UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

CENTRO UNIVERSITARIO UAEM VALLE DE MÉXICO

Modelo Computacional para la Estimación de Oxígeno Disuelto en Estanques de Producción Acuícola Empleando Redes Neuronales Artificiales

TESIS

Que para obtener el Grado de

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

P r e s e n t a

Ing. Carlos Julián Torres González

Tutor Académico:
Dr. Víctor Manuel Landassuri Moreno
Tutores Adjuntos:
Dr. José Martín Flores Albino
Dr. José Juan Carbajal Hernández

Centro Universitario UAEM Valle de México

Atizapán de Zaragoza, Edo. de México, enero de 2017

Agradecimientos

Quiero agradecer al Dr. Víctor Manuel Landassuri Moreno por brindarme el apoyo y consejos durante todo este tiempo, ya que sin él no hubiera podido lograr esto.

También agradezco al Dr. José Martín Flores Albino y al Dr. José Juan Carbajal Hernández por haberme apoyado, brindado su tiempo y guiarme durante este tiempo.

Un agradecimiento especial por el tiempo de cómputo brindado por parte del Clúster del Centro Universitario UAEM Valle de México (CLU-CUUAEM-VM), de la Universidad Autónoma del Estado de México (UAEM).

Al Centro de Investigación en Computación (CIC) del Instituto Politécnico Nacional por permitirme el intercambio de ideas y conocimientos.

Quiero agradecer a la Universidad Autónoma del Estado de México por abrirme las puertas para concluir este posgrado el cual me costó mucho esfuerzo.

Finalmente, al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo ofrecido con la beca otorgada, al cual le brindo mis agradecimientos.

Dedicatoria

Quiero dedicar esta tesis a mis padres y a mi familia en general por ser un apoyo fundamental en toda esta etapa de mi vida, tanto en lo personal como en los consejos que me brindaron. Lo más importante comienza a partir de ahora para aprovechar todo lo que hicieron por mí.

“El mejor momento para plantar un árbol fue hace 20 años. El segundo mejor momento es ahora”

Resumen

En la actualidad, la población mundial está en constante aumento, lo que tiene como consecuencia, entre otras cosas, un mayor consumo de alimentos. Así pues, la acuicultura se ha convertido en el sector alimenticio con mayor crecimiento a nivel mundial (Lekang, 2013). Sin embargo, llevar a cabo esta actividad implica controlar diversos parámetros como el oxígeno, temperatura, salinidad, nitritos y nitratos entre otros, para mantenerlos en rangos adecuados o lo más parecido a los que se encontrarían en la naturaleza, permitiendo así obtener producciones acuícolas exitosas donde los organismos no se estresen, enfermen o mueran, y a la vez se tenga el máximo rendimiento en reproducción y crecimiento.

El oxígeno disuelto es principal indicador de la calidad del agua; por ello, los acuicultores presentan especial atención a las concentraciones de este parámetro. Para evitar las fluctuaciones de este gas, inherentes a la dinámica natural de los sistemas acuícolas y los problemas que esto ocasiona a los organismos de cultivo, los productores generalmente utilizan aireación artificial a máxima potencia (potencia nominal) para complementar los suministros de oxígeno necesarios a lo largo del día (Tucker, 2005). Sin embargo, como lo sugiere Boyd (1998), el uso de la aireación máxima para lograr la mayor producción posible es menos rentable que la aireación moderada, cuando se trata de mejorar la calidad del agua y la eficiencia de conversión alimenticia. Así, una aireación convencional (máxima) trae consigo un uso, en la mayoría de las ocasiones, ineficiente del oxígeno disuelto, además de un significativo incremento en el consumo de energía de los equipos y el posible deterioro de estos al estarse activando y desactivando constantemente durante periodos prolongados de tiempo, además de posibles problemas relacionados al estrés de los organismos que esto provoca.

Los sistemas de cultivo actuales, tienen la finalidad de una mayor producción de organismos en un menor espacio de cultivo, por lo que se han comenzado a desarrollar nuevas técnicas de control y formas de predicción para integrarlas dentro de sistemas de automatización comerciales con bajo costo, mínimo impacto ecológico y fácil de usar.

Las mediciones de oxígeno disuelto que son tomadas cada determinado tiempo generan una serie tiempo la cual oscila estacionalmente y durante un periodo de 24 horas. Derivado de las múltiples variables que influyen en él, presenta un comportamiento complejo y no lineal, generalmente con niveles de concentración

por la mañana y por la noche, contrastando en la tarde, donde se suelen encontrar niveles altos.

En años recientes, las redes neuronales artificiales (RNAs) se han utilizado en problemas de estimación y predicción de series temporales en distintas disciplinas; sin embargo, son pocos los trabajos en los que se han aplicado para problemas de calidad del agua (y todos los parámetros relacionados). Su uso en predicción de series temporales de oxígeno disuelto puede permitir, entre otras cosas, encontrar las relaciones no lineales entre las variables de entrada (principalmente valores pasados de la misma serie de tiempo y valores de otras variables que influyen en la serie) y las variables de salida (valores futuros de la serie).

En este trabajo de tesis se propone el desarrollo de un modelo computacional para la estimación de oxígeno disuelto utilizando RNAs, las cuales realizarán predicciones para conocer las concentraciones de este parámetro en periodos futuros de tiempo.

El diseño de las RNAs está basado en Algoritmos Evolutivos (AEs), particularmente en el algoritmo llamado: Selección de Características en el Algoritmo de Programación Evolutiva de Redes Neuronales Artificiales del Inglés Feature Selection of Evolutionary Programming of Artificial Neural Networks (FS-EPNet) (Lopez et. al., 2013; Landassuri et. al., 2013), el cual determinará la arquitectura de la red, donde la función objetivo será la predicción en un lapso determinado de tiempo.

Aunque el análisis de la calidad del agua se ve afectado por varios parámetros, este trabajo considera únicamente la predicción del oxígeno, utilizando dos formas de predicción: predicción a un paso adelante y predicción iterada. Esto permitirá sentar las bases para futuras investigaciones sobre predicciones multiparamétricas, análisis del estado de la calidad del agua y control predictivo de la calidad del agua.

Abstract

At present, the world population is constantly increasing, which implies a greater consumption of food. Thus, aquaculture has become the fastest growing food sector worldwide (Lekang, 2013). However, carrying out this activity involves controlling various parameters such as oxygen, temperature, salinity, nitrites and nitrates among others, to maintain them in adequate ranges or the most similar to those that would be found in nature, allowing to obtain successful aquaculture productions where organisms do not stress, become ill or die, and at the same time have maximum performance in reproduction and growth.

The dissolved oxygen is the main indicator of water quality; for this reason, aquaculturists are especially interested in the concentrations of this parameter. To avoid fluctuations of this gas, inherent in the natural dynamics of aquatic systems and problems caused by growing organisms, producers generally use artificial aeration at maximum power (the nominal power) to achieve the required supplies of oxygen throughout the day (Tucker, 2005). However, as Boyd (1998) suggests, using maximal aeration to achieve the highest possible production is less cost effective than moderate aeration when it comes to improving water quality and feed conversion efficiency. Inefficient use of dissolved oxygen is present, in most cases, at conventional (maximum) aeration, in addition to a significant increase in the energy consumption of the equipment and their possible deterioration when activated and deactivated constantly during long periods of time, in addition to possible problems related to the stress of the organisms that this provokes.

Current aquaculture systems have the purpose of a greater production of organisms in a smaller cultivation space, reason why they have begun to develop new techniques of control and forms of prediction to integrate them within commercial automation systems with low cost, minimum ecological impact and easy to use.

Dissolved oxygen measurements that are taken at regulars periods of time generate a time series which oscillates seasonally and over a period of 24 hours. Derived from the multiple variables that influence it, it presents a complex and non-linear behavior, usually with low levels of concentration in the morning and at night, whereas high levels are present at afternoon.

In recent years, artificial neural networks (ANNs) have been used in problems of estimation and prediction of time series in different disciplines; however, few works

have been applied to water quality problems (and all their parameters related). Its use in predicting dissolved oxygen time series may allow, among other things, to find the nonlinear relationships between the input variables (mainly past values of the same time series and values of other variables that influence the series) and the output variables (future values of the series).

This thesis proposes the development of a computational model for the estimation of dissolved oxygen using ANNs, which will make predictions to know the concentrations of this parameter in future periods of time.

The design of the ANNs is based on Evolutionary Algorithm (EAs), particularly the algorithm called: Feature Selection of Evolutionary Programming of Artificial Neural Networks (FS-EPNet) (Lopez et. al., 2013; Landassuri et. al., 2013), which will determine the architecture of the network, where the objective function will be the prediction in a determined time lapse.

Although the analysis of water quality is affected by several parameters, this work considers only the prediction of oxygen, using two forms of prediction: direct prediction and iterated prediction. This will provide the basis for future research on multi-parameter predictions, analysis of water quality status and control aspects of water quality.

Índice de contenido

Índice de figuras

Índice de tablas

Lista de acrónimos

CAPÍTULO 1

Introducción

1.1 Antecedentes

Para satisfacer la creciente demanda de productos pesqueros en todo el mundo tales como peces, camarones, ostiones, entre otros; cada vez se tiene que producir una mayor cantidad de ellos en ambientes controlados, así como en granjas acuícolas (Areerachakul et. al., 2011; Bhatnagar & Devi, 2013). Uno de los factores más influyentes en la producción de los sistemas acuícolas es la calidad del agua, la cual afectará en mayor o menor medida el desarrollo de los organismos de acuerdo a la especie, por lo que su manejo apropiado juega un papel importante para el éxito de las operaciones acuícolas (Datta, 2012).

Múltiples factores pueden interactuar entre sí para modificar las propiedades físico-químicas del agua como lo son: el pH, temperatura, oxígeno, alcalinidad, turbidez, nitratos, cloro, amonio, etc. Sin embargo, hay parámetros que tienden a desestabilizar más el ecosistema acuático que otros, y en consecuencia son varias las afectaciones que esto ocasiona al cultivo. Por ejemplo, un cambio repentino de la temperatura o de la concentración de oxígeno disuelto en el agua puede resultar en una mortalidad masiva de la producción en sus etapas tempranas. Cambios menos drásticos tienden a afectar la capacidad de los animales para resistir organismos causantes de enfermedades presentes en el agua del cultivo.

Es importante conocer el rango de tolerancia de estos parámetros en las especies de cultivo, establecer niveles críticos, monitorearlos constantemente y estar preparado para actuar si se produce un problema. Si llegan a haber periodos prolongados con condiciones por debajo de las deseadas, esto podría resultar en un ritmo lento de crecimiento y una mayor tasa de mortalidad de los organismos de cultivo, en la mayoría de las ocasiones, entre otras. Es por lo anterior, que para lograr una buena producción, es necesario mantener las condiciones ambientales del agua dentro de los límites de tolerancia para la especie cultivada y esto se logrará cuando todos los factores que influyen sobre el desarrollo del organismo se acercan a su punto óptimo (Meyer, 2004).

Dentro de todos los parámetros para establecer la calidad del agua, se puede establecer que por varias razones, el nivel de oxígeno disuelto (OD) es el mejor y más importante indicador del estado general del cultivo acuícola. Son múltiples los factores que intervienen en los niveles de concentración de OD, al igual que las interacciones que este tiene con los demás parámetros que pueden afectar rápidamente a los organismos de cultivo. Entre otras cosas, tiene una influencia directa en la ingesta de alimento, la resistencia a enfermedades y metabolismo (Boyd, 2001; Lekang, 2013).

Regularmente en los estanques acuícolas el oxígeno se puede agotar más rápidamente que en un ambiente natural debido a la densidad de los organismos, plantas y microorganismos la cual es comparativamente mayor y dependiendo de qué tan baja esté la concentración de oxígeno disuelto en el agua y que tanto tiempo permanezca así, los organismos pueden alimentarse menos y entonces, crecer más lentamente, tener una conversión de alimento menos eficiente, ser más susceptibles a enfermedades infecciosas, estresarse e incluso morir. Por el contrario, si los niveles presentes en el estanque acuícola son adecuados, se reducirán factores de riesgo a enfermedades e incrementará la rentabilidad de la granja en general (Hargreaves & Tucker, 2002; Lekang, 2013; Meyer, 2004). Por lo anterior, mantener los niveles de OD estables, es una de las tareas prioritarias de la producción acuícola, el cual determinará en gran medida la cantidad y la calidad en los organismos de cultivo.

La tendencia actual en los sistemas de cultivo intensivos es intentar producir una gran cantidad de organismos con un bajo consumo de agua por kilogramo de cultivo (Bhatnagar & Devi, 2013). Por ello, actualmente se están implementando métodos para hacer un uso eficiente del oxígeno disuelto por medio de los aireadores en los estanques con variadas técnicas, una de ellas es la aplicación de sistemas computacionales, los cuales son una herramienta auxiliar que le da solución a distintos problemas basándose en técnicas de inteligencia artificial, bases de datos y la bioestadística, entre las más utilizadas. Ahora bien, la inteligencia artificial se está convirtiendo en una técnica útil como enfoque alternativo a las ya convencionales o también es utilizada como componente de los sistemas integrados. Se ha utilizado para resolver problemas prácticos en diversas áreas y cada vez es más empleada en los sistemas acuícolas. En este trabajo de tesis se usará una RNA como herramienta para hacer estimación de OD en estanques acuícolas, en donde se referirá dicha estimación como predicción.

1.2 Planteamiento del Problema

Algunos estudios y sistemas de aireación se han realizado para controlar mediante diversas técnicas el oxígeno disuelto y mantenerlo a niveles constantes (Doaa et. al., 2012; Juan et. al., 2014; Huan et. al., 2013), sin embargo, presentan inconvenientes como la necesidad de estar monitoreando constantemente el OD con mediciones continuas (uso de sensores) para realizar el control de aireadores en tiempo real, además, de un alto consumo de energía eléctrica por parte de los aireadores al estar funcionando constantemente. Dichos sistemas pueden controlar la cantidad de aire suministrado al agua; sin embargo, la respuesta del OD al igual que otros parámetros, tienen un cierto retardo para llegar al nivel deseado de control cuando se presentan valores fuera de rango. Esto traería problemas a los organismos de cultivo ya que los niveles tardarían un determinado tiempo en restablecerse.

Una necesidad presente en los sistemas acuícolas es poder conocer la relación del OD con los demás factores y cómo los demás factores y parámetros afectan a este, lo cual podría dar una idea de los problemas que pudieran surgir a futuro, pudiendo realizar tareas preventivas. Se han realizado trabajos como (Areerachakul et. al., 2011), el cual presenta un modelo de estimación para conocer las concentraciones de OD en el mismo instante de tiempo, de acuerdo a una serie de parámetros de entrada que influyen en su comportamiento. En (Miao et. al., 2010) se realiza predicción del OD, pero el tiempo y trabajo de diseño de la red corren a cargo del diseñador. Es así, que resultaría de gran ayuda contar con un modelo computacional diseñado de forma automática que se pueda ajustar a la dinámica particular del problema en cuestión.

1.3 Objetivos

1.3.1 Objetivo general

Desarrollar un modelo computacional para la predicción del oxígeno disuelto en granjas acuícolas empleando redes neuronales artificiales, en el cual, el diseño de la arquitectura estará a cargo del algoritmo evolutivo FS-EPNet. Para realizar las predicciones, los valores previos del oxígeno disuelto serán usados como parámetros de entrada (predicción univariada) y así obtener a la salida los valores predichos.

1.3.2 Objetivos específicos

- Obtener un conjunto de datos inicial, el cual se divida en un conjunto de entrenamiento y un conjunto de prueba, para entrenar, probar y validar el modelo neuronal.
- Aplicar el algoritmo evolutivo FS-EPNet para diseñar la arquitectura de una RNA que permita estimar la concentración de oxígeno disuelto, esto a través del ingreso de la misma variable (predicción uni-variable).
- Realizar ajustes necesarios al modelo neuronal obtenido, así como efectuar distintos experimentos con diversas formas de hacer la predicción (i.e. MSP, SSP, diferentes horizontes de predicción), con el fin de obtener un rendimiento adecuado en la estimación del oxígeno disuelto.

1.4. Hipótesis

El uso del algoritmo evolutivo FS-EPNet permitirá encontrar una estructura de red neuronal artificial adecuada de acuerdo a determinada configuración elegida en la predicción. Con esta red neuronal evolucionada, se podrá realizar pruebas para estimar valores futuros de oxígeno disuelto usando el mismo indicador como entrada para el modelo. Las predicciones realizadas con redes evolucionadas con este algoritmo, permitirán obtener una mejor predicción del OD y disminuir el trabajo en el diseño de la red, comparándolo con predicciones con redes neuronales hechas a mano.

1.5. Justificación

Los sistemas de control de oxígeno disuelto tradicionales utilizados en países como China, Indonesia y Estados Unidos, se han empezado a utilizar en algunas granjas acuícolas tecnificadas en México, sin embargo, debido al elevado costo (sensores de oxigeno de $15,000.00 M.N., por mencionar un ejemplo) y al consumo de energía eléctrica que generan, se vuelven una alternativa poco viable de aplicarse en la mayoría de las granjas acuícolas del país, por lo que es de particular interés el aplicar métodos alternativos que ayuden a hacer más eficientes a estos sistemas.

Es por lo anterior que en el modelo computacional propuesto se busca aplicar una combinación del uso de RNAs y EAs, con el fin de sentar las bases para que en un futuro trabajo de investigación, mediante las predicciones, se pueda realizar el control en varios estanques acuícolas, minimizando el número de sensores y el uso de los equipos de aireación, lo que se traduciría en una disminución del costo del

equipo de aireación y por ende un incremento de la rentabilidad de la granja acuícola; características que no presentan los sistemas de producción actuales en la acuicultura. Así, al predecir las concentraciones de oxígeno disuelto, se podrá disminuir la potencia de los aireadores, trayendo consigo un ahorro en el consumo de energía eléctrica y mejorando la rentabilidad de la granja, algo que principalmente preocupa mucho a los productores.

Obtener los resultados esperados en esta tesis, permitirá realizar aportaciones futuras a la industria acuícola, tales como:

- La predicción del oxígeno permitirá alertar sobre problemas que puedan surgir en periodos futuros y tomar medidas preventivas de control.
- Minimizar el trabajo en sitio del acuicultor en el control del oxígeno.
- Disminuir el estrés en los organismos al llevar a cabo un mejor control.
- Entender mejor el comportamiento del sistema acuícola debido a que el conocer el comportamiento del OD permite determinar la relación entre varios parámetros que tienen que ver en el entorno del oxígeno disuelto.
- Desarrollar una tarjeta de adquisición de datos para el monitoreo y control de parámetros en diferentes especies o sistemas con nulos o mínimos cambios en ella.
- Poder incrementar la eficiencia de los sistemas de aireación en la acuicultura, evitando el desgaste innecesario de los equipos.
- Controlar el OD mediante la predicción del mismo, permitirá mejorar las condiciones de vida de los organismos de cultivo (i.e. disminución de estrés) ocasionadas por las características de los sistemas de aireación actuales.

1.6 Delimitación o alcances de la investigación

El objetivo principal de esta investigación es determinar con qué aproximación se puede hacer la predicción del OD usando RNAs generadas con el FS-EPNet. Para realizar la predicción se utilizan valores pasados de la misma variable (OD) para predecir el siguiente valor. Se usan dos formas diferentes de hacer predicción: predicción a un simple paso (del Inglés single-step prediction - SSP) y usando predicción a múltiples pasos (Multi-step prediction - MSP).

1.7. Publicación derivada de esta investigación

Derivado del trabajo de investigación, se publicó un artículo de revista:

Carlos Julián Torres González, Víctor Manuel Landassuri-Moreno, José Juan Carbajal Hernández, José Martín Flores Albino. Predicción de Oxígeno Disuelto en Acuacultura Semi-intensiva con Redes Neuronales Artificiales. Advances in Computing Science. Research in Computing Science. Instituto Politécnico Nacional. (2016). (Artículo aceptado, en espera que salga publicado).

1.8. Organización de la tesis

El trabajo de tesis está organizado de la siguiente manera:

En el capítulo II se comentarán los antecedentes acerca del oxígeno disuelto y de la teoría básica de las redes neuronales artificiales. Se mencionarán los inconvenientes de éstas, lo que dará paso a explicar posteriormente las bases teóricas de los algoritmos evolutivos para entender el algoritmo de optimización utilizado en este trabajo, el algoritmo FS-EPNet. Así, se explicará el diseño y funcionamiento del algoritmo FS-EPNet, indicando la lógica que usa para tratar de optimizar las arquitecturas de las RNAs. Finalmente, se explicarán los trabajos encontrados en la literatura que presentan similitudes con el realizado en esta tesis.

El capítulo III presentará la forma en que se recolectaron los datos y se presentará la configuración experimental para los escenarios planteados en esta tesis.

En el capítulo IV se mostrarán los resultados de las predicciones llevadas a cabo, donde se resaltan las predicciones que dieron mejores aproximaciones de acuerdo a determinada configuración. También, se presentará el resultado de la predicción con una red neuronal diseñada a mano, en donde se verá la viabilidad y ventajas de realizar las RNAs mediante el algoritmo FS-EPNet para predecir el OD.

Las conclusiones de acuerdo a los resultados obtenidos, se presentarán en el capítulo V, así como también las líneas o trabajos de investigación a futuro.

CAPÍTULO 2
Marco teórico y estado del arte

2.1 Estudio del oxígeno disuelto en cultivo acuícola

El oxígeno disuelto presenta un comportamiento dinámico el cual fluctúa cada dia, principalmente por la fotosíntesis y respiración del fitoplancton (ver Figura 2.1). Debido a ello, el nivel máximo de OD ocurrirá en la tarde con la acumulación de O_2 (oxigeno) por el proceso fotosintético. Sin embargo, a medida que el fitoplancton (algas microscópicas) consume la mayoría de este O_2 junto a la respiración que siguen teniendo los peces y a que la fotosíntesis se detiene durante la noche, los niveles de OD comienzan a disminuir hasta encontrar un punto mínimo en la madrugada (Boyd, 2001; Simbeye & Yang, 2014; Tucker, 2005). Una de las razones más frecuentes por la que el OD baja a niveles críticos, es la muerte masiva de algas, ya que la descomposición consecuente de estas células muertas por bacterias demanda grandes cantidades de oxígeno (Bhatnagar & Devi, 2013) (Boyd, s.f).

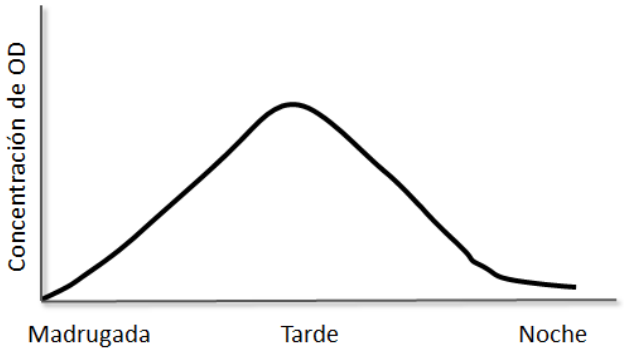

Figura 2.1. El ciclo diario de oxígeno en un estanque.

Esto hace, que el manejo del equilibrio de la fotosíntesis y respiración al igual que el crecimiento de las algas sea una tarea determinante en el trabajo diario de un productor. Por otro lado, es casi imposible que la distribución de oxígeno disuelto

en un estanque sea totalmente uniforme, entre otras cosas, porque la cantidad luz (necesaria para la fotosíntesis) se reduce rápidamente de forma natural al aumentar la profundidad (Boyd & Tucker; William, 2011), 1998). Esto ocasiona que haya una mayor concentración de oxígeno en la superficie, por lo que se suelen utilizar estanques relativamente poco profundos.

La temperatura es otro de los parámetros que afecta la cantidad de oxígeno contenido en el agua, principalmente en el cultivo de peces, debido que a altas temperaturas la demanda de oxígeno se incrementa, por lo que el oxígeno disponible disminuye (tabla 2.1) y también disminuye la cantidad de oxígeno que puede transferirse al agua mediante una aireación mecánica y al mismo tiempo, la cantidad de energía requerida para la aireación aumenta considerablemente (Areerachakul et. al., 2011; Lekang, 2013). Además, la capacidad del oxígeno para disolverse en el agua se ve afectada por la presencia de sales disueltas, debido a que las moléculas de sal ocupan lugares en el agua donde pueden estar presentes las moléculas de oxígeno, reduciendo la capacidad del oxígeno para disolverse; esto se aprecia en el agua de mar, la cual presenta menos oxigeno que el agua dulce.

Tabla 2.1. Solubilidad del oxígeno en función de la temperatura y la salinidad.

Temperatura °C	Solubilidad del Oxigeno (mg/l)		
	Sal (0.30 %)	Sal (9.055 %)	Sal (18.080 %)
0	14.621	13.728	12.888
5	12.77	12.024	11.32
10	11.288	10.656	10.058
15	10.084	9.541	9.027
20	9.092	8.621	8.174
25	8.263	7.85	7.457
30	7.559	7.194	6.845
35	6.95	6.624	6.314
40	6.412	6.121	5.842
50	5.477	5.242	5.016

Normalmente, se suministra oxigeno adicional por medio de equipos mecánicos, inclusive algunos productores prefieren utilizar oxígeno puro por su mejor solubilidad en agua (Tucker, 2005). Sin embargo, la adición continua de demasiado oxígeno no sólo ocasiona gastos innecesarios sino que los peces responden a tal abundancia produciendo menos glóbulos rojos y haciéndose más susceptibles a las enfermedades (Boyd, 1998; Doaa et. al., 2012; Tucker, 2005).

2.1.1 Saturación del oxígeno

El oxígeno disuelto se puede expresar en miligramos por litro (mg/L) o en porcentaje de saturación (%) (Meyer, 2004). La primera, mg/l, representa la masa de oxígeno por litro de agua, mientras la segunda se expresa como el porcentaje de la concentración de saturación, que se define como la cantidad de oxígeno en el agua con relación a la cantidad máxima de oxígeno que puede tener a la misma temperatura y presión (Corral et. al., 1999; Tucker, 2005; Boyd, 1998).

La concentración o saturación de oxígeno en el agua, ecuación 2.1, depende de la presión atmosférica que empuja el aire cargado de oxígeno sobre la superficie del agua permitiendo su disolución hasta el punto de saturación.

$$(\%OD) = \frac{OD^{out}}{OD^{*}} \times 100 \tag{2.1}$$

En donde OD^{out} es el oxígeno disuelto medido y OD^{*} es el porcentaje de oxígeno disuelto en saturación. Se entiende que si la presión atmosférica baja (tempestad o lluvias entre otras) bajará el nivel de saturación de oxígeno en el agua, por lo que se debe aumentar la aireación (Tautenhahn, 2014).

De acuerdo a los niveles en las concentraciones de OD, los peces experimentan distintos comportamientos y consecuencias (Tabla 2.2); sin embargo, no todos las especies responden de la misma manera, por ejemplo, peces de agua fría como las truchas y salmones tienden a ser más sensibles a niveles de hipoxia (condiciones de bajo OD) que los peces de agua caliente, por lo que necesitan más oxígeno para un correcto desarrollo. Para ellos se recomiendan los siguientes niveles de saturación:

- Huevos y juveniles(peces jóvenes): cerca de 100%
- Adultos:
  --Mínimo.- 5 mg/l (50%)
  --Recomendable.- 8 mg/l (70%)

Un fenómeno muy importante que se produce en algunos sistemas es la sobresaturación, también llamada “hiperoxia” (condiciones de sobresaturación de OD), la cual se da entre el 140-300% de saturación de oxígeno en el agua (Tautenhahn & Karg, 2007; William, 2011). De igual forma, el agua sobresaturada de oxigeno también es perjudicial para los peces, ya que pueden presentar enfermedades, debido al incremento de la presión del gas (oxigeno) en el agua (Cañon, 2012).

Tabla 2.2. Rangos de concentración de DO y consecuencias ecosistémicas.

[DO] mg/l	Condición	Consecuencias
0	Anoxia	Muerte masiva de los organismos
0-5	Hipoxia	Desaparición de organismos o especies sensibles
5-8	Aceptable	Condiciones adecuadas para la vida de la mayoría de los organismos acuáticos
8-12	Buena	
>12	Hiperoxia	Podría ser dañino, si las condiciones ocurren en todo el estanque.

