serie NN5-109. Los errores obtenidos de todos los experimentos realizados con las diversas series se presentan en el apéndice A.

Al momento de la preparación de este documento, no es pública la descripción

detallada de los métodos que obtuvieron los primeros lugares en la competencia. Sin embargo, podría suponerse que estos métodos mejores al propuesto utilizan una combinación de diferentes estrategias. Debe hacerse notar que en esta competencia no considera en su análisis tiempos de aprendizaje y ejecución de los modelos involucrados.

Tabla 4.7: Comparación de los errores de predicción entre series empleando el promedio de 12 experimentos por serie. Los resultados de la serie *NN5* se presentan como el promedio de las 11 series analizadas.

Serie de Tiempo	**Precisión**	**Predictores**		
		Feedforward	***Recurrente***	***Arquitectura Propuesta***
Sumsin	MSE	0.094 ± 0.038	0.131 ± 0.140	0.068 ± 0.039
	MASE	59.989 ± 13.044	63.255 ± 27.553	54.243 ± 15.551
Promedio de las series del *NN5 reducido*	MSE	250.122 ± 226.046	198.693 ± 131.121	34.053 ± 20.116
	%sMAPE	49.276 ± 12.357	60.755 ± 13.050	27.217 ± 8.268
	MASE	517.505 ± 1079.671	546.314 ± 1218.950	194.992 ± 387.221

Tabla 4.8: Se promedian los tiempos de entrenamiento en segundos para cada serie. Los promedios toman los tiempos de 12 experimentos por cada serie.

Serie de Tiempo	***Feedforward***	***Recurrente***	***A. Propuesta***
Sumsin	116.293	12189.334	4172.256
NN5-101	120.622	14377.539	4485.296
NN5-102	111.185	14362.120	4478.583
NN5-103	112.319	14394.535	4242.688
NN5-104	111.709	14241.316	4400.140
NN5-105	111.440	14360.129	4070.943
NN5-106	116.950	14193.278	4210.881
NN5-107	122.672	14265.567	4457.868
NN5-108	119.079	14061.191	4396.104
NN5-109	118.294	13494.753	4479.746
NN5-110	116.916	13545.116	4327.870
NN5-111	109.260	15956.157	4017.856

Tabla 4.9: Se promedian los tiempos de predicción en segundos para cada serie. Los promedios toman los tiempos de 12 experimentos por cada serie.

Serie de Tiempo	***Feedforward***	***Recurrente***	***A. Propuesta***
Sumsin	1.849	3.614	1.917
NN5-101	1.934	4.116	2.788
NN5-102	1.679	4.273	2.634
NN5-103	1.758	4.387	2.181
NN5-104	1.743	3.190	2.940
NN5-105	1.392	4.199	1.841
NN5-106	1.655	4.138	2.326
NN5-107	1.171	2.907	2.258
NN5-108	1.706	2.797	2.719
NN5-109	1.032	3.746	2.754
NN5-110	1.277	4.899	2.024
NN5-111	1.046	3.351	2.388

Tabla 4.10: Errores de predicción del conjunto de prueba de las series NN5 empleando el promedio de 12 experimentos por serie.

Serie de tiempo	**MSE**	**%sMAPE**	**MASE**
NN5-101	17.181 ± 3.709	13.694 ± 1.770	117.338 ± 14.302
NN5-102	33.116 ± 2.124	27.691 ± 1.329	80.617 ± 5.437
NN5-103	4.642 ± 0.117	28.042 ± 0.514	25.698 ± 0.571
NN5-104	65.191 ± 7.775	26.370 ± 2.252	194.042 ± 18.518
NN5-105	30.685 ± 3.539	29.175 ± 1.290	13.219 ± 0.762
NN5-106	26.313 ± 9.330	24.140 ± 5.256	1348.732 ± 223.985
NN5-107	60.352 ± 91.831	40.543 ± 25.482	159.467 ± 109.365
NN5-108	33.950 ± 6.119	30.262 ± 2.270	104.184 ± 10.044
NN5-109	31.554 ± 3.517	20.365 ± 1.221	37.315 ± 2.304
NN5-110	59.789 ± 10.251	40.530 ± 2.489	37.886 ± 2.850
NN5-111	11.805 ± 1.541	18.574 ± 0.921	26.415 ± 1.517

Tabla 4.12: Resultados de la predicción de 56 valores del conjunto de 11 series del conjunto *NN5* (Crone, 2009).

Lugar	**Autor**	**sMAPE promedio**
1	Rauch	19.00 %
2	Dyakonov	19.90 %
3	Andrawis	20.50 %
4	Luna	21.10 %
5	Hung	21.30 %
6	Gutierrez	21.90 %
7	Wichard	22.40 %

Tabla 4.12: Resultados de la predicción de 56 valores del conjunto de 11 series del conjunto *NN5* (Crone, 2009) (*continuación página previa*).

Lugar	**Autor**	**sMAPE promedio**
8	Vogel	22.40 %
9	Puma-Villanueva	23.10 %
10	Gao	23.20 %
11	Pasero	23.20 %
12	Teddy	24.70 %
13	Fillon	25.40 %
14	Dang	25.40 %
15	Coyle	25.90 %
16	Rabie	28.20 %
17	Adeodato	29.90 %
18	undisclosed	30.20 %
19	undisclosed	30.60 %
20	undisclosed	31.50 %
21	Tung	34.60 %
22	undisclosed	34.70 %
23	undisclosed	35.50 %
24	undisclosed	37.60 %
25	Carbajal	40.10 %
26	undisclosed	40.20 %
27	undisclosed	40.50 %
28	undisclosed	47.40 %
29	undisclosed	50.30 %

Tabla 4.11: Errores de predicción del conjunto de validación de las series NN5 empleando el promedio de 12 experimentos por serie.

Serie de tiempo	**MSE**	**%sMAPE**	**MASE**
NN5-101	41.286 ± 15.872	25.858 ± 4.685	235.125 ± 38.597
NN5-102	52.740 ± 34.474	35.193 ± 9.268	743.213 ± 160.499
NN5-103	7.220 ± 0.183	24.830 ± 0.460	42.185 ± 0.790
NN5-104	105.199 ± 34.533	32.892 ± 6.608	108.496 ± 19.738
NN5-105	22.244 ± 2.502	23.054 ± 1.717	31.453 ± 2.343
NN5-106	77.115 ± 177.037	31.406 ± 34.135	91.796 ± 94.911
NN5-107	133.190 ± 216.226	53.168 ± 44.841	1488.116 ± 1339.254
NN5-108	38.840 ± 16.915	33.955 ± 9.569	86.905 ± 22.072
NN5-109	46.989 ± 4.188	18.507 ± 1.579	36.927 ± 2.454
NN5-110	206.702 ± 4.793	37.681 ± 0.648	53.937 ± 0.984
NN5-111	25.369 ± 1.304	23.644 ± 0.736	64.573 ± 1.998

4.5. Resumen del capítulo

En este capítulo se abordaron los experimentos realizados durante el desarrollo de la investigación, los cuales incluyen pruebas con un total de 12 series de tiempo: 11 del concurso de predicción NN5 y una empleada comúnmente en la literatura para probar las capacidades del análisis *wavelet*, conocida como *Sumsin*. Los experimentos con la serie *Sumsin*, además del mejor y peor caso de las 11 series de NN5 fueron descritos en este apartado. En el siguiente capítulo se hablará de las principales contribuciones de esta investigación y de los trabajos futuros.

