Pronóstico de series de tiempo con redes neuronales regularizadas y validación cruzada

Time series forecasting with neural networks regularized and cross validation

Juan David Velásquez H.*
Yulieth Fonnegra R**.
Fernán Alonso Villa G.***

Fecha recepción: 17 de marzo de 2013
Fecha aceptación: 30 de abril de 2013

Resumen

En este trabajo se propone usar integralmente la estrategia de regularización de descomposición de pesos y validación cruzada con el fin de controlar integralmente el problema del sobreajuste en redes neuronales tipo perceptrón multicapa para el pronóstico de series de tiempo. Con el fin de evaluar la capacidad de la propuesta, se pronostica una serie de tiempo tradicional de la literatura. Los resultados evidencian que la combinación de ambas técnicas permite encontrar modelos con mejor capacidad de generalización que aproximaciones tradicionales.

Palabras Clave

Pronóstico, Series de tiempo, regularización, validación cruzada, descomposición de pesos.

* Universidad Nacional de Colombia. Medellín, Antioquia, Colombia. jdvelasq@unal.edu.co

** Universidad de Medellín Medellín Antioquia, Colombia. yfonnegrar@unal.edu.co

** Universidad Nacional de Colombia. Medellín, Antioquia, Colombia favilla0@unal.edu.co

Abstract

In this paper we propose the use of weight decay regularization strategy and cross-validation in order to control integrally the problem of overfitting in Multilayer Perceptron neural networks for time series forecasting. In order to evaluate the ability of the proposal, we made experiments with some traditional series. The results show that the combination of both techniques allows to find models with better generalization ability that the traditional approaches.

Keywords

Forecasting, time series, regularization, cross validation, weight decay

1. Introducción

El pronóstico de series de tiempo es considerado un problema común en muchas disciplinas del conocimiento [1]. Por ejemplo, en la administración de la producción y sistemas de inventario se realizan frecuentemente pronósticos con el fin de facilitar la toma de decisiones de corto, mediano y largo plazo sobre procesos de control de calidad, análisis de inversiones, planeación financiera, mercadeo, entre otros [2]. Además, el interés en este campo ha aumentado gradualmente a través del tiempo en diversas áreas de la ciencia, ingeniería y finanzas [1].

Una serie de tiempo es una secuencia de observaciones de un fenómeno determinado, ordenadas secuencialmente y registradas, usualmente, en igual intervalo de tiempo. El modelado de una serie consiste en construir sistemáticamente una representación matemática que permita capturar, total o parcialmente, el proceso generador de los datos; una vez se construye un modelo, es posible realizar el pronóstico de la serie para un horizonte determinado, es decir, estimar sus valores futuros [2, 3, 4].

Este problema ha sido tratado con diferentes tipos de modelos estadísticos y matemáticos [1, 5]; los cuales, en un sentido amplio, se pueden categorizar en lineales y no lineales con base en un comportamiento supuesto para una serie de tiempo [1].

Según Palit y Popovic [1], tradicionalmente se han usado varios tipos de RNA para el pronóstico de series de tiempo, entre estos se encuentran: Perceptrón multicapa (MLP) [25, 1]; Funciones de base radial (RBF) [26, 27]; FNN - Fuzzy Neural Networks [28, 29]; Feedforward and Recurrent Networks [3, 30, 31]. Zhang et al. [24] y Palit et al. [1] coinciden en que el tipo de red neuronal MLP es uno de los modelos más influyentes para el modelado y pronóstico de series de tiempo.

En el contexto general del modelado y pronóstico de series de tiempo, los MLP presentan serias limitaciones debido a que su proceso de especificación es difícil debido a la gran cantidad de pasos metodológicos que requiere (selección de la entradas al modelo, cantidad de neuronas en la capa oculta, etc.) [32]. Además, al igual que los modelos tradicionales (no paramétricos y no lineales), los MLP pueden adolecer del fenómeno del sobreajuste, y memorizar los datos de entrada degradando su capacidad de pronóstico [23].

Con el fin de controlar el problema del sobreajuste, Tikhonov en [33] propuso la metodología de regularización para resolver problemas mal condicionados, similares al problema de estimación de parámetros de

una RNA. La idea principal del método es estabilizar la solución usando algún tipo de función para penalizar la función objetivo, también llamada estrategia de regularización. No obstante, la aplicación de la metodología es compleja, dado que, el problema no es solamente seleccionar una determinada estrategia de regularización entre las disponibles, sino que también es necesario determinar qué tanto debe incidir tal estrategia sobre el entrenamiento de la red [22]. Por su simplicidad, una de las técnicas más usadas es la de descomposición de pesos propuesto por Hinton [34].

Suponiendo que se regulariza la red neuronal, aún quedan varios interrogantes por resolver, uno de ellos es ¿Cómo dividir el conjunto de datos (serie de tiempo), tal que el subconjunto de entrenamiento contenga la información suficiente del fenómeno que se desea modelar?; esto conlleva a la siguiente pregunta ¿Cómo evaluar la capacidad de generalización del modelo?, es decir, ¿Cómo validar el modelo? Se dice que un modelo de red neuronal generaliza bien cuando el mapeo de entrada-salida de la red es cercano al conjunto de validación, el cual no fue usado para el entrenamiento [25].