El efecto del ciclo diario del oxígeno sobre los peces y camarones es poco conocido, pero en general, un buen crecimiento se logra cuando las concentraciones de oxígeno no descienden más de un 30 o 40% de saturación durante la noche, y siempre y cuando dicho nivel de concentración de oxígeno no perdure más de 1 o 2 horas (Boyd, 2001; Boyd & Tucker, 1998).

2.2 Redes Neuronales Artificiales

Cuando se resuelve un problema del mundo real utilizando cómputo secuencial, se suelen encontrar diversos problemas, debido a que los sistemas biológicos poseen una arquitectura distinta (paralelismo masivo) a la de una computadora actual y es por esta razón que se han tratado de simular algunas características fisiológicas del cerebro humano, para elaborar métodos que den solución a estos problemas. En ello se centra el funcionamiento de las redes neuronales artificiales, debido a su característica de trabajar con un cómputo paralelo (Freeman & Skapura, 1991; Kriesel, 2005).

Una red neuronal artificial (RNA) es entendida como un modelo matemático inspirado en los sistemas neuronales de organismos biológicos. (Haykin, 2005; Kriesel, 2005) para tratar de imitar el funcionamiento del cerebro humano. En un enfoque biológico se toman parámetros de entrada y se combina para producir una salida (Kriesel, 2005), sin embargo esta combinación se hace de manera automática e irracional. En este sentido, las RNA son un modelo que permite buscar la forma en que se combinan dichos parámetros y a la vez aplicar este resultado. Encontrar la combinación de valores que mejor se ajuste al problema es lo que se conoce como “entrenar” la RNA (pasando por etapas como diseño de la arquitectura, entrenamiento y validación de la red.). Una vez entrenada, la red se puede aplicar

al problema para la cual fue diseñada, en caso de querer aplicar la RNA a otra tarea distinta, es necesario hacer un diseño nuevo para ello.

2.2.1 La neurona: unidad básica de procesamiento

El elemento básico de un sistema neuronal biológico es la neurona (ver Figura 2.2). Un sistema neuronal biológico está compuesto por millones de neuronas organizadas en capas. Una red neuronal artificial es la emulación de dicho sistema neuronal biológico, en el que se puede establecer una estructura jerárquica similar a la existente en el cerebro.

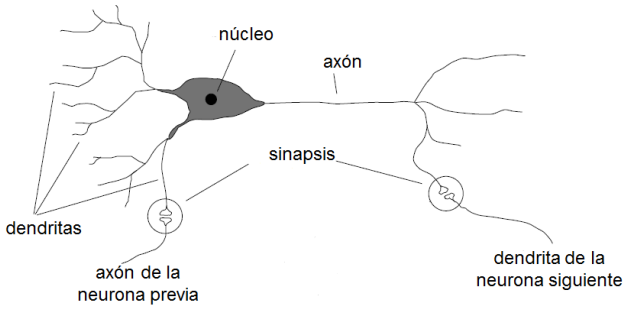

Figura 2.2. Neurona. Unidad fundamental de un sistema neuronal biológico (Krisel, 2005).

El elemento esencial en el que basan su funcionamiento las RNAs es la neurona, donde se ha diseñado una neurona artificial (o nodo) el cual trata de simular el comportamiento de la neurona biológica. Estos nodos se pueden organizar en capas (en el caso de una red multi-layer) para constituir un sistema neuronal artificial (RNA), donde hay que notar que un solo nodo no es suficiente para resolver muchos problemas, pero el agrupamiento de varios de ellos, permite aproximar una función deseada, característica que suele identificar a estos modelos computacionales, i.e., permiten aproximar una función compleja, donde el diseño, entrenamiento y validación de la RNA juegan un papel importante.

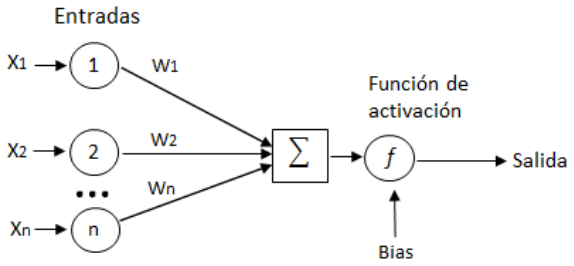

Figura 2.3. Neurona artificial. Unidad fundamental de una RNA.

Así, un nodo es un elemento formado por una o varias entradas las cuales son asociadas a un peso (peso sináptico – haciendo la similitud con la sinapsis) con un cierto peso cada una (ver Figura 2.3), donde dichos pesos esta relacionados directamente con las conexiones de la RNA. Si la suma total de esas entradas multiplicadas por cada peso es mayor que un determinado umbral (depende de la función de activación), la salida del perceptrón (el nodo) generalmente es uno, o si es menor, la salida es cero, similar a lo que ocurre con una neurona biológica (Haykin, 2005; Krisel, 2005).

Dados un conjunto de entradas X_j y unos pesos sinápticos w_{ij}, con $j = 1 \ldots . . n$, es definida una regla de propagación R_i. La regla de propagación más comúnmente utilizada consiste en combinar linealmente tanto entradas como los pesos sinápticos, de la siguiente manera:

$$R_i\left(x_1, \ldots, x_n, w_{i_1}, \ldots, w_{i_n}\right) = \sum_{i=1}^{n} w_{ij} x_j + b \tag{2.2}$$

donde w_{ij} es el peso o valor de cada conexión (suponiendo que hay varias) conectada a la neurona, x_j es la entrada a la neurona y b es el vías, donde el vías juega un papel de ajuste extra. Desde otro punto de vista, se puede apreciar la similitud de la ecuación 2.2 con la ecuación de la recta para entender mejor el papel que juega el parámetro bias (*b*). Posteriormente, una función de activación, denotada como y, representa la salida de la neurona y el estado de activación que tendrá, como se puede apreciar en la figura 2.3.

$$y = f(R_i) = f\left(\sum_{i=1}^{n} w_{ij} x_j + b\right) \tag{2.3}$$

Esta función de activación permitirá a las neuronas cambiar de nivel de disparo (transmitir un valor de salida a otros nodos) a partir de las señales que reciben. El conjunto de estados de activación que toma una nodo, normalmente tiene un rango de [0, 1] o de [-1, 1], en donde el nodo puede estar totalmente inactiva (0 o -1) o activa (1). Estos valores están directamente correlacionados con los rangos de funcionamiento de las diversas funciones de activación mostradas en la sección 2.2.4.

La ecuación es escrita de forma matricial como sigue:

$$y = f(Wx + b) \tag{2.4}$$

y donde *W* representa la matriz de pesos y $\boldsymbol{x}$ el vector de entradas. Los subíndices en la matriz de pesos *W* representan los términos involucrados en la conexión, el primer subíndice representa el nodo destino y el segundo, representa el nodo origen de la señal que alimenta al nodo actual.

$$\boldsymbol{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ \vdots & \vdots & & \vdots \\ w_{S,1} & w_{S,2} & \cdots & w_{S,R} \end{bmatrix}$$

Esta notación es especialmente útil cuando se tiene una capa con varias entradas y nodos.

2.2.2 Clasificación de redes neuronales

En problemas reales, para proporcionar una mayor capacidad de procesamiento, se combinan varias capas de neuronas formando complejos sistemas neuronales artificiales. Estas neuronas al estar organizan definen la topología o arquitectura de la red neuronal, donde sus parámetros principales nos: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas. Donde una capa, la cual suele ser llamada capa oculta, en un conjunto de nodos ubicados al mismo nivel, como se puede ver en la figura 2.4, mientras que la conectividad define la forma en que están conectados dichos nodos, lo que implica directamente la forma de transmitir la información internamente en la red. De esta forma, la distribución de neuronas en una RNA se realiza ordenando los nodos que tienen comportamientos similares en capas.

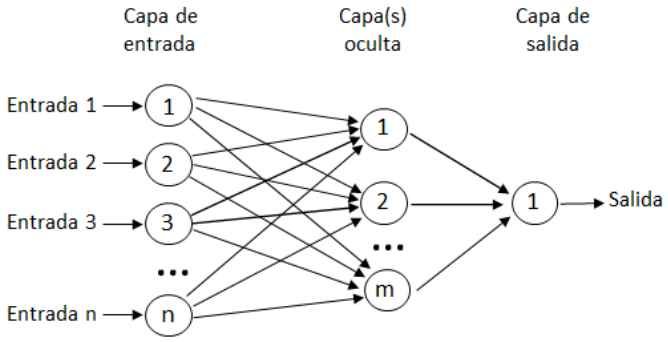

Figura 2.4. Red neuronal típica multicapa.

La gran mayoría de problemas requieren de redes que tengan varias capas para una correcta solución. Estas pueden ser agrupadas en dos categorías:

- *Redes feedforward*.- En las cuales las conexiones entre neuronas solamente están permitidas a neuronas de la siguiente capa, i.e. no hay una retroalimentación de capas delanteras hacia las anteriores.
- *Redes recurrentes o feedback*.- Son redes que no siempre tienen explícitamente definidas las neuronas de entrada y salida, ya que disponen de conexiones de las neuronas de las capas posteriores a la entrada de capas anteriores, de forma que la información puede circular entre las distintas capas en cualquier sentido, incluso de salida a entrada.

En general, estos dos tipos de redes se pueden clasificar de acuerdo al número de capas: mono-capas y multicapas. En cuanto a la conectividad se pueden encontrar RNAs parcialmente conectadas (o dispersamente conectadas), totalmente conectadas, conexión uno a uno, entre otras configuraciones. En las redes feedforward, entre los modelos más conocidos, se encuentran el perceptrón monocapa, perceptrón multicapa (MLP) y redes de función de base radial (Haykin, 2005; Praveen & Varalakshmi, 2015). En cuanto a las redes recurrentes/feedback se encuentran las redes competitivas, mapas auto-organizados de Kohonen (SOM), redes de Hopfield y los modelos de Teoría de Resonancia Adaptativa (ART) (Haykin, 2005; Jain & Mao, 1996).

Cuando se aborda un problema utilizando redes neuronales artificiales, uno de los primeros pasos a realizar es el diseño de la arquitectura de red. Este proceso implica la determinación de la función de activación a emplear, el número de neuronas y el número de capas de la red.

2.2.3 Consideraciones para el diseño

En relación al número de neuronas y capas, algunos de los parámetros vienen dados por el problema en cuestión (reglas heurísticas), pero otros tienen que ser elegidos por el experto, por ejemplo, en tareas de clasificación usualmente se proporciona la base de datos a entrenar y clasificar, donde ya se tienen identificadas ciertas variables, como: estatura, peso, pH de la sangre, temperatura corporal, entre otros parametros, las cuales dependen del problema o tarea a resolver. De esta forma, tanto el número de neuronas de la capa de entrada como el número de neuronas en la capa de salida, vienen dados en gran medida por las características del problema en cuetión.

Así es posible notar que algunas tareas no representan mayor problema y se sabe de antemano el número adecuado de entradas y salidas. No obstante, existen otras

tareas en los que el número de variables de entrada relevantes para el problema no se conoce con exactitud, i.e., el caso de la predicción uni-variable, donde la mayor interrogante es saber cuanta información del pasado es necesaria para hacer predicciones adecuadas, donde escoger erróneamente ese valor tendrá efectos importantes en el desempeño del algoritmo.

Esto implica que al disponer de un gran número de variables, es posible que algunas de ellas podrían no aportar información relevante a la red (redundancia o ruido) y su utilización podría complicar el aprendizaje (aspecto de las RNAs mostrado más adelante), lo cual implicaría arquitecturas de gran tamaño con alta conectividad. En estas situaciones, es conveniente realizar un análisis previo de las variables de entrada más relevantes al problema y descartar aquellas que no aportan información a la red (sensitividad). Este análisis puede llegar a ser una tarea complicada y requerir técnicas avanzadas, como técnicas basadas en análisis de correlación (Battiti, 1994), análisis de componentes principales (Cadima & Jollife, 2009), análisis de sensibilidad de redes y neuronas (Mao, 2002; Zurada et. al., 1997) y técnicas basadas en algoritmos genéticos (Guo & Uhrig, 1992), entre otras.

Es posible encontrar en la literatura métodos matemáticos que tratan de aproximar el número de nodos y capas ocultas en un solo paso, haciendo operaciones con el número de datos de entrada (si se conoce) y el número de patrones en el conjunto de entrenamiento y prueba, sin embargo, a la fecha no existe un método matemático que domine este aspecto, y por lo mismo no son totalmente confiables, siendo ahí los métodos de búsqueda en los que se ha centrado la atención. Por lo que se puede afirmar que no existe un método único para determinar el número óptimo de nodos y capas ocultas, y mucho del trabajo es a prueba y error (si es que no se considera algún otro método de optimización para ello). Así, partiendo de la arquitectura ya entrenada, se realizan cambios graduales aumentando y disminuyendo el número de neuronas ocultas y el número de capas hasta conseguir una arquitectura adecuada para resolver el problema dado, que pudiera no ser la óptima, pero proporcionaría una solución adecuada de acuerdo a las necesidades del usuario. La generalización es la capacidad que tiene una RNA a resolver patrones de entrada de forma adecuada, sin que hayan sido usados en dichos patrones en el entrenamiento.

Así, dentro de los algoritmos de búsqueda, se tienen trabajos centrados en la determinación automática del número de neuronas ocultas, así como capas ocultas, para cada problema en particular; algunos de ellas se basan en la utilización de técnicas evolutivas, donde a partir de un espacio de búsqueda, tratan de encontrar arquitecturas y configuraciones, donde la búsqueda es guiada normalmente por el rendimiento de las redes (Miller et. al., 1989; Peralta et. al., 2012; Yao & Lin, 1997).

2.2.4 Funciones de activación

Después de ver las consideraciones de diseño, en cuanto a nodos y capas ocultas, se continúa con el diseño de las RNAs, pero en cuanto a las funciones de transferencia.

La regla de propagación de información en las RNAs está dada por el tipo de funciones de transferencia usadas. Estas toman como entrada la combinación de las entradas, pesos y bias para transformarlo en la salida del nodo. En la figura 2.5 se muestran las funciones de transferencia más utilizadas.

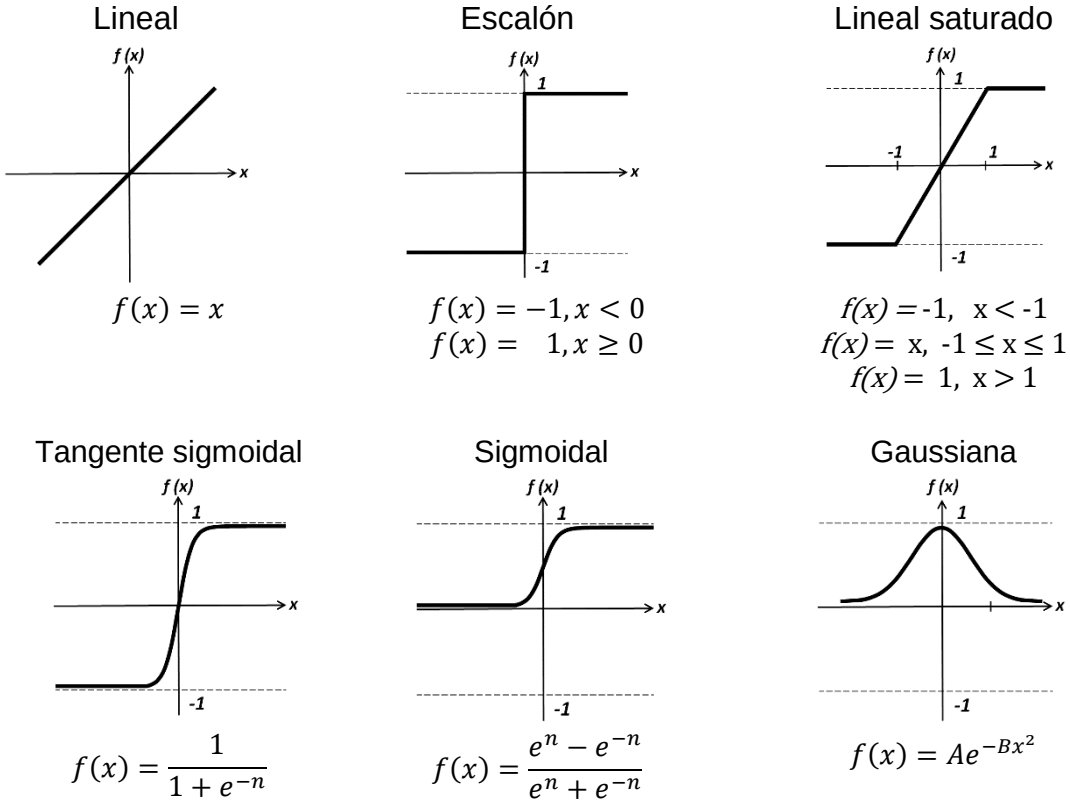

Figura 2.5. Funciones de transferencias mayormente utilizadas.

La función de transferencia calcula el estado de actividad de una neurona, transformando la entrada global (menos el umbral o bías, si lo hay) en un valor (estado) de activación. De esta manera, la neurona puede estar activa (excitada) o inactiva (no excitada); es decir, que tiene un “estado de activación”. Las neuronas

pueden tomar diferentes estados de activación dentro de un rango determinado (i.e. valores reales); algunas de ellas solamente dos (cuando se trabaja con datos binarios).

Existen algunas redes que transforman su estado de activación en una salida binaria y para se usa la función escalón, en otras, los algoritmos de aprendizaje requieren que la función de desempeño cumpla la condición de ser derivable. Estas últimas meten un factor de no linealidad entre entradas y salidas, y deben ser derivables para cumplir con los requisitos de los algoritmos de aprendizaje de primer orden de convergencia, como el Backpropagation (BP), así como los de segundo orden de convergencia, como el algoritmo de Levenberg Marquardt, entre otros.

De esta forma es fácil determinar la función de transferencia necesaria. Cuando en el problema se tenga que usar valores binarios (en entradas-salidas), la función escalón es la adecuada. Al contrario, si se desea que la red pueda dar una salida en cualquier rango, i.e., que no esté limitada en el rango de valores, es posible usar la función lineal, la cual es comúnmente usada en la literatura para las salidas. Para el caso de los nodos ocultos, no ha sido posible encontrar alguna publicación que determine si una función un otra es adecuada para un tarea en particular, donde las funciones sigmoidales o tangente sigmoidal son de las más comúnmente usadas en las capas ocultas (Abdallah et. al., 2012; Gomes te. al., 2016; Odim et al., 2016).

2.2.5 Entrenamiento

Un punto interesante, es que en lugar de seguir un conjunto de reglas especificadas por expertos humanos, las RNA parecen aprender reglas (como relaciones de entrada-salida) de la colección dada de ejemplos representativos (conjunto de entrenamiento), lo que se convierte en una de las principales ventajas de las redes neuronales sobre los sistemas expertos tradicionales. Así, una red neuronal debe aprender a calcular la salida correcta para cada vector de entrada (patrón de entrada) dado todo el conjunto de entrenamiento, el cual estará formado por cientos o miles de patrones de entrada y salida. Este proceso se denomina: *proceso de entrenamiento o acondicionamiento supervisado*.

Un proceso de aprendizaje en el contexto de la RNA puede considerarse como el problema determinación de los pesos de las conexiónes para que una red pueda realizar una tarea específica de manera eficiente, esto se traduce en un proceso de adaptación, para mejorar el rendimiento mediante cada actualización iterativa. A pesar de que la arquitectura es importante, los algoritmos de aprendizaje desarrollados hasta el momento consideran una arquitectura fija para adaptar los

pesos, dejando la modificación de la arquitectura de lado, en manos del usuario o de otro algoritmo.

Los cambios en los pesos a causa del entrenamiento se pueden ver como la eliminación, modificación y creación de conexiones entre las neuronas. De esta forma, la creación de una nueva conexión quiere decir que el peso de la misma tiene un valor diferente de cero. De la misma manera, una conexión se elimina cuando su peso pasa a ser cero. Finalmente, este proceso termina (la red ha aprendido) cuando los valores de los pesos permanecen estables, esto está dado por la ecuación 2.5:

$$\frac{dw_{ij}}{dt} = 0 \tag{2.5}$$

Por ende, el conocer cómo se modifican los valores de los pesos se vuelve un aspecto importante en el aprendizaje de las redes neuronales, es decir, saber qué reglas de aprendizaje rigen el proceso de actualización. Para ello se debe tener un modelo como forma de aprendizaje, diseñado como un algoritmo el cual usa estas reglas para ajustar los pesos. Existen tres formas principales de entrenamiento:

- **Aprendizaje supervisado:** este aprendizaje se lleva a cabo mediante un entrenamiento controlado por un agente externo (experto) que determina la salida que debería generar la red (target, valor objetivo o patrón de salida) a partir de una entrada determinada (patrón de entrada), extrayendo el error y modificando los pesos de las conexiones, para aproximar el resultado deseado.
- **Aprendizaje no supervisado.-** Las redes con aprendizaje no supervisado (también conocido como auto-supervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique, si la salida generada, producida por dicha entrada, es o no correcta, por lo que deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten. Los más utilizados aquí son el *aprendizaje hebbiano* y el *aprendizaje competitivo* (Naves & Gerstner, 2015; Sanger, 1989; Xu et. al., 2016).
- **Aprendizaje por refuerzo.-** En este caso, no se tiene un ejemplo del comportamiento deseado, por lo que el experto únicamente indica si la salida se ajusta al target (i.e. éxito = 1, fracaso = -1) y en función de ello se ajustan los pesos mediante un mecanismo de probabilidades.

Cabe señalar, como un criterio adicional para diferenciar las reglas de aprendizaje, el considerar si la red puede aprender durante su funcionamiento real, conocido como on-line (en etapa de trabajo), o si el aprendizaje supone la desconexión de la red, es decir, su inhabilitación hasta que el proceso termine, llamado off-line (fuera de trabajo). Esto es, asumiendo que la red está en el entorno de funcionamiento (resolviendo la tarea para la cual fue creada), al momento de dar respuesta a los patrones entrantes, la red continua aprendiendo (on-line) dichos patrones. Por el contrario, se puede entrenar la red, y ya que está lista, se pone en funcionamiento (off-line) para resolver el problema, considerando que si llegan patrones nunca antes vistos en el conjunto de entrenamiento, esto no se agregan al conocimiento de la red, de esta forma, la única opción es ponerla en modo off-line de nuevo y realizar un entrenamiento completo, para captar la nueva información recibida.

Antes de entrenar la red se debe dividir el conjunto de datos total en dos partes; una parte se utiliza como conjunto de entrenamiento para el aprendizaje de la red y la otra parte, llamada conjunto de prueba, se utiliza para estimar el error de aprendizaje y generalización. El conjunto de entrenamiento suele a su vez dividirse en dos conjuntos, el conjunto de entrenamiento y validación para ajustar el modelo, el cual sirve para no sobreentrenar la red. Normalmente, se suele utilizar el 80% de los datos disponibles para entrenar y un 20% como prueba, aunque no es una regla general.

Término de entrenamiento

Al realizar el proceso de entrenamiento, hay que prestar esencial cuidado de no generar un sobre-aprendizaje, ya que provocará la perdida de la capacidad de generalización de la red, lo cual puede suceder cuando la cantidad de ciclos (épocas) de entrenamiento tiende a ser muy alta. Esto se observa cuando al proporcionar los patrones de entrenamiento a la red, esta da una respuesta muy buena (errores que son prácticamente de cero) pero cuando se presentan nuevos patrones, es muy pobre el rendimiento dando un error de aprendizaje muy alto, por lo tanto, su generalización suele ser muy pobre. Para ello, se tiene que establecer un método para detener el proceso de aprendizaje en el momento antes de empezar a sobreentranarse. Regularmente, esto es cuando el cálculo del error cuadrático sobre todos los ejemplos de entrenamiento ha alcanzado un mínimo o cuando para cada uno de los ejemplos dados, el error observado está por debajo de un determinado umbral.

Otra forma de detener el entrenamiento a tiempo es cuando se hayan cumplido un cierto número de ciclos de entrenamiento, sin que cambien significativamente los pesos. Así mismo, se tiene el método de Early Stopping, el cual usa el conjunto de

validación anteriormente comentado, para evaluar el rendimiento de la red durante la etapa de entrenamiento, i.e. por un lado se está entrenando la red, y por otro lado se está validando su precisión con el conjunto de validación. Así, es de esperarse que el error de validación descienda a medida que el error de entrenamiento lo hace, sin embargo, si la red empieza a perder generalización, el error de validación empezará a aumentar, punto exacto para terminar el entrenamiento.

2.2.6 Reglas de aprendizaje

Normalmente, dada una arquitectura de red neuronal, ésta debería entrenarse hasta un punto óptimo en el que el error de generalización sea mínimo. Para ello, se utiliza un procedimiento llamado validación cruzada (cross validation) que consiste en entrenar y validar al mismo tiempo para detenerse en el punto óptimo (similar al Early Stopping), el cual es ampliamente utilizado en la fase de desarrollo de una red neuronal supervisada especialmente en las redes Multi-Capa (del inglés Multi-layered perceptron – MLP). Así mismo es un método usado cuando se tiene poca información para entrenar. Donde la principal diferencia con el método de Early Stopping radica que el segundo sólo usa un conjunto de validación, y la validación cruzada suele generar varios conjuntos de entrenamiento y validación, los cuales se van alternando durante el proceso.

Independientemente del tipo de aprendizaje usado, i.e. supervisado o no supervisado, se debe de tomar en cuenta la regla de aprendizaje a usar, la cual indica cómo se modifican los pesos de las conexiones en función de los datos usados en la entrada. De ahí, que la regla Delta merece una mención especial, en la cual se basa el algoritmo Backpropagation (BP), y el cual con una modificación (MBP), se utiliza en este trabajo para el entrenamiento de la red neuronal. Donde la modificación se comentará junto con la descripción del algoritmo FS-EPNet.

También llamada Regla del Mínimo Error Cuadrado (LMS), la regla Delta es utilizada para entrenar redes de una capa con un patrón de entradas, modificando los pesos para reducir la desviación a la salida respecto al valor deseado, tomando en consideración a todas las neuronas predecesoras que tiene la neurona de salida. Con el propósito de reducir el error cuadrático medio. Así, el error calculado es igualmente repartido entre las conexiones de las neuronas anteriores.

2.2.7 La regla delta generalizada - Backpropagation

La regla delta era la más utilizada anteriormente, sin embargo, dado que no funcióna en redes multicapa, fue creada la regla Delta Generalizada, también llamada

Backprogagation (Rumelhart et. al., 1986), para redes de varias capas y para funciones de transferencia no lineales y diferenciables.

Con un aprendizaje supervisado, esta regla consiste en ajustar pesos y bias tratando de minimizar la suma del error cuadrático. Esto se realiza de forma continua, cambiando dichas variables en la dirección contraria a la pendiente del error, técnica denominada *descenso de gradiente*.

Redes Neuronales entrenadas mediante esta regla, dan resultados adecuados cuando se presentan datos nunca antes vistos por la red. Donde la salida otorgada por la red será similar a una salida que haya visto la red durante entrenamiento. Así, una de las propiedades distintivas de este tipo de métodos es el de alcanzar una buena *capacidad de generalización.*

El funcionamiento general de este algoritmo es el siguiente:

1- Se aplica un vector de entrada y se calculan sus correspondientes salidas.
2- Se determina una medida de error.
3- Establecer en qué dirección se deben cambiar los pesos para minimizar el error.
4- Se determina la cantidad en que es preciso cambiar cada peso.
5- Se modifican los pesos.
6- Se repiten los pasos del 1 al 5 con todos los patrones de entrenamiento, hasta que el error para todos los vectores del conjunto de entrenamiento quede reducido a un valor aceptable.

Este algoritmo es utilizado también para entrenar redes feed-forward, que necesitan entre otros, aproximar funciones de regresión no lineal y clasificación de patrones, en donde el objetivo es minimizar la función de error.

Tasa de aprendizaje

Elegir un incremento adecuado de los pesos es importante en los algoritmos de gradiente descendente como el Backpropagation, ya que influye en la velocidad de convergencia del algoritmo. Esta velocidad se controla mediante la tasa de aprendizaje, que por lo general se escoge como un número pequeño, para asegurar que la red encuentre una solución. Un valor pequeño en la tasa de aprendizaje significa que la red tendrá que hacer un gran número de iteraciones, sin embargo, si se toma un valor muy grande, los cambios en los pesos serán muy grandes, avanzando muy rápidamente por la superficie de error, con el riesgo de saltar el valor mínimo del error y estar oscilando alrededor de él, pero sin poder alcanzarlo. Es recomendable aumentar este valor a medida que disminuye el error de la red

durante la fase de entrenamiento, para garantizar así una rápida convergencia, teniendo la precaución de no tomar valores demasiado grandes que hagan que la red oscile alejándose demasiado del valor mínimo. Sin embargo con el uso de BP, no se asegura en ningún momento que el error mínimo alcanzado sea el global, ya que una vez la red se asiente en un mínimo, sea local o global, se detendrá el aprendizaje aunque el error siga siendo alto.

