



RECIBIDO EL 28 DE SEPTIEMBRE DE 2019 - ACEPTADO EL 4 DE ENERO DE 2020

ESTUDIO COMPARATIVO DE LOS ALGORITMOS BACKPROPAGATION (BP) Y MÚLTIPLE LINEAR REGRESSION (MLR) A TRAVÉS DEL ANÁLISIS ESTADÍSTICO DE DATOS APLICADO A REDES NEURONALES ARTIFICIALES

BACKPROPAGATION (BP) AND MULTIPLE LINEAR REGRESSION (MLR) ALGORITHMS COMPARATIVE STUDY OF THE THROUGH STATISTICAL DATA ANALYSIS APPLIED TO ARTIFICIAL NEURAL NETWORKS

144

Iván Mesías Hidalgo Cajo¹

Saul Yasaca Pucuna² ESPOCH

Byron Geovanny Hidalgo Cajo³

Diego Patricio Hidalgo Cajo⁴ UNACH

Nelly Baltazara Latorre Benalcázar⁵ MEC



RESUMEN

El objetivo de la investigación es comparar el algoritmo Backpropagation desarrollado por el usuario bajo software libre Java y el algoritmo Multiple Linear Regression, dicha comparación demanda del análisis estadístico decriptivo basado en redes neuronales artificiales. Se utilizó específicamente dos modelos de algoritmos de predicción aplicados a 451 patrones o registros a procesar que están repartidos en las primeras 401 filas para entrenamiento de la red neuronal y los otros 50 registros para validación y prueba, conformado por 4 variables de entrada (Height above sea level, Fall, Net fall, Flux) y 1 variable a predecir (Power turbine), para las diferentes pruebas los parámetros de entrenamiento y selección con los mejores resultados son: Architecture of the neural network, Type of scaling of data, Initial range of weight and thresholds, Learning rate and Momentum, Batched / online, Number of training epochs. Entre los resultados de comparación de los algoritmos analizados se determinó que el error en mayor iteraciones es menor que son respuestas de los 50 patrones de prueba. En el algoritmo Multiple Linear Regression la variable real es el valor de la variable a predecir, esta variable es la suministrada a predecir por el

usuario y es el valor que se predijo de la red neuronal, la variable prediction es la diferencia que se hace de la resta de los anteriores errores y se lo realiza para calcular el error y el total error es el valor mínimo a obtener que representa el error calculado de todos los datos, es decir el porcentaje de error de la red neuronal de back-propagation. Entre más bajo es este porcentaje mejor será la red, porque menor será su porcentaje de error.

PALABRAS CLAVE: Backpropagation, Multiple Linear Regression, Redes neuronales artificiales.

ABSTRACT

The objective of the research is to compare the Backpropagation algorithm developed by the user under free Java software and the Multiple Linear Regression algorithm, the comparison demands of the descriptive statistical analysis based on artificial neural network. Specifically, two models of prediction algorithms applied to 451 patterns or records to be processed that are distributed in the first 401 rows for training of the neural network and the other 50 records for validation and testing, consisting of 4 input variables (Height above sea level, Fall, Net fall, Flux) and 1 variable to predict (Power turbine), for the different tests the training and selection parameters with the best results are: Architecture of the neural network, Type of scaling of data, Initial range of weight and thresholds, Learning rate and Momentum, Batched / online, Number of training epochs. Among the comparison results of the algorithms analyzed, it was determined that the error in the highest iterations is less than the responses of the 50 test patterns. In the Multiple Linear Regression algorithm the real variable is the value of the variable to be predicted, this variable is the one provided to be predicted by the user and is the predicted value of the neural network, the variable prediction is the difference that is made the subtraction of the previous errors and it is done to calculate the

1 ORCID <https://orcid.org/0000-0002-9059-0272>
Máster Universitario en Ingeniería Informática: Seguridad Informática y Sistemas Inteligentes, ihidalgo@esPOCH.edu.ec, (593)999706462, Escuela Superior Politécnica de Chimborazo (ESPOCH), Riobamba, Ecuador.