En este orden de ideas se han planteado una diversidad de técnicas para la validación de modelos, entre las más tradicionales se tiene: *SplitSample* [35] y *Cross Validation (*validación cruzada*)* [36, 37, 38]; una de las ventajas de validación cruzada es que permite controlar el problema del sobreajuste, mediante la selección adecuada de un conjunto de entrenamiento que posea la información suficiente para modelar la serie de tiempo con una red neuronal.

Entonces, este artículo tiene los siguientes objetivos: exponer algunos de los problemas del pronóstico de series de tiempo con redes neuronales, y presentar el sobreajuste como uno de los principales; para controlar integralmente tal problema se propone usar simultáneamente la estrategia de regularización descomposición de pesos y de validación cruzada en los perceptores multicapa; además, analizar experimentalmente el efecto de realizar tal integración al pronosticar una serie de tiempo tradicional de la literatura; también, con este trabajo se pretende contribuir tanto conceptual como metodológicamente, a la solución de algunos de los problemas que se presentan en la predicción de series de tiempo con redes neuronales.

Con el fin de alcanzar los objetivos, este trabajo está estructurado como sigue: en la Sección 2 se realiza una breve introducción a los perceptrones multicapa para el pronóstico de series de tiempo y sus principales bondades y dificultades; seguidamente en la 3, se presenta la regularización como una manera de controlar el sobreajuste en las redes neuronales; sin embargo, con la regularización no es suficiente, dado que no permite controlar el tamaño del conjunto de entrenamiento; entonces en la 4, se revisa la técnica de validación cruzada para controlar el tamaño del conjunto de entrenamiento. Finalmente, en la 5 se propone usar integralmente validación cruzada y regularización para controlar integralmente el sobreajuste al pronosticar una serie de tiempo conocida en la literatura, y se concluye que tal propuesta permite encontrar modelos con adecuada capacidad de generalización.

2. Perceptrones multicapa para el pronóstico de series de tiempo

Para modelar series de tiempo con comportamiento supuesto como lineal se han usado ampliamente modelos como: AR, MA, ARMA y ARIMA (Box y Jenkins, 1976; Montgomery et al. , 1990; Wei, 2006). Sin embargo, éste tipo de modelos no es suficiente, dado que la gran mayoría de series de tiempo en ingeniería, finanzas y econometría presentan un comportamiento aparentemente no lineal [1].

En la literatura más relevante se han propuesto diversos modelos no lineales entre los que se encuentran: Bilineales [6]; Autorregresivos de umbral (TAR) [7]; De Hete-

rocedasticidad condicional autorregresiva (ARCH) [8]; Autorregresivos de transición suave (STAR) [9, 10, 11]; De Heterocedasticidad condicional autorregresiva generalizada (GARCH) [12]. Adicionalmente, Tong [13], De Gooijer y Kumar [14], Peña [15], Tjostheim [16], Hardle et al. [17] y Tong [11] realizan una amplia recopilación donde examinan otros modelos.

Aunque los modelos no lineales tradicionales han demostrado ser útiles en problemas particulares, no son adecuados para la mayoría de los casos, dado que suponen una forma de no linealidad preestablecida en la serie, es decir, los datos se deben adaptar a la estructura no lineal definida por el modelo; de este modo, muchas veces no representan adecuadamente el comportamiento de la serie, véase a [18]. Además, para definir cada familia de estos modelos, es necesario especificar un tipo apropiado de no linealidad; esto es una tarea difícil comparado con la construcción de modelos lineales; la cantidad posible de funciones para definir el tipo de no linealidad es amplia [19, 20].

Por otro lado, desde la Inteligencia Computacional se han propuesto diversas técnicas para el modelado y pronóstico de series de tiempo; de las disponibles, las redes neuronales artificiales (RNA) han mostrado ser más robustas que otras técnicas tradicionales, especialmente en la representación de relaciones complejas que exhiben comportamientos no lineales, por ejemplo véase [21, 22]. Masters [23], recomienda utilizar RNA en vez de alguna técnica tradicional por las siguientes razones:

- Poseen una amplia capacidad para aprender relaciones desconocidas a partir de un conjunto de ejemplos.
- Tienen una alta tolerancia a patrones extraños de ruido y componentes caóticas presentes en un conjunto de datos.
- Son suficientemente robustas para procesar información incompleta, inexacta o contaminada.
- No restringen el tipo de no linealidad de la serie de tiempo a la estructura matemática del modelo de red neuronal.

Respecto al pronóstico de series de tiempo con RNA, Zhang *et al*. realizaron una revisión general del estado del arte donde resaltan tanto éxitos y fracasos reportados de las redes neuronales (especialmente con los perceptrones multicapa) [24]; incluyendo las publicaciones más relevantes y los tópicos de investigación más influyentes hasta 1996. Sin embargo, en la última década se ha producido un considerable número de contribuciones en múltiples campos como metodologías de aprendizaje, selección de entradas relevantes, neuronas ocultas, entre otros, cuya influencia no ha sido evaluada ni reportada en la literatura.

Un MLP es un tipo de red neuronal que imita la estructura masivamente paralela de las neuronas del cerebro. Básicamente, es un conjunto de neuronas (nodos) que están lógicamente ordenadas en tres o más capas; generalmente, posee una capa de entrada, una oculta y una de salida, cada una de éstas tiene al menos una neurona. Entre la capa de entrada y la capa de salida, es posible tener una o varias capas ocultas; aunque se ha demostrado que para la mayoría de problemas es suficiente con una sola capa oculta [1]; mientras que para el pronóstico de series de tiempo es suficiente con una neurona en la capa de salida [39].