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

En la presente tesis se desarrolló una arquitectura neuronal capaz de predecir valores futuros de series de tiempo inspirada en la red neuronal híbrida compleja (HCNN) y basada en subredes generadoras de señales reconstruidas de coeficientes *wavelets*. La precisión de las predicciones hechas por la arquitectura propuesta fue analizada utilizando las medidas de error MSE, sMAPE y MASE.

La arquitectura propuesta se comparó con arquitecturas *de alimentación progresiva* y totalmente recurrentes empleando las series *Sumsin* y *NN5*. En total se realizaron 432 experimentos con las 3 arquitecturas utilizando 11 series de NN5 y la serie Sumsin. Los resultados obtenidos señalan que nuestro método es superior a los modelos clásicos de redes neuronales alimentadas hacia adelante y totalmente recurrentes en la predicción de 56 valores del conjunto de prueba de las series antes mencionadas. Al utilizar el predictor con el conjunto de validación del *"benchmark NN5 reducido"* pudo observarse que los resultados obtenidos por la arquitectura propuesta son razonables, ya que el promedio de error sMAPE de las 11 series de *NN5 reducido* es igual a 30.926 %, que se encuentra entre los lugares 19 y 20 del concurso de predicción internacional (Crone, 2008). Por lo cual, nuestra apro-

ximación empleando información de las señales reconstruidas con *wavelets* puede considerarse adecuada para este tipo de problema. Otra conclusión importante es que el algoritmo de entrenamiento RTRL-EKF presentó un mejor comportamiento que el BPTT en pruebas simples de entrenamiento de redes totalmente recurrentes.

La aportación principal de esta investigación ha sido la creación de un modelo de predicción cuya estructura toma ventaja de la información de descomposición del análisis *wavelet* y de las capacidades oscilatorias de las redes neuronales recurrentes. A diferencia del modelo HCNN que emplea la solamente la información de frecuencias producto del análisis de Fourier, el modelo propuesto emplea información tanto de frecuencia como de tiempo. Si bien el modelo propuesto no obtiene los mejores resultados, las predicciones siguen la tendencia de los valores reales de las series.

5.2. Trabajo futuro

Con base en los resultados obtenidos, existen algunas variantes que se podrían seguir como trabajo futuro. Primeramente, los resultados de la arquitectura neuronal podrían ser mejorados si se incluye la subred con la información de la señal reconstruida que no se consideró durante los experimentos (ver sección 3.2.2). Sin embargo, habrá que considerar el costo computacional de agregar más nodos a la red, ya que en la realización de los experimentos mostrados aquí se requirió alrededor de 1.2 horas de tiempo de cómputo por cada uno de ellos. Otra posible línea para mejorar los resultados es emplear una suma ponderada con las salidas de las redes neuronales entrenadas, con la finalidad de que se aproveche el conocimiento del conjunto de redes entrenadas.

Bibliografía

Addison P. S. (2002). *The Illustrated Wavelet Transform Handbook: Introductory Theory and Applications in Science, Engineering, Medicine and Finance*. IOP Publishing, UK.

Akay M. (2000). *Nonlinear Biomedical Signal Processing Vol. II: Dynamic Analysis and Modeling*. Wiley-IEEE Press, New York, NY.

Alarcon-Aquino V., Garcia-Trevino E. S., Rosas-Romero R., y Ramirez-Cruz J. F. (2005). Learning and approximation of chaotic time series using wavelet-networks. En *Sixth Mexican International Conference on Computer Science, 2005. ENC 2005*, pp. 182–188.

Beliaev I. y Kozma R. (2007). Time series prediction using chaotic neural networks on the cats benchmark. *Neurocomputing*, 70(13-15):2426–2439.

Burges C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167.

Cai X., Zhang N., Venayagamoorthy G. K., y Wunsch II D. C. (2007). Time series prediction with recurrent neural networks trained by a hybrid PSO-EA algorithm. *Neurocomputing*, 70(13-15):2342–2353.

Catsigeras E. (2001). La teoría matemática del caos determinista. *Lapzus de cultura*, 2:22–28.

Cernansky M. (2008). Michal Cernansky's Homepage. Downloads. Disponible en `http://www2.fiit.stuba.sk/~cernans/main/download.html`. Última consulta realizada en enero de 2009.

Crone S. F. (2008). NN5 forecasting competition for artificial neural networks & computational intelligence. Disponible en `http://www.neural-forecasting-competition.com/`. Última consulta realizada en Marzo de 2009.

Crone S. F. (2009). NN5 forecasting competition results. Disponible en `http://www.neural-forecasting-competition.com/NN5/results.htm`. Última consulta realizada en Julio de 2009.

Crone S. F., Guajardo J., y Weber R. (2006). The impact of preprocessing on support vector regression and neural networks in time series prediction. En *Proceedings of the 2006 International Conference on Data Mining, DMIN 2006, Las Vegas, Nevada, USA, June 26-29, 2006*, pp. 37–44.

Doya K. (2002). *Handbook of Brain Theory and Neural Networks, 2nd edition.*, cap. Recurrent Networks Learning: Algorithms, pp. 1–18. MIT Press, Cambridge.

Edmonds A. N. (1996). *Time Series Prediction Using Supervised Learning and Tools from Chaos Theory*. Tesis doctoral, University of Luton.

Espinoza M., Suykens J., y De Moor B. (2005). Short term chaotic time series prediction using symmetric LS-SVM regression. En *Proc. of the 2005 International Symposium on Nonlinear Theory and Applications (NOLTA)*, pp. 606–609.

Gao H. (2008). *Distributed Learning in Sensor Networks: an online-trained spiral recurrent neural network, guided by an evolution framework, making duty-cycle reduction more robust*. Tesis doctoral, Department of Database and Information Systems, Institute for Computer Science, University of Munich.

Gao H., Sollacher R., y Kriegel H.-P. (2007). Spiral recurrent neural network for online learning. En *European Symposium on Artificial Neural Networks*, pp. 483–488.

Garcia-Treviño E. S. y Alarcon-Aquino V. (2006). Chaotic time series approximation using iterative wavelet-networks. En *CONIELECOMP '06: Proceedings of the 16th International Conference on Electronics, Communications and Computers*, pp. 19, Washington, DC, USA. IEEE Computer Society.

Gómez-Gil P. (1998). *The Effect of Non-linear Dynamic Invariants in Recurrent Neural Networks for Prediction of Electrocardiograms*. Tesis doctoral, Texas Tech University.