2 ORCID <https://orcid.org/0000-0001-8851-8258>
Magister en Informática Educativa, syasaca@esPOCH.edu.ec, (593)988514153, Escuela Superior Politécnica de Chimborazo (ESPOCH), Riobamba, Ecuador.

3 ORCID <https://orcid.org/0000-0002-5526-1676>
Máster Universitario en Ingeniería Computacional y Matemática, bhidalgo@esPOCH.edu.ec, (593)984050453, Universidad Nacional de Chimborazo (UNACH), Riobamba, Ecuador.

4 ORCID <https://orcid.org/0000-0002-1937-0752>
Magister en Educación Matemática, diegohidalgomat@hotmail.es, (593)94804236, Universidad Nacional de Chimborazo (UNACH), Riobamba, Ecuador.

5 ORCID <https://orcid.org/0000-0003-2618-7814>
Licenciada en Ciencias de la Educación profesora de Educación Básica, la.torrenelly0625@gmail.com, (593)996748799, Ministerio de Educación (MEC), Riobamba, Ecuador.



error and the total error is the minimum value to be obtained which represents the calculated error of all the data, that is, the error percentage of the back-propagation neural network. The lower this percentage is, the better the network will be, because the lower the percentage of error.

KEY WORDS: Backpropagation, Multiple Linear Regression, Artificial neural networks.

1. INTRODUCCIÓN

Redes Neuronales Artificiales (NN), son inspiradas en las redes neuronales biológicas, aunque poseen otras funcionalidades y estructuras de conexión distintas a las vistas desde la perspectiva biológica. Las características principales de las NN son las siguientes:

- Auto-Organización y Adaptabilidad: utilizan algoritmos de aprendizaje adaptativo y auto-organización, por lo que ofrecen mejores posibilidades de procesamiento robusto y adaptativo.
- Procesado no Lineal: aumenta la capacidad de la red para aproximar funciones, clasificar patrones y aumenta su inmunidad frente al ruido.
- Procesado Paralelo: normalmente se usa un gran número de nodos de procesamiento, con alto nivel de interconectividad.

El elemento básico de computación (modelo de neurona) se le llama habitualmente nodo o unidad. Recibe un input desde otras unidades o de una fuente externa de datos. Cada input tiene un peso asociado w , que se va modificando en el llamado proceso de aprendizaje. Cada unidad aplica una función dada f de la suma de los inputs ponderadas mediante los pesos $y_i = \sum w_{ij}$ (Figura 1).

El resultado puede servir como output de otras unidades (Balbontín Noval, Martín Mateos, & Ruiz Reina, 2013).

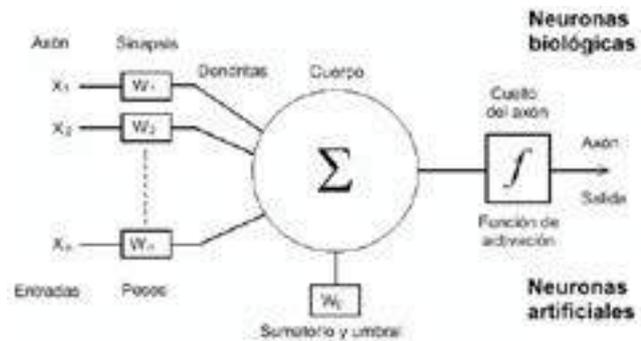


Figura 1. Modelo de Neurona

Fuente: Introducción a Redes Neuronales

1.1 Justificación/Problema

Los diversos algoritmos que se utilizan para calcular el error mínimo en las Redes Neuronales Artificiales entre la salida obtenida de la red y la óptima de un conjunto de datos con un grupo de patrones de entrenamiento, se da la necesidad de construir el algoritmo Backpropagation y realizar las respectivas pruebas con el algoritmo Multiple Linear Regression, para determinar que algoritmo se obtiene el menor error posible.