Como se mencionó en la introducción, los MLP son uno de los tipos de red que más ha tenido influencia en la literatura, su éxito se debe a que: desde un punto de vista matemático, un MLP tiene la capacidad de aproximar cualquier función continua definida en un dominio compacto con una precisión arbitraria previamente establecida [40, 41, 42]. En la práctica, los MLP se han caracterizado por ser muy tolerantes a información incompleta, inexacta o contaminada con ruido [23]

Para pronosticar una serie de tiempo con un MLP, se toma como punto de partida, que

una serie se define como una secuencia de T observaciones ordenadas en el tiempo:

$$y_t = \{y_i\}_1^T \quad (1)$$

para la cual se pretende estimar una función que permita explicar y_t en función de sus rezagos $\{y_{t-1}, y_{t-2},\ldots,y_{t-p}\}$; es posible especificar matemáticamente la función y_t como un MLP, así:

$$y_t = \beta_* + \sum_{h=1}^{H} \beta_h \times g\left(\alpha_h + \sum_{p=1}^{P} w_{p,h} \times y_{t-p}\right) + \varepsilon_t \quad (2)$$

La Ecuación (2) equivale a un modelo estadístico no paramétrico de regresión no lineal [43]; para esta ecuación se tienen en cuenta los siguientes aspectos:

- Se supone que εt sigue una distribución normal con media cero y varianza desconocida σ2.
- H representa el número de neuronas en la capa oculta.
- P es el número máximo de rezagos considerados (neuronas de entrada).
- es la función de activación de las neuronas de la capa oculta.
- Los parámetros $W = [\beta^*, \beta_h, \alpha_h, w_{p,h}]$, con h = 1, 2,…, H y p = 1, 2, …, P, son estimados usando el principio de máxima verosimilitud de los residuales, el cual equivale a la minimización del error cuadrático medio.
- En el contexto de las series de tiempo, el modelo puede ser entendido como una combinación lineal ponderada de la transformación no lineal de varios modelos Autorregresivos.

La estimación de los parámetros W del modelo definido en (2) es un problema numérico de optimización [23], mientras que desde un punto de vista estadístico, es un proceso de estimación no paramétrica funcional [44]. Para resolverlo se han propuesto diversas técnicas de optimización:

- Basadas en gradiente, tales como Backpropagation [45], y Rprop - Resilient Backpropagation [45, 46];
- Heurísticas, como estrategias evolutivas [47], entre otras.

En general, RPROP es considerado como uno de los algoritmos basados en gradiente más apropiados para entrenar redes neuronales artificiales [47, 45, 46].

Sin embargo, el problema no es simplemente estimar cada modelo para una serie en particular. Mientras en el caso lineal hay una importante experiencia ganada, existen muchos problemas teóricos, metodológicos y empíricos abiertos sobre el uso de modelos no lineales. En el caso del MLP, su proceso de especificación es difícil debido a la gran cantidad de pasos metodológicos que requiere:

- Seleccionar cuáles son las entradas al modelo o rezagos (neuronas capa de entrada).
- Determinar la cantidad de neuronas en la capa oculta.
- Seleccionar la función de activación.
- Seleccionar cuál es la función objetivo que se desea optimizar (SSE, MSE, RMSE, MAE, GRMSE).
- Estimar los parámetros del modelo con alguna técnica de optimización.
- Cómo evaluar la capacidad de generalización del modelo, es decir, validar que el modelo estimado representa adecuadamente el comportamiento de la serie.

A lo anterior, se suma la dificultad de que los criterios sobre cómo abordar cada paso son subjetivos [48]. La falta de identificabilidad estadística del modelo es uno de los aspectos que dificultan su especificación.

Los parámetros óptimos no son únicos para una especificación del modelo (número de entradas o rezagos, cantidad de neuronas en la capa oculta, funciones de activación, etc.), y un conjunto de datos dado. Esto se debe a que [49]:

- Se puede obtener múltiples configuraciones que son idénticas en comportamiento cuando se permutan las neuronas de la capa oculta, manteniendo vinculadas las conexiones que llegan a dichas neuronas.
- Cuando las neuronas de la capa oculta tienen funciones de activación simétricas alrededor del origen, la contribución neta de la neurona a la salida de la red neuronal se mantiene igual si se cambian los signos de los pesos que entran y salen de dicha neurona.
- Si los pesos de las conexiones entrantes a una neurona oculta son cero, es imposible determinar el valor del peso de la conexión de dicha neurona oculta a la neurona de salida.
- Si el peso de la conexión de una neurona oculta hacia la neurona de salida es cero, es imposible identificar los valores de los pesos de las conexiones entrantes a dicha neurona oculta.

Otro inconveniente que se debe tener en cuenta, al igual que los modelos tradicionales, los MLP pueden adolecer del fenómeno del sobreajuste, básicamente por tres causas: la primera está relacionada con la existencia de datos extremos (*outliers*) en el conjunto de entrada, esto hace que la varianza de los parámetros de la red sea alta; la segunda con la cantidad de neuronas en la capa de entrada y oculta, es decir, el tamaño óptimo de la red. Si se selecciona una cantidad alta o inadecuada de entradas, se sobreparametriza la red neuronal, y esta memoriza los datos de entrenamiento en vez de aprender el comportamiento de la serie, esto se evidencia cuando se produce un error de entrenamiento muy pequeño y un error de validación muy alto [22]; la tercera, el subconjunto de entrenamiento no posee la cantidad suficiente de información que represente la estructura del proceso generador de los datos [50].