Gómez-Gil P. (2007). Long term prediction, chaos and artificial neural networks. Where is the meeting point? *Engineering Letters*, 15(1).

Graps A. (1995). An introduction to wavelets. *Computing in Science and Engineering*, 2(2):50–61.

Gujarati D. N. (2002). *Basic Econometrics*. Mcgraw-Hill, Boston.

Haykin S. (1998). *Neural Networks: A Comprehensive Foundation (2nd Edition)*. Canada. Prentice Hall.

Hyndman R. J. (2006). Another look at forecast accuracy metrics for intermittent demand. *Foresight: The International Journal of Applied Forecasting*, (4):43–46.

Ito K. (1987). *Encyclopedic Dictionary of Mathematics: Second Edition*. The MIT Press.

Kantz H. y Schreiber T. (2003). *Nonlinear Time Series Analysis*. Cambridge University Press, New York, NY, USA.

Lendasse A., Oja E., Simula O., y Verleysen M. (2007). Time series prediction competition: The CATS benchmark. *Neurocomputing*, 70(13-15):2325–2329.

Lendasse A., Wertz V., Simon G., y Verleysen M. (2004). Fast Bootstrap applied to LS-SVM for Long Term Prediction of Time Series. En *IJCNN 2004, International Joint Conference on Neural Networks*, pp. 705–710, Budapest (Hungary).

Lillekjendlie B., Kugiumtzis D., y Christophersen N. (1994). Chaotic time series part II: System identification and prediction. *Modeling, Identification and Control*, 15(4):225–243.

Makridakis S. G., Wheelwright S. C., y McGee V. E. (1983). *Forecasting: Methods and Applications*. John Wiley & Sons, Inc., New York, NY, USA.

Mallat S. G. (1989). A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(7):674–693.

Mandic D. P. y Chambers J. (2001). *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability*. John Wiley & Sons, Inc., New York, NY, USA.

Merry R. (2005). Wavelet theory and applications: a literature study. Dct 2005.053, Eindhoven: Technische Universiteit Eindhoven.

Misiti M., Misiti Y., Oppenheim G., y Poggi J. M. (2008). *Wavelet Toolbox 4 User's Guide*. The MathWorks, 4.3 edition.

Mukherjee S., Osuna E., y Girosi F. (1997). Nonlinear prediction of chaotic time series using support vector machines. En *IEEE Workshop on Neural Networks for Signal Processing VII*, pp. 511–519. IEEE Press.

Oldham W. (1989). Artificial neural networks, an introduction. Course Notes. Texas Tech University, Spring 1989. Internal Document. Lubbock TX USA.

Oldham W. y Gómez-Gil P. (1997). Application of recurrent neural networks for the prediction on the behavior of biological oscillators. En *Proceedings of the*

VII International Conference on Electronics, Communications and Computers. CONIELECOMP 97, Puebla, México.

Paggi H. (2003). Predicción de series temporales utilizando redes neuronales. Tesis de Maestría, Instituto de Computación. Facultad de Ingeniería. Universidad de la República. Montevideo, Uruguay.

Palit A. K. y Popovic D. (2005). *Computational Intelligence in Time Series Forecasting: Theory and Engineering Applications (Advances in Industrial Control).* Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Pearlmutter B. A. (1989). Learning state space trajectories in recurrent neural networks. *Neural Comput.*, 1(2):263–269.

Rosas-Orea M. C. E. (2006). Método para la reducción de ruido en imágenes utilizando la transformada wavelet compleja con un algoritmo de umbral óptimo. Tesis de Maestría, Universidad de las Américas Puebla. Escuela de Ingeniería y Ciencias Departamento de Computación, Electrónica, Física e Innovación.

Särkkä S., Vehtari A., y Lampinen J. (2007). CATS benchmark time series prediction by Kalman smoother with cross-validated noise density. *Neurocomputing*, 70(13-15):2331–2341.

Sarmiento H. O. y Villa W. M. (2008). Artificial intelligence in forecasting demands for electricity: an application in optimization of energy resources. *Revista Colombiana de Tecnologías de Avanzada*, 2(12):94–100.

Smola A. J. y Schölkopf B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222.

Suykens J. A. K. (2000). The K.U. Leuven competition data: a challenge for advanced neural network techniques. En *8th European Symposium on Artificial Neural Networks. ESANN2000. Proceedings*, pp. 299–304.

Welch G. y Bishop G. (1995). An introduction to the Kalman Filter. Reporte técnico, Chapel Hill, NC, USA.

Werbos P. J. (1994). *The roots of backpropagation: from ordered derivatives to neural networks and political forecasting*. Wiley-Interscience, New York, NY, USA.

Williams R. J. y Zipser D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.*, 1(2):270–280.

Williams R. J. y Zipser D. (1995). Gradient-based learning algorithms for recurrent networks and their computational complexity. pp. 433–486.

Wong C.-C. C., Chan M.-C., y Lam C.-C. (2000). Financial time series forecasting by neural network using conjugate gradient learning algorithm and multiple linear regression weight initialization. Computing in Economics and Finance 2000 61, Society for Computational Economics.

Zemke S. (2003). *Data Mining for Prediction. Financial Series Case*. Tesis doctoral, The Royal Institute of Technology, Sweden.

Zhang Q. y Benveniste A. (1992). Wavelet networks. *IEEE Transactions on Neural Networks*, NN-3(6):889–898.

Apéndice A

Tablas detalladas de resultados

A continuación se presentan los resultados de las predicciones de 56 valores de las series de tiempo descritas en las sección 4.2.

Tabla A.1: Resultados de la predicción de 56 valores de la serie *NN5-101*. Los 56 valores pertenecen al conjunto de prueba.

Número de	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	47.206	20.868	184.888	180.561	59.222	367.342	16.452	13.736	119.909
2	39.839	20.790	175.622	54.026	30.009	217.607	15.329	13.167	112.955
3	524.732	89.431	689.952	202.748	61.176	416.582	18.247	13.914	118.203
4	100.759	38.757	296.362	79.639	30.724	241.738	18.026	13.583	118.443
5	392.787	99.409	587.614	163.028	55.051	385.836	19.059	14.177	122.646
6	79.038	32.198	257.055	75.480	36.498	269.020	17.221	14.878	124.706
7	423.899	94.074	644.393	76.286	34.838	247.056	11.703	10.849	93.275
8	66.309	27.364	220.039	78.579	40.124	272.467	16.987	12.818	111.072
9	61.296	24.593	212.365	176.491	55.143	389.739	14.163	11.747	104.390
10	70.210	29.015	246.543	217.470	64.371	408.380	16.067	13.953	114.846
11	318.681	92.562	528.065	38.565	21.005	176.716	15.757	13.375	113.743
12	59.073	23.305	201.810	56.834	25.101	213.245	27.161	18.130	153.873
Promedio	181.986	49.364	353.725	116.642	42.772	300.477	17.181	13.694	117.338
Desv. Est.	178.436	33.305	197.235	65.459	15.324	86.718	3.709	1.770	14.302

Tabla A.2: Resultados de la predicción de 56 valores de la serie *NN5-102*. Los 56 valores pertenecen al conjunto de prueba.