1.2 Revisión de la literatura

Las redes neuronales son modelos que intentan reproducir el comportamiento del cerebro (Castillo Ron, Cobo Ortega, & Gutiérrez, 1999). Los mismos constan de dispositivos elementales de proceso, las neuronas, a partir de ellas, se pueden generar representaciones específicas, de tal forma que un estado conjunto de ellas puede significar una letra, un número u otro objeto. Generalmente se pueden encontrar tres tipos de neuronas:

Aquellas que reciben estímulos externos relacionados con el aparato sensorial, que tomarán la información de entrada. Dicha



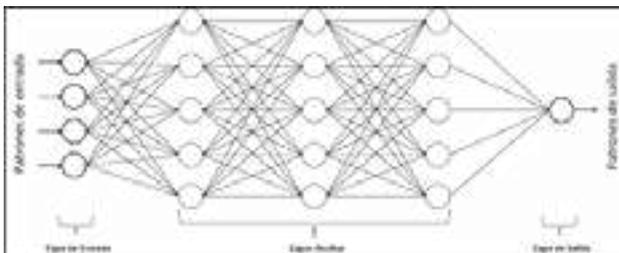
información se transmite a ciertos elementos internos que se ocupan de su procesamiento (Noriega Borge, 2017).

Es en las sinapsis y neuronas correspondientes a este segundo nivel donde se genera cualquier tipo de representación interna de información. Como no tienen relación directa con la información de entrada ni con la salida, estos elementos se denominan unidades ocultas (Borquez Ramírez & Villanueva Ramos, 2006).

Una vez finalizado el período de procesamiento, la información llega a las unidades de salida, cuya misión es dar la respuesta al sistema.

A continuación, se puede observar en la Figura 2, el esquema de una red neuronal (Hilera & Martínez, 2000):

Figura 2. Esquema de una red neuronal



Fuente: Redes Neuronales Artificiales

El esquema está constituido por neuronas interconectadas y arregladas en tres capas (esto último puede variar). Los datos ingresan por medio de la “capa de entrada”, pasan a través de la “capa oculta” y salen por la “capa de salida”. Cabe mencionar que la capa oculta puede estar constituida por varias capas a su vez (Valencia Reyes, Yáñez Márquez, & Sánchez Fernán, 2006).

Cada **entrada** a la red tiene un peso asociado w , que se va modificando en el llamado proceso de aprendizaje (Galván & Valls, 2010). Esta es la fórmula:

La función de activación se elige de acuerdo con la tarea realizada por la neurona (Figura 3).

Entre las más comunes dentro del campo de las RNAs podemos destacar:

Figura 3. Función de Activación de la Neurona

| | Función | Range | Gráfica |
|-----------|--|-----------------------------|---------|
| Identidad | $y = x$ | $[-\infty, +\infty]$ | |
| Escalón | $y = \text{sign}(x)$ $y = H(x)$ | $\{-1, +1\}$ $\{0, +1\}$ | |
| Sigmoides | $y = \frac{1}{1+e^{-x}}$ $y = \text{tanh}(x)$ | $[0, +1]$ $[-1, +1]$ | |
| Gausiana | $y = Ae^{-bx^2}$ | $[0, +1]$ | |
| Senoidal | $y = A \text{sen}(ax + \phi)$ | $[-1, +1]$ | |

Fuente: Redes Neuronales una visión superficial

La **Fase de Entrenamiento**, es un conjunto de datos o patrones para determinar los pesos que definen el modelo de la red neuronal (Buendía-Rodríguez, Vargas-Pérez, Leyva-Ovalle, & Terrazas-Domínguez, 2012). Se calculan de manera iterativa con el objetivo de minimizar y el error cometido entre la salida obtenida por la red neuronal y la salida deseada, es decir la salida por la red es 1 respuesta que se genera de los patrones de entrada. Y la salida deseada es otra respuesta que es la variable a predecir (Match, 2010).

La **Fase de Prueba o Validación**, es un segundo grupo de datos diferentes a los de entrenamiento, que permiten controlar el proceso de aprendizaje. En la fase de entrenamiento van a ver ajustes, que son los pesos en multiplicación con los patrones, y es donde se presentan particularidades cuando la red se asocia a los valores porque va perdiendo su habilidad de generar aprendizaje en casos nuevos, como si se acostumbrara a los datos que van entrando y luego cuando se agregan otros datos no los aprende. Entonces esta fase es para hacer el control de ese proceso y así



poder que la red aprenda cualquier dato que se ingresen (Gestal Pose , 2009).