Las primeras dos causas se pueden controlar mediante el uso de la regularización, mientras que la tercera mediante la selección de una técnica adecuada de validación. Sin embargo, en la literatura más relevante no se ha considerado usar integralmente regularización y validación cruzada para controlar efectivamente el sobreajuste en redes neuronales.

3. La regularización para controlar el sobreajuste

Con el fin de controlar el problema del sobreajuste, Tikhonov en [33] propuso la metodología de regularización para resolver problemas mal condicionados. La idea principal del método es estabilizar la solución usando algún tipo de función para penalizar la función objetivo. En general, el método de regularización tiene como objetivo realizar un intercambio equilibrado entre la fiabilidad de los datos de entrenamiento y las bondades del modelo. En procedimientos de aprendizaje supervisado, el intercambio se realiza a través de la minimización el riesgo total [25], dado por la expresión:

$$R(W) = \xi_s(W) + \lambda\xi_c(W) \qquad (3)$$

La ecuación (3) corresponde a un caso general del método de regularización de Tikhonov [33] para solucionar problemas mal condicionados (como lo es el entrenamiento de una red neuronal), en este, ξ_s (W) se conoce como la medida estándar de rendimiento, acostumbra utilizar el error cuadrático (SSE) o el error cuadrático medio (MSE); éste término corresponde a (3), de este modo R(W) puede definirse como:

$$R(W) = \sum_{t=1}^{T} (\hat{y}_t - y_t)^2 + \lambda\xi_c(W) \quad (4)$$

ξ_c (W) es la penalización compleja, también conocida como estrategia de regularización

o término de regularización, que para una red en general, está dado por la integral de suavizado de orden k [25]. Mientras que, λ $\xi_c(W)$ es el parámetro o factor de regularización que controla el nivel de incidencia de $\xi_c(W)$ sobre el entrenamiento de la red, en secciones posteriores de éste documento se discutirá sobre este factor.

$$\xi_c(w,k) = \frac{1}{2}\int \left\| \frac{\partial^k}{\partial w^2} F(w,m) \right\|^2 \mu(w)dw \quad (5)$$

En la Ecuación (5), F(w,m) es el mapeo de entrada–salida realizado por el modelo, $\mu(w)$ es alguna función de ponderación que determina la región del espacio de entrada sobre la cual la función F(w,m) es requerida para ser suavizada.

Dada la Ecuación (4), desde el punto de vista de optimización numérica, el método de regularización es una especie de penalización que se impone sobre la función objetivo definido en (3). A continuación, se describe una de las técnicas más usadas, descomposición de pesos propuesto por Hinton [34].

3.1 La descomposición de pesos (DP) - (Weight Decay)

El procedimiento de descomposición de pesos propuesto por Hinton [34], opera sobre algunos pesos sinápticos de la red forzándolos a tomar valores cercanos a cero y permitiendo a otros conservar valores relativamente altos. Esta discriminación permite agrupar los pesos de la red en: pesos que tienen poca o ninguna influencia sobre el modelo; y pesos que tienen influencia sobre el modelo, llamados pesos de exceso. Para ésta estrategia el procedimiento la penalización de complejidad se define como:

$$\xi_c(w) = \left\| w_{p,h} \right\|^2 = \sum_{h=1}^{H}\sum_{p=1}^{P} w_{p,h}^2 \quad (6)$$

En la Ecuación (6), $w_{p,h}$ son los pesos de la entrada p a la neurona h, es decir, los pesos entre la capa de entrada y la oculta. Todos los pesos son tratados igual, es decir, se parte del supuesto que la distribución de los pesos en el espacio estará centrada en el origen.

La descomposición de pesos es una de las estrategias de regularización más utilizadas en la literatura [51]; dado que su implementación es computacionalmente sencilla, no depende de parámetros adicionales y permite mejorar la capacidad de generalización de la red neuronal.

Por otro lado, en problemas de ajustes de curvas también se le conoce con el nombre de regresión de borde (Ridge Regression) [52], porque su efecto es similar a la técnica de regresión del mismo nombre propuesta por Hoerl y Kennard [53]. Además, en aprendizaje Bayesiano, es posible hallar la correspondiente función de distribución de prioridad para esta estrategia, la cual depende tanto de los pesos como de su agrupación (neuronas, también llamadas hiperparámetros) [52].

Suponiendo que se usa la regularización por descomposición de pesos, para controlar el sobre ajuste, aún falta abordar ¿cómo dividir el conjunto de datos (serie de tiempo), tal que el subconjunto de entrenamiento contenga la información suficiente del fenómeno que se desea modelar?; esto conlleva a la siguiente pregunta ¿cómo evaluar la capacidad de generalización del modelo?