2.3 Series de tiempo

La red MLP al igual que otras redes, son modelos de redes de neuronas considerados estáticos (Hilera, 2000; Tabares et. al., 2006), ya que su arquitectura está diseñada para encontrar patrones de entrada y salida independientemente de la variable tiempo. Sin embargo, las RNAs pueden ser utilizadas para tratar información temporal, el cual es el caso abordado en este trabajo de investigación mediante el problema de predicción de series de tiempo (ST). Se pueden encontrar series temporales en distintos ámbitos como en la física, economía, demografía, marketing, telecomunicaciones y transporte, entre otros temas.

Uno de los puntos a resolver en el estudio de las series de tiempo es el de predicción. Es decir, dada una serie $\{x(t_1), \dots, x(t_n)\}$, tratar de describir el comportamiento de la serie, determinar el mecanismo generador de la serie temporal y buscar posibles patrones en el tiempo que permitan superar la incertidumbre del futuro.

De esta forma, una *serie de tiempo* se define como un conjunto de mediciones estadísticas de proceso en intervalos de tiempo regulares. Se denota por $\{x(t_1), x(t_2), \dots, x(t_n)\} = \{x(t): t \in T \subseteq R$, con $x(t_i)$ como el valor de *x* en el instante t_i.

Los principales objetivos que se tienen en cuenta para el análisis de una serie de tiempo son:

- Saber si los datos se presentan en forma creciente (tendencia).
- Determinar alguna influencia de ciertos periodos de cualquier unidad de tiempo (estacionalidad).
- La aparición de outliers (observaciones extrañas o discordantes).
- Simular valores futuros de la serie, bajo condiciones o restricciones definidas por políticas o criterios nuevos, para así supervisar y controlar los cambios que se producen en la serie.

Terminología en series de tiempo

A continuación se mencionan algunos términos importantes para entender el comportamiento y el análisis de una serie de tiempo.

El término *variable aleatoria,* se explica como una variable que toma valores numéricos determinados por el resultado de un experimento aleatorio. Por otro lado, un *proceso estocástico,* se describe como una secuencia de datos (variables aleatorias) que evolucionan en el tiempo. Luego entonces, las series temporales pueden ser definidas como un caso particular de los procesos estocásticos. Dentro de los aspectos estadísticos, encontramos la *Varianza, siendo una* medida que nos permite identificar la diferencia promedio que hay entre cada uno de los valores respecto a su punto central. La covarianza, en el caso de dos variables, es la media aritmética de los productos de las desviaciones de cada una de las variables respecto a sus medias respectivas.

La *Función de correlació*n entre dos variables aleatorias *X, Y* mide el grado al cual tienden a moverse conjuntamente, es una medición sobre el co-movimiento que manifiestan. Así el *Coeficiente de correlación* mide el grado de relación lineal entre dos variables, y que toma valores entre -1 y 1. Este mismo concepto se traslada al caso de una serie de tiempo, surgiendo los coeficientes de autocorrelación y de correlación cruzada

- Coeficiente de autocorrelación.- El cálculo de este coeficiente es similar al caso de variables relacionadas, pero ahora evaluado entre pares de valores de la misma serie.
- Coeficiente de autocorrelación cruzada.- Se calcula de manera similar al coeficiente de autocorrelación, pero este considera valores de dos series relacionadas, en lugar de una, por lo que sirve para cuantificar el grado de interdependencia entre dos procesos o señales. Es este trabajo, se le nombra “correlación” y se utiliza para medir la similitud entre las series de tiempo de OD original y predicha.

2.3.1 Componentes de la serie de tiempo

De acuerdo a la forma de estudiar una serie temporal, esta puede ser dividida en:

Determinística. Se considera determinística cuando se quiere hacer un estudio descriptivo de series temporales, la cual se basa en la idea de descomponer la variación de una serie en varias componentes básicas. Esta forma de estudiarlas

es conveniente cuando en la serie se observa cierta tendencia o cierta periodicidad. Las componentes de variación que se consideran habitualmente son las siguientes:

- *Tendencia*: representa el comportamiento predominante de la serie. Esta puede ser definida como el cambio de la media a lo largo de un periodo largo de tiempo
- *Ciclo*: se define por oscilaciones alrededor de la tendencia con una larga duración, aunque sus factores no son claros.
- *Estacionalidad*: es un movimiento periódico que se producen dentro de un periodo corto y conocido.
- *Aleatoriedad*: son movimientos no bien definidos que no siguen un patrón específico y que obedecen a causas diversas, el cual es prácticamente impredecible y lo representan todos los tipos de movimientos de una serie de tiempo que no son tendencia, variaciones estacionales ni actuaciones cíclicas.

Aleatoria, determina el nivel de error que obtendremos en la predicción. En este, se consideran cuatro tipos de series temporales:

- *Estacionarias*: su media no aumenta, no tienen tendencia y la varianza de los puntos no cambia.
- *No estacionarias en la media*: son series con tendencia, pero las varianzas dentro de los ciclos estacionarios son iguales.
- *No estacionarias en la varianza*: son series que presentan una tendencia en las que la varianza se va haciendo más marcada en cada ciclo.
- *Caóticas*: las más complejas de predecir, se rigen por la "*Teoría del Caos"*, con ecuaciones de funciones recursivas con gran error asociado.

Ahora bien, las RNAs pueden ser vistas como grafos dirigidos ponderados en el cual las neuronas artificiales son nodos, los arcos dirigidos (con pesos) son las conexiones entre neuronas de entrada y neuronas de salida.

2.3.2 Métodos de predicción de la serie de tiempo

Existen variados métodos para realizar estimaciones, entre los métodos clásicos existen dos tipos principales: cualitativos y cuantitativos.

Los *métodos cualitativos* son procedimientos que estiman los pronósticos basándose en juicios de expertos, se utilizan la experiencia, intuición, habilidad y sentido común de personas entrenadas para enfrentar este tipo de problemas. Por ende, las técnicas estadísticas no juegan papel importante en su implementación.

Los métodos más comunes se detallan en (Bowerman & Connell, 2007; Gerstenfeld, 1971; Rowe & Wright, 2001).

Los *métodos cuantitativos.-* necesitan el estudio de información histórica para estimar los valores futuros de la variable de interés. Estos, son basados en modelos estadísticos o matemáticos y se caracterizan por que una vez elegido un modelo o una técnica de pronóstico, es posible obtener nuevos pronósticos automáticamente. Estos modelos se pueden agrupar en dos clases: univariados y causales.

- *Univariados*. Predicen el futuro de una serie con base en su comportamiento histórico propio; son muy útiles si el patrón detectado en el pasado se mantiene hacia el futuro, de lo contrario no son aconsejables. Los modelos Integrated Autoregressive Moving Average Model (modelo ARIMA) son representativos de este grupo (Box & Jenkins, 1976; Pindick & Rubinfield, 1991).
- *Causales*. Requieren la identificación de otras variables que se relacionan de la manera causa efecto con la variable que se desea predecir. Una vez identificadas estas variables relacionadas, se construye un modelo estadístico que pretende describir la relación entre estas variables y la variable que se desea pronosticar. Los modelos de regresión lineal simple y los modelos de regresión lineal múltiple son los más conocidos de este grupo.

En años recientes, se han desarrollado otros métodos no convencionales para la predicción y la disminución de esta incertidumbre. Dichos métodos presentan un nivel mucho más alto de complejidad, en su mayoría surgieron de la imitación de procesos de la naturaleza. Algunos de estos métodos no convencionales de predicción son: redes neuronales artificiales (Hernández et. al., 2014; Hippert, Pedreira, & Souza, 2001; Senjyu, Takara, Uezato , & Funabashi, 2002), Máquinas de soporte vectorial (Kjoung, 2003; Sapankevych & Sankar, 2009), los algoritmos genéticos (Anufriev et at., 2012; Kishtawal el. al., 2003; Mahtfoud & Ganesh, 2010) y Filtros de Kalman (Kalman, 1960; Louka, y otros, 2008) entre otros.

2.3.3 Predicción de series temporales con RNAs

La predicción de series temporales puede ser considerada como un problema de modelamiento entre las entradas y la salida. Dado esto, puede ser usado este enfoque para predecir los valores futuros basado en los valores previos. Sin embargo, para ello se debe conocer la relación (función) entre los datos entrada-salida para poder diseñar un método matemático, i.e. una ecuación diferencial que permita modelar la ST. Donde, dada la dinámica de la ST podría requerir una función no lineal, si fuera un sistema caótico el que diera origen a ella. No obstante, es

común que no se conozca dicha relación entre los datos (función que los puede modelar o predecir), y es ahí donde las RNAs son de utilidad al poder aproximar dicha función desconocida, manipulando sus elementos estructurales (entradas, nodos, número de capas ocultas, etc.) así como buscando pesos adecuados para aproximarlo (algoritmo de entrenamiento).

Dada la no linealidad inherente de las RNA, es posible aplicarlas a una gran cantidad de problemas de interés para los humanos (predicción en este caso), ya que muchos de los problemas resolver, de una mejor forma o de una manera más precisa, son no lineales.

Antes de poder analizar y predecir una ST el primer paso es analizar y eliminar los datos erróneos y posteriormente un estudio de correlaciones entre las variables de entrada (y de salida). Tras ello, preferentemente se debe filtrar la entrada (datos) para eliminar en cierta medida el ruido (aunque se sabe que las RNAs trabajan bien en presencia de ruido). Posteriormente, se recomienda normalizar los valores (dependiendo de la función de transferencia que se pretenda usar, normalmente en un rango de [0, 1] o al [−1. 1]).

Cabe mencionar, que se debe establecer el tipo de predicción que se pretende realizar, esto es:

- *Valores*. Predicción de los valores numéricos.
- *Componentes*. Predecir la tendencia, variaciones de ciclo estacional, entre otros.

Ello implica una preparación diferente de datos en cada caso seleccionado. Donde se puede dividir la serie temporal en sus componentes (por ejemplo con una técnica de descomposición de señales: Fourier, Ondeletas o Descomposición Empírica en Modos entre otros) y usar una RNA para el tratamiento de cada una (Coughlin & Tung, 2004; Li & Wang, 2008; Mabrouk et. al., 2008; Saha et. al., 2006).

Por lo anterior, se puede definir la red neuronal en el contexto de series de tiempo como modelos no lineales que permiten: A) Realizar conexiones entre los valores pasados y presentes de una serie de tiempo, o bien B) Extraer estructuras y relaciones escondidas que gobiernan el sistema de información, incluso entre varias series.

Utilizar las redes neuronales en problemas de predicción de series temporales se ha vuelto de gran utilidad y una viable alternativa a los métodos comúnmente utilizados en ST caóticas, con dinámicas complejas no-lineales o estacionarias. Es

por ello que a continuación se presentan los mecanismos para abordar el problema de predicción de ST utilizando RNAs, para un pronóstico a un determinado horizonte de tiempo.

2.3.4 Predicción en un paso en el tiempo

Para poder hacer predicción con RNAs, primero se tiene que determinar cuanta información del pasado es útil para ingresar al modelo, así como saber si esa información es consecutiva o espaciada. Se tiene como ejemplo, predecir una serie periódica la cual tiene ciclos semanales.

Se podría asumir que para predecir el valor de un lunes, es necesario los datos de entrada de la semana pasada (Domingo, Sábado, Viernes,…, Martes y Lunes), donde la primera interrogante nace aquí ¿se debe considerar o no al lunes anterior? O ¿solo se debe tomar hasta el martes? así mismo, se puede pensar que una semana anterior no es suficiente, y se puede pensar en ingresar dos semanas anteriores para predecir el siguiente lunes. Todas las posibles combinaciones que podamos tener, sería la resultante de pensar en cuanta información del pasado es útil para el sistema neuronal.

La otra variable que falta describir son los espacios entre los datos de entrada, los cuales pueden ser llamados retardos. Así, para el mismo problema se puede pensar que no toda la información consecutiva es necesaria, es decir, una posible entrada valida a la RNA podría ser los datos de la semana pasada de la siguiente forma: Domingo, Viernes, Miércoles y Lunes, donde se podría decir que tenemos un retardo de dos, porque estamos tomando como entrada cada dos valores.

Por otro lado, asumiendo que no es suficiente una semana sino 5 semanas anteriores, y asumiendo que solo queremos ingresar la información del mismo día que vamos a predecir (retardo igual a 8), esta sería la entrada correcta: lunes -1, lunes -2,…, lunes –n, donde $1 < n < 5$. Así, se ha identificado dos variables importantes, el número de entradas al modelo y los retardos entre dichas entradas.

Con ello en mente, se tiene la forma de predicción a corto plazo o SSP (Single Step Prediccion), en este tipo de predicción, se pronostica el valor inmediato siguiente de los valores muestreados de la serie de tiempo. De esta forma, considerando el vector $(x(t), x(t-1), \dots, x(t-r))$ como patrón de entrada, las redes de neuronas pueden utilizarse para aproximar la función que relaciona las entradas con las salidas, con el siguiente modelo de predicción:

$$y(t+1) = F(x(t), x(t-1), \dots, x(t-r)) \tag{2.6}$$

Donde la función F representa una aproximación al problema, la cual será modelada con la RNA y $y(t+1)$ es la predicción (salida de la RNA) por el modelo en el instante de tiempo $t+1$, dado un patrón de entrada proveniente de la serie de tiempo en los $r+1$ instantes anteriores.

Tabla 2.3. Patrones de entrenamiento. Predicción a un paso en el tiempo.

	Entrada	**Predicción**
Patrón 1	$x(t), x(t-1), \dots, x(1), x(0)$	$y(t+1)$
Patrón 2	$x(t+1), x(t), \dots, x(2), x(1)$	$y(t+2)$
Patrón 3	$x(t+2), x(t+1), \dots, x(3), x(2)$	$y(t+3)$
Patrón 4	$x(t+3), x(t+2), \dots, x(4), x(3)$	$y(t+4)$
…	…	…
Patrón N - r	$x(N-1), x(r-2), \dots, x(N-r), x(r-1)$	$y(N)$

Los patrones de entrenamiento de la red se obtienen del conjunto de muestra disponible de la serie temporal del siguiente modo: dado el conjunto $\{x(t)\}_{t=0,\dots,N}$

De este modo, cada patrón de entrada recoge la información temporal necesaria para predecir el valor de la serie en el instante de tiempo *t+1,* $\forall$ *t=r,…,N-1.*
con *N* como el número máximo como posible entrada de la serie.
Al realizar el entrenamiento, se pretende que la red capture la relación entre el valor de la serie en un instante de tiempo y los *r* valores pasados. El ajuste de los pesos se realiza para minimizar el error cuadrático medio en la salida, es decir:

$$e(t+1) = \frac{1}{2}(x(t+1) - x(t+1))^2 \ \forall \ t = r, \dots, N-1 \qquad (2.7)$$

Una vez realizado el entrenamiento de la red, el modelo puede realizar predicción en un paso en el tiempo, presentando a la red los $r+1$ instantes anteriores de tiempo.

2.3.5 Predicción de múltiples pasos en el tiempo

Un problema de predicción de *múltiples pasos adelante* (también llamada “a largo plazo” o Multi-Step Prediction - MSP) podría ser considerada una de las formas intuitivas de hacer predicción. Donde de manera recurrente se aplica el modelo neuronal construido para la predicción en un paso de tiempo (similar a SSP). Aquí la estimación de los h valores $[y(t+h+1), \dots, y(t+1)]$ de una serie de tiempo

$[x(t), \ldots, x(t-r)]$ compuestos de r valores pasados. Nótese que se sigue aplicando el concepto de entradas y retardos para hacer la predicción.

A diferencia de la predicción de series temporales en SSP, la predicción en MSP se enfrenta a problemas como la acumulación de errores. Esto es así, porque al basarse en una predicción a un paso adelante, para minimizar el error local (ecuación 2.6), a partir de la segunda iteración la red recibe los valores de la serie predichos en instantes anteriores de tiempo, de manera que los errores cometidos en la predicción de estos son propagados hacia las predicciones futuras, pudiendo influir negativamente en predicciones posteriores.

Por lo anterior, en este estrategia iterativa, primero un modelo de predicción a un paso adelante es realizado (ecuación 2.5). Subsecuente a esto, se usa el valor ya predicho como parte de las variables de entrada para predecir el siguiente paso (usando el mismo modelo de un paso adelante). Se continuará de esta forma hasta que se alcance el horizonte de predicción.

La tabla 2.3 muestra las entradas y salidas para realizar las predicciones con un modelo MSP.

A pesar de estas limitaciones, la estrategia iterativa se ha utilizado con éxito para pronosticar muchas series temporales del mundo real utilizando diferentes modelos de aprendizaje, como redes neuronales recurrentes (Saad et, al., 1998), vecinos más cercanos (Bontempi et. at., 1999; McNames, 1998) y máquinas de soporte vectorial (Sorjamaa et. al., 2007; Yukun et. al., 2013).

Tabla 2.4. Predicción a múltiples pasos del tiempo. Predicción iterada.

	Entrada	**Predicción**
Patrón 1	$f(x(t), x(t-1), \ldots, x(t-r))$	$y(t+1)$
Patrón 2	$f(x(t+1), x(t), x(t-1), \ldots, x(t-r+1))$	$y(t+2)$
Patrón 3	$f(x(t+2), x(t+1), x(t), \ldots, x(t-r+2))$	$y(t+3)$
…	…	…
Patrón 4	$f(x(t+h), x(t+h-1), \ldots, x(t+1), x(t), \ldots, x(t-r+h))$	$y(t+h+1)$

Otra forma de abordar el problema de predicción en múltiples pasos consiste en construir un modelo para predecir directamente cada horizonte independientemente de los otros. Es decir, h modelos son aprendidos (uno para cada horizonte) de las

series $[x(t), \dots, x(t-r)]$. De manera que el valor a predecir en el instante $t+h+1$ a partir de lainformacion disponible en el instante actual t es:

$$y(t+h+1) = f(x(t), (t-1), \dots, x(t-d)) \tag{2.8}$$

donde f es una función no lineal que relaciona el valor de la serie en el instante $t+h+1$ con una secuencia finita de valores disponibles en el instante de tiempo actual t. Los patrones de entrenamiento de la red se obtienen de las muestras disponibles de la serie temporal, $\{x(t)\}_{t=0,\dots,N}$ y considrando como salida deseada, en el instante t, el valor de la serie en el instante $t+h+1$ (ver Tabla 2.5).

Tabla 2.5. Patrones de entrenamiento. Predicción en múltiples pasos en el tiempo utilizando predicción directa.

	Entrada	**Predicción**
Patrón 1	$x(d), x(d-1), \dots, x(1), x(0)$	$y(d-h+1)$
Patrón 2	$x(d+1), x(d), \dots, x(2), x(1)$	$y(d-h+2)$
Patrón 3	$x(d+2), x(d+1), \dots, x(3), x(2)$	$y(d-h+3)$
Patrón 4	$x(d+3), x(d+2), \dots, x(4), x(3)$	$y(d-h+4)$
…	…	…
Patrón N -d- h	$x(N-h-1), x(d-h-2), \dots, x(N-1-h-d)$	$y(N)$

Esta estrategia no utiliza valores aproximados para calcular las predicciones como en la tabla 2.3, minimizando así el error a la salida en dicho instante como sigue:

$$e(t+h+1) = \frac{1}{2}(x(t+h+1) - x(t+h+1))^2 \; \forall \; t = d, \dots, N-h-1 \tag{2.9}$$

Diferentes modelos de aprendizaje automático se han utilizado para implementar la estrategia directa para tareas de pronóstico a largo plazo, por ejemplo redes neuronales (Kline, Methods for multi-step time series forecasting with neural networks, 2004), vecinos más cercanos (Sorjamaa et al., 2007) y árboles de decisión (Tran et. al., 2009). Se han implementado otras estrategias de predicción las cuales combinan aspectos de las estrategias iterada y directa. La estrategia *DirRec* (Sorjamaa et. al., 2006), estima un conjunto de h modelos de prediccion, uno en cada paso, y cada modelo predicho lo introduce como parte del conjunto de entradas para la futura predicción. Se han desarrollado estrategias que permiten conservar, entre los valores predichos, la dependencia estocástica que caracteriza las series temporales, como lo son la estrategia *MIMO* (Multi-Entradas Multi-Salidas) y la estrategia *DIRMO*, que a diferencia de las estrategias anteriores en las que los modelos devuelven un valor escalar, estas proporcionan un vector de valores

futuros en un solo paso y combinan características de las anteriores (Bontempi, 2008; Bontempi & Taieb, 2011; Kline, 2004) (ver Figura 2.6).

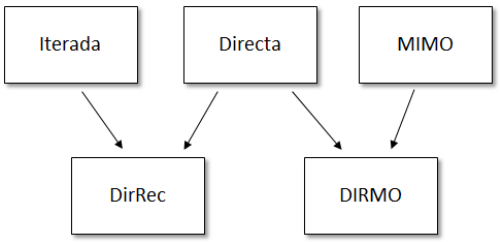

Figure 2.6: Diferentes estrategias de predicción con las relaciones entre ellas.

Con lo anterior, se puede ver que dependiendo de la estrategia seleccionada, se requerirá un número y tipo de modelos diferentes. En general, las RNAs están demostrando tener un mejor desempeño que los modelos matemáticos convencionales, especialmente con problemas de series de tiempo caóticos Por ello, las RNAs se pueden ver como una opción efectiva para el problema de predicción de la calidad del agua, la cual incluye hacer predicciones independientes de los sus variables relacionadas (temperatura, OD, pH, etc.), aún más, considerando que pueden ser tolerantes a fallos, como podrían ser datos inválidos. No obstante, aún falta tener un mecanismo que permita encontrar automáticamente la arquitectura de las redes, lo que incluye determinar el valor adecuado del número de entrada así como de retardos entre ellas, parámetros que podrán ser encontrados con los Algoritmos Evolutivos (AEs) mencionados posteriormente.

2.3.6 Medidas de rendimiento de la predicción

Tomando en cuenta que diferentes medidas de error miden aspectos diferentes de lo que queramos medir, existen varias métricas de error dentro de la predicción de una ST como la velocidad de cálculo, la interpretabilidad o la calidad de la predicción (Armstrong & Collopy, 1992; Kamaev et. al., 2012; Mahmoud, 1984; Owoeye et. al., 2013; Tyukov et. al., 2012; Yokuma & Armstrong, 1995). Las medidas de error de predicción o precisión de pronóstico son las medidas más importantes en la solución de problemas prácticos (Yokuma & Armstrong, 1995). Por lo general, las mediciones de error de pronóstico utilizadas regularmente, se aplican para estimar la calidad de los métodos de predicción y para elegir el mejor mecanismo de pronóstico (en caso de múltiples objetos). A continuación, se mostraran las medidas de error de predicciones que se utilizan para el pronóstico de series de tiempo, las cuales, se dividen en grupos de acuerdo con el método que utilizan para calcular un valor de error para cierto tiempo t.

Medidas de error absoluto

Este grupo está basado en el cálculo del error absoluto e incluye estimaciones basadas en el error e_t

$$e_t = (y_t - Y_t) \tag{2.10}$$

donde y_t es el valor real en el tiempo t, y Y_t el valor predicho en el tiempo t.

El MAE (error absoluto medio) es una indicación de la desviación media de los valores predichos a los valores observados correspondientes y puede presentar información sobre el comportamiento a largo plazo de los modelos; cuanto menor es el MAE, mejor es la predicción del modelo a largo plazo.

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_t| \tag{2.11}$$

Esta métrica se puede ver en la ecuación 2.11, donde n es el horizonte de predicción.

El MSE (Error Cuadrático Medio) es el más estable de las métricas de error. Sin embargo, interpretar el valor MSE puede ser engañoso, ya que para el error cuadrado medio incrementará errores en predicciones a largo plazo.

$$MSE = \frac{1}{n}\sum_{t=1}^{n}({e_t}^2) \tag{2.12}$$

La Raíz Cuadrada del Error Cuadrático Medio (RMSE), es otro método usado, y en este caso es aplicar la raíz cuadrada al MSE, lo que significa que los errores no serán valores tan grandes como pudiera ser el MSE, su ecuación es la mostrada a continuación:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}({e_t}^2)} \tag{2.13}$$

Otras medidas

Este grupo incluye medidas para mejorar la dependencia de escala. Una medida en este grupo es la Raíz Cuadrada Normalizada del Error Cuadrático Medio (NRMSE), la cual es usado como medida de rendimiento en este trabajo y se calcula por:

$$NRMSE = \sqrt{\frac{\sum_{t=1}^{n}({e_t}^2)}{\sum_{t=1}^{n}(y_t - \bar{y})^2}} \tag{2.14}$$

donde $\bar{y}$ es la media de y_t . Es de señalar, que esta estimación es afectada por los valore atípicos, si tienen un valor mucho más grande que el máximo valor “normal”.

2.4. Algoritmos evolutivos

Los algoritmos evolutivos, son a grandes rasgos, una serie de modelos computacionales basados en la evolución y supervivencia del más apto (Fraser, 1957, Bremermann, 1958; Holland, 1975). Ofrecen un método de búsqueda efectivo para problemas complejos de optimización, principalmente cuando resulta costoso realizar un gran número de iteraciones o donde existen dos o más funciones objetivo, ya que se adaptan excepcionalmente bien a problemas multi-objetivo. Por lo tanto, los AE están indicados para resolver todo tipo de problemas que puedan ser expresados en forma de problema de optimización de una o varias funciones sujetas a un número variable de restricciones y a una o más restricciones del entorno.

Dado que en una red neuronal el proceso de aprendizaje puede ser entendido principalmente como un proceso de optimización para encontrar los pesos que mejor se ajusten al problema (minimizando el error), los algoritmos evolutivos se vuelven una alternativa a los métodos de entrenamiento tradicionales, ya que al no quedar detenidos en mínimos locales permiten aumentar la velocidad de convergencia a la solución (Chacraborty, 2010). Esto lo llevan a cabo, debido a que realizan la búsqueda simultánea de un resultado dentro de un conjunto de posibles soluciones (individuos). Reproducen genéticamente una población de individuos a través de una cantidad de generaciones y de esta manera buscan una solución (Bodenhofer, 2004). Esto lo llevan a cabo mediante los operadores básicos de los AEs, los cuales permiten que vaya aumentando gradualmente la calidad de la solución, hasta alcanzar un resultado que cumpla con las condiciones requeridas (Forrest 1996).

No obstante, los AEs son sensibles a la manera en que codifiquemos a los individuos de la población, i.e., la codificación utilizada puede influir sensiblemente en las posibilidades de convergencia. Esta codificación (fenotipo) es la forma de representar el problema en términos que los algoritmos pueden manipular (genotipo). Tanto es así, que algunas variantes de AEs se diferencian precisamente en elegir una u otra forma de codificación interna. Por tanto, la tarea más importante en un AE será encontrar la representación adecuada para las soluciones. La segunda tarea crítica será elegir correctamente la función que guiará la búsqueda, y que puede ser única, o bien, formarse de la combinación ponderada de varias funciones. Esta función suele recibir el nombre de función objetivo o función de fitness (adaptabilidad).

En general, el funcionamiento de un AE es el siguiente: dada una población de soluciones candidatas (soluciones potenciales), y en base al valor de la función objetivo para cada uno de los individuos de dicha población, se seleccionan los mejores (los que minimizan la función objetivo, i.e. error más bajo) y se combinan para generar nuevos individuos. Este proceso se repite cíclicamente, como se muestra en la figura 2.7 donde se aprecia el esquema general de un algorítmico evolutivo. En primer lugar, se procede a la inicialización de la población. Para cada individuo de la población se selecciona un valor que puede ser completamente aleatorio. También puede considerarse tomar como valores iníciales una aproximación a la solución, i.e. introduciendo conocimiento del experto, lo cual ayuda a acelerar la convergencia. Después se aplica a cada individuo la función objetivo para que tan bueno es un individuo como solución, lo que da una medida de lo adaptado que está cada con respecto a los demás.