El **Algoritmo Backpropagation (BP)**, es tan solo un *Entrenamiento Supervisado*, en este algoritmo de Backpropagation se considera una etapa de funcionamiento donde se presenta, ante la red entrenada, un **patrón de entrada** y éste se transmite a través de las sucesivas capas de neuronas hasta obtener una **salida** y, luego una etapa de entrenamiento o aprendizaje donde se modifican los pesos de la red de manera que coincida la salida deseada con la salida obtenida por la red (Piloto Rodríguez, 2017).

La constante (**learning rate**) se coloca para acelerar o ralentizar el aprendizaje, si es necesario varia de 0 a 1. Así que se lo puede variar entre esos valores (Valencia Reyes, Yáñez Márquez, & Sánchez Fernán, 2006)

Regresión lineal o **ajuste lineal**, es un modelo matemático usado para aproximar la relación de dependencia entre una variable dependiente Y_t , las variables independientes X_i y un término aleatorio ϵ . Este modelo puede ser expresado como:

$$Y_t = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p + \epsilon$$

$Y_t = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p + \epsilon$

donde:

Y_t : variable dependiente, explicada o regresando.

X_1, X_2, \dots, X_p : variables explicativas, independientes o regresores.

$\beta_0, \beta_1, \beta_2, \dots, \beta_p$: parámetros, miden la influencia

que las variables explicativas tienen sobre el regrediendo.

Donde β_0 es la intersección o término “constante”, las β_i son los parámetros respectivos a cada variable independiente, y p es el número de parámetros independientes a tener en cuenta en la regresión. La regresión lineal puede ser contrastada con la regresión no lineal (Montero Granados , 2016).

1.3 Propósito

Entrenar una red neuronal para minimizar el error entre la salida obtenida por la red y la salida deseada por el usuario ante la presentación de un conjunto de patrones denominado grupo de entrenamiento

2. MÉTODO

Tipos de investigación

Investigación documental: Búsquedas en diversas fuentes de información como son: bases de datos digitales, libros, revistas, manuales, internet, entre otros.

Método

Científico: Es un estudio sistemático, lógico y organizado de la proposición planteada para adquirir conocimientos y brindar una solución.

Descriptivo: Se realizó un estudio descriptivo que consistió en llevar a conocer situaciones relevantes a través de la descripción de las variables de investigación para exponer de manera cuidadosa los resultados a fin de extraer generalizaciones significativas.



2.1 Instrumentos y materiales

- Lenguaje de Programación Java.
- Entorno de desarrollo integrado libre Netbeans.
- Microsoft Office Excel 2016.
- Sistema Operativo Windows 10 Professional.

2.2 Procedimiento

Los datos del muestreo que se usaron son 451 patrones o registros, que están repartidos en los primeros 401 registros para entrenamiento de la red neuronal y los últimos 50 patrones son para validación y prueba. Cada patrón está conformado por 5 columnas: 4 variables de entrada y 1 variable a predecir.

Funcionamiento del algoritmo:

1. Calcular el error de la neurona de salida
2. Cambiar los pesos de la capa de salida
3. Calcular (retro-propaga) los errores de la capa oculta
4. Cambiar los pesos de la capa oculta

Código fuente desarrollado en Java (Figura 4):

Declaración e inicialización de variables (parámetros):

El primera variable es una constante cuyo valor es 4 y representa el NÚMERO DE NEURONAS DE ENTRADAS. La segunda variable es el NÚMERO DE NEURONAS QUE HAY EN LA CAPA OCULTA, puede cambiar de valor y sirve para las respectivas pruebas. La tercera variable su valor es constante con una valor de 1 y es el NÚMERO DE SALIDAS. La cuarta variable es el NÚMERO DE CAPAS OCULTAS, este valor puede cambiar. La quinta variable es LEARNING RATE, varía en un rango de 0 a 1. El número 1000 son los EPOCH, representa el número total de iteraciones que se ejecutan en la etapa de entrenamiento y prueba, el resto del código corresponde a la asignación del número de neuronas, tipo, patrones de entrenamiento y pruebas, otras variables del programa, y en la parte final encontramos la declaración e invocación a la clase Backpropagation (proceso de la red neuronal).