4. La validación cruzada para seleccionar un conjunto apropiado de entrenamiento

En la literatura más relevante se han planteado una diversidad de técnicas para la validación de modelos de redes neuronales, entre las más tradicionales se tiene:

- *SplitSample* [35] el cual consiste en dividir el conjunto de datos en entrenamiento y validación, algunos autores recomiendan dividirlo en entrenamiento, prueba y validación; en ambos casos el conjunto de

validación nunca se usa para estimar los parámetros del modelo. La principal crítica de esta técnica es que no se ha definido un criterio sobre cómo se debe dividir el conjunto tal que no se pierda información valiosa para la estimación de los parámetros [54].

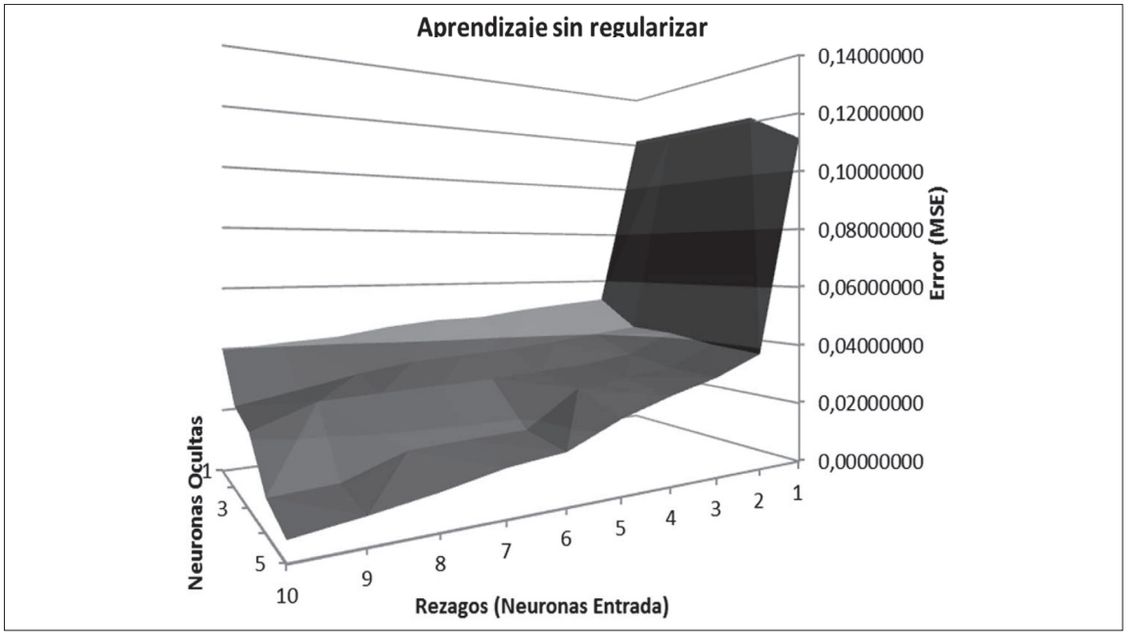
- *Cross Validation (*validación cruzada*)* [36, 37, 38], en la literatura existen varios tipos de validación cruzada, la más utilizada es la de k-iteraciones, la cual consiste en dividir el conjunto de datos en k subconjuntos. Uno de los subconjuntos se utiliza como datos de validación y el resto (k–1) como datos de entrenamiento. El proceso de validación cruzada es repetido durante k iteraciones, con cada uno de los posibles subconjuntos de datos de validación. Finalmente, se selecciona el que mayor capacidad de generalización posea.

Dado que la validación cruzada permite controlar el problema del sobreajuste, mediante la selección adecuada de un conjunto de entrenamiento que posea la información suficiente para modelar la serie de tiempo; en este trabajo se propone integrar en las redes MLP la técnica de regularización de descomposición de pesos y de validación cruzada con el fin de controlar integralmente el sobreajuste. A continuación, se revisan experimentalmente los efectos de la propuesta.

5. Control integral del sobreajuste

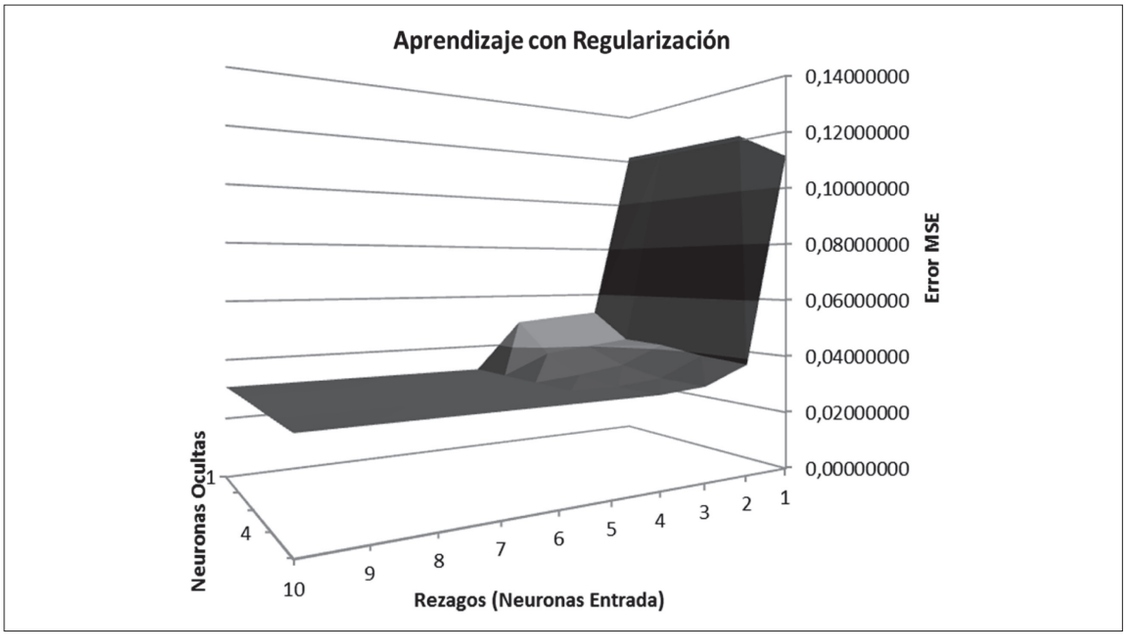
En esta sección se propone el uso integrado de regularización por descomposición de pesos y de validación cruzada en la especificación de la arquitectura de red MLP; para pronosticar la serie de tiempo de Linces Canadienses, ampliamente usada en la literatura; en esta serie se encuentra registrada la cantidad de linces capturados anualmente, desde 1821 hasta 1934, en los alrededores del río Mackenzie ubicado en el distrito de Northem, Canadá, fue estudiada por [55], [56], y [57]. Los datos de la serie se transformaron utilizando la función logaritmo base 10; a diferencia de estudios anteriores,