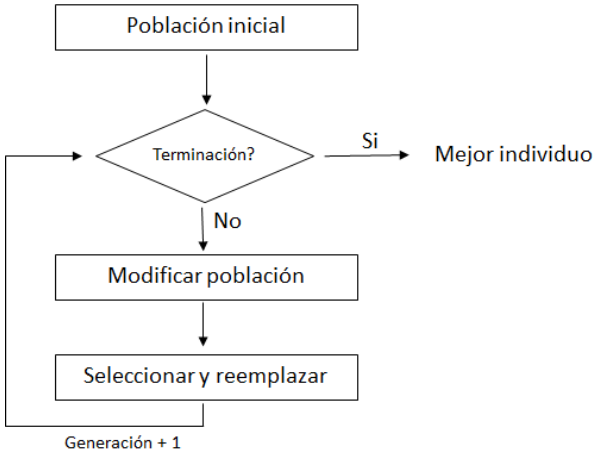

Figura 2.7.- Diagrama general de un algoritmo evolutivo.

En función del valor obtenido, se ordena la población, quedando así en primer lugar los individuos más adaptados. Se seleccionan entonces los individuos que se van a cruzar y mutar para generar la nueva población (de hijos), i.e. la población que remplazará a la actual.

Elegir uno u otro método de selección determinará la estrategia de búsqueda del Algoritmo. Si se opta por un método con una alta presión de selección, se centra la búsqueda de las soluciones en un entorno próximo a las mejores soluciones actuales. Por el contrario, optando por una presión de selección menor se deja el camino abierto para la exploración de nuevas regiones del espacio de búsqueda. Lo que puede implicar mezclar mejores con mejores, o permitirle al algoritmo combinar mejores con peores. Cabe recalcar que los operadores de cruzamiento y mutación también influyen en la forma que el algoritmo estará buscando nuevas soluciones, de ahí que deberán ser operadores adecuados el tipo de problema en cuestión.

2.4.1. Terminología

Un individuo representará una posible solución al problema que se pretenda resolver por medio de un algoritmo evolutivo. La representación del individuo puede variar según la codificación utilizada, pero en general se trata de una serie de parámetros cuyo conjunto se llama ***cromosoma***, el cual comprende un conjunto de estructuras individuales llamadas ***genes***. Este conjunto de cromosomas de un individuo en concreto se llama **genotipo**. El valor que representa el genotipo se denomina **fenotipo**. Cada gen codifica una característica de un individuo, y la ubicación del gen (locus) determina la característica particular que representa el gen. Los diferentes valores de un gen, se denominan alelos.

Para hallar una solución se parte de un conjunto inicial de ***individuos***, generado aleatoriamente, esto es lo que se conoce como población. De esta manera lo que vamos a tener es un conjunto de posibles soluciones a nuestro problema que vamos a hacer evolucionar con base en una serie de operadores de manera que en cada iteración del algoritmo habremos obtenido una población distinta, es decir, una nueva ***generación***.

2.4. 2 Métodos de selección

Entre los métodos de selección más usuales encontramos:

- Selección basada en rango: Según este criterio se seleccionan los k individuos mejor adaptados. Aunque puede usarse también para eliminar

toda tendencia a seleccionar a mejores o peores, como lo hace el FS-EPNet adelante mencionado.

- Selección por ruleta: Consiste en dar a cada individuo una probabilidad de ser seleccionado proporcional a su *fitness*. En este tipo de selección un individuo puede seleccionarse dos veces, incluso cruzarse consigo mismo.
- Selección por torneo: La idea principal de este método consiste en realizar la selección basándose en comparaciones directas entre individuos. Existen dos tipos de selección mediante torneo y dependiendo de la elección que se realice, puede ser de forma *determinista* o *probabilística*. Al final, el individuo con la mejor adaptabilidad es la que gana (es seleccionado).
- Selección a medida (ad-hoc): Muchas veces, la naturaleza del problema o de los datos de entrada requieren que la selección se realice siguiendo criterios específicos al problema en cuestión.

2.4.3 Operadores evolutivos

En investigación frecuentemente es necesario definir operadores genéticos queque se ajustan mejor a las características del problema. Un operador especializado suele se convierte en una buena opción si es posible su implementación.

Cruce

El operador de cruce, normalmente representa el mecanismo más importante y está en casi todos los AEs, ya que en este paso es donde se produce el intercambio de información genética. Aunque existen muchas variantes a la operación de cruce (crossover del Inglés) entre las más usuales encontramos:

- Cruce de un punto. Es la técnica más sencilla, y consiste en cortar dos cromosomas por un punto determinado aleatoriamente y se intercambian estos segmentos para generar dos hijos, como se muestra en la figura 2.8.

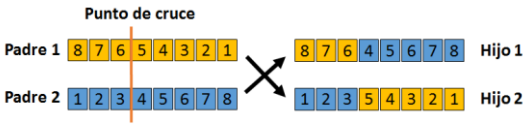

Figura 2.8.- Operación de cruce por un punto.

- Cruce de varios puntos: los dos cromosomas se cortan por dos o más puntos, y el material genético situado entre ellos se intercambia (ver Figura 2.9).

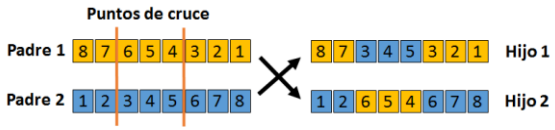

Figura 2.9.- Operación de cruce por varios puntos.

- Cruce uniforme. Cada gen de la descendencia tiene las mismas probabilidades de pertenecer a uno u otro padre. Se genera un patrón aleatorio de 1s y 0s (mascara), y se intercambian los genes de los dos cromosomas padres que coincidan donde hay un 1 en el patrón. También, se puede generar un número aleatorio para cada bit, y si supera una determinada probabilidad se intercambia ese bit entre los dos cromosomas (ver Figura 2.10).

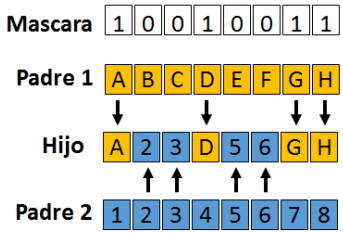

Figura 2.10.- Operación de cruce uniforme.

Mutación

Mientras el operador de cruce mezcla la información genética de los individuos para llevar a cabo la explotación de soluciones a través del espacio de búsqueda, el operador mutación introduce un mecanismo de diversidad en la búsqueda, facilitando la exploración de áreas aún no tratadas. Por lo tanto, se puede decir que este operador introduce perturbaciones aleatorias a los individuos seleccionados. Además, su funcionamiento permite recuperar características de los individuos que por medio de la selección y el cruce podrían haberse perdido, cayendo en el olvido y que, aunque siguieran siendo buenas características, sería imposible recuperar sin la acción del operador mutación.

Cuando una cadena resulta afectada por este operador, se selecciona un lugar de mutación. El valor de dicho lugar se intercambia o sustituye por otro de los posibles que se están utilizando. Las opciones mayormente utilizadas para realizar la mutación son: A) *Mutación estándar o por intercambio*, siendo el operador más común en la codificación binaria, así se seleccionan dos genes aleatoriamente con una probabilidad muy baja (entre 0.1 y 1%), y según la probabilidad obtenida se muta o no el bit, intercambiando unos por ceros y viceversa; B) *Mutación uniforme*,

este operador reemplaza el genoma, ya sea inferior o superior con un número aleatorio de acuerdo al rango de entrada; C) *Mutación no uniforme*, fue desarrollado para codificaciones con números reales, donde este operador realiza una búsqueda uniforme en el espacio, aunque más local en las generaciones finales. La probabilidad de crear una solución cercana al padre es mayor que la de crearla a él y a medida que crecen las iteraciones, aumenta dicha probabilidad; D) En el Cambio del orden se seleccionan dos genes de forma aleatoria intercambiando sus posiciones, generalmente se utiliza en permutaciones (ver Figura 2.11).

Figura 2.11.- Operadores de mutación. a) Estándar. b) Cambio de orden.

2.4.4 Tipos de algoritmos evolutivos

Los algoritmos evolutivos están considerados dentro de las técnicas no convencionales de optimización para problemas reales o los que requieran una codificación binaria. A partir de la creación de estas estrategias evolutivas aparecieron diferentes paradigmas como lo son: la Programación Evolutiva (Fogel et. al., 1966) Algoritmos Genéticos (Holland, 1975), Programación Genética (Koza, 1990), y Estrategias de Evolución (Rechenberg/Schwefel, 1973), usando los operadores genéticos descritos anteriormente para llevar acabo la evolución, i.e. la búsqueda de una solución adecuada al problema en cuestión. Los tipos de algoritmos evolutivos se describen a continuación:

Programación Evolutiva: Técnica en donde la representación del problema se realiza mediante números reales, y emplea mecanismos de mutación y selección. A diferencia de las estrategias evolutivas, la programación evolutiva no emplea el cruzamiento como una operación.

Algoritmos Genéticos: Son los más utilizaos actualmente. Modelan el proceso de evolución como una sucesión de frecuentes cambios en los genes. Como método de selección emplea el mecanismo de la ruleta y siempre se usa el operador de elitismo, el cual evita perder la mejor solución encontrada.

Estrategias Evolutivas: Diseñada originalmente para problemas de optimización discretos y continuos. Trabaja con números reales en forma de vectores, que codifican las posibles soluciones de problemas numéricos. Utiliza cruce (aritmético),

mutación y operación de selección, ya sea determinística o probabilística, elimina las peores soluciones de la población y no genera copia de aquellos individuos con una aptitud por debajo de la aptitud promedio.

Programación genética: Aquí, los individuos son programas o autómatas de longitud variable, descritos mediante Expresiones-S de LISP representadas habitualmente como árboles, los cuales emplean cruce y mutación además de mecanismos de selección.

En los últimos años han proliferado todo tipo de soluciones híbridas que permiten que no haya una línea clara entre los diferentes algoritmos y puedan tomar las características buenas de uno y otro. La lógica borrosa, las redes neuronales y los paradigmas evolutivos son metodologías complementarias en los trabajos de diseño e implementación de sistemas inteligentes que implementan técnicas hibridas. Para aprovechar las ventajas y eliminar sus desventajas, en aplicaciones operativas reales, se están proponiendo la integración de varias de estas metodologías.

2.4.5. Función de evaluación y función de aptitud (fitness)

La función de aptitud (fitness) es la función objetivo en los problemas de optimización. Ésta, se debe maximizar y/o minimizar, encontrando valores para los diferentes parámetros que resulten óptimos al ser reemplazados en la función objetivo. La función objetivo debe reflejar los aspectos más relevantes del problema; estableciéndose las condiciones que restringen los resultados proporcionados por el algoritmo. El valor de la Función de Aptitud representa la calidad de la solución o el fitness de cada individuo.

2.4.6 Evolución de redes neuronales mediante algoritmos evolutivos

La construcción de una RNA no es una tarea trivial, siendo en si un problema combinatorio el cual es considerado dentro de los problema NP-Completos, por todas las variables que se tienen que ajustar. La evolución de la arquitectura de la red implica una adaptación de la topología de la red que responda mejor al problema sin intervención del experto, lo que representa un acercamiento al diseño automático de RNAs, dado que los pesos y la arquitectura se pueden evolucionar.

Elección de la codificación y representación

En la evolución de los pesos de conexión, así como en la evolución de la arquitectura de la red, los operadores genéticos son fundamentales. Sin embargo, la forma en la que se apliquen estos operadores depende de cómo las redes son

representadas dentro del algoritmo evolutivo. Por ejemplo, una codificación directa (transcripción uno-a-uno de la estructura del genotipo al fenotipo) en la cual se suele representar a las RNAs por una matriz binaria de conectividad y en donde para el caso de esta tesis, el algoritmo FS-EPNet ocuparía dicha configuración. Por el contrario, es posible emplear una codificación indirecta, donde pueden encontrar enfoques alternativos, tales como gramáticas o autómatas celulares. Por ejemplo, para representar una RNA se tiene el algoritmo NEAT (Kenneth & Risto, 2002) el cual trabaja bajo el principio de complexificación, esto es, ir aumentando gradualmente el tamaño de las redes, donde para cada nodo se tiene un listado de conexiones entrantes así como conexiones salientes.

Evolución de RNAs

Anteriormente a los AEs, el diseño de la arquitectura de RNAs había sido hecho manualmente, por un experto por medio de un proceso de ensayo y error. Para ello, los enfoques más ampliamente estudiados eran el incremental donde consistía en una capa oculta e ir incrementando gradualmente los nodos en ella, o bien y los algoritmos de poda, donde se iniciaba con RNAs demasiado grandes, y se iban podando los nodos hasta llegar a una arquitectura adecuada. Se asume que en cada modificación (para ambos enfoques) se reentrenaba la red para determinar si el cambio había tenido efecto positivo o no, y para determinar si se debía continuar con el proceso de adaptación.

Así, una red con muy pocas neuronas puede experimentar la dificultad de aprender la tarea requerida, mientras que una red con demasiadas neuronas puede superar sus salidas a los patrones de entrenamiento, disminuyendo su capacidad de generalización cuando se presentan nuevos patrones.

El diseño de una arquitectura óptima puede formularse como un problema de búsqueda en el espacio de las arquitecturas, donde cada punto representa una posible arquitectura de red. Este problema de optimización puede resolverse, más fácilmente usando un AE que usando métodos incrementales o de poda, por las siguientes razones:

- El espacio de búsqueda es demasiado grande, al punto de hacerlo intratable con algoritmos convencionales. Dado que el número de nodos y conexiones no se conoce de previamente.
- La función de error no es diferenciable, ya que los cambios en el número de nodos o conexiones son discretos.
- Arquitecturas similares pueden tener capacidades diferentes de aproximación al problema.

- La superficie de la función de error es multimodal, ya que arquitecturas diferentes pueden tener capacidades similares de aproximar el problema.

Varios autores han propuesto métodos que hacen evolucionar simultáneamente tanto los pesos como la arquitectura, mejorando los resultados obtenidos. Así, Liu y Yao (Liu & Yao, 1996) aplicaron la programación evolutiva (PE) para la evolución de RNAs con el algoritmo EPNet, el cual entre otras cosas, realiza evolución de vectores de números reales para desarrollar RNAs.

2.4.7 Algoritmo EPNet

El algoritmo descrito e implementado como parte de esta tesis se basa fundamentalmente en el algoritmo EPNet (Programación Evolutiva de Redes Neuronales). El algoritmo EPNet fue diseñado, implementado y utilizado para los estudios empíricos por Xin Yao y Yong Liu (Liu & Yao, 1996). Este algoritmo híbrido (dado que usa AEs para evolucionar RNAs) está enfocado en evolucionar redes del tipo Feedforward. Combina la evolución de arquitecturas de RNAs con el aprendizaje de los pesos, donde este último es llevado a cabo con una modificación del algoritmo de Backporpagation (BPM). A través de operadores de mutación (sin usar cruzamiento) se hace evolucionar el número de los nodos y conexiones incluyendo el bías, donde después de cada modificación se entrena la red para volver a medir la adaptabilidad de ella. Así, son usados cinco operaciones de mutación: 1) entrenamiento híbrido usando el algoritmo BPM y recocido simulado (Simulated Annealing - SA); 2) eliminación de nodo; 3) eliminación de conexión; 4) adición de nodo y 5) adición de conexión. Notar que la primera mutación está enfocada a la modificación de los pesos y las otras 4 a la arquitectura de la red.

El énfasis del algoritmo está enfocado en mantener el acoplamiento de comportamiento entre el padre y la descendencia, el cual es llevado a cabo mediante el uso de la programación evolutiva, y un entrenamiento parcial después de cada mutación. Donde dicho entrenamiento parcial es útil para compensar cualquier error introducido a la red después de una modificación, así como para acelerar la convergencia del algoritmo. Este algoritmo está enfocado en eliminar antes de agregar elementos neuronales, y como resultado no se requiere un factor de penalización en la complejidad de la arquitectura de la red al momento de calcular la adaptabilidad, donde al mismo tiempo se mantiene su enfoque en evolucionar arquitecturas pequeñas, con el máximo rendimiento posible.

El algoritmo EPNet se compone de los siguientes pasos:

1) Generar una población inicial de *M* redes inicializadas de forma aleatoria. El número de nodos ocultos y la densidad de conexión inicial para cada red se generan aleatoriamente de forma uniforme, dentro de ciertos rangos establecidos previamente. Los pesos iniciales aleatorios están uniformemente distribuidos dentro de un pequeño rango que van de [-0.5, 0.5].
2) Entrenar parcialmente con un número pequeño (K_o) de épocas a cada red de la población con el BPM, el cual cuenta con tasas de aprendizaje auto adaptables. El número de épocas K_o, es especificado por el experto. El valor de error e (ec. 2.15) de cada red en el conjunto de validación se comprueba después de un entrenamiento parcial. Si e no se ha reducido significativamente, entonces la suposición es que la red está atrapada en un mínimo local y la red está marcada con "falla". De lo contrario, la red está marcada con "éxito".

$$e = 100\frac{O_{max} - O_{min}}{T.n.}\sum_{t=1}^{T}\sum_{i=1}^{n}(Y_i(t) - Z_i(t))^2 \qquad (2.15)$$

   donde O_{max} y O_{min} son los valores máximo y mínimo de los coeficientes de salida en la representación del problema, n es el número de nodos de salida, $Y_i(t)$ y $Z_i(t)$ son las salidas actuales y deseadas del nodo i.

3) Clasifique las redes en la población de acuerdo con sus valores de error, de lo mejor a lo peor.

4) Si la mejor red encontrada es aceptable o se ha alcanzado el número máximo de generaciones, detener el proceso evolutivo y vaya al paso 11, de lo contrario, continuar.

5) Utilice la selección basada en el rango (donde se elimina toda tendencia de selección) para elegir una red de la población. Si su marca es "éxito", vaya al paso 6, o bien vaya al paso 7.

6) Entrenar parcialmente la red principal para las épocas K_1 usando el BPM para obtener una red de descendencia (hijo) y marcarla de la misma manera que en el Paso 2, donde K_1 es un parámetro especificado por el usuario. Reemplace la red principal con la descendencia de la población actual y vaya al paso 3.

7) Entrenar a la red seleccionada con un algoritmo de recocido simulado (*SA*) para obtener una red de descendientes. Si el algoritmo *SA* reduce el error e (dado por

la ecuación 2.15) significativamente, marque a la descendencia con "éxito", reemplace a su padre por ella en la población actual y luego vaya al paso 3, de lo contrario descartar esta descendencia y vaya al paso 8.

8) Primero decida el número de nodos ocultos N_{hidden} a eliminar generando un número aleatorio uniformemente distribuido entre uno y un número máximo especificado por el usuario. N_{hidden} es normalmente muy pequeño en los experimentos, no más de tres en la mayoría de los casos. A continuación, elimine los nodos ocultos N_{hidden} de la red principal uniformemente al azar. Entrenar parcialmente la red que acaba de ser podada, con el algoritmo BPM. Si la red es mejor que la peor red de la población actual, reemplace lo peor por la descendencia y vaya al Paso 3, o de lo contrario, deseche la descendencia y vaya al paso 9.

9) Calcule la importancia aproximada de cada conexión en la red principal de acuerdo a (Liu & Yao, 1996). Decida el número de conexiones a eliminar de la misma manera que se describe en el paso 8. Eliminar aleatoriamente las conexiones de la red principal de acuerdo con la importancia calculada. Entrenar parcialmente la red podada mediante el BPM. Si la red es mejor que la peor red de la población actual, reemplace lo peor por la descendencia y vaya al Paso 3, en caso contrario, deséchela y vaya al paso 10.

10) Decida el número de conexiones y nodos a añadir. Calcule la importancia aproximada de cada conexión virtual con peso cero (Liu & Yao, 1996). Agregar aleatoriamente las conexiones a la red principal para obtener $descendencia_1$ según su importancia. Generar la $descendencia_2$ al agregar los nodos seleccionados, donde la adición de cada nodo se implementa dividiendo un nodo oculto seleccionado al azar en la red principal. Posteriormente, Entrenar a ambos hijos parcialmente $descendencia_1$ y $descendencia_2$ con el algoritmo de Backpropagation Modificado. Reemplace la peor red en la población actual por el mejor de los dos descendientes (selección por torneo) y vaya al Paso 3.

11) Después del proceso evolutivo, entrene la mejor red en el conjunto combinado de entrenamiento y validación usando el BPM hasta que "converga".

La figura 2.12 muestra el diagrama general del algoritmo EPNet, donde la figura 2.12a presenta el diagrama general y la figura 2.12b las mutaciones de él. Como se explicó anteriormente el EPNet usa una codificación directa donde se tiene una matriz binaria para representar los nodos y las conexiones, a su vez se tiene una

matriz del mismo tamaño para representar los pesos de ella, tanto en la matriz binaria como en la de pesos se tiene representado el parámetro de bias. La función de transferencia es sigmoide según lo indicado por los autores.

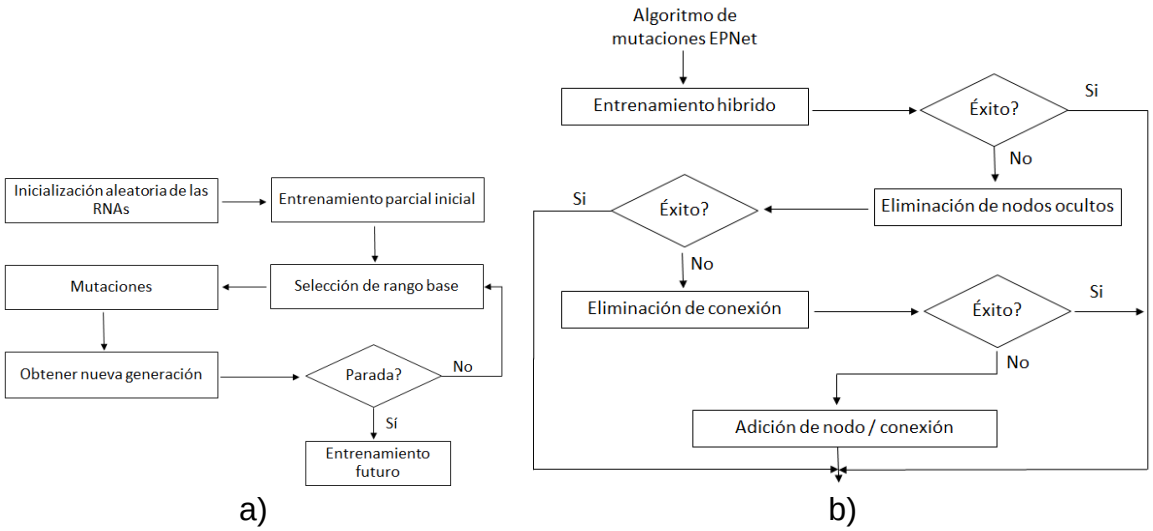

Figura 2.12.- Algoritmo EPNet. Procedimiento general (figura 2.12 a) y mutaciones del EPNet (figura 2.12 b)

2.4.8 Multi-Layer Perceptron Generalizado

El algoritmo EPNet usa una representación de Perceptrón Generalizado Multicapa (GMLP) para representar la red, donde algunas conexiones entre nodos son permitidas en forma feedforward. Se diferencia del MLP, en que en éste, los nodos están organizados por capas y solo conexiones entre capas consecutivas están permitidas. Sin embargo, GMLPs pueden permitir arquitecturas más generales, y potencialmente más poderosas, que las MLP, permitiendo conexiones de cualquier nodo anterior a uno delantero (asumiendo que están numerados de forma consecutiva). Así es posible tener conexiones de entradas a nodos de salida, incluso nodos de salida que están conectados a otros nodos de salida. Permitiendo tener redes muy compactas y con menos conexiones y nodos que con las MLP.

De forma general, un GMLP, consta de una capa de entrada, una capa de salida y un número de nodos ocultos interconectados entre ellos. La representación de un GMLP se puede ver en la figura 2.13. Ahí se nota que el ith nodo, siempre que no sea un nodo de entrada, tiene conexiones desde cada jth nodo, $j < i$. Por lo que permite conexiones desde entradas a nodos ocultos (IH), entradas a salidas (IO), nodos ocultos a nodos ocultos (HH), nodos ocultos a salidas (HO) y nodos de salida

a salida (OO). Donde se han tomado las siglas en Inglés para mantener la misma notación: Input (I), Hidden (H) y output (O).

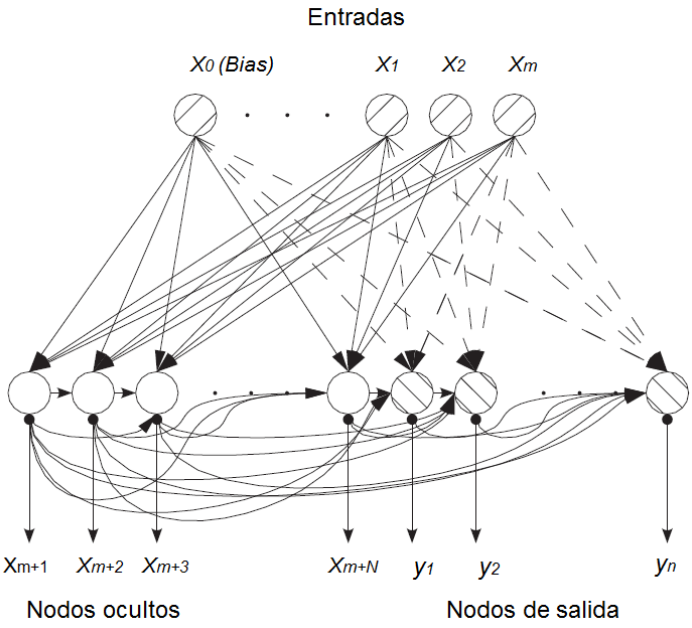

Figura 2.13.- Representación del Perceptrón Multicapa Generalizado.

2.4.9 Algoritmo FS-EPNet

El algoritmo EPNet (Yao, 1997; Landassuri et. al., 2009) se enfoca únicamente en evolucionar arquitecturas en RNAs (nodos ocultos y conexiones), lo cual presenta limitantes si se quiere adaptar entradas y retardos. Así, el algoritmo Feature Selection EPNet (Landassuri et. al., 2012; López et. al., 2013) evoluciona las arquitecturas de RNAs incluyendo las entradas y retardos entre ellas. Esto con 4 mutaciones más que el EPNet: agregación y/o eliminación de entradas y retardos. Notar que sin esos 4 operadores genéticos del FS-EPNet se obtienen las mutaciones originales del algoritmo EPNet (ver Figura. 2.12b). Se debe tener en cuenta que el algoritmo general para el FS-EPNet es el mismo que 2.12a. Por otro lado, y de forma similar al EPNet, si existe una modificación de entradas o retardo, hay que volver a reajustar los pesos de la red con un entrenamiento parcial.

Dado que la modificación de entradas y retardos puede afectar significativamente el rendimiento de la red, hay que mantener el rango de mutaciones lo más bajo posible, por ello cuando se mutan estos dos parámetros, se toma un valor aleatorio, del tipo entero entre el rango [1, 2]. Así, a lo más se pueden modificar (agregar o

eliminar) 2 entradas, y lo mismo sucede para los retardos. Notar también que si alguno de estos dos parámetros cambia, también lo hacen los patrones de entrenamiento, de ahí que una modificación de estos valores implicar reajustar los patrones de entrenamiento para que se ajusten a la nueva arquitectura de la RNA.

Como puede verse en la figura. 2.14, al final de las mutaciones de eliminación se introduce las eliminaciones de entradas y retardos, esto con el fin de permitir que la arquitectura primero se adapte antes de hacer una modificación mayor. Así mismo se puede asumir que existe un mayor número de combinaciones entre nodos ocultos y su conectividad, en comparación con las entradas y retardo, de ahí que se le da mayor importancia a los primeros de ser adaptados. Por el contrario, las mutaciones de agregación de entradas y retardo se introducen al mismo nivel de las otras agregaciones, i.e. al nivel donde se aplica el torneo de los hijos.