Número de Experimento	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
	MSE	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	1,664.262	106.620	539.760	425.361	87.504	297.438	32.931	28.982	85.202
2	216.432	59.615	237.032	48.332	26.460	84.935	31.701	28.195	83.079
3	965.868	91.113	379.860	33.326	29.500	85.512	36.837	30.191	89.560
4	83.023	40.960	131.957	189.609	43.631	152.825	29.797	25.854	70.948
5	334.743	59.883	255.018	72.843	37.809	126.030	33.659	27.211	76.774
6	1,488.246	124.436	567.891	244.359	72.191	277.307	32.184	26.088	74.611
7	692.609	79.593	336.179	91.182	44.865	142.914	33.362	28.520	83.162
8	88.181	34.443	120.750	241.020	56.346	239.464	36.484	28.980	86.327
9	182.377	65.642	219.248	40.161	28.373	85.061	32.022	27.740	81.246
10	164.613	57.480	213.094	372.308	79.449	292.811	33.415	26.829	80.523
11	131.447	38.788	148.950	39.506	23.234	71.502	30.486	26.318	74.789
12	689.809	90.360	344.318	64.360	33.018	107.452	34.514	27.384	81.188
Promedio	558.468	70.744	291.171	155.197	46.865	163.604	33.116	27.691	80.617
Desv. Est.	554.458	28.152	148.083	137.724	22.056	88.079	2.124	1.329	5.437

Tabla A.3: Resultados de la predicción de 56 valores de la serie *NN5-103*. Los 56 valores pertenecen al conjunto de prueba.

Número de	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	8.259	35.965	32.297	53.633	68.094	78.854	4.563	27.675	25.318
2	12.918	42.471	43.723	34.895	70.667	69.913	4.556	27.479	25.155
3	14.199	40.490	43.840	11.251	50.067	38.426	4.589	27.127	24.664
4	6.364	32.422	30.594	51.576	75.190	73.789	4.851	28.801	26.540
5	5.942	30.470	28.780	32.575	51.779	60.226	4.690	27.898	25.661
6	6.443	33.394	31.561	13.467	55.431	42.753	4.741	28.356	26.162
7	10.718	37.276	38.322	12.054	37.518	39.097	4.658	28.336	25.880
8	9.640	36.244	36.132	10.940	48.472	36.101	4.654	28.246	25.915
9	14.059	56.003	40.757	7.951	35.337	32.059	4.489	27.925	25.451
10	16.722	44.067	49.268	27.554	71.362	56.979	4.656	28.436	26.124
11	6.512	28.241	27.828	22.950	65.582	53.625	4.795	28.667	26.392
12	8.709	37.630	37.060	29.547	106.760	65.685	4.463	27.556	25.112
Promedio	10.040	37.889	36.680	25.699	61.355	53.959	4.642	28.042	25.698
Desv. Est.	3.664	7.380	6.758	15.623	19.467	16.067	0.117	0.514	0.571

Tabla A.4: Resultados de la predicción de 56 valores de la serie *NN5-104*. Los 56 valores pertenecen al conjunto de prueba.

Número de	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	808.803	160.063	762.961	415.420	72.200	549.928	56.779	23.254	166.904
2	257.480	56.347	415.403	925.702	67.727	679.385	54.304	23.307	167.808
3	68.501	30.043	197.573	689.839	81.148	650.071	70.401	26.302	200.736
4	415.651	60.495	457.886	339.694	50.985	430.595	57.388	23.905	176.314
5	135.617	43.963	310.713	281.147	46.721	381.486	62.267	26.331	191.320
6	142.535	42.139	294.340	561.224	71.015	570.273	74.811	30.038	219.225
7	162.798	39.882	302.167	263.986	70.185	425.743	65.697	28.009	204.610
8	111.519	38.288	288.846	180.877	50.485	342.340	79.345	28.935	222.375
9	142.143	42.654	310.137	932.472	59.602	615.948	73.102	28.912	212.600
10	51.250	23.214	166.432	575.013	91.616	600.306	61.113	25.962	187.116
11	112.221	35.356	263.586	364.337	56.230	421.119	62.752	25.252	188.502
12	399.656	55.952	419.459	845.874	99.602	734.206	64.338	26.237	190.991
Promedio	234.015	52.366	349.125	531.299	68.126	533.450	65.191	26.370	194.042
Desv. Est.	216.072	35.617	156.067	265.987	16.566	128.812	7.775	2.252	18.518

Tabla A.5: Resultados de la predicción de 56 valores de la serie *NN5-105*. Los 56 valores pertenecen al conjunto de prueba.

Número de	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	27.862	29.618	13.231	199.697	67.178	30.311	34.038	30.183	13.944
2	103.890	49.593	24.457	111.100	56.451	25.104	30.939	29.461	13.371
3	59.528	37.629	18.135	127.423	63.228	26.533	27.437	29.045	12.817
4	31.972	30.291	13.577	215.394	73.992	35.231	30.559	29.752	13.348
5	48.952	31.776	15.023	86.553	55.637	22.677	32.924	29.561	13.478
6	46.852	32.606	15.149	169.511	76.942	33.176	25.979	27.689	12.161
7	82.825	42.431	20.481	75.497	50.930	19.928	30.293	29.366	13.309
8	76.181	41.500	20.358	161.475	70.463	29.191	35.671	30.660	14.146
9	31.190	30.052	13.244	191.566	74.853	33.452	25.988	27.396	12.209
10	67.459	39.011	18.956	117.049	74.714	27.121	27.192	27.010	12.168
11	220.634	54.157	29.435	144.842	75.571	29.014	36.094	31.230	14.436
12	34.441	30.584	13.847	64.168	47.472	19.359	31.109	28.741	13.235
Promedio	69.315	37.437	17.991	138.689	65.619	27.592	30.685	29.175	13.219
Desv. Est.	53.192	8.204	5.084	50.027	10.556	5.170	3.539	1.290	0.762

Tabla A.6: Resultados de la predicción de 56 valores de la serie *NN5-106*. Los 56 valores pertenecen al conjunto de prueba.