Figura 1: Error cuadrático medio de entrenamiento (MSE). Aprendizaje sin regularizar y validación cruzada.



Fuente: elaboración propia

Figura 2. Error cuadrático medio de entrenamiento (MSE). Aprendizaje con regularización y validación cruzada



Fuente: elaboración propia

donde de sus 114 datos se tomaron los 100 primeros para entrenamiento y los últimos 14 para validación, en este estudio se ha usado validación cruzada de k iteraciones, con k incremental.

Se realizó el pronóstico de la serie con diferentes modelos de MLP regularizados y no regularizados, incrementando la cantidad de neuronas en la capa de entrada (1 a 10) y oculta (1 a 5) una a la vez, los parámetros de cada modelo se estimaron usando RPROP y se usó validación cruzada de k iteraciones incremental para evaluar la capacidad de generalización de cada modelo. Como caso de control se tomó el pronóstico sin regularizar, el resumen de los resultados de entrenamiento se presenta en la Figura 1; en esta se puede observar que el error cuadrático medio (MSE) de entrenamiento decrece a medida que se aumenta tanto la cantidad de neuronas en la capa de entrada como en la capa oculta; de este modo se evidencia que la sobreparametrización de la red ocasiona que se memoricen los datos de entrenamiento y se obtenga una pobre capacidad de generalización, pero se ha garantizado mediante validación cruzada que los datos usados para entrenamiento contienen la información suficiente del proceso que generó la serie de tiempo.

Con el fin de controlar el sobreajuste producido por la sobreparametrización de la red neuronal, en la Figura 2, se resumen los resultados de pronosticar con los mismos modelos de la Figura 1 usando descomposición de pesos como técnica de regularización y validación cruzada. Los resultados evidencian que la combinación de ambas técnicas permite controlar el sobreajuste, dado que el error de entrenamiento se estabiliza, sin importar si se siguen agregando neuronas en la capa de entrada o en la capa oculta, de este modo se puede obtener una adecuada capacidad de generalización.

Finalmente, es necesario seleccionar cuál es el modelo más adecuado para pronosticar la serie de tiempo en cuestión; en este tra-

Tabla 1: Valor de para varios modelos de MLP.

	P: Neuronas en la Capa de Entrada									
H	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	0,0000	0,0000	0,0000	0,0000	0,0010	0,0001	0,0001	0,0001	0,0100	0,0100
2	0,0000	0,0000	0,0000	0,0000	0,0100	0,0001	0,0010	0,0010	0,0010	0,0100
3	0,0000	0,0000	0,0000	0,0000	0,0001	0,0010	0,1000	0,0100	0,0010	0,0100
4	0,0000	0,0000	0,0000	0,0000	0,0010	0,0001	0,1000	0,0100	0,0010	0,0010
5	0,0000	0,0000	0,0000	0,00001	0,0001	0,0001	0,0001	0,0001	0,0001	0,0001

bajo se propone usar los valores obtenidos del factor de regularización , los cuales se presentan en la Tabla 1, en esta se puede observar que para modelos a partir de 4 neuronas en la capa de entrada (rezagos) y 5 en la oculta es necesario aplicar regularización, dado que el factor es diferente de cero; lo que indica que estos modelos no están sobreajustados y tienen una adecuada capacidad de generalización, los cuales corresponden a la región plana de la Figura 2. Por el principio de parsimonia (*Ockham's razor*) el modelo más simple es el mencionado anteriormente (H=4 y P=5).

6. Conclusiones

Realizar la combinación de regularización mediante descomposición de pesos y técnicas de validación cruzada permite encontrar modelos con mejor capacidad de generalización que aproximaciones tradicionales, como usar validación cruzada sin regularización y viceversa.

Como se puede apreciar en la Tabla 1 el uso de validación cruzada permite estabilizar el parámetro de regularización (término de penalización), permitiendo usar este factor como criterio para seleccionar el modelo que representa adecuadamente la serie de tiempo del experimento.

7. Agradecimientos

Los autores expresan sus agradecimientos a los evaluadores anónimos cuyos comentarios permitirán mejorar ampliamente la calidad de este trabajo.