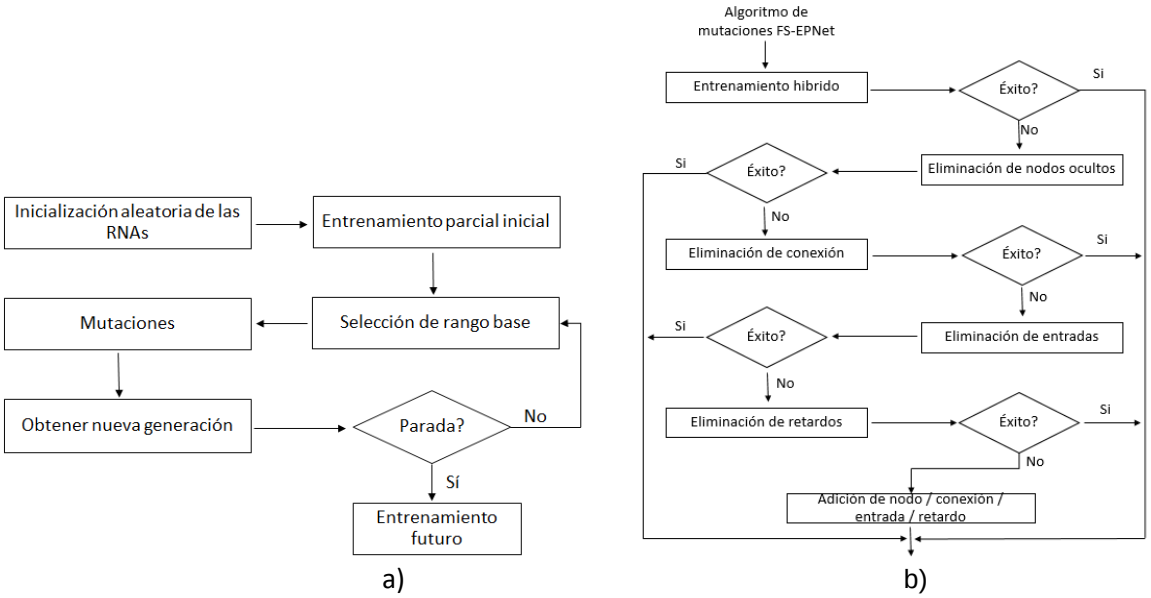

Figura 2.14.- Algoritmo de Selección futura EPNet (FS-EPNet). Procedimiento general (figura 2.14 a) y mutaciones FS-EPNet (figura 2.14 b)

Similar el EPNet, el algoritmo FS-EPNet no cuenta con operador de cruzamiento, y la búsqueda de nuevas soluciones las lleva a cabo con estos 9 operadores de mutación, donde de nueva cuenta, la primera mutación (entrenamiento híbrido) es la única mutación que modifica los pesos, y el resto se enfoca al aspecto arquitectónico. Es así, que este algoritmo está orientado a eliminar antes de agregar elementos neuronales (entradas, retardos, nodos ocultos o conexiones), controlando el crecimiento de las redes (Lopez et. al., 2013).

2.5 Estado del Arte

En esta sección se describen las investigaciones más recientes y actuales de los sistemas de control y predicción de oxígeno disuelto. Donde cabe notar que este trabajo de tesis presenta algunas diferencias significativas respecto a los encontrados en la literatura.

En la figura 2.15 se muestra el sistema de control del OD en Daa et al. (2012), donde determinan la demanda bioquímica de oxígeno; suponen que se tiene un sensor de OD permanentemente en el estanque, no obstante, en los sistemas acuícolas no siempre es así, por lo que si se contara con un sensor exclusivo para el estanque, este sistema de control tendría grandes inconvenientes, por ende el control lo efectúan con los datos de entrada del sensor en tiempo real. Para el control solo se basan en determinar la debanda bioquímica de oxígeno y con esto establecen la potencia necesaria de trabajo de los aireadores necesitada.

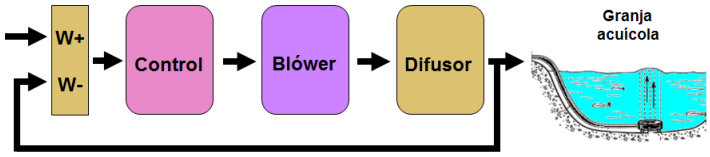

Figura 2.15.- Diagrama de bloques de control FLC para oxígeno disuelto.

En Huan et al. (2013) el sistema de monitoreo mide temperatura, corriente, PH y nivel del agua. Mediante un control Proporcional, Integral y Derivativa (PID) (ver Figura 2.16) realizan el control de los parámetros. Sin embargo, para monitorear el OD y todos los demás parámetros este sistema necesita tener tantos sensores realizando las mediciones constantemente como distintos paramentos se midan.

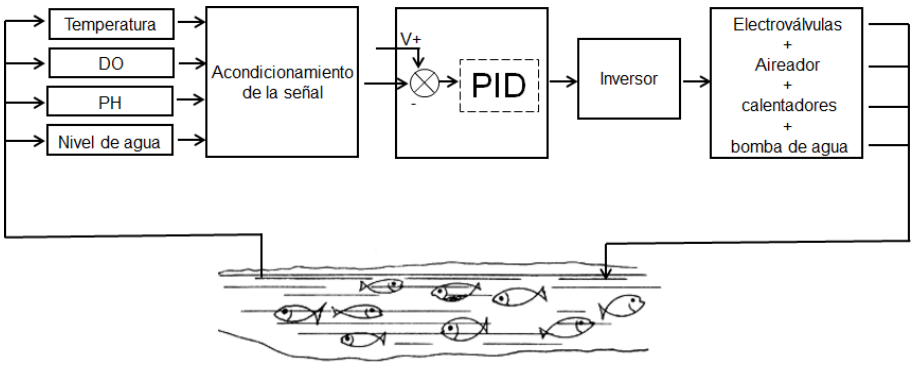

Figura 2.16.- Sistema de control basado en PID.

Una diferencia respecto al sistema propuesto en esta tesis, es que el control no está basado en predicciones, por lo que puede haber ocasiones que no sea necesario tener en funcionamiento el sistema de control dado que, por el comportamiento dinámico del sistema, los parámetros se podrían reestablecer en un futuro, siendo innecesario tenerlos funcionando continuamente.

En Johan et al. (2014) se monitorea en tiempo real mediante una tarjeta electrónica el PH, Humedad y Oxigeno en un invernadero, asimismo, dichos parámetros distan mucho de los que hay en un ambiente acuícola por lo que solo se toman algunas características semejantes en el diseño electrónico (ver Figura 2.18). .

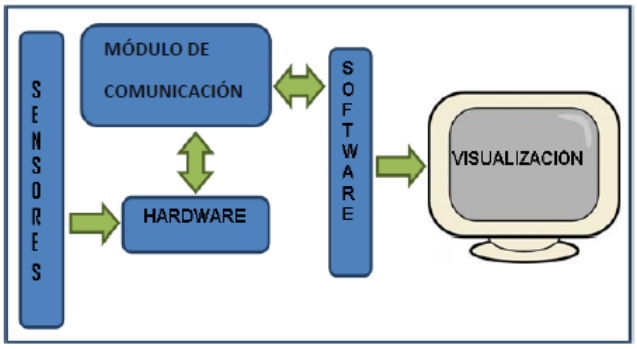

Figura 2.17.- Diagrama a bloques del sistema.

El estudio de Areerachakul et al. (2013) se basa en una red neuronal (ver Figura 2.18) para la predicción de oxígeno disuelto, el cual toma 11 parámetros de entrada

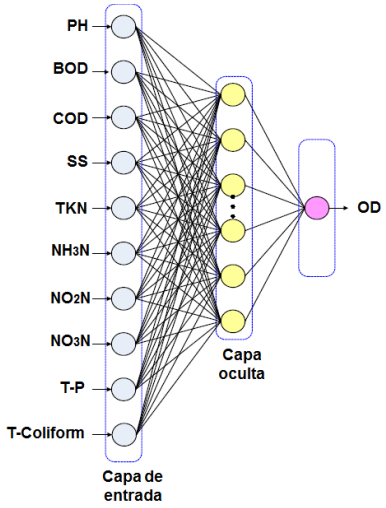

Figura 2.18.- Ejemplo de una arquitectura de red neuronal para la estimación de las variables.

para la red neuronal, sin embargo, en él se realiza una “estimación” y no una predicción a futuro, ya que con los parámetros (excepto el OD) de entrada se calcula la concentración de OD pero en ese momento.

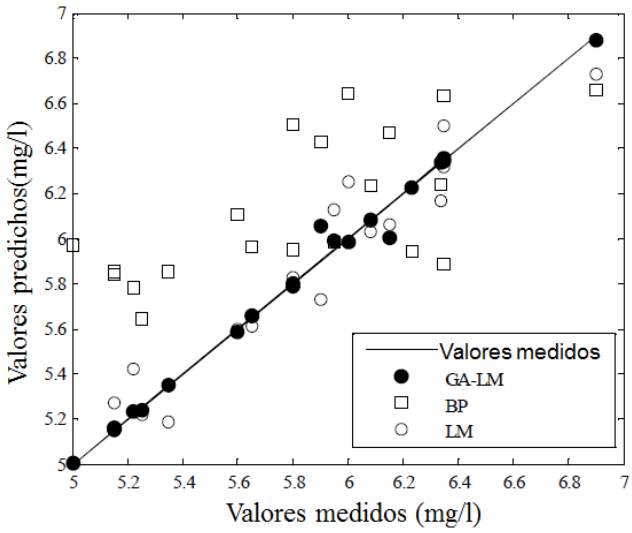

Figura 2.19.- Comparación entre los valores predichos con cada modelo de predicción.

En Miao et al. (2010) se realiza predicción del OD, solo que la realizan a un paso adelante. Además, el diseño de la arquitectura para entrenar la red se realiza de forma manual y el entrenamiento es realizado mediante una combinación del algoritmo Levenberg-Marquart con algoritmos genéticos. Realizan también una comparación de la predicción con un entrenamiento con el BackPropagation y predicción usando el algoritmo Levenberg-Marquart sin algoritmos genético (ver Figura 2.19).

CAPÍTULO 3
Diseño de modelos de predicción

En esta sección se presenta el modelo computacional a diseñar en este trabajo, donde se puede observar la metodología para la obtención de los datos y el uso de las RNAs para la predicción de OD. Como se comentó, el algoritmo evolutivo FS-EPNet se utiliza para optimizar o diseñar la arquitectura de red, usando el algoritmo Backpropagation Modificado para hacer la actualización de pesos, esto permite obtener una red compacta (pocos elementos), con buena adaptabilidad y un rendimiento adecuado en la predicción para los fines de este trabajo, donde la adaptabilidad se va a determinar bajo el nivel de generalización y la adecuada predicción estará en función del tipo de aproximación de la predicción contra los datos reales.

3.1 Diagrama a bloques de modelo computacional

En la figura 3.1 se presenta el diagrama de bloques del sistema de control predictivo, del cual forma parte el modelo computacional para la predicción del OD. Este sistema de control se explica solo de manera informativa ya que no se llevará a cabo en esta tesis, está pensado para un trabajo a futuro.

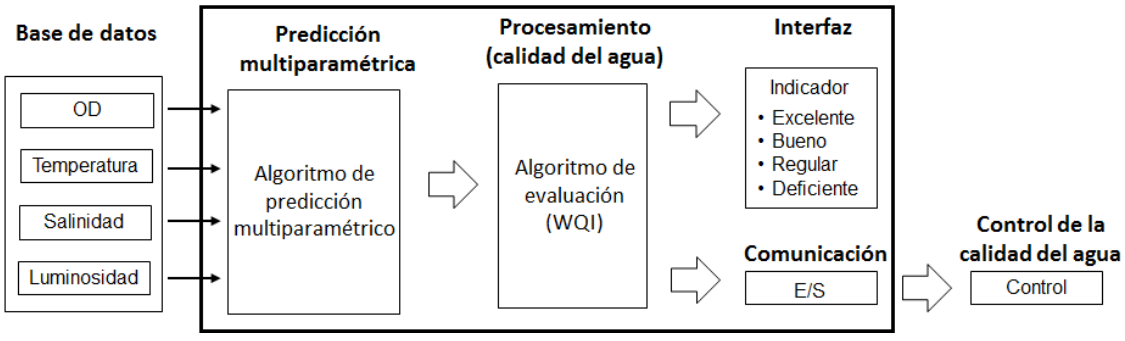

Figura 3.1.- Diagrama de bloques de sistema de control predictivo.

La sección de *base de datos* contiene los registros tomados en campo, los cuales serán la entrada del algoritmo de predicción multiparamétrica. La salida de predicción será procesada por el módulo de *evaluación de calidad del agua* el cual proporcionara un índice del deterioro global de esta. El valor de estas variables se podrá observar en una *interfaz de usuario* tendrá características como mostrar en tiempo real es el estado de los parámetros y de la calidad del agua. La salida del módulo de calidad del agua se comunicara con el *sistema de control*, que determina cuándo y por cuanto tiempo permanecerán funcionando los equipos para controlar los parámetros desestabilizadores.

Dentro del módulo de predicción paramétrica se realizará la predicción del OD, en esta parte se enfoca el modelo computacional realizado en esta tesis (predicción univariada), donde la información del pasado del oxígeno disuelto será tomada como entrada de la RNA para predecir la misma variable. Esto es mostrado en la figura 3.2.

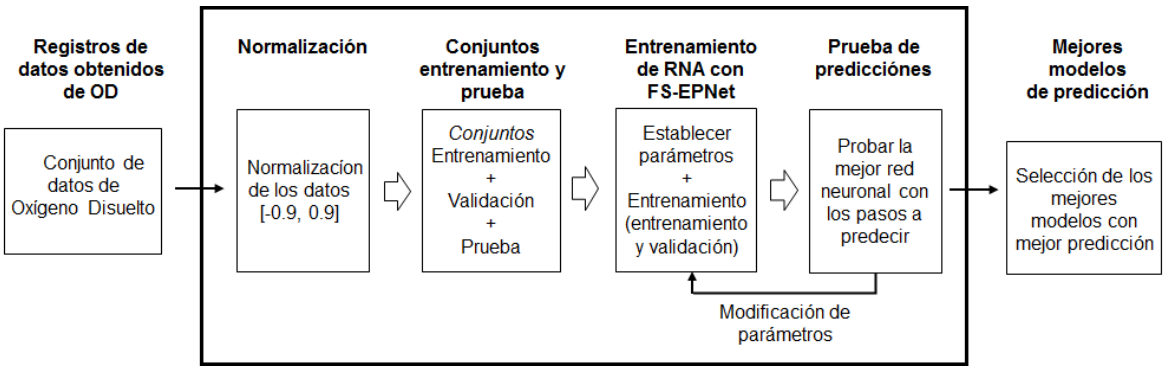

Figura 3.2.- Diagrama de bloques de modelo computacional de OD.

3.2 Recolección de muestras

Con el fin de crear una base de datos para probar el modelo, se midió el oxígeno disuelto mediante un sensor en un estanque de pruebas de la finca Rancho Chapo ubicada en Sonora, México (ver Figura 3.3). El período de seguimiento de las muestras fue de 15 minutos a lo largo de tres meses de mediciones; esto es, el conjunto de datos está formado por 8736 valores (junio, julio y agosto de 2007). Sin embargo, las fallas en el sensor generaron registros falsos, por lo que fueron borrados del conjunto de datos obtenidos, quedando 5998 registros para este trabajo.

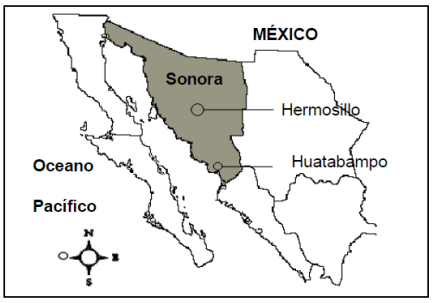

Figura 3.3. Ubicación de lugar de recolección de datos en Huatabampo, Sonora, México

El comportamiento en general del periodo de cultivo se puede apreciar en la figura 3.4 (se muestra solo el comportamiento representativo de todo el conjunto de datos tomados), donde los límites permisibles son sobrepasados, lo que genera malas condiciones de calidad del agua.

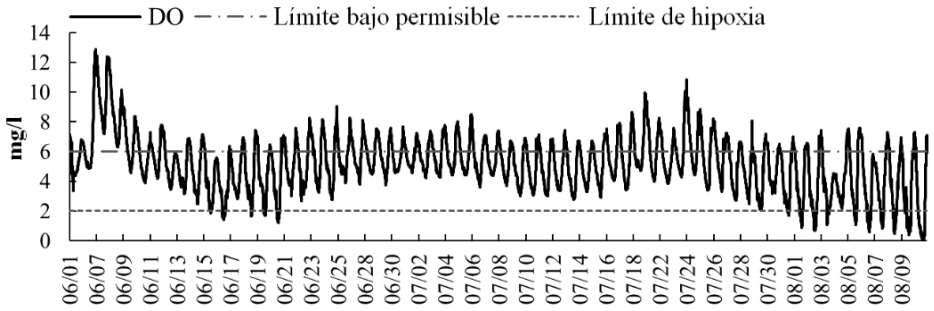

Figura 3.4. Comportamiento diurno del oxígeno durante un periodo de cultivo.

Es importante remarcar que el oxígeno disuelto presenta concentraciones de hipoxia, lo que es un problema extremadamente peligroso en cualquier tipo de sistema de cultivo acuícola.

3.3 Preprocesamiento de los datos y consideraciones preliminares

Las redes neuronales trabajan con funciones de transferencia en el rango de 0 a 1 o de -1 a 1, por lo que introducir datos en otros rangos, como los de OD de la figura 3.4, puede traer problemas cuando la información pase por dichas funciones, ya que valores más grandes tendrían un mayor impacto y pueden provocar una tendencia en el resultado final. Por ello lo ideal es normalizar los datos a un rango deseado, lo cual contribuye a que el aprendizaje sea de forma suavizada.

Existen diversas formas para normalizar datos, las cuales se determinan de acuerdo

al problema en cuestión (Senjyu et. al., 2002; Zhang et. al., 1998), en este trabajo se normaliza el conjunto de datos en su totalidad utilizando la expresión matemática de la ecuación 3.1.

$$X^{\omega} = (N\max - N\min)\,\frac{X_i^k - \min(X^k)}{\max(X^k) - \min(X^k)} + N\min \qquad (3.1)$$

la cual permite una normalización con un valor máximo de +1 y mínimo de -1, en donde $\max(X^k)\, y\ \min(X^k)$ son los valores máximo y mínimo de la serie, $N\text{máx}$ y $N\min$ son los valores máximo y mínimo del rango deseado de normalización respectivamente y $\boldsymbol{X}^{\omega}$ es el valor normalizado de ${X_i}^k$. Para esta tesis los valores fueron normalizados en un rango de [-0.9, 0.9]. El conjunto de datos ya normalizados se divide en dos subconjuntos tomados secuencialmente. Los primeros para realizar el entrenamiento y los siguientes para probar el modelo de predicción; la cantidad de ellos se determina de acuerdo a la configuración de cada experimento, los cuales se verán más adelante.

3.4 Configuraciones iniciales

La función de activación implementada es Tangente Hiperbólica para las capas intermedias y Lineal para la capa de salida (ver tabla 3.1), esto es porque de acuerdo a la literatura, estas funciones de activación responden mejor en problemas de predicción.

Tabla 3.1 Tipos de funciones de transferencia utilizadas

Nodos	Funciones de transferencia
Ocultos	Tangente hiperbólica
Salidas	Lineal

En lo que se refiere al algoritmo, fueron utilizadas las librerías que ya existen del algoritmo FS-EPNet (Victor & Bullinaria, 2011) para evolucionar las redes y encontrar la mejor para la predicción. Para ello, se realizaron pruebas con 20, 21, 30, 50, 100, 500, 1000 y 1500 pasos adelante en la predicción, y con diversos parámetros de configuración como generaciones, número de individuos en la población, entre otros. Para ello, se realizaron pruebas con 20, 21, 30, 50, 100, 500, 1000 y 1500 pasos adelante en la predicción, y con diversos parámetros de

configuración como generaciones, número de individuos en la población, entre otros.

Algunos parámetros se establecieron para que quedaran fijos en el algoritmo durante todos los experimentos. Algunos de ellos son las entradas y nodos ocultos máximos para la arquitectura de red encontrada (tabla 3.2).

Tabla 3.2 Valores máximos de entradas y nodos ocultos para la arquitectura de red.

.

Parámetro	Valor
Máximo total de entradas	50
Máximo total de nodos ocultos	100
Máximo total de salidas	1

En el algoritmo se deben especificar valores que serán aleatorios durante el proceso evolutivo; para ello, se fijan rangos en los que variaran estos parámetros (tabla 3.3). Esto permite generar una población inicial, la cual va cambiando en cada generación, lo que permite generar redes con diferentes poblaciones iniciales y así tener un espacio de búsqueda mayor.

Tabla 3.3 Configuración de parámetros inicializados aleatoriamente por la red.

.

Parámetro	Valor	
	Mínimo	Máximo
Entradas	1	3
Retardos	1	5
Nodos ocultos	1	10

El algoritmo FS-EPNet es utilizado para evolucionar las entradas, nodos y retardos de la red para optimizar la tarea de encontrar una red adecuada. Para encontrar las configuraciones de red que tengan una mejor respuesta en la predicción se deben realizar varias pruebas, en donde se establecen ciertos parámetros los cuales se cambiaran en determinadas pruebas, para ir generando diferentes resultados en la predicción, estos valores pueden ocasionar que el algoritmo converja más rápido.

Tabla 3.4 Configuración de parámetros dentro del AE.

.

Parámetro	Valor
Épocas dentro del conjunto de entrenamiento	200
Épocas fuera del EP-Net	2000
Ejecuciones repetidas del experimento	30
Número de individuos de la población	20
Generaciones	4000

Los parámetros escogidos de épocas son 200 dentro del conjunto de entrenamiento y 2000 fuera del evolutivo, este último para asegurar un entrenamiento adecuado y completo de todos los individuos de la población.

Al ser un algoritmo evolutivo, se necesita ejecutar un cierto número de ocasiones el mismo experimento para que se incremente la probabilidad de que la red escogida haya evolucionado correctamente, en la literatura se recomienda ejecutar 30 veces el experimento para que los resultados sean aceptables. Un valor adecuado para la población se establece en 20 individuos. Dicha población de individuos es sugerida por Yao y Liu (1999). Las generaciones de acuerdo a otros experimentos anteriores se fijan en 4000. Tanto los individuos y las generaciones no se incrementan ya que un aumento de ellos incrementa en significativamente el tiempo de procesamiento sin sustanciales mejoras.

3.5 Modelos de predicción utilizados

Cabe señalar, como se mencionó, que se utilizan ambos modelos de predicción SSP y MSP. Estos modelos consideraron entradas pasadas únicamente del OD para realizar la predicción del valor futuro de OD, donde el objetivo de usar ambos es poder determinar la dificultad que tienen las redes para predecir los datos. Aquí, con el modelo MSP se podrá ver si es posible extender el horizonte de predicción y con SSP simplemente se mostrará si se pueden hacer predicciones precisas a un paso adelante.

3.5.1 Configuración en el modelo SSP

El primer modelo de predicción es con SSP, tomando los primeros 3000 valores del conjunto disponible de datos para el entrenamiento. Las configuraciones realizadas en este modelo se muestran en la tabla 3.5.

Tabla 3.5 Configuración experimental con SSP.

Modelo	Patrones de entrenamiento	DT	Nombre de la configuración
SSP	3000	1	A

En este modelo solo se realiza una configuración, ya que el modelo SSP solo se utiliza para predicción a corto plazo y se utiliza como forma de validar el modelo final de predicción únicamente a un paso adelante. Parámetros como el número de neuronas, número de conexiones, capas ocultas y por ende la arquitectura de la red, son evolucionadas por el FS-EPNet. Este proceso se hace junto con el entrenamiento con el BPM, por lo que los resultados arrojados al final de la ejecución del algoritmo FS-EPNet son tanto los parámetros antes mencionados, la arquitectura de las redes así como los resultados de la predicciones del modelo.

3.5.2 Configuraciones en el modelo MSP

Para extender la predicción del OD, primero un modelo a múltiples pasos adelante (MSP) es realizado mediante varias configuraciones agrupadas por los patrones de entrenamiento (1000 y 3000 datos), modificando el *DT* en 1 y 4 (véase tabla 3.5). Se realiza de esta manera, porque las predicciones tanto en el modelo SSP como en MSP se llevan a cabo modificando el horizonte de la predicción en cada configuración (pasos adelante).

Tabla 3.5 Configuraciones experimentales con MSP

.

Modelo	Patrones de entrenamiento	DT	Nombre de la configuración
MSP	1000	1	B
		4	C
	3000	1	D
		4	E

En el capítulo 4 donde se exponen los resultados en los experimentos más representativos de las predicciones de OD. De igual forma que como se explicó en el inciso anterior, los resultados del algoritmo evolutivo y la arquitectura de red se presentan en el capítulo 4. La métrica de error usada para determinar la precisión en la predicción de las RNAs fue el NRMSE (*Normalized Root Mean Square Error).*

CAPÍTULO 4
Resultados experimentales

4.1 Conjunto de datos de prueba

Los datos de prueba para el modelo fueron tomados de forma continua a los datos de entrenamiento. La cantidad de datos de prueba se realiza de acuerdo a la configuración de la siguiente forma.

Tabla 4.1 Cantidad de datos del conjunto de prueba.

Nombre de la configuración	Conjunto de prueba
A, D y E	1500 datos
B y C	1000 datos

4.2 Pruebas de predicción con RNA en SSP (configuración A)

La figura. 4.1 muestra la mejor RNA encontrada por el algoritmo evolutivo para la configuración experimental anteriormente descrita. Esta red presenta una topología (7-7-2-1), esto es: 7 nodos de entrada, 7 en la primera capa oculta, 2 en la segunda capa oculta y 1 en la capa de salida.

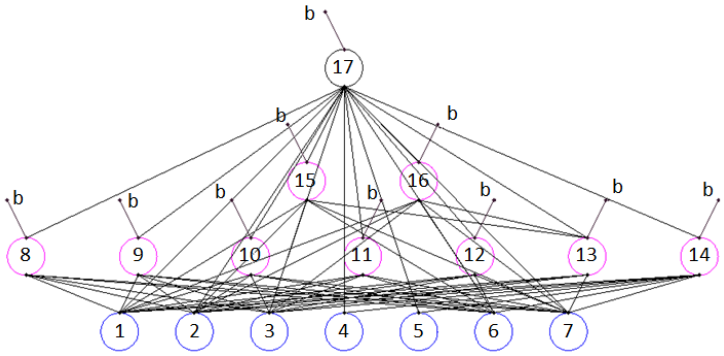

Figura 4.1.- Mejor RNA encontrada para la predicción con la configuración A y 1200 pasos adelante.

Es bueno recordar que esta arquitectura es obtenida con el algoritmo FS-EPNet usando un modelo GMLP, y las 7 entradas (nodos 1 al 7) de esta red corresponden

a 7 valores anteriores al valor a predecir del OD, es decir los nodos del 1 al 7 toman valores pasados para predecir el siguiente. En su contraparte, los nodos del 8 al 18 son ocultos repartidos en 2 capas ocultas y el nodo 17 provee la salida de la RNA. Se indica con una “b” los nodos que tienen un bias asociado, para este caso, todos los nodos lo tienen. También cabe resaltar que redes del tipo GMLP puede ser representadas en la forma redes multi-capa, como se muestra en la figura 4.1, donde simplemente son separados por capa, nodos que tengan o no una dependencia de un nodo anterior.

En la figura 4.2 se presenta la predicción con esta configuración, la cual muestra la aproximación de los valores reales en la configuración con la predicción con SSP, presentando un NRMSE = 0.12941. Asimismo, se muestra una respuesta aceptable y una alta correlación de 0.9916. Donde es bueno resaltar que la correlación no es una medida valida de error para predicciones, sin embargo se presenta aquí, junto al NRMSE, para tener otro punto de comparación, entre datos reales y predicciones.

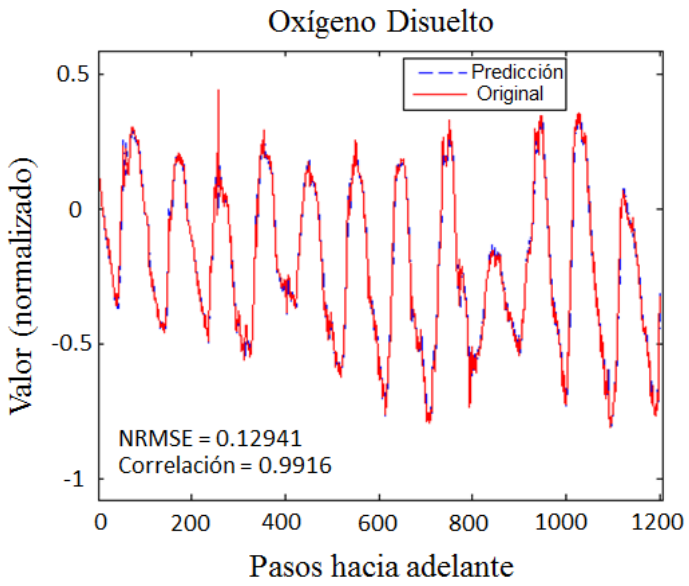

Figura 4.2.- Predicción de la mejor red encontrada con la configuración A con 1200 pasos adelante. Comparación de valores medidos de OD y predichos de la red neuronal.