Número de Experimento	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
	MSE	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	75.476	39.847	2,246.898	179.970	55.402	3,178.198	35.896	30.636	1,626.646
2	639.756	74.532	5,421.208	77.801	46.244	2,105.596	19.611	20.980	1,230.992
3	441.944	95.163	5,061.793	314.501	88.509	4,809.764	15.654	17.390	1,057.496
4	251.618	111.690	4,349.505	536.058	90.217	5,975.170	22.010	21.846	1,237.902
5	179.235	55.551	3,203.803	377.504	85.695	5,203.830	23.845	21.635	1,261.794
6	114.309	48.673	2,947.729	192.000	68.654	3,426.619	25.996	24.258	1,363.529
7	277.991	57.794	3,808.288	448.811	101.522	5,808.814	20.098	20.697	1,247.844
8	323.923	71.870	4,309.647	122.642	53.112	2,624.611	18.850	19.244	1,142.392
9	600.468	99.661	5,819.825	419.124	71.205	5,048.187	47.670	33.841	1,805.430
10	193.928	85.210	3,735.753	213.918	70.052	3,718.135	35.502	30.035	1,585.691
11	10.506	15.295	826.331	284.701	65.479	4,162.369	20.542	20.715	1,178.766
12	131.149	50.168	3,125.771	266.652	72.792	4,271.659	30.084	28.401	1,446.301
Promedio	270.025	67.121	3738.046	286.140	72.407	4194.413	26.313	24.140	1348.732
Desv. Est.	200.346	27.822	1401.661	138.704	16.558	1227.983	9.330	5.256	223.985

Tabla A.7: Resultados de la predicción de 56 valores de la serie *NN5-108*. Los 56 valores pertenecen al conjunto de prueba.

Número de	*Redes FeedForward*			*Redes Recurrentes*			*Arquitectura Propuesta*		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	528.029	NaN	380.038	92.314	66.644	209.365	28.890	30.847	102.930
2	90.422	46.692	186.778	160.606	49.585	223.028	38.683	31.090	109.282
3	71.740	46.798	188.327	34.227	26.738	93.687	29.976	27.301	91.510
4	1,284.521	NaN	678.001	58.811	42.609	147.720	42.278	32.074	114.671
5	590.117	NaN	388.519	77.539	54.687	191.577	28.048	28.408	94.309
6	635.970	NaN	401.375	127.955	60.917	219.806	31.174	29.829	102.532
7	1,712.267	89.302	826.922	29.836	28.116	97.024	45.235	32.422	116.728
8	675.775	NaN	469.962	169.066	75.296	281.572	29.536	27.529	93.033
9	420.364	NaN	342.393	127.887	NaN	253.392	27.615	29.003	97.221
10	1,885.739	NaN	864.295	55.022	43.550	155.093	35.395	29.294	102.883
11	67.909	43.485	171.920	33.266	41.608	115.235	30.620	30.054	101.613
12	1,334.158	NaN	807.577	195.109	79.872	287.056	39.950	35.291	123.495
Promedio[a]	774.751	56.569	475.509	96.803	51.784	189.546	33.950	30.262	104.184
Desv. Est.	632.297	21.876	256.524	57.927	17.613	67.887	6.119	2.270	10.044

[a]Los promedios y desviaciones estándar de columnas con valores NaN fueron calculados excluyendo dichos valores

Tabla A.8: Resultados de la predicción de 56 valores de la serie *NN5-110*. Los 56 valores pertenecen al conjunto de prueba.

Número de	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	85.838	40.598	42.592	178.674	66.466	62.340	67.346	43.669	41.489
2	83.606	39.914	48.674	268.831	62.676	75.607	59.944	42.422	37.429
3	86.373	38.743	47.728	203.487	73.359	71.071	52.711	38.872	36.795
4	104.945	38.156	49.478	145.179	45.508	51.969	58.281	39.095	37.897
5	143.077	47.576	61.712	118.606	49.023	47.037	76.076	42.408	42.080
6	83.183	40.681	48.563	320.675	69.614	77.090	53.642	40.778	36.196
7	41.252	25.986	30.495	166.294	79.829	61.914	78.873	44.035	42.770
8	169.626	49.243	68.157	322.745	54.778	82.374	47.670	36.795	34.471
9	122.627	42.950	55.997	267.467	68.883	72.785	48.661	40.620	34.248
10	90.097	35.925	43.764	80.934	39.850	41.735	50.433	36.338	35.737
11	87.467	40.485	48.373	293.514	91.451	80.073	59.476	41.771	38.514
12	99.373	37.544	48.492	176.841	62.439	59.273	64.349	39.556	37.011
Promedio	99.789	39.817	49.502	211.937	63.656	65.272	59.789	40.530	37.886
Desv. Est.	32.869	5.848	9.478	80.848	14.742	13.374	10.251	2.489	2.850

Tabla A.9: Resultados de la predicción de 56 valores de la serie *NN5-111*. Los 56 valores pertenecen al conjunto de prueba.

Número de	***Redes Feedforward***			***Redes Recurrentes***			***Arquitectura Propuesta***		
Experimento	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**	**MSE**	**% sMAPE**	**MASE**
1	937.377	84.232	166.243	239.835	116.345	119.113	11.716	18.886	26.931
2	538.691	56.957	110.311	211.871	83.259	104.969	10.005	16.966	23.972
3	39.446	28.916	44.416	113.725	84.234	87.116	9.685	17.950	24.796
4	290.564	108.456	112.673	466.204	100.222	152.930	13.192	19.770	27.916
5	28.311	29.138	40.506	180.253	81.365	96.448	11.960	18.338	26.165
6	423.140	56.966	89.868	266.006	86.509	114.845	13.319	19.439	27.935
7	73.396	42.933	65.379	218.725	133.173	125.163	14.392	19.905	28.843
8	12.856	18.086	25.246	95.977	53.299	71.286	13.549	19.468	27.939
9	12.991	18.604	25.454	198.719	97.196	106.690	11.084	17.824	25.480
10	11.281	16.536	22.874	107.835	70.245	79.815	9.911	17.710	24.877
11	38.961	35.975	49.299	171.439	72.659	102.587	11.465	18.376	26.334
12	45.731	33.596	50.717	263.115	64.006	108.044	11.383	18.258	25.793
Promedio	204.395	44.200	66.916	211.142	86.876	105.751	11.805	18.574	26.415
Desv. Est.	292.765	28.287	44.255	98.976	22.312	21.690	1.541	0.921	1.517

Tabla A.10: Resultados de la predicción del conjunto de validación de la serie *NN5-101* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	35.964	24.038	224.010
2	32.096	22.981	210.422
3	35.420	24.586	222.491
4	34.359	23.808	215.782
5	43.934	26.676	243.202
6	36.672	23.289	219.193
7	33.764	23.343	213.274
8	33.441	23.629	215.561
9	32.544	23.041	210.539
10	86.833	37.703	331.492
11	34.076	24.009	218.879
12	56.334	33.190	296.652
Promedio	41.286	25.858	235.125
Desv. Est.	15.872	4.685	38.597

Tabla A.11: Resultados de la predicción del conjunto de validación de la serie *NN5-102* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	49.603	34.614	746.282
2	48.102	32.742	733.505
3	48.614	33.594	727.875
4	161.285	64.141	1,240.656
5	44.520	32.132	689.311
6	41.688	32.847	693.595
7	42.601	33.836	718.218
8	44.048	33.787	718.374
9	37.398	28.827	639.748
10	35.052	30.010	632.424
11	39.323	32.100	680.242
12	40.650	33.688	698.322
Promedio	52.740	35.193	743.213
Desv. Est.	34.474	9.268	160.499