8. Referencias

[1] A. K. Palit y D. Popovic, Computational Intelligence in Time Series Forecasting, London: Springer, 2005.

[2] D. C. Montgomery, L. A. Johnson y J. S. Gardiner, Forecasting & Time Series Analysis, Segunda ed., Singapure: McGraw-Hill, Inc., 1990.

[3] R. Gençay y T. Liu, «Nonlinear modelling and prediction with feedforward and recurrent networks,» *Physica D: Nonlinear Phenomena,* vol. 108, nº 1-2, pp. 119-134 , 1997.

[4] B. L. Bowerman, R. T. O'Connell y A. B. Koehler, Forecasting, Time Series, and Regression: An Applied Approach, Cuarta ed., Ohio: Cengage Learning Brooks Cole, 2006.

[5] N. Kasabov, Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering, 2da ed., Massachusetts: The MIT Press Cambridge, 1998.

[6] C. Granger y A. Anderson, An Introduction to Bilinear Time Series Models, Gottingen: Vandenhoeck and Ruprecht, 1978.

[7] H. Tong y K. Lim, «Threshold autoregressive, limit cycles and cyclical data,» *Journal of the Royal Statistical Society Series B, 42 (3). Pág. 245–292,* vol. 42, nº 3, p. 245–292, 1980.

[8] R. Engle, «Autoregressive conditional heteroskedasticity with estimates of the variance of UK inflation,» *Econometrica,* vol. 50, p. 987–1008, 1982.

[9] K. S. Chan y H. Tong, «On estimating thresholds in autoregressive models,» *Journal of Time Series Analysis,* nº 7, pp. 178-190, 1986.

[10] D. Dick van Dijk, T. Teräsvirta y F. Franses, «Smooth Transition Autoregressive Models - A Survey Of Recent Developments,» *Econometric Reviews,* nº 21, pp. 1-47, 2002

[11] H. Tong, «Threshold models in time series analysis —30 years on (with discussions by P.Whittle, M.Rosenblatt, B.E.Hansen, P.Brockwell, N.I.Samia & F.Battaglia),» *Statistics & Its Interface,* nº 4, pp. 107-136, 2011.

[12] T. Bollerslev, «Generalised autoregressive conditional heteroscedasticity,» *Journal of Econometrics,* vol. 31, pp. 307-327, 1986.

[13] H. Tong, Nonlinear Time Series: A Dynamical System Approach, Oxford: Oxford University Press, 1990.

[14] I. De Gooijer y K. Kumar, «Some Recent Developments in Non- Linear Modelling, Testing, and Forecasting,» *International Journal of Forecasting,* vol. 8, pp. 135-156, 1992.

[15] D. Peña, «Second-generation time-series models: a comment on 'Some advances in non-linear and adaptive modelling in time-series analysis' by Tiao and Tsay,» *Journal of Forecasting,* vol. 13, pp. 133-140, 1994.

[16] D. Tjostheim, «Nonlinear time series: a selective review,» *Scand. J. Statist.,* vol. 21, pp. 97-130, 1994.

[17] W. Hardle, H. Liitkepohl y R. Chen, «A review of non-parametric time series analysis.,» *Int. Statist. Rev.,* vol. 65, pp. 49-72, 1997.

[18] C. Granger y T. Teräsvirta, Modelling Nonlinear Economic Relationships, Oxford: Oxford University Press, 1993.

[19] C. Granger, «Strategies for modelling nonlinear time-series relationships,» *The Economic Record,* vol. 69, nº 206, p. 233–238, 1993.

[20] P. Zhang, B. Patuwo y M. Hu, «A simulation study of artificial neural networks for nonlinear time-series forecasting,» *Computers & Operations Research,* vol. 28, nº 4, pp. 381-396, 2001.

[21] M. Ghiassi, H. Saidane y D. Zimbra, «A dynamic artificial neural network model for forecasting time series events,» *International Journal of Forecasting,* vol. 21, nº 2, pp. 341-362, 2005.

[22] F. A. Villa, J. D. Velásquez y R. C. Souza, «Una aproximación a la regularización de redes cascada-correlación para la predicción de series de tiempo.,» *Investigación Operacional.,* nº 28, pp. 151-161, 2008.

[23] T. Masters, Practical neural network recipes in C++, New York: Academic Press, 1993.

[24] G. Zhang, B. E. Patuwo y M. Y. Hu, «Forecasting with artificial neural networks: the state of the art,» *International Journal of Forecasting,* vol. 14, nº 1, pp. 35-62, Marzo 1998.

[25] S. Haykin, Neural Networks: A Comprehensive Foundation, New Jersey: Prentice Hall, 1999.

[26] D. Zhang, Y. Han, X. Ning y .. Liu, «A Framework for Time Series Forecasts,» *Proceedings ISECS International Colloquium on Computing, Communication, Control, and Management,* vol. 1, pp. 52-56, 2008.

[27] X.-B. Yan, Z. Wang, S.-H. Yu y Y.-J. Li, «Time Series Forecasting with RBF Neural Network,» *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics,* pp. 4680-4683, 2005.

[28] M. Rast, «Forecasting Financial Time Series with Fuzzy Neural Networks,» *IEEE lnternational Conference on Intelligent Processing Systems,* pp. 432-434, 1997.