En su contraparte, la figura 4.3a – 4.3c, muestra el promedio de las conexiones (4.3a), nodos ocultos (4.3b) y entradas (4.3c) durante las 4000 generaciones de evolución. A medida que aumentan las generaciones hay un incremento en el número de esas 3 variables, al final la figura 4.3c alcanza 6 nodos de entrada en promedio sobre la población entera de individuos a evolucionar; sin embargo, difiriere de las 7 entradas que presenta la mejor RNA obtenida (figura 4.1).

Observando el comportamiento de las figura 4.3a – 4.3c, se nota un incremento de todas las variables mientras que el error promedio durante la evolución disminuye y no aumenta, lo cual es una ventaja de usar un algoritmo evolutivo como el usado en esta tesis, en lugar de diseñar las arquitecturas de RNAs a mano por el experto humano, lo que se conoce como: Hand Design Neural Networks.

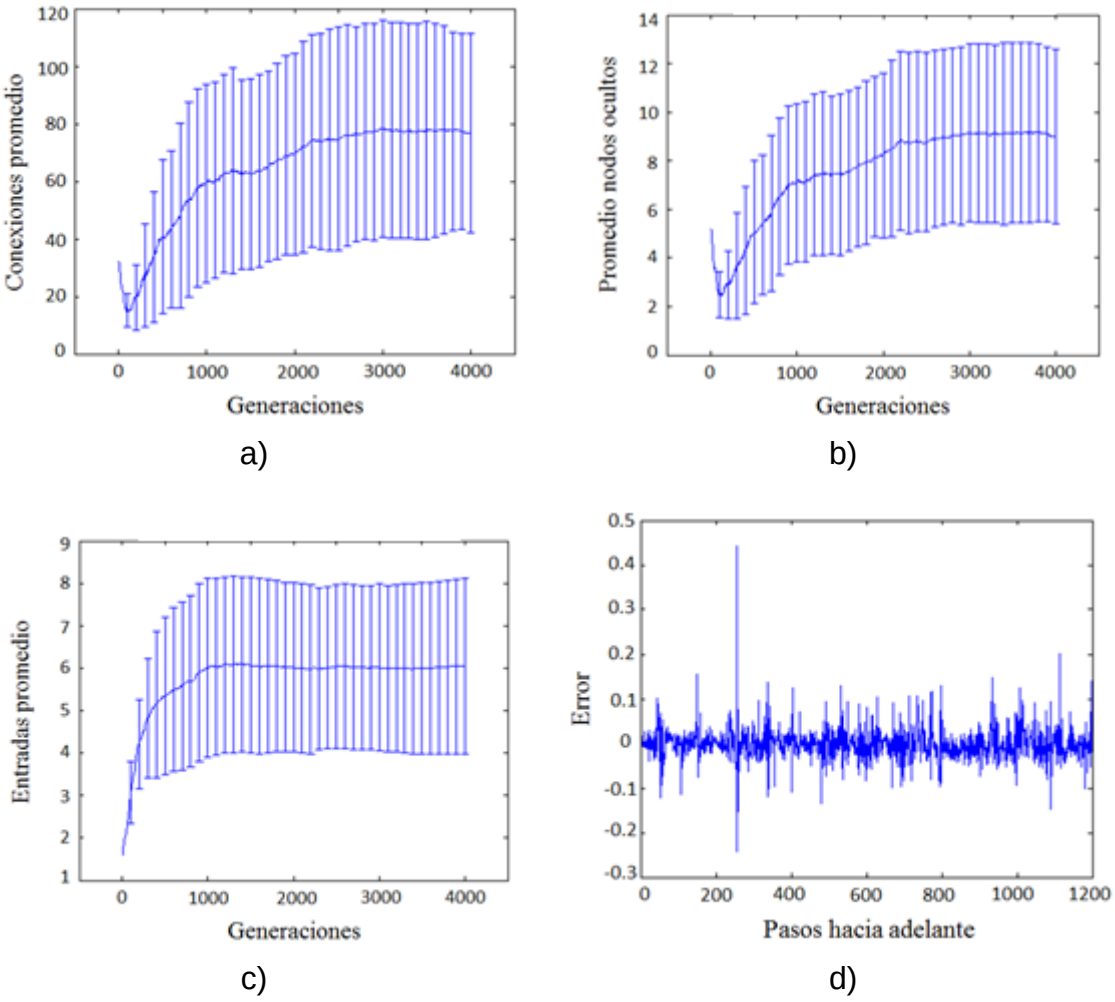

Figura 4.3.- Barras de Error (desviación estándar) durante evolución de: conexiones (a), nodos ocultos (b), entradas (c). La figura (d) muestra el error entre las mediciones reales y los predichos por la red neuronal. Configuración A y 1200 pasos adelante

En la figura 4.3 d se muestra el error entre la predicción y los datos reales. Se puede ver que a medida que van avanzando los pasos a predecir se mantiene el error cercano a cero, sin mucha variación, entre las predicciones y los valores reales, esto

es debido al modelo de predicción utilizado (SSP) utiliza los valores originales sin importar que tantos pasos adelante se hayan avanzado, por lo que el error no se acumula.

Tabla 4.2 Valores mínimos, máximos y promedio de parámetros generales con la configuración A y 1200 pasos adelante.

Parámetro	Promedio	Mínimo	Máximo
Número de entradas	6.2333	2	11
Número de retardos	5.8333	3	9
Número de nodos ocultos	9.4	2	19
Número de conexiones	83.7333	20	224
Error en el conjunto de validación	0.1222	0.1197	0.1247
Error del conjunto de prueba	0.1311	0.1294	0.1337

La tabla 4.2 muestra que en promedio para las 30 corridas del experimento en esta configuración se obtuvo un NMRSE de 0.1311, el cual es muy cercano a 0 tanto el mínimo, máximo y el promedio, esto debido a la que la predicción es solo un paso adelante. El número de entradas oscila entre 2 y 11, un poco más amplia que lo que marca la figura 4.3c, esto es porque esta tabla marca los valores para las 30 ejecuciones independientes en total. También es posible notar como el error del conjunto de prueba aumenta en comparación con el error del conjunto de validación, en donde también es posible notar que el error del conjunto de prueba no es desproporcionadamente más grande que el de validación, lo que deja notar que no hay sobre entrenamiento, la red tiene una generalización adecuada, la cual se puede apreciar al medir el error en esos dos conjuntos de datos.

4.3 Pruebas de predicción con RNA en MSP (configuración B)

La figura 4.4 muestra la mejor RNA encontrada por el algoritmo evolutivo para la configuración experimental B con 1200 pasos adelante. Esta red presenta una topología (11-3-1), esto es: 11 nodos de entrada, 3 en la capa oculta y 1 en la capa de salidas. Las 11 entradas (nodos 1 al 11) de esta red corresponden a 11 valores anteriores al valor a predecir del OD.

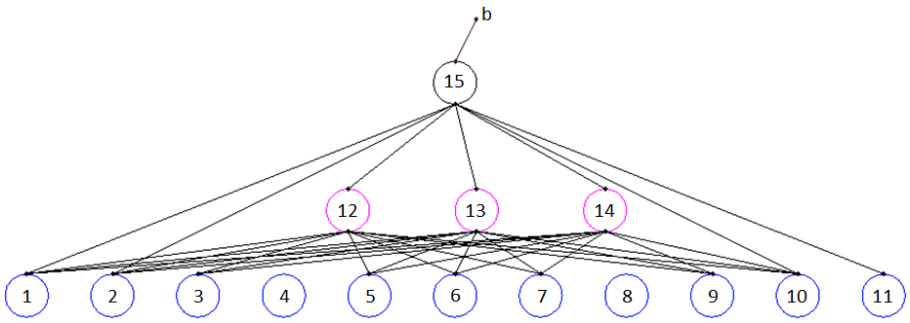

Figura 4.4.- Mejor RNA encontrada para la predicción con la configuración B. 1200 pasos adelante.

En la figura 4.5 se presenta la predicción realizada en configuración B con 1200 pasos hacia adelante, la cual muestra un buen acercamiento a los valores reales presentando un NRMSE = 0.42078. Asimismo, se muestra una respuesta aceptable y una alta correlación, sin embargo, se puede observar que la curva de predicción parece estar marcando una ligera tendencia del comportamiento de los datos originales, esto es, posiblemente por la gran cantidad de datos de entrenamiento y los pasos adelante a predecir (1200). No obstante, para cuestiones prácticas, se podría considerar una predicción valida, dado que los puntos de inflexión se pueden predecir, aunque no con una precisión mayor, es decir, el algoritmo permite indicar cuando va a haber un aumento o decremento del OD.

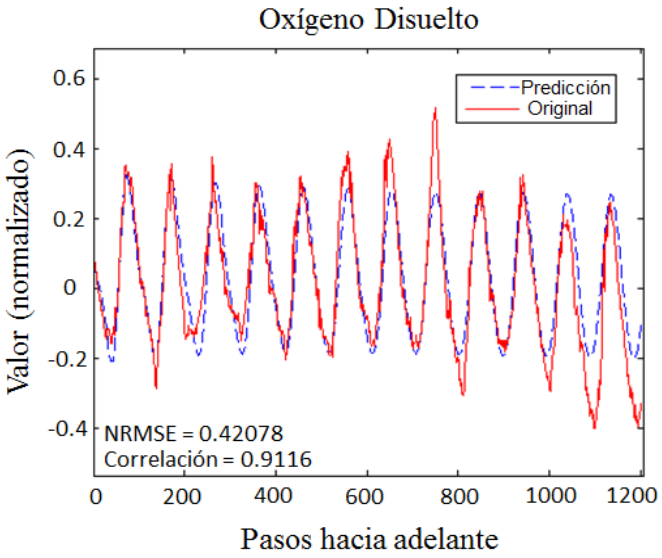

Figura 4.5.- Predicción de la mejor red encontrada con la configuración B con 1200 pasos adelante. Comparación de valores medidos de OD y predichos de la red neuronal.

El punto importante a notar aquí, es que el MSP tiene una mayor dificultad al hacer las predicciones, y esta dificultad se incremente a medida que se agranda el horizonte de predicción, así se considera un resultado adecuado dado el método usado (MSP) y los pasos a predecir (1200).

En su contraparte, la figura 4.6a – 4.6c, muestra el promedio de las conexiones (4.6a), nodos ocultos (4.6b) y entradas (4.6c) durante las 4000 generaciones de evolución.

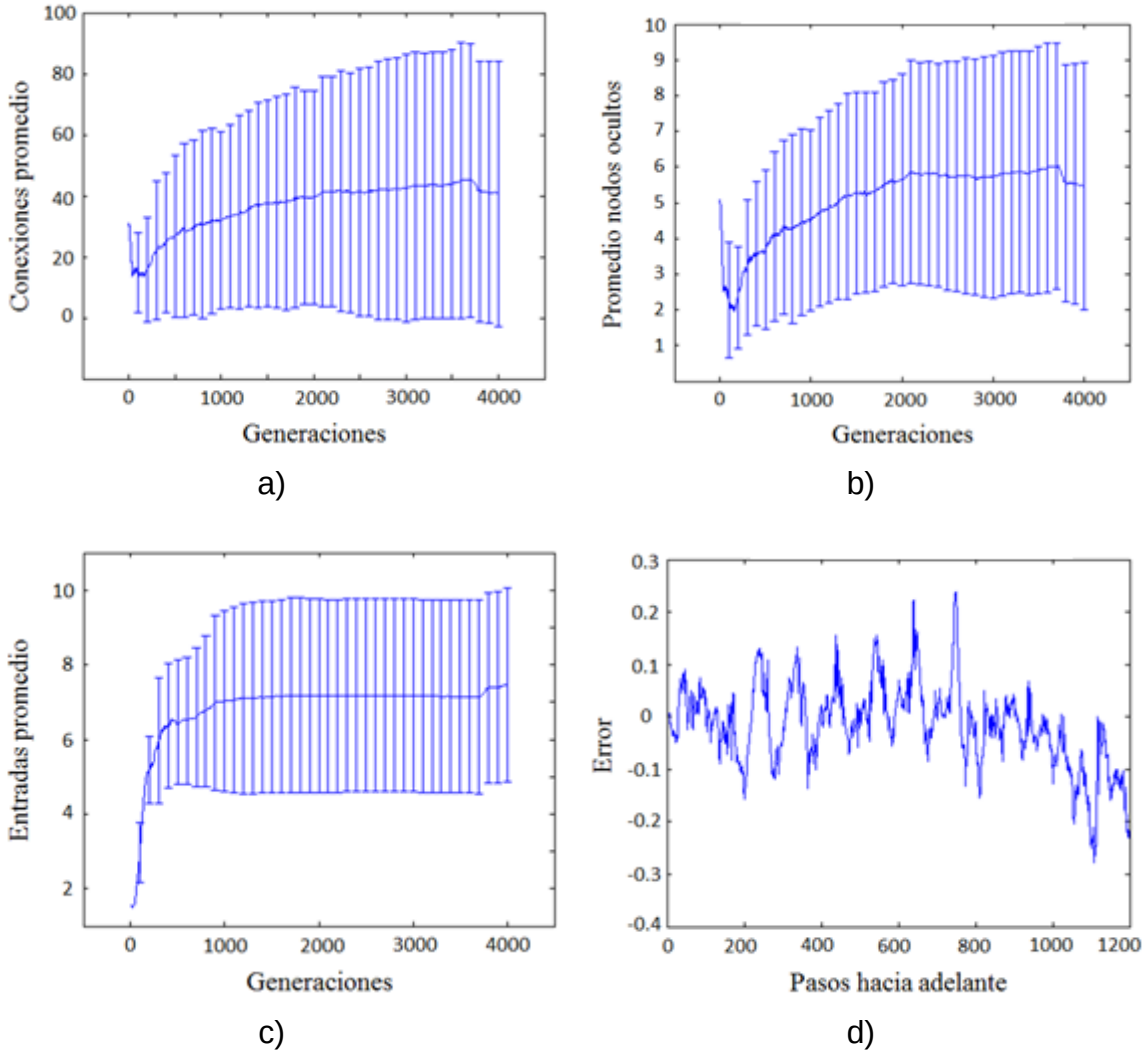

Figura 4.6.- Barras de Error (desviación estándar) durante evolución de: conexiones (a), nodos ocultos (b), entradas (c). La figura (d) muestra el error entre las mediciones reales y los predichos por la red neuronal. Configuración B y 1200 pasos adelante

A medida que aumentan las generaciones hay un incremento en el número de esas 3 variables, al final la figura 4,6c alcanza 7 nodos de entrada en promedio sobre la población entera de individuos a evolucionar; sin embargo, la mejor RNA presenta 11 neuronas de entrada como se mostró anteriormente. En la figura 4d se muestra el error de la predicción y de los datos reales. Ahí se puede ver que a medida que van avanzando los pasos a predecir se va presentando más disparidad entre las predicciones y los valores reales, en donde se puede apreciar que el error es cercano a cero al inicio de las predicciones y a medida que se incrementan el horizonte de predicción el error va aumentado, sin ser demasiado grande, esto es debido al modelo de predicción utilizado (MSP), el cual predice los siguientes valores usando valores predichos previamente (esto es, puede haber un error que se va arrastrando conforme se avanza en el horizonte de predicción).

La tabla 4.3 muestra que en promedio para las 30 corridas del experimento en esta configuración se obtuvo un NMRSE de 0.6382, claramente más alto que en SSP, algo que de igual forma sucede con el error de validación. El número de entradas oscila entre 4 y 13 y el número de conexiones también es un poco mayor que en SSP.

Tabla 4.3.- Valores mínimos, máximos y promedio de parámetros generales con la configuración B y 1200 pasos adelante.

Parámetro	Promedio	Mínimo	Máximo
Número de entradas	7,5000	4	13
Número de retardos	5,9333	3	10
Número de nodos ocultos	5,9000	1	16
Número de conexiones	46,5333	8	256
Error en el conjunto de validación	0,4516	0,3273	0,5168
Error del conjunto de prueba	0,6382	0,4208	0,9675

4.4 Pruebas de predicción con RNA en MSP (configuración C)

Después de varios experimentos el algoritmo evolutivo encuentra la mejor RNA (ver Figura 4.4) para la configuración experimental C con 1200 pasos adelante. Esta red presenta una topología (5-6-1-1), esto es: 5 nodos de entrada, 6 y 1 nodo en la primera y segunda capa oculta y 1 en la capa de salidas.

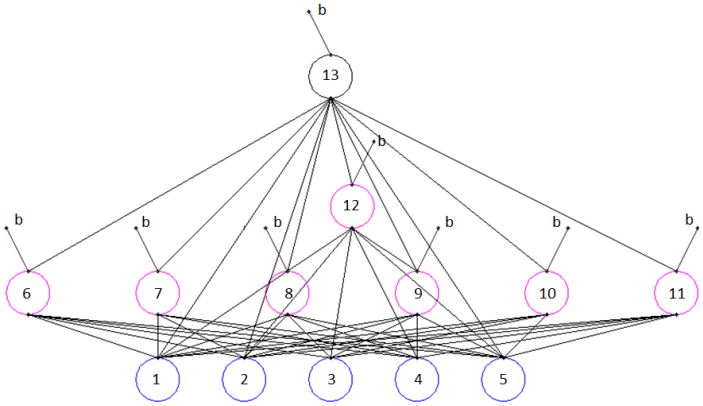

Figura 4.7.- Mejor RNA encontrada para la predicción con la configuración C. 1200 pasos adelante.

En esta configuración la predicción de OD (ver Figura 4.8) muestra un buen acercamiento a los valores reales presentando un NRMSE = 0.44327. Asimismo, se muestra una respuesta aceptable y una buena correlación, aunque al final de los 1200 valores parece incrementarse el error.

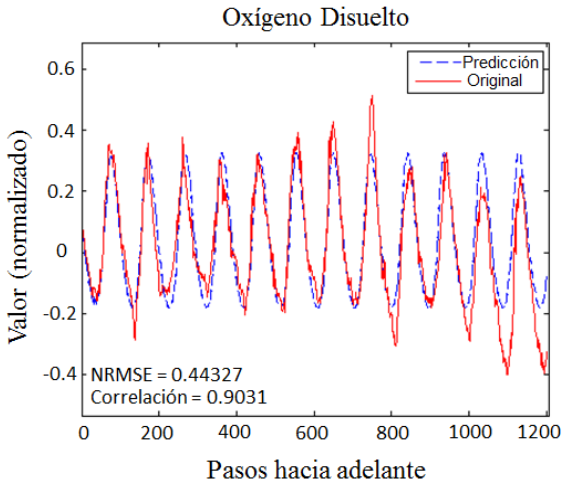

Figura 4.8.- Predicción de la mejor red encontrada con la configuración C con 1200 pasos adelante. Comparación de valores medidos de OD y predichos de la red neuronal.

Las gráficas de desviación estándar (barras de error) son mostradas en la figura 4.9a – 4.9c, la cual muestra el promedio de las conexiones (4.9a), nodos ocultos (4.9b) y entradas (4.9c) durante las 4000 generaciones de evolución. A medida que aumentan las generaciones hay un incremento en el número de esas 3 variables, al final la figura 4,9c prácticamente los mismos 5 nodos de entrada que se tienen en

la mejor red encontrada (ver Figura 4.7). En la figura 4.9d se muestra el error de la predicción y de los datos reales. Ahí se puede ver que a medida que van avanzando los pasos a predecir se va presentando más disparidad entre las predicciones y los valores reales que lo que se tiene en la figura 4.6d.

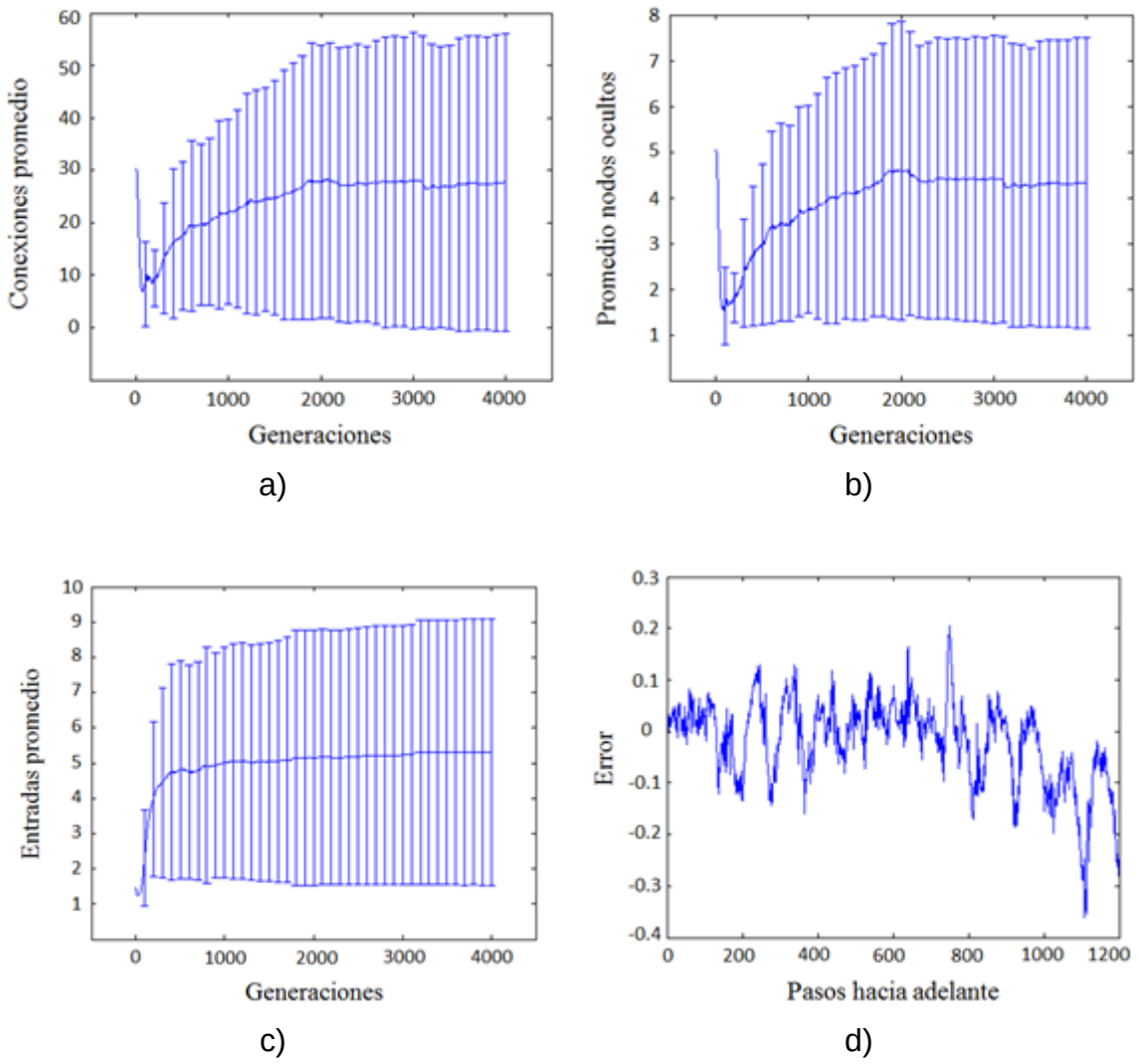

Figura 4.9.- Barras de Error (desviación estándar) durante evolución de: conexiones (a), nodos ocultos (b), entradas (c). La figura (d) muestra el error entre las mediciones reales y los predichos por la red neuronal. Configuración C y 1200 pasos adelante

La tabla 4.4 muestra que en promedio para las 30 corridas del experimento en esta configuración se obtuvieron errores más altos que para la configuración B. Se puede

apreciar lo contrario en el número de entradas nodos ocultos y conexiones el cual es menos que en la configuración B.

Tabla 4.4.- Valores mínimos, máximos y promedio de parámetros generales con la configuración C y 1200 pasos adelante.

Parámetro	Promedio	Mínimo	Máximo
Número de entradas	5,4000	1,0000	14,0000
Número de retardos	4,1000	1,0000	9,0000
Número de nodos ocultos	4,2667	1,0000	12,0000
Número de conexiones	27,1667	2,0000	126,0000
Error en el conjunto de validación	0,6886	0,2704	1,1369
Error del conjunto de prueba	0,8480	0,4433	2,6287

4.5 Pruebas de predicción con RNA en MSP (configuración D)

4.5.1 Configuración D, 1200 Pasos Adelante

Se realizaron pruebas con la configuración D y 1200 pasos adelante y la mejor red obtenida se muestra en la figura 4.10. Esta red presenta una topología (8-8-1-1-1), esto es: 8 nodos de entrada, 8 en la primera capa oculta, 1 en la segunda capa oculta, 1 en la tercera capa oculta y 1 en la capa de salida.

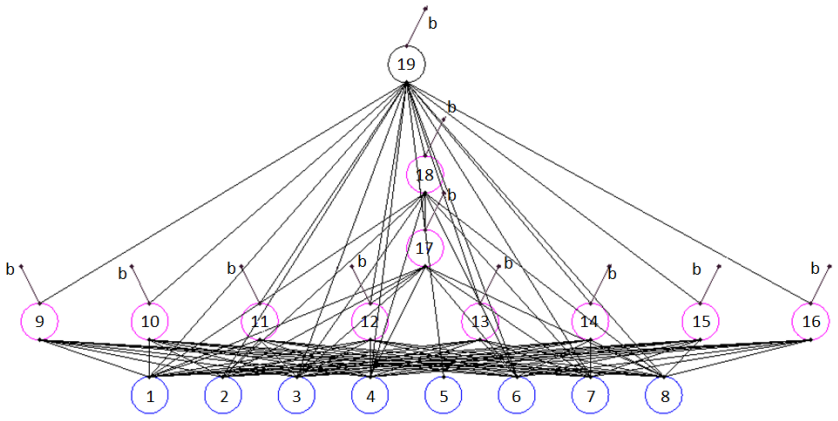

Figura 4.10.- Mejor RNA encontrada para la predicción con la configuración D. 1200 pasos adelante.

Come se mencionó anteriormente, esta arquitectura obtenida con el algoritmo FS-EPNet permite predecir el OD con 8 valores previos en los nodos de entrada. Como se aprecia, todos los nodos presentan un parámetro de bias asociado.

En la figura 4.11 se presenta la predicción realizada con 1200 pasos adelante, la cual muestra que el ajunte de la curva predicha a la real es más deficiente, presentando un NRMSE = 0.7956, esto comparado con las configuraciones C Y D, debido muy probablemente a que la cantidad de datos es mayor (4500 datos totales, 3000 de entrenamiento y 1500 de prueba en la configuración B y C). Asimismo, se muestra una correlación de 0.8395, sin embargo, la curva de predicción parece estar marcando solo una tendencia, y una disparidad mayor a la curva real.

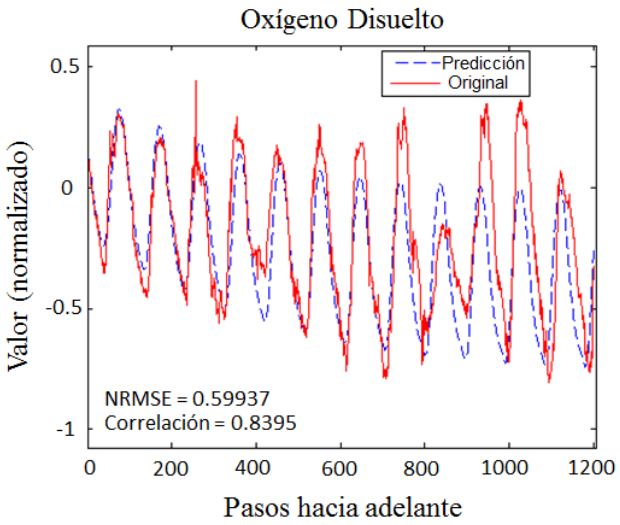

Figura 4.11.- Predicción de la mejor red encontrada con la configuración D con 1200 pasos adelante. Comparación de valores medidos de OD y predichos de la red neuronal.