Tabla A.12: Resultados de la predicción del conjunto de validación de la serie *NN5-103* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	7.259	25.095	42.495
2	6.922	24.362	41.368
3	6.992	24.149	40.986
4	7.259	24.827	42.212
5	7.428	25.819	43.987
6	7.353	24.544	41.975
7	7.464	25.283	43.012
8	7.404	24.958	42.521
9	7.099	24.326	41.449
10	7.317	24.722	42.091
11	7.056	24.919	42.073
12	7.087	24.953	42.056
Promedio	7.220	24.830	42.185
Desv. Est.	0.183	0.460	0.790

Tabla A.13: Resultados de la predicción del conjunto de validación de la serie *NN5-104* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	57.226	23.498	79.266
2	63.119	25.284	84.441
3	147.833	43.173	137.334
4	72.248	26.647	89.621
5	126.498	37.367	124.755
6	122.233	38.678	119.505
7	80.197	28.387	96.618
8	160.427	39.116	129.101
9	97.185	30.401	102.370
10	121.684	36.550	120.911
11	80.589	27.448	92.871
12	133.151	38.160	125.154
Promedio	105.199	32.892	108.496
Desv. Est.	34.533	6.608	19.738

Tabla A.14: Resultados de la predicción del conjunto de validación de la serie *NN5-105* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	23.001	23.832	32.748
2	21.680	22.564	30.285
3	19.372	21.418	29.119
4	21.730	22.682	31.113
5	27.624	27.006	36.145
6	21.220	21.656	30.032
7	22.613	23.617	32.185
8	22.468	23.595	32.241
9	21.857	22.824	30.513
10	25.845	24.892	34.931
11	21.142	21.940	30.098
12	18.379	20.619	28.019
Promedio	22.244	23.054	31.453
Desv. Est.	2.502	1.717	2.343

Tabla A.15: Resultados de la predicción del conjunto de validación de la serie *NN5-106* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	48.047	34.829	88.819
2	20.331	18.108	56.656
3	18.032	18.547	57.160
4	23.786	19.939	61.729
5	23.800	19.205	60.407
6	21.394	19.027	60.727
7	23.919	18.709	60.053
8	21.152	18.495	57.854
9	638.553	138.461	391.483
10	41.177	30.576	81.384
11	21.749	19.608	60.237
12	23.446	21.370	65.048
Promedio	77.115	31.406	91.796
Desv. Est.	177.037	34.135	94.911

Tabla A.16: Resultados de la predicción del conjunto de validación de la serie *NN5-108* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	30.665	30.723	78.625
2	59.265	46.225	114.810
3	23.904	24.709	63.420
4	50.129	41.828	104.735
5	28.097	27.146	71.720
6	27.702	27.710	72.671
7	78.430	54.490	133.919
8	24.770	25.838	67.717
9	25.419	26.532	69.357
10	39.410	33.194	88.648
11	31.161	28.636	76.098
12	47.125	40.430	101.139
Promedio	38.840	33.955	86.905
Desv. Est.	16.915	9.569	22.072

Tabla A.17: Resultados de la predicción del conjunto de validación de la serie *NN5-110* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	212.293	38.551	55.152
2	207.459	37.338	53.550
3	206.808	37.963	54.436
4	206.886	38.170	54.635
5	194.457	36.591	52.271
6	210.003	37.585	53.639
7	211.174	38.594	55.451
8	206.444	37.565	53.722
9	203.251	37.039	52.977
10	206.634	37.272	53.307
11	211.164	38.383	54.984
12	203.854	37.122	53.117
Promedio	206.702	37.681	53.937
Desv. Est.	4.793	0.648	0.984

Tabla A.18: Resultados de la predicción del conjunto de validación de la serie *NN5-111* utilizando la arquitectura propuesta.

No. de Experimento	**MSE**	**% sMAPE**	**MASE**
1	25.902	24.045	65.742
2	24.611	23.350	63.771
3	27.712	24.663	67.279
4	24.214	22.773	62.399
5	24.891	23.271	63.488
6	23.984	23.402	63.800
7	26.738	24.112	66.122
8	25.670	23.844	64.998
9	23.538	22.383	60.997
10	25.880	24.265	65.872
11	26.899	24.656	67.466
12	24.392	22.969	62.945
Promedio	25.369	23.644	64.573
Desv. Est.	1.304	0.736	1.998

Apéndice B

Gráficas del mejor y peor caso

A continuación se presentan las gráficas del mejor y peor caso de las predicciones hechas por el método propuesto.