Figura 4. Código fuente, declaración de variables

```
public class BackPropagation {

    static int NumNeuronCapaEntrada = 4;
    static int NumNeuronCapaOculta = 3;
    static int NumNeuronCapaSalida = 1;
    static int NumCapasOcultas = 5;
    static double LearningRate = 0.4/1000;
    static int epoch = 1000;

    static int Neuronas = 451;
    static int Tipo = 5;
    static int PatronesEntrenamiento = 401;
    static int PatronesPrueba = 50;

    static double DatosEntrada[][] = new double[Neuronas][Tipo];
    static double DatosEntradaOculta[][] = new double[Neuronas][Tipo];

    static double DatosEntrenamiento[][] = new double[PatronesEntrenamiento][NumNeuronCapaEntrada];
    static double DatosEntrenamientoOculta[][] = new double[PatronesEntrenamiento];

    static double ValidacionPatrones[][] = new double[PatronesPrueba][NumNeuronCapaEntrada];
    static double ValidacionPrediccionOculta[][] = new double[PatronesPrueba];
    static double DatosPrediccion[] = new double[PatronesPrueba];
    static double DatosPrediccionNormalizado[] = new double[PatronesPrueba];

    public static void main(String[] args) {

        BackPropagation red = new BackPropagation(NumNeuronCapaEntrada, NumNeuronCapaOculta, NumNeuronCapaSalida,
            NumCapasOcultas, LearningRate); //BackPropagation red = new BackPropagation(4, 3, 1, 5, 0.4);

        red.InicializaEntrada();
    }
}
```



2.3 Resultados

CASO PRÁCTICO CON 1000 ITERACIONES

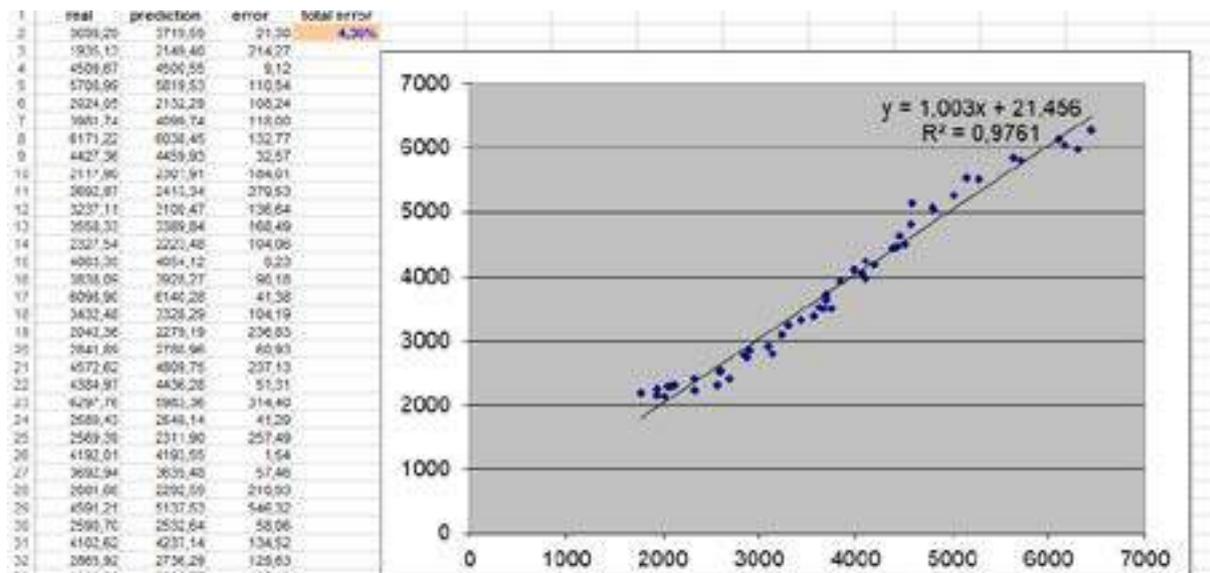
Para el análisis de datos con el algoritmo Backpropagation se realizó 1000 iteraciones (epochs) con 4 neuronas de entrada, 5 neuronas en la capa oculta, 1 salida, 5 capas ocultas y 0.1 del valor de la tasa de aprendizaje, ya que este valor acepta solamente entre el rango de 0 a 1, se obtiene un error mínimo de **4,35%** en un tiempo de 10 segundos (Figura 5).