En su contraparte, la figura 4.12a – 4.12c, muestra el promedio de las conexiones (4.12a), nodos ocultos (4.12b) y entradas (4.12c) durante las 4000 generaciones de evolución. A medida que aumentan las generaciones hay un incremento en el número de esas 3 variables (más pronunciado en el 4.12b – 4.12c que en la configuración C y D), al final la figura 4.12c alcanza 6 nodos de entrada en promedio sobre la población entera de individuos a evolucionar; difiriendo un poco de las 8 neuronas de entrada que presenta la mejor RNA encontrada. En la figura 4.12d se muestra el error de la predicción y de los datos reales. Ahí se puede ver que a medida que van avanzando los pasos a predecir se va presentando más disparidad entre las predicciones y los valores reales, en donde se puede apreciar que el error es cercano a cero al inicio de las predicciones y se va presentando una mayor

amplitud en las zonas de cambio del valor de la curva, comparándolo con los de la configuración C y D.

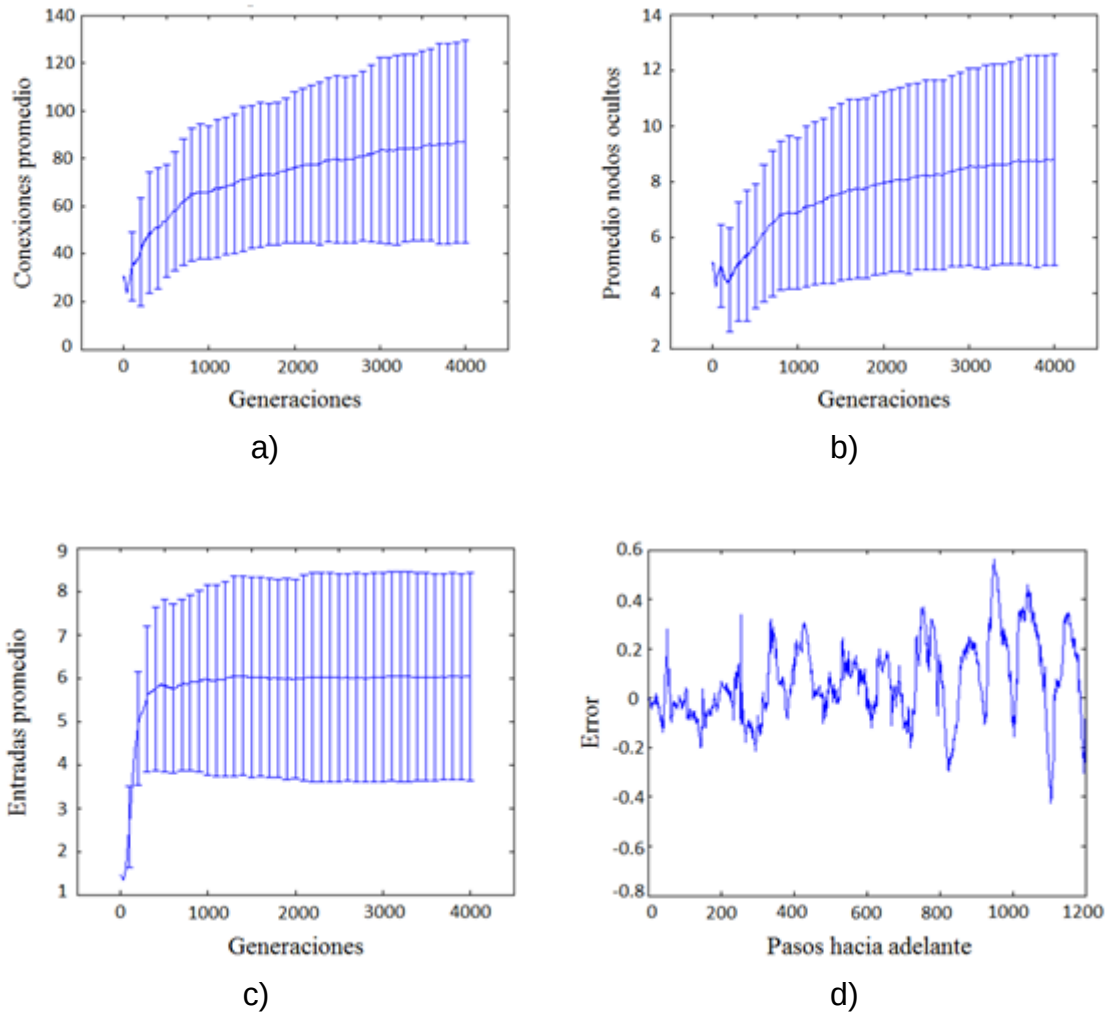

Figura 4.12.- Barras de Error (desviación estándar) durante evolución de: conexiones (a), nodos ocultos (b), entradas (c). La figura (d) muestra el error entre las mediciones reales y los predichos por la red neuronal. Configuración D y 1200 pasos adelante

La tabla 4.5 muestra que en promedio para las 30 corridas del experimento en esta configuración se obtuvieron errores más altos que para la configuración B y C. Se puede apreciar lo mismo en el número de entradas nodos ocultos y conexiones el cual es menos que en la configuración B y C, lo que se apreció también en la arquitectura de red de la figura 4.10.

Tabla 4.5.- Valores mínimos, máximos y promedio de parámetros generales con la configuración D y 1200 pasos adelante.

Parámetro	Promedio	Mínimo	Máximo
Número de entradas	5.9	1	12
Número de retardos	4.6333	1	11
Número de nodos ocultos	8.8667	3	21
Número de conexiones	88.3667	21	255
Error en el conjunto de validación	0.5183	0.4501	1.0173
Error del conjunto de prueba	0.8671	0.5994	1.1096

Para eliminar la aparente tendencia en la predicción con 1200 pasos delante de las configuraciones B, C y D se realizaron pruebas con menos puntos a predecir.

4.5.2 Configuración D, 21 Pasos Adelante

La mejor respuesta con predicciones menores a 1200 pasos adelante se obtiene con 21 pasos sobre en dicho horizonte, la arquitectura con la cual se realiza el entrenamiento se presenta en la figura 4.13. Esta red presenta una topología (11-4-5-1), esto es: 11 nodos de entrada, 4 en la primera capa oculta, 5 en la segunda capa oculta y 1 en la capa de salida. Los 11 nodos de entrada recibirán los 11 valores de DO para predecir el siguiente en el nodo 21.

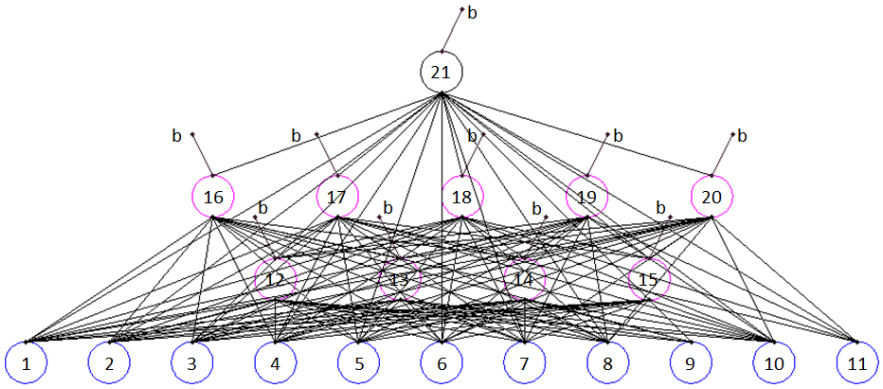

Figura 4.13.- Mejor RNA encontrada para la predicción con la configuración D y 21 pasos adelante.

La gráfica de predicción se muestra en la figura 4.14, en donde la curva de datos original corresponde a los últimos 21 puntos (1179-1200) de la gráfica de datos original de la figura 4.10. Nótese que la escala es diferente a las figuras anteriores, y el error obtenido es alto considerando solo 21 puntos a predecir. Si lo comparamos con las predicciones de los experimentos anteriores, en algunos lados se tenían errores de predicción altos, pero en otros eran muy precisos, lo que hacia disminuir el error total, el cual hay que recordar que está normalizado.

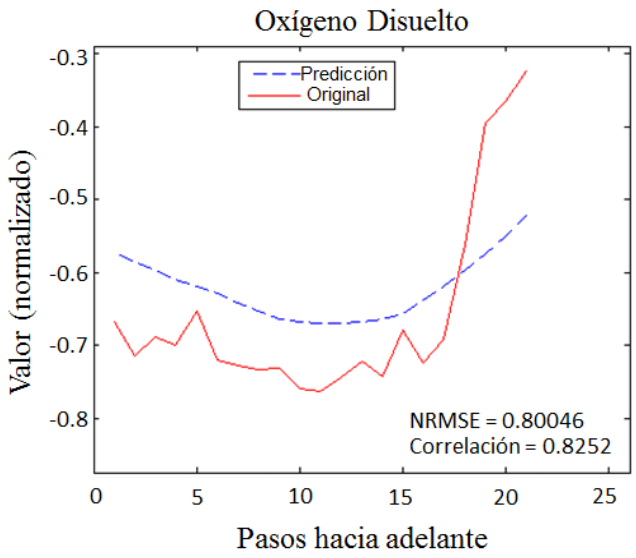

Figura 4.14.- Predicción de la mejor red encontrada con la configuración D con 21 pasos adelante. Comparación de valores medidos de OD y predichos de la red neuronal.

Las gráficas de desviación estándar se presentan en la figura. 4.15a – 4.15c, muestra el promedio de las conexiones (4.15a), nodos ocultos (4.15b) y entradas (4.15c) durante las 4000 generaciones de evolución. A medida que aumentan las generaciones hay un incremento en el número de esas 3 variables. Se aprecia que la figura 4.15c alcanza 5 nodos de entrada en promedio sobre la población entera de individuos a evolucionar, contrastando con las 11 de entrada de la mejor red encontrada como en la figura 4.13. En la figura 4.15d se muestra el error de la predicción y de los datos reales. Puede verse que al final, entre el paso 16 y 21 el error se incrementa notoriamente.

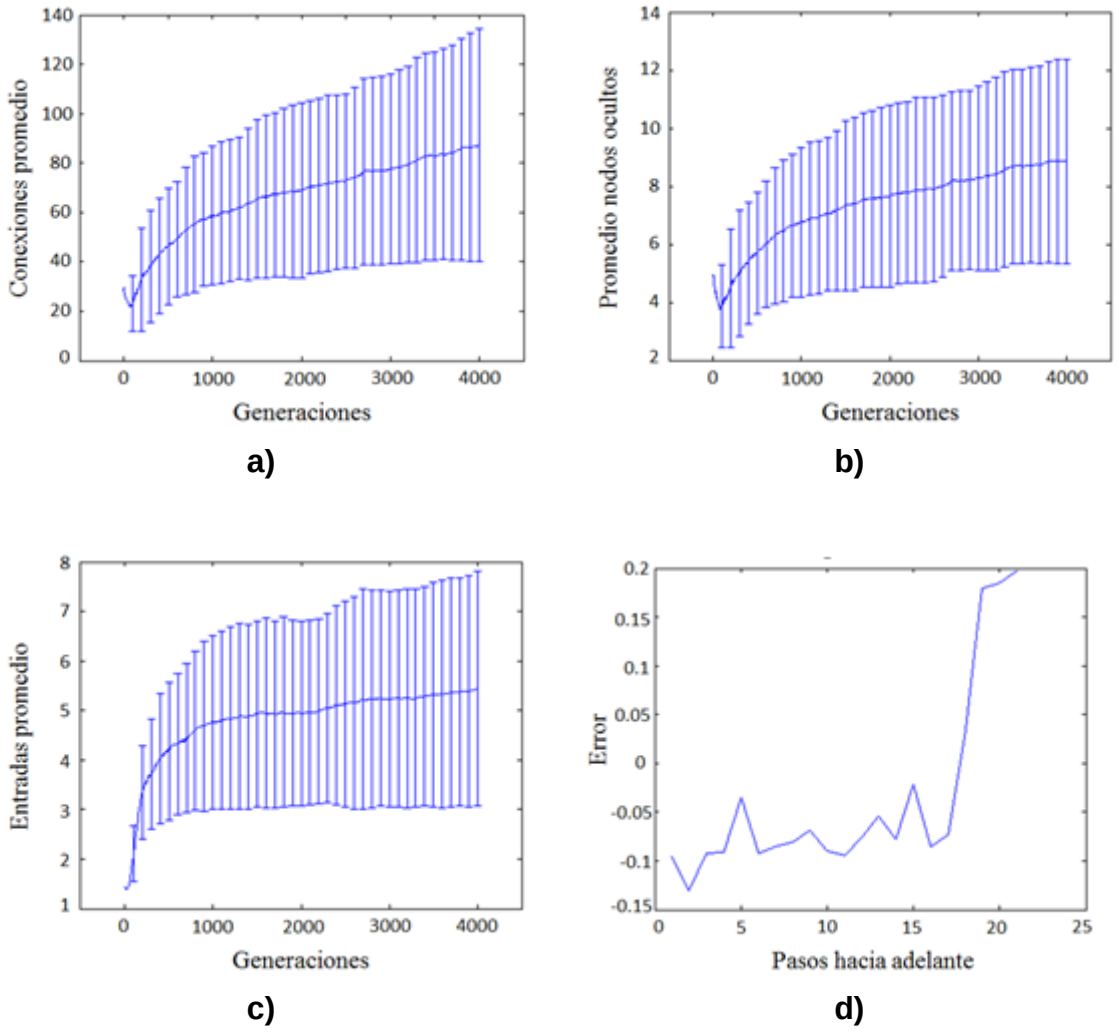

Figura 4.15.- Barras de Error (desviación estándar) durante evolución de: conexiones (a), nodos ocultos (b), entradas (c). La figura (d) muestra el error entre las mediciones reales y los predichos por la red neuronal. Configuración D y 21 pasos adelante

La tabla 4.6 muestra que en promedio para las 30 corridas del experimento en esta configuración se obtuvieron errores más altos que para 1200 pasos adelante. Se puede apreciar lo mismo en el número de entradas nodos ocultos y conexiones el cual es menos que en la configuración D a 1200 pasos adelante, lo que se apreció también en la arquitectura de red de la figura 4.13.

Tabla 4.6 Valores mínimos, máximos y promedio de parámetros generales con la configuración D y 21 pasos adelante.

Parámetro	Promedio	Mínimo	Máximo
Número de entradas	6	1	12
Número de retardos	3.4	1	8
Número de nodos ocultos	9.1	1	18
Número de conexiones	94.2	4	276
Error en el conjunto de validación	0.3679	0.2517	1.2991
Error del conjunto de prueba	1.1089	0.8005	1.2237

4.6 Pruebas de predicción con RNA en MSP (configuración E)

4.6.1 Configuración E, 1200 Pasos Adelante

Después de varios experimentos el algoritmo evolutivo encuentra la mejor RNA (ver Figura 4.16) para la configuración experimental E con 1200 pasos adelante. Esta red presenta una topología (6-6-2-3-1), esto es: 6 nodos de entrada, 6 nodos en la primera, 2 nodos en la segunda capa oculta, 3 nodos en la tercera capa oculta y 1 nodo en la capa de salida. Todos los nodos presentan un bias asociado a ellos.

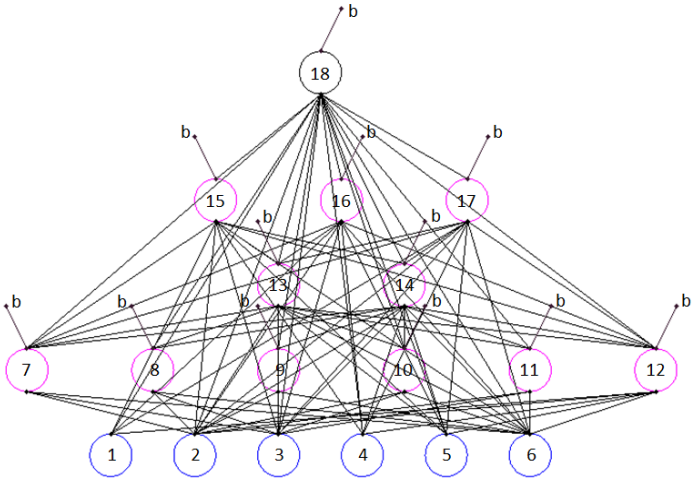

Figura 4.16.- Mejor RNA encontrada para la predicción con la configuración E y 1200 pasos adelante.

En la figura 4.17 se presenta la predicción realizada con 1200 pasos adelante, la cual muestra un ajuste de la curva predicha a la real parecido al de la configuración D con 1200 pasos adelante, aunque con un NRMSE = 0.81982, el cual es menor, al de la configuración D. Asimismo, presenta una correlación de 0.8074, también menor que en la configuración D, sin embargo, la curva de predicción también parece estar marcando solo una tendencia, y una disparidad mayor a la curva real, esto es.

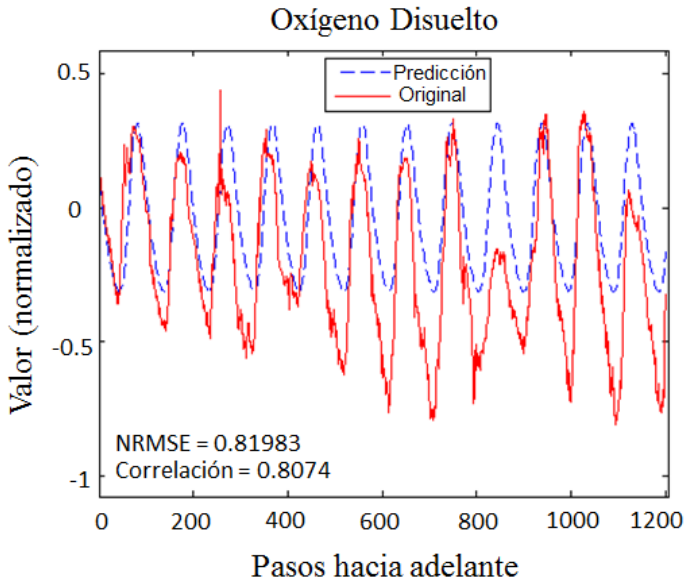

Figura 4.17.- Predicción de la mejor red encontrada con la configuración E con 1200 pasos adelante. Comparación de valores medidos de OD y predichos de la red neuronal.

En la figura 4.18 – 4.18c, muestra el promedio de las conexiones (4.18a), nodos ocultos (4.18b) y entradas (4.18c) durante las 4000 generaciones de evolución. A medida que aumentan las generaciones hay un incremento de las conexiones y los nodos ocultos, no así en las entradas la cual permanece casi desde el principio constante has el final en 3 nodos de entrada en promedio sobre la población entera de individuos a evolucionar, contrastando con la mejor RNA la cual presenta 6 neuronas de entrada como se mostró anteriormente. En la figura 4.18d se muestra el error de la predicción y de los datos reales.

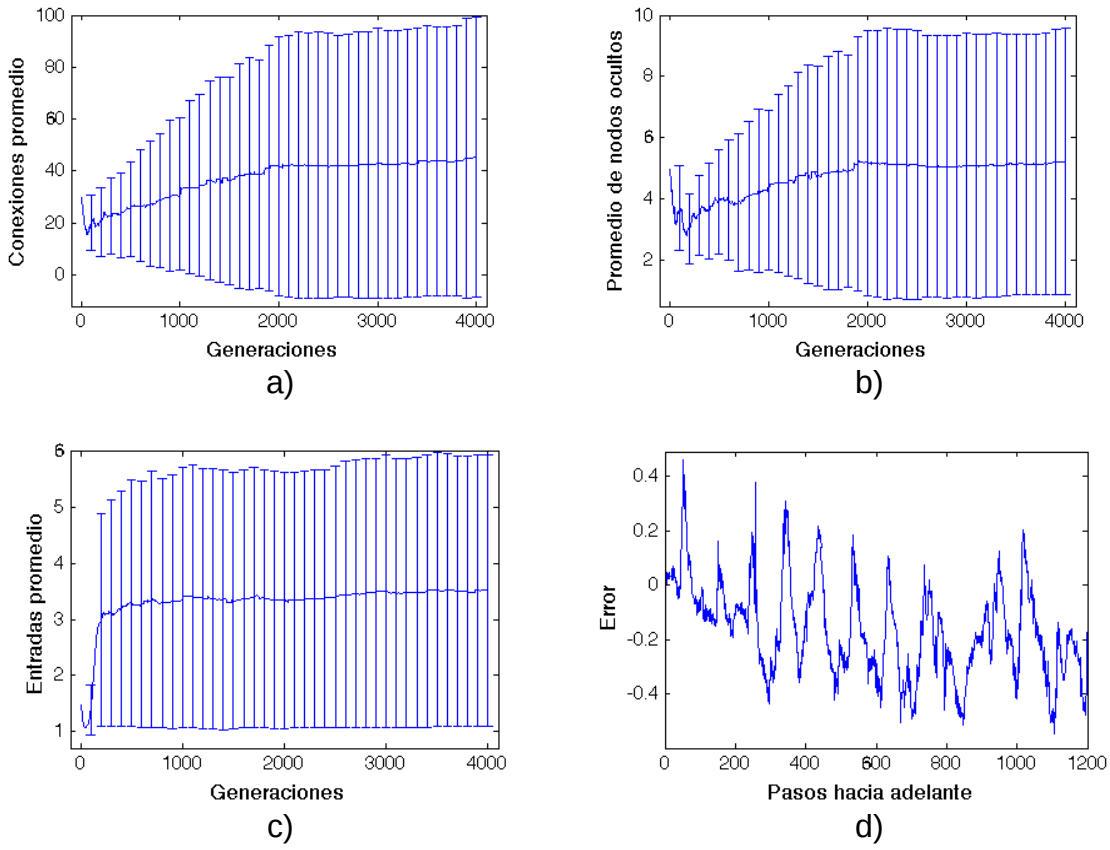

Figura 4.18.- Barras de Error (desviación estándar) durante evolución de: conexiones (a), nodos ocultos (b), entradas (c). La figura (d) muestra el error entre las mediciones reales y los predichos por la red neuronal. Configuración E y 1200 pasos adelante

Tabla 4.7.- Valores mínimos, máximos y promedio de parámetros generales con la configuración E y 1200 pasos adelante.

Parámetro	Promedio	Mínimo	Máximo
Número de entradas	3.3667	1	6
Número de retardos	3.1333	1	4
Número de nodos ocultos	5.8667	1	15
Número de conexiones	50.4667	2	173
Error en el conjunto de validación	0.7975	0.5818	1.0257
Error del conjunto de prueba	0.9517	0.8198	1.2373

La tabla 4.7 muestra que en promedio para las 30 corridas del experimento en esta configuración se obtuvieron errores un poco más altos que para la configuración D con 1200 pasos adelante.

4.6.2 Configuración E, 21 Pasos Adelante

De igual forma a como se hizo en la configuración D, se muestra la arquitectura de la mejor RNA con la configuración E con 21 pasos sobre en dicho horizonte (ver Figura 4.19).

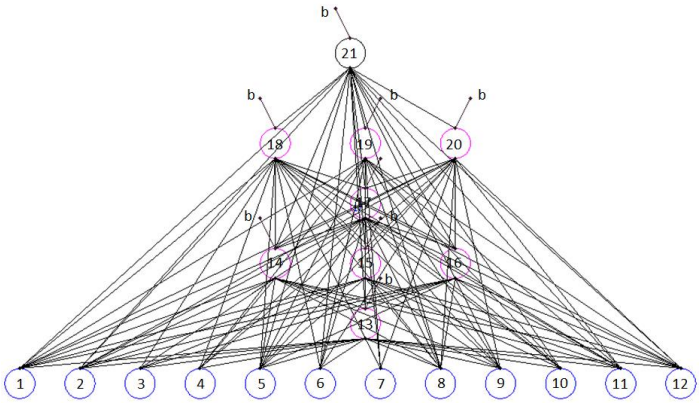

Figura 4.19.- Mejor RNA encontrada para la predicción con la configuración E y 21 pasos adelante.

Esta red presenta una topología (12-1-3-1-3-1), esto es: 12 nodos de entrada, 1 en la primera capa oculta, 3 en la segunda capa oculta, 1 en la tercera capa oculta, 3 en la cuarta capa oculta y 1 en la capa de salida. Los 12 nodos de entrada recibiran

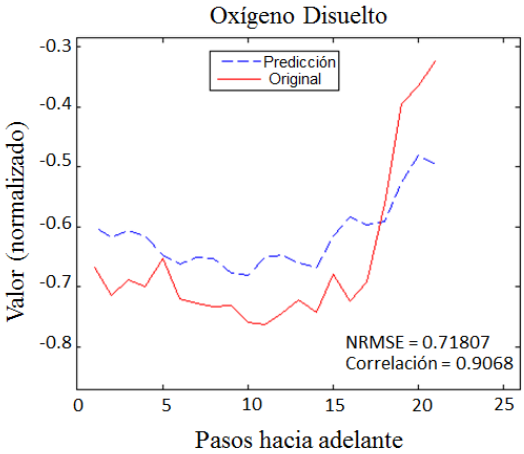

Figura 4.20.- Predicción de la mejor red encontrada con la configuración E con 21 pasos adelante. Comparación de valores medidos de OD y predichos de la red neuronal.

los 12 valores de DO para predecir el siguiente valor en el nodo 21. Todos los nodos tienen un bias asociado. La precisión en predicción se muestra en la figura 4.20, en donde la curva de datos original corresponde a los últimos 21 puntos (1179-1200) de la gráfica de datos original de la figura 4.17. Este experimento es el que presenta una mayor correlación de los presentados con un DT =4.

Las gráficas de desviación estándar se presentan en la figura 4.21a – 4.21c, muestra el promedio de las conexiones (4.21a), nodos ocultos (4.15b) y entradas (4.21c) durante las 4000 generaciones de evolución.A medida que aumentan las generaciones hay un incremento en el número de esas 3 variables. Se aprecia que la figura 4.21c alcanza 8 nodos de entrada en promedio sobre la población entera de individuos a evolucionar, contrastando con las 12 de entrada de la mejor red

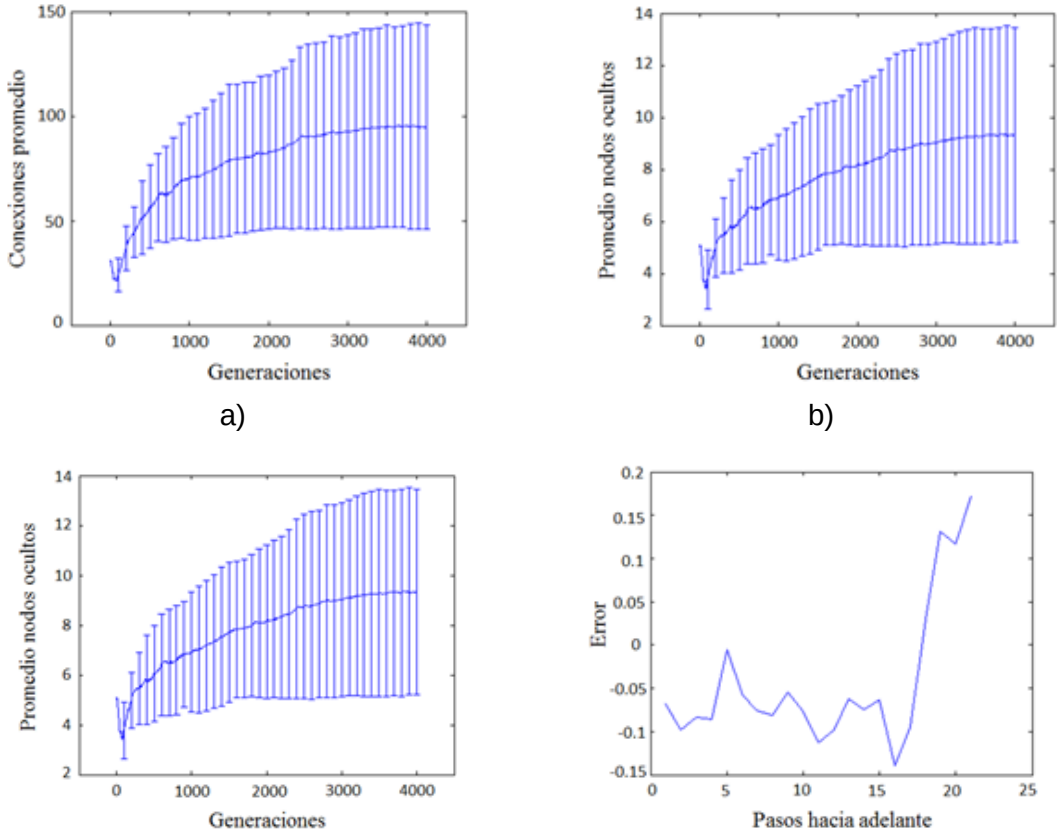

Figura 4.21.- Barras de Error (desviación estándar) durante evolución de: conexiones (a), nodos ocultos (b), entradas (c). La figura (d) muestra el error entre las mediciones reales y los predichos por la red neuronal. Configuración E y 21 pasos adelante.

encontrada como en la figura 4.13. En la figura 4.21d se muestra el error de la predicción y de los datos reales. Puede verse que al final, entre el paso 16 y 21 el error se incrementa notoriamente.