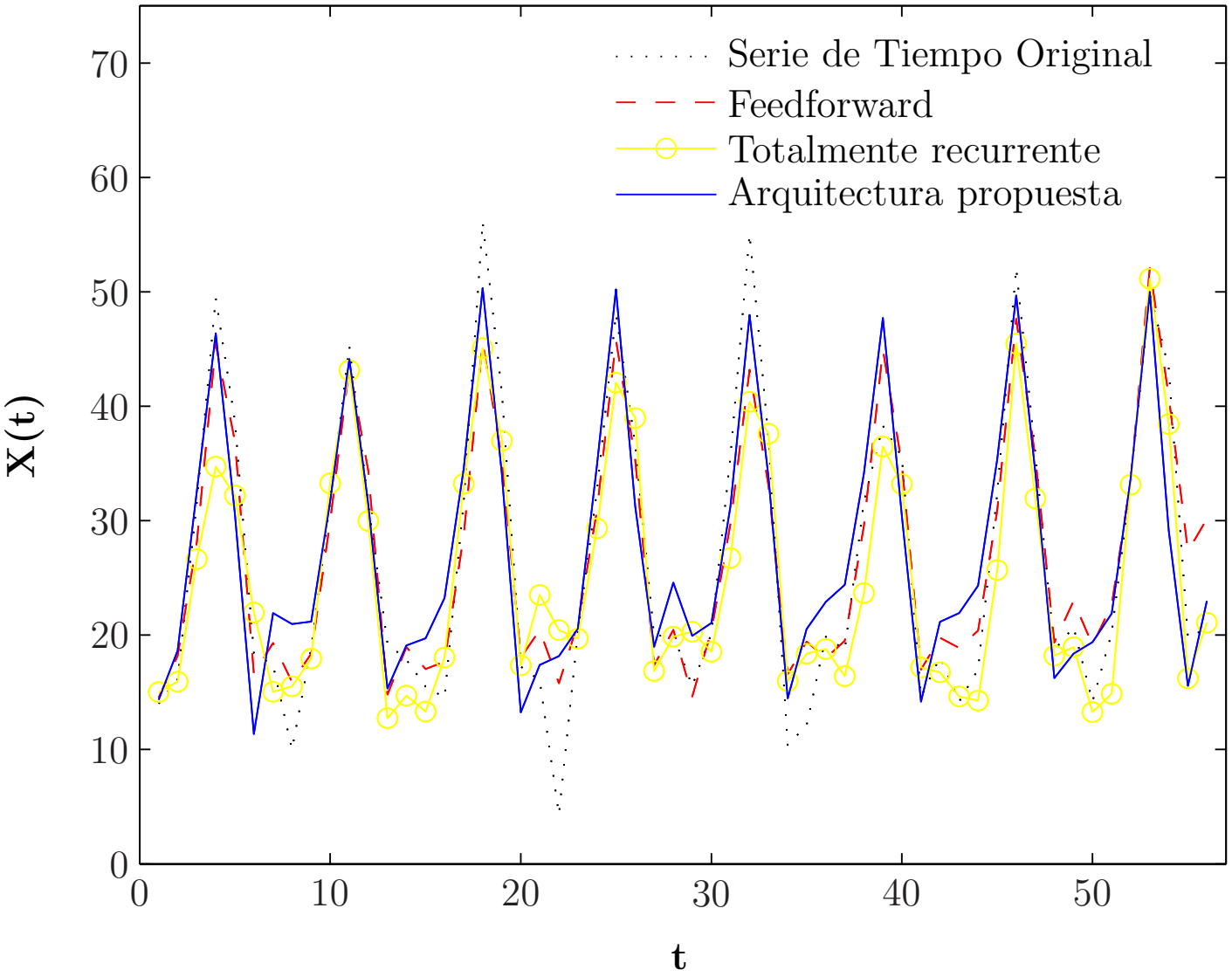


Figura B.1: Gráfica de las predicciones del conjunto de prueba de la serie NN5-109. Las mejores predicciones hechas por la red *de alimentación progresiva*, red totalmente recurrente y la arquitectura propuesta son mostradas.

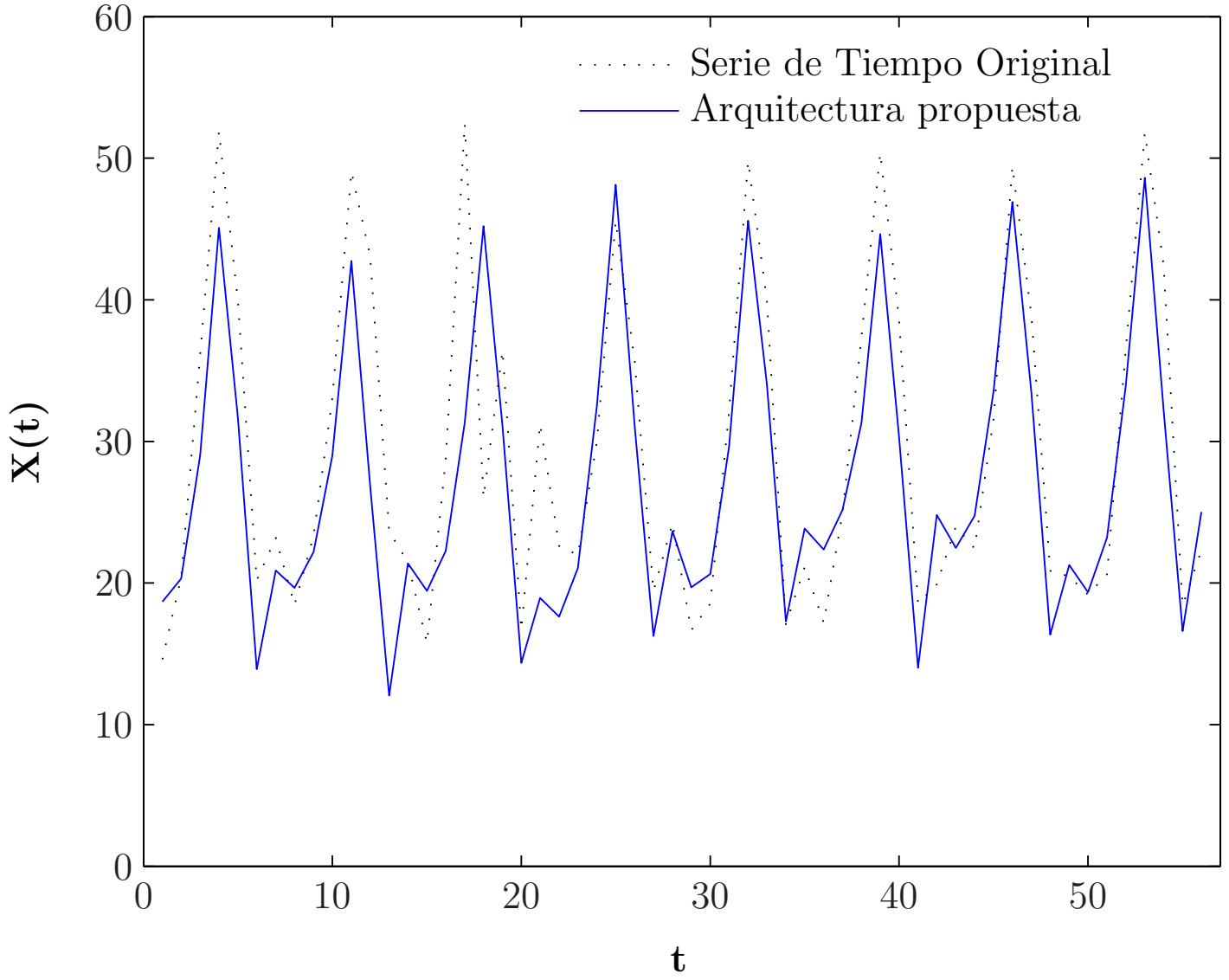


Figura B.2: Predicciones del conjunto de validación de la serie NN5-109. La mejor predicción hecha por la arquitectura propuesta es mostrada.