Figura 5. Análisis de datos con 1000 iteraciones según algoritmo Backpropagation.

Por otro lado, al realizar el análisis de datos con el algoritmo Multiple Linear Regression con

Figura 6. Análisis de datos con 1000 iteraciones según algoritmo Multiple Linear Regression



1000 iteraciones (epochs) con 4 neuronas de entrada, 5 neuronas en la capa oculta, 1 salida, 5 capas ocultas y 0.1 del valor de la tasa de aprendizaje, nos muestra los datos a predecir y el error mínimo a obtener que es de **4,36%**, según la Figura 6, se observa que los datos son muy cercanos, no se dispersan demasiado y se obtiene un valor mínimo de error.

real es el valor de la variable a predecir, esta variable es suministrada por el usuario.

prediction es el resultado de la ejecución del algoritmo.

total error: es el valor mínimo a obtener y es el error calculado de todos los datos correspondiente a los 4 patrones, es decir el porcentaje de error de la red neuronal de Back-Propagation. Entre más bajo es este porcentaje mejor será la red, porque menor será su porcentaje de error.



CASO PRÁCTICO CON 5 ITECCIONES

Análisis de datos con el algoritmo Backpropagation con 5 iteraciones (epochs) con 4 neuronas de entrada, 5 neuronas en la

capa oculta, 1 salida, 5 capas ocultas y 0.1 del valor de la tasa de aprendizaje, ya que este valor acepta solamente entre el rango de 0 a 1, se obtiene un error mínimo de **28,05%** en un tiempo de 0 segundos (Figura 7).

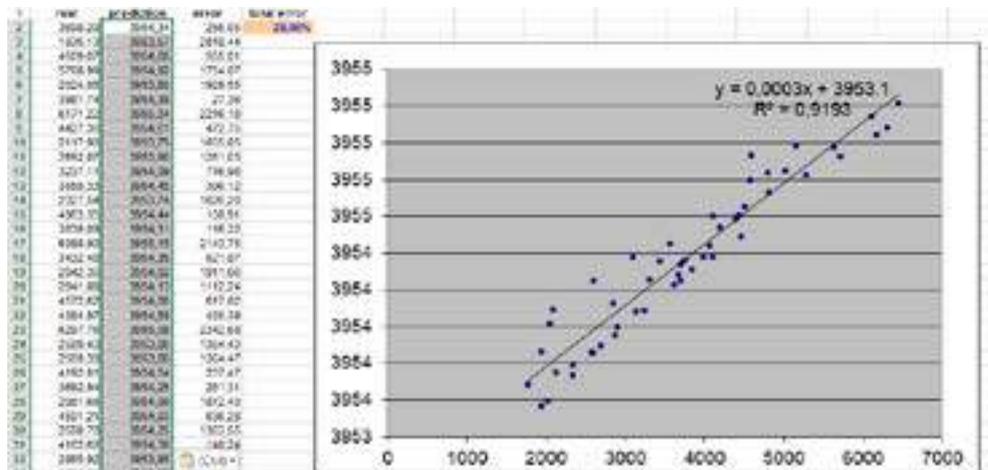
Figura 7. Análisis de datos con 5 iteraciones según algoritmo Backpropagation



Análisis de datos con el algoritmo Multiple Linear Regression con 5 iteraciones (epochs) con 4 neuronas de entrada, 5 neuronas en la capa oculta, 1 salida, 5 capas ocultas y 0.1 del valor de la tasa de aprendizaje