[29] V. Kadogiannis y A. Lolis, «Forecasting financial time series using neural network and fuzzy system-based techniques,» *Neural Computing and Application,* vol. 11, pp. 90-102, 2002.

[30] A. Parlos, O. Rais y A. Atiya, «Multi-step-ahead prediction using dynamic recurrent neural networks,» *Neural Networks,* vol. 13, pp. 765-786, 2000.

[31] S. Mishra y S. Patra, «Short term load forecasting using a novel recurrent neural network,» *International Journal of Computational Intelligence: Theory and Practice,* vol. 4, nº 1, pp. 39-45, 2009.

[32] P. Sánchez y J. D. Velásquez, «Problemas de Investigación en la Predicción de Series de Tiempo con Redes Neuronales Artificiales,» *Revista Avances en Sistemas e Informática,* vol. 7, nº 3, pp. 67-73, 2010.

[33] A. Tikhonov, «On Solving Incorrectly Posed Problems and Method of Regularization,» *Docklady Academii Nauk,* vol. 151, pp. 501-504, 1963.

[34] G. Hinton, «Connectionist learning procedures,» *Artificial Intelligence,* nº 40, p. 185–243, 1989.

[35] E. W. Steyeberg, Clinical Prediction Models, New York: Springer, 2009.

[36] S. Geisser, Predictive Inference, New York, NY: Chapman and Hall, 1993.

[37] R. Kohavi, «A study of cross-validation and bootstrap for accuracy estimation and model selection,» *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence,* vol. 2, nº 12, p. 1137–1143, 1995.

[38] P. A. Devijver y J. and Kittler, Pattern Recognition: A Statistical Approach, London, GB: Prentice-Hall, 1982.

[39] F. A. Villa y J. D. Velásquez, «Regularización de Redes Cascada Correlación con Regresión en Cadena,» Quinto Congreso Colombiano de Computación (Abril 14 – 16), Cartagena, 2010.

[40] K. Hornik, M. Stinchcombe y H. White, «Multilayer feedforward networks are universal approximators,» *Neural Networks,* vol. 2, p. 359–366, 1989.

[41] G. Cybenko, «Approximation by superpositions of a sigmoidal function,» *Mathematics of Control: Signals and Systems,* vol. 2, p. 202–314, 1989.

[42] K. Funahashi, «On the approximate realization of continuous mappings by neural networks,» *Neural Neworks,* vol. 2, p. 183–192, 1989.

[43] W. Sarle, «The 19th Annual SAS Users Group Int. Conference,» de *Neural networks and statistical models*, Cary, North Carolina, 1994.

[44] T. W. S. Chow y S. Cho, Neural networks and computing: learning algorithms and applications, vol. 7, London: Imperial College Press, 2007, p. 309.

[45] M. Riedmiller y H. Braun, «A direct adaptive method for faster backpropagation learning: The RPROP algorithm,» de *Proceedings of the IEEE International Conference on Neural Networks*, IEEE Press, 1993, p. 586–591

[46] M. Riedmiller, «Advanced supervised learning in multi-layer perceptrons – from backpropagation to adaptive learning algorithms,» *Computer Standards and Interfaces,* vol. 16 , p. 265–278, 1994.

[47] D. M. Ortíz, F. A. Villa y J. D. Velásquez, «Una Comparación entre Estrategias Evolutivas y RPROP para la Estimación de Redes Neuronales,» *Avances en Sistemas e Informática,* vol. 4, nº 2, p. 135–144, 2007}

[48] I. Kaastra y M. Boyd, «Designing a neural network for forecasting financial and

economic series,» *Neurocomputing,* nº 10, pp. 215-236, 1996.

[49] U. Anders y O. Korn, «Model selection in neural networks,» *Neural Networks,* nº 12, pp. 309-323, 1999.

[50] V. Cherkassky y Y. Ma, «Another look at statistical learning theory and regularization,» *Neural Networks,* vol. 7, nº 22, pp. 958-969, 2009.

[51] C. Leung, H. Wang y J. Sum, «On the selection of weight decay parameter for faulty networks,» *IEEE Transactions on Neural Networks* , vol. 8, nº 21, pp. 1232-1244 , 2010.

[52] C. M. Bishop, Neural Networks for Pattern Recognition, New York: Oxford University Press Inc., 1994, p. 482.

[53] A. E. Hoerl y R. W. Kennard, «Ridge Regression: Biased Estimation for Nonorthogonal Problems,» *Technometrics.,* vol. 12, nº 1, p. 55–67, 1970.

[54] G. Karystinos y D. Pados, «On overfitting, generalization, and randomly expanded training sets,» *IEEE Transactions on Neural Networks,* vol. 11, nº 5, pp. 1050-1057 , 2000.

[55] M. J. Campbell y A. M. Walker, «A survey of statistical work on the mackenzie river series of annual canadian lynx trappings for the years 1821–1934 and a new analysis,» *Journal of the Royal Statistical Society,* vol. 140, nº 4, p. 411–431, 1977.

[56] T. S. Rao y M. Gabr, «An introduction to bispectral analysis and bilinear time series models,» *Lecture Notes in Statistics,* vol. 24, p. 528–535, 1984.

[57] G. Zhang, «Time Series forecasting using a hybrid ARIMA and neural network model,» *Neurocomputing,* vol. 50, p. 159–175, 2003.