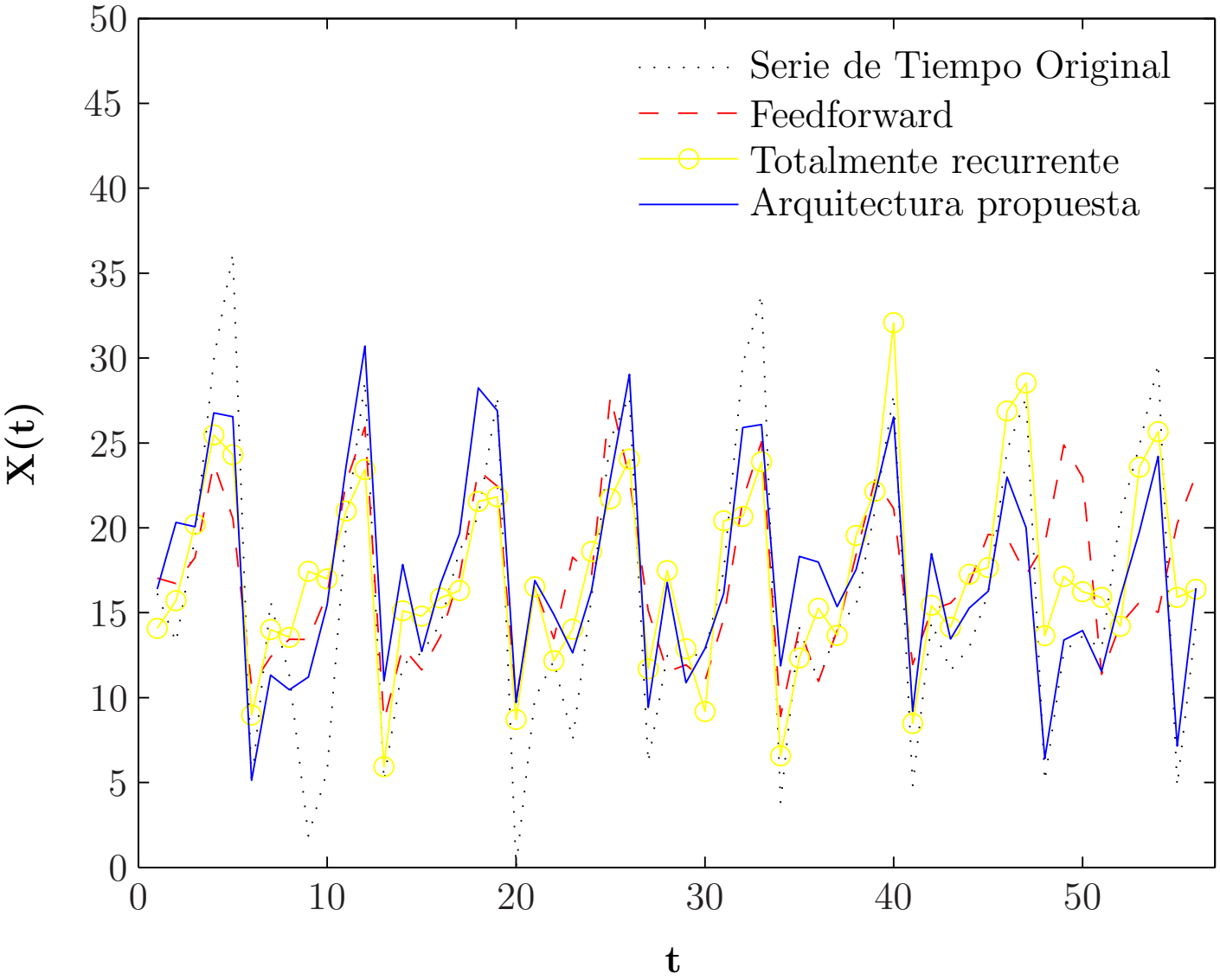


Figura B.3: Gráfica de las predicciones del conjunto de prueba de la serie NN5-107. En la gráfica se muestran las mejores predicciones realizadas por la red *de alimentación progresiva*, red totalmente recurrente y la arquitectura propuesta

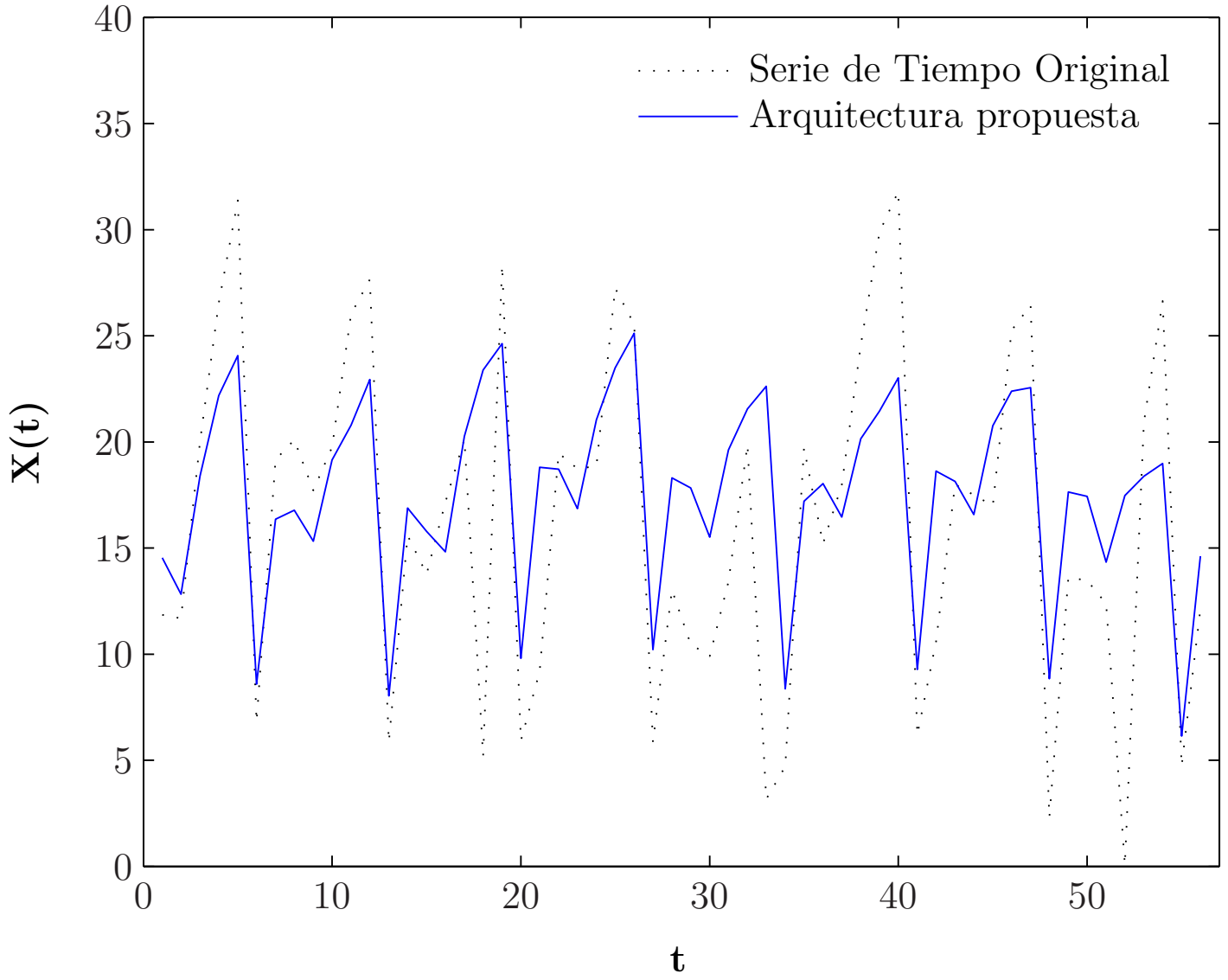


Figura B.4: Predicciones del conjunto de validación de la serie NN5-107. En la gráfica se muestra la mejor predicción hecha por la arquitectura propuesta.