La tabla 4.8 muestra que en promedio para las 30 corridas del experimento en esta configuración se obtuvieron errores más altos que para 1200 pasos adelante y para las demás configuraciones con DT = 4. Se puede apreciar lo mismo en el número de entradas, nodos ocultos y conexiones de todas las configuraciones B, C, D Y E.

Tabla 4.8.- Valores mínimos, máximos y promedio de parámetros generales con la configuración E y 21 pasos adelante.

Parámetro	Promedio	Mínimo	Máximo
Número de entradas	7.6333	2	14
Número de retardos	6.6333	1	14
Número de nodos ocultos	9.8667	3	25
Número de conexiones	104.2333	30	304
Error en el conjunto de validación	0.2996	0.2013	0.4053
Error del conjunto de prueba	0.9228	0.7181	1.2096

4.7 Red Neuronal hecha manualmente sin FS-EPNet

Para verificar los beneficios de implementar el FS-EPNet en el diseño de la RNA, se realizaron experimentos en donde la arquitectura de la red se realizaba con distintos parámetros establecidos manualmente.

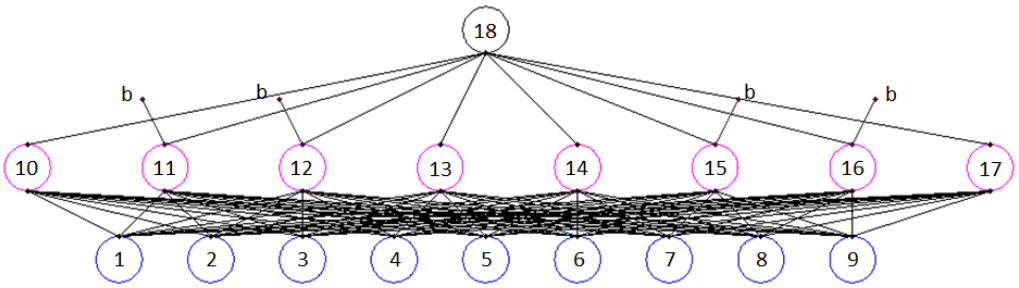

Figura 4.22.- RNA obtenida manualmente para la predicción con la configuración E. 1200 pasos adelante.

La red que presenta los mejores resultados se muestra en la figura 4.22. Se realiza con los parámetros de la configuración E a 1200 pasos adelante. Esta red presenta una topología (9-8-1), esto es: 9 nodos de entrada, 8 en la primera capa oculta y 1 en la capa de salida. Los 9 nodos de entrada recibirán los 9 valores de DO para predecir el siguiente valor en el nodo 18. No todos los nodos tienen un bias asociado.

En la figura 4.23 se presenta la predicción realizad, la cual muestra un ajuste de la curva predicha a la real más deficiente que en los experimentes utilizando en algoritmo FS-EPNET, con un NRMSE = 1.1944. El índice de correlación de forma contraria, es el menor que con el algoritmo FS-EPNET con una correlación de 0.0662. Aproximadamente a partir de los 50 pasos adelante, se comienza a marcar una tendencia cíclica de la predicción.

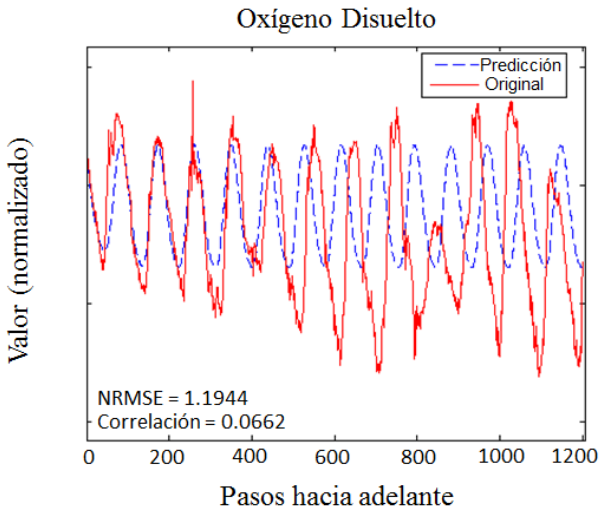

Figura 4.23.- Predicción con RNA hecha manuelmente. Para MSP y 1200 pasos adelante.

En la figura 4.24 se muestra el error entre la predicción y los datos reales. Se puede ver que a medida que van avanzando los pasos se va incrementando el error. Sin embargo, se puede ver algo interesante, hasta los 50 pasos el error comienza a variar pero se mantiene razonable hasta los 400 pasos, en donde su fluctuación es muy mercado y esto se puede ver en la diferencia en las predicciones de la figura 4.23.

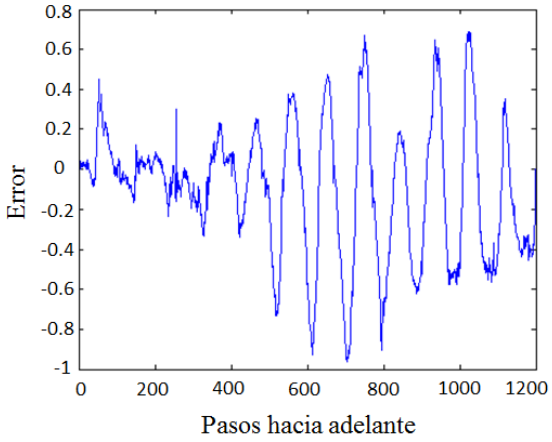

Figura 4.24.- Error entre las mediciones reales y los predichos por la red neuronal hecha manualmente

Como se puede apreciar en la tabla 4.9 muestra el número de conexiones es muy elevado para la cantidad de nodos que hay. Los errores tanto de entrenamiento como de validación y prueba son mayores que en las demás configuraciones donde se utiliza el algoritmo FS-EPNet para diseñar la red.

Tabla 4.9.- Parámetros generales de la RNA configurada a mano.

Parámetro	Valor
Número de entradas	9
Número de retardos	4
Número de nodos ocultos	8
Conexiones	84
Error en el conjunto de validación	0.789749
Error del conjunto de prueba	1.194365

4.8 Discusión de resultados

En esta tesis se demostró que es posible predecir la evolución de redes neuronales mediante el algoritmo FS-EPNet, lo cual permite una buena predicción del oxígeno disuelto en el cultivo acuícola.

Los resultados obtenidos demuestran que la buenos acercamientos en las predicciones utilizando un modelo SSP; sin embargo, en un entorno real y en el ámbito acuícola este no se utilizaría ya que lo importante es la predicción a un lapso de tiempo por lo que el modelo MSP puede ser mas implementado en el ambiente acuicola. Es muy importante para la predicción los parámetros establecidos en el algoritmo. En general, predicciones con 2000 datos, entre entrenamiento y prueba, proporcionan mejores resultados. Se puede pudo observar que no hay una significativa diferencia entre las predicciones con DT=1 y DT=4 dando muy semejantes.

El uso del algoritmo FS-EPNet es útil en problemas de optimización de la arquitectura de red, ya que aunque el espacio de soluciones es muy grande, se pueden encontrar arquitecturas adecuadas para realizar predicciones; en este caso se lograron hacer predicciones hasta con 1500 pasos adelante con resultados ciertamente aceptables. Esto puede verse también en las pruebas realizadas con el diseño de la red manualmente en la sección 4.7, las cuales ofrecen resultados en la predicción más deficientes en comparación con las redes evolucionadas.

En la literatura hay varios artículos que tratan la predicción de series de tiempo, sin embargo, ningún trabajo encontrado en la literatura había realizado predicción con estos datos obtenidos. En trabajos parecidos como en Areerachakul et al., 2013, se utiliza una red neuronales para estimar en un tiempo presente el nivel de OD mediante una serie de parámetros de entrada, el problema es que no realizan una predicción como y el diseño de la red neuronal es realizada mediante parámetros que fija el diseñador; esto es algo que se evita con el uso de el algoritmo FS-EPNet. El hacerlo manualmente tiene el inconveniente que tienen que realizar varias pruebas para presentar resultados en las estimaciones más o menos aceptables.

La similitud los resultados de esta tesis se presentan en el trabajo de Miao et. al., 2010, ya que en él se utilizan redes neuronales entrenadas mediante algoritmos genéticos. El problema es que el diseño de la red lo realizan manualmente presentando complicaciones en la predicción y el tiempo de diseño. Estos inconvenientes son los problemas que principalmente se solucionan al hacer uso del FS-EPNet.

Los resultados de predicción mediante el modelo propuesto en esta tesis permiten incluso realizar investigaciones sobre otros parámetros y trabajar sobre la calidad del agua con mayor exactitud en las predicciones y disminución en el tiempo de diseño.

CAPÍTULO 5
Conclusiones

El estudio de la calidad del agua cobra gran relevancia para los cultivos acuícolas y por ello ha sido estudiada ampliamente.

La importancia de la concentración de OD no puede ser subestimada, ya que es probablemente uno de los parámetros más importantes que influye en el bienestar del organismo acuático. Donde es un problema su control en el día, pero lo es aún más por la noche, y es ahí donde los procedimientos de manejo preventivos (predicción de él con métodos de la inteligencia artificial) pueden ayudar a prevenir esta situación.

El modelo con RNA propuesto en este trabajo permite realizar una predicción adecuada como se mostró en el capítulo 4 de resultados, donde se puede concluir que dichas predicciones pueden ser utilices al momento de tomar decisiones en la auicultura. En otro aspecto, al usar el algoritmo evolutivo FS-EPNet permite automatizar la creación de ellas (arquitecturas), reduciendo el trabajo y tiempo para encontrar la estructura de la red neuronal adecuada. Así las RNAs evolucionadas permiten abordar de una mejor manera el problema de no linealidad del OD y así anticipar con mayor precisión el estado de la calidad del agua del cultivo, dado que el proceso evolutivo permite ajustar de una mejor forma valores que serían difíciles de ajustar por un experto (Hand Design Neural Networks). Por lo anterior se concluye que es conveniente el uso del FS-EPNet al diseñar RNAs como alternativa para controlar el estado de la calidad del agua.

Así, el uso del algoritmo FS-EPNet en el diseño de las RNAs permite obtener una arquitectura optimizada con un rendimiento aceptable, y dado el proceso evolutivo y el entrenamiento parcial con el BPM se evita caer en mínimos locales, todo esto sin la intervención de un experto. No es posible hacer una comparación directa con los trabajos citados anteriormente dado el origen diverso de los datos, sin embargo se espera que en trabajos futuros se puedan comparar.

5.1. Trabajo a futuro

A partir de lo concluido en esta tesis se piensa realizar en futuros trabajos investigaciones las cuales se compongan de: Realizar predicción multivariada de varios parámetros de la calidad del agua. Asimismo, se plantea explorar otras técnicas de predicción para los parámetros de la calidad del agua. Con la predicción de estos parámetros se hará posible realizar una evaluación de la calidad del agua proporcionando un índice del deterioro de esta relacionado a la estimación y predicción de dichas variables.

Dichas aproximaciones, permitirán que el experto pueda tomar decisiones como medidas preventivas y predictivas de la calidad del agua, como iniciar el proceso de aireación para adecuar los nivele de OD, así como también determinar si algún otro parámetro está afectando o afectará posteriormente la concentración presente de OD o si de igual forma está en un rango no deseado, adecuarlos realizando un control y de esta forma poder mantener en un nivel estable la calidad del agua en el cultivo acuícola..

Para ello, se planea diseñar una tarjeta de adquisición de datos medioambientales. Esto para sentar las bases de estos trabajos futuros al tener un prototipo funcional que permita controlar y monitorear los parámetros de mayor impacto involucrados en el cultivo acuícola permitirá su futuro desarrollo.

Referencias

Abdallah, C. T., Docampo, D., & Hush, D. R. (2012). Constructive function approximation: theory and practice. *Electrical & Computer Engineering Faculty Publications*, 1-24.

Anufriev, M., Hommes, C., & Makarewicz, T. (2012). *Learning-To-Forecast with Genetic Algorithms.* Center for Nonlinear Dynamics in Economics and Finance at University of Amsterdam, Amsterdam.

Areerachakul, S., Junsawang, P., & Pomsathit, A. (2011). Prediction of Dissolved Oxygen Using Artificial Neural Network. *Proc.of CSIT, 5*, 524-528.

Armstrong, J. S., & Collopy, F. (1992). Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting, 8*(1), 69-80.

Battiti, R. (1994). Using mutual information for selecting features in supervised neural net learning. *IEEE Trans. on Neural Networks, 5*(4), 537-550.

Bhatnagar, A., & Devi, P. (2013). Water quality guidelines for the management of pond fish culture. *Int. J. Environ, 3*(6), 1980-2009.

Bhatnagar, A., & Devi, P. (Junio de 2013). Water quality guidelines for the management of pond fish culture. *International journal of environmental sciences, 3*(6), 1980-2009.

Bontempi, G. (Febrero de 2008). Long term time series prediction with multi-input multi-output local learning. *In Proceedings of the 2nd European Symposium on Time Series Prediction (TSP)*, 145-154.

Bontempi, G., & Taieb, S. B. (Septiembre de 2011). Conditionally dependent strategies for multiple-step-ahead prediction in local learning. *International Journal of Forecasting, 27*(3), 689-699.

Bontempi,, G., Birattari,, M., & Bersini., H. (1999). Local learning for iterated time-series prediction. *Machine Learning: Proceedings of the Sixteenth International Conference*, 32-38.

Bowerman , B. L., & Connell, R. T. (2007). *Pronósticos, series de tiempo y regresión.* México: Torzon.

Box, G. P., & Jenkins, G. M. (1976). *Time series analysis, forecasting and control.* San Francisco: Holden Day Inc.

Boyd, C. E. (1998). Pond water aeration systems. *Aquacultural Engineering, 18*, 9-40.

Boyd, C. E. (2001). *Water Quality Standards: Dissolved Oxygen.* sustainable aquaculture practices: Global Aquaculture Alliance.

Boyd, C. E. (s.f.). *Consideraciones sobre la calidad del agua y del suelo en el cultivo de camarón.* Department of Fisheries and Allied Aquacultures. Auburn University, Alabama, USA.

Boyd, C. E., & Tucker, C. S. (1998). *Pond Aquaculture Water Quality Management.* Boston, Massachusetts. USA: Kluwer Academic Publishers.

Brebels, A., Shcherbakov, M. V., & Kamaev, V. A. (2010). Mathematical and statistical framework for comparison of neural network models with other algorithms for prediction of Energy consumption in shopping centres. *In the Proceedings of the 37 Int. Conf. Information Technology in Science Education Telecommunication and Business, suppl. to Journal Open Education*, 96-97.

Cadima, J., & Jollife, I. (2009). On Relationships Between Uncentred and Column-Centred Principal Component Analysis. *Pakistan Journal of Statistics, 25*(4), 473-503.

Cañon, J. (2012). Deformidades, daño físico y conducta como indicadores de bienestar en peces. *SalmonXpert, 1*, 25-29.

Corral, L., Grizel, H., Montes, J., & Polanco, E. (1999). *LA ACUICULTURA. Biología, regulación, fomento, nuevas tendencias y estrategia comercial.* España: Fundación Alfonso Martín Escudero.

Coughlin, K. T., & Tung, K. K. (2004). 11-Year solar cycle in the stratosphere extracted by the empirical mode decomposition method. *Advances in Space Research, 34*(2), 323-329.

Doaa, M. A., Faten, H. F., Ninet, M. A., & Hassen, T. D. (2012). Design and Control Strategy of Diffused Air Aeration System. *International Journal of Computer, Electrical, Automation, Control and Information Engineering, 6*(3), 385-389.

Fogel, L. J., Owens , A. J., & Walsh, M. J. (1966). *Artificial intelligence through simulated evolution.* New York: Wiley.

Freeman, J., & Skapura, J. (1991). *Redes Neuronales - Algoritmos, aplicaciones y técnicas de programación.* Addison-Wesley.

G, Z., B, P., & M, H. (1998). Forecasting with artificial neural networks. The state of art. *International Journal of Forecasting, 14*, 35-62.

Gerstenfeld, A. (1971). *Technological forecasting.* 44(1): Journal of Business.

Gomes, F. L., Machado, M. A., Caldeira, A. M., Santos, J. D., & do Nacimento, W. J. (2016). Time Series Forecasting with Neural Networks and Choquet Integral. *Procedia Computer Science*, 1119-1129.

Goyenola, G. (2007). *Oxigno disuelo.* Uruguay: RED MAPSA.

Guo, Z., & Uhrig, R. (1992). Using genetic algorithms to select inputs for neural networks. (D. a. Schafer, Ed.) *in Proc. of the Int't Workshop on Combinations of Genetic Algorithms and Neural Networks* , 223-234.

Hargreaves, J. A., & Tucker, C. S. (2002). *Measuring Dissolved Oxygen Concentration in Aquaculture.* Southern Regional Aquaculture Center, Detartment of Agriculture. Mississippi, U.S.: SRAC Publication No. 4601.

Haykin, S. (2005). *Neural Networks. A comprensive foundation* (2 ed.). Delhi, India: Pearson.

Hernández, L., Baladrón, C., Aguiar, J. M., Calavia, L., & Carro, B. (2014). Artificial Neural Network for Short-Term Load Forecasting in Distribution Systems. *Energies, 7*, 1576-1598.

Hilera, J. (2000). Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones. *Alfaomega,*, 132-153.

Hippert, H. S., Pedreira, R. C., & Souza, R. C. (2001). Neural networks for short-term load forecasting: a review and evaluation. *IEEE Transactions on Power Systems, 16*(1), 44-55.

Holland, H. J. (1975). *Adaptation in Natural and Artificial Systems.* MI: Univ. of Michigan Press.

Huan, J., Liu, X., Wang, H., & Gui, F. (2013). DO Control System Base on PID Control Algorithm for Aquaculture. *Journal of Information & Computational Science, 10*(11), 3519-3528.

Jain, A. K., & Mao, J. (1996). *Artificial Neural Networks: A Tutorial.* IEEE.

Juan, H., Xingqiao, L., Hui, L., Hongyan, W., & Xiaowei, Z. (2014). A Monitoring and Control System for Aquaculture via Wireless Network and Android Platform. *Sensors & Transducers, 4*, 250-256.

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering, 82*(D), 35-40.

Kamaev, V. A., Shcherbakov, M. V., Panchenko, D. P., Shcherbakova, N. L., & Brebels, A. (2012). Using Connectionist Systems for Electric Energy Consumption Forecasting in Shopping Centers. *Automation and Remote Contro, 73*(6), 1075-1084.

Kenneth , O., & Risto, M. (2002). Evolving Neural Networks through Aumenting Topologies. *Evolutionary Computation, 10*(2), 99-127.

Kishtawal, C. M., Basu, S., Patadia, F., & Thapliyal, P. K. (2003). Forecasting summer rainfall over India using genetic algorithm. *AN AGU JOURNAL, 30*(23), 16-20.

Kjoung, K. (Septiembre de 2003). Financial time series forecasting using support vector machines. *Neurocomputing, 55*(1-2), 307-319.

Kline, D. M. (2004). Methods for multi-step time series forecasting with neural networks. (P. Zhang, Ed.) *Neural Networks in Business Forecasting*, 226–250.

Kline, D. M. (2004). Methods for multi-step time series forecasting with neural networks. (P. Zhang, Ed.) *Neural Networks in Business Forecasting*, 226-250.

Kolassa, S., & Martin, R. (2011). Percentage errors can ruin your day (and rolling the dice shows how). *Foresight, (Fall)*, 21-27.

Koza, J. R. (1990). *Genetic Programming: A Paradigm for Genetically Breeding Populations of Computer Programs to Solve Problems.* Standford University Computer Science Department technical report: STAN-CS-90-1314.

Kriesel, D. (2005). *A Brief Introduction to Neural Networks.* dkkriesel.

Landassuri, M. V., Bustillo, H. C., Carbajal, H. J., & Sanchez, F. L. (2013). Single-Step-Ahead and Multi-Step-Ahead Prediction with Aplications. *Lecture Notes in Computer Science*, 65-72.

Lekang, O. I. (2013). *Aquaculture engineering.* Ucrania: Wiley-Black Well.

Li, R., & Wang, Y. (2008). Short-term wind speed forecasting for wind farm based on empirical mode descomposition. *International Conference on Electrical Machines and Systems*, 2521-2525.

Liu, Y., & Yao, X. (1996). Evolutionary design neural networks with different nodes. *Conf. on Evolutionary Computation*, 670-675.

Lopez, M. D., Landasuri, M. V., Quintana, L. M., & Bustillo, H. C. (2013). Predicción multivariada de fenómenos meteorológicos usando el algoritmo fs-epnet. *IEEE Advancing Technology for Humanity*.

Louka, P., Galanis, G., Siebert, N., Kariniotakis, G., Katsafados, P., Pytharoulis, I., & Kallos, G. (Diciembre de 2008). Improvements in wind speed forecasts for wind power prediction purposes using Kalman filtering. *Journal of Wind Engineering and Industrial Aerodynamics, 96*(12), 2348-2362.

Mabrouk, A. B., Abdallah, N. B., & Dhifaoui, Z. (Mayo de 2008). Wavelet decomposition and autoregressive model for time series prediction. *Applied Mathematics and Computation, 199*(1), 334-340.

Mahmoud, E. (1984). Accuracy in forecasting: A survey. *Journal of Forecasting, 3*(2), 139-159.

Mahtfoud, S., & Ganesh, M. (Noviembre de 2010). Financial forecasting using genetic algorithms. *10*(6), 543-566.

Mao, K. Z. (2002). RBF neural network center selection based on Fisher ratio class separability measure. *IEEE Transactions on Neural Networks, 13*(5), 1211-1217.

McNames, J. (Belgium de 1998). A nearest trajectory strategy for time series prediction. *In Proceedings of the InternationalWorkshop on Advanced Black-Box Techniques for Nonlinear Modeling*, 112-128.

Meyer, D. E. (2004). Introduccion a la acuicultura. Escuela Agricola Panamericana Zamorano, Honduras.

Miao, X., Deng, C., Li, X., Gao, Y., & He, D. (2010). A Hybrid Neural Network and Genetic Algorithm Model for Predicting Dissolved Oxygen in an Aquaculture Pond. *2010 International Conference on Web Information Systems and Mining*, 416-419.

Miller, G., Todd, P., & Hedge, S. (1989). Designing neural networks using genetic algorithms. (J. Schaffer, Ed.) *Procedings of th Third International Conference on Genetic Algorithms*, 379-384.

Naves, C. S., & Gerstner, W. (2015). Nonlinear Hebbian learning as a universal principle in unsupervised feature learning. *Deep Learning Workshop*, Lille, Francia.

Odim, M. O., Gbadeyan, J. A., & Sadiku, J. S. (2016). Modelling the Multi-Layer Artificial Neural Network for Internet Traffic Forecasting: The Model Selection Design Issues. *CoRI'16*, 10-16.

Owoeye, D., Shcherbakov, M., & Kamaev, V. (2013). A photovoltaic output backcast and forecast method based on cloud cover and historical data. *In the Proceedings of the The Sixth IASTED Asian Conference on Power and Energy Systems (AsiaPES 2013)*, 28-31.

Peralta, J., Cortez, P., Sanchis, A. M., & Gutierrez, G. S. (2012). Evolving time-lagged feedforward neural networks for time series forecasting. (ACM, Ed.) *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, 163-164.

Pindick, R. S., & Rubinfield, D. I. (1991). *Econometric models and economic forecasts.* Sinagpore: McGraw Hill.

Praveen, J., & Varalakshmi, P. (2015). A SURVEY ON NEURAL NETWORK MODELS FOR DATA ANALYSIS. *ARPN Journal of Engineering and Applied Sciences , 10*(11), 4872-4876.

Rechenberg, I. (1973). *Evolutions strategie.* Friedrich Frommann Verlag.

Rowe, G., & Wright, G. (2001). *Expert opinions in forecasting: The role of the Delphi Technique.* Boston:: Kluwar.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning Representations by Bak-Propagation Errors. *Nature, 323*, 533-536.

Saad, E., Prokhorov, D., & Wunsch, D. (1998). Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *Neural Networks, 9*(6), 1456-1470.

Saha, A. K., Chowdhury, S., Chowdhury, S. P., Song , Y. H., & Taylor, G. A. (2006). Application of wavelets in power system load forecasting. *IEEE Power Engineering Society General Meeting*, 6-16.

Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks, 2*(6), 459-473.

Sapankevych, N. I., & Sankar, R. (Mayo de 2009). Time Series Prediction Using Support Vector Machines: A Survey. *IEEE Computational Intelligence Magazine, 4*(2), 24-38.

Senjyu, T., Takara, H., Uezato , K., & Funabashi, T. (Febrero de 2002). One-hour-ahead load forecasting using neural network. *IEEE Transactions on Power Systems, 17*(1), 113-118.

Simbeye, D. S., & Yang, S. F. (Abril de 2014). Water Quality Monitoring and Control for Aquaculture Based on Wireless Sensor Networks. *JOURNAL OF NETWORKS, 9*(4), 840-849.

Sorjamaa , A., Hao,, J., Reyhani, N., Ji, Y., & Lendasse, A. (Octubre de 2007). Methodology for long-term prediction of time series. *Neurocomputing, 70*(16-18), 2861-2869.

Tabares, H., Branch, J., & Jaime , V. (Septiembre de 2006). Generación dinámica de la topología de una red neuronal artificial del tipo perceptron multicapa. *Revista Facultad de Ingeniería*, 146-162.

Talavera, V., & Zapata, L. M. (1998). *Monitoreo del oxígeno disuelto en estanques de cultivo de camarón.* Nicovita. Perú: Tumpis.

Tautenhahn, A., & Karg, U. (2007). *Niveles fiables de oxígeno en piscifactorías con LDO.* Vizcaya, España: Hatch Lange.

Tran, V. T., Yang,, B. S., & Tan, A. C. (2009). Multi-step ahead direct prediction for the machine condition prognosis using regression trees and neuro-fuzzy systems. *Expert Syst, 36*(5), 9378-9387.

Tucker, C. (2005). *Pond Aeration.* Southern Regional Aquaculture Center, Detartment of Agriculture. Texas, U.S.: SRAC Publication No. 3700.

Tyukov, A., Brebels, A., Shcherbakov, M., & Kamaev, V. (2012). A concept of web-based energy data quality assurance and control system. *In the Proceedings of the 14th International Conference on Information Integration and Web-based Applications & Services*, 262-271.

Tyukov, A., Shcherbakov, M., & Brebels, A. (2011). Automatic two way synchronization between server. *In the Proceedings of The 13th International Conference on Information Integration and Web-based Applications & Services*, 467-470.

Victor, M. L., & J, A. B. (2011). Blasing the evolution of modular neural networks. *in Procedings of the 2011 IEEE Congress on Evolutionary Computation* , 1952-1959.

William, A. W. (2011). *Low oxigen and pond aereation.* Kentuky State University Cooperative Extension Program.

Xu, B., Shen, F., & Zhao, J. (2016). Density Based Self Organizing Incremental Neural Network for data stream clustering. *2016 International Joint Conference on Neural Networks Conference on Neural Networks*, 2654-2661.

Yao, X., & Lin, Y. (1997). Anew evolutionary system for evolving artifitial neural networks. *IEEE Transactions on Neural Networks, 8*(3), 694-713.

Yokuma, J. T., & Armstrong, J. S. (1995). Beyond accuracy: Comparison of criteria used to select forecasting methods. *International Journal of Forecasting, 11*(4), 591-597.

Yukun, B., Xiong, T., & Hu, Z. (2013). Multi-Step-Ahead Time Series Prediction using Multiple-Output Support Vector Regression. *Neurocomputing*, 1-39.

Zurada, J. M., Malinowski, A., & Usui, S. (1997). Perturbation method for deleting redundant inputs of Perceptron network. *Neurocomputing,, 14*(2), 177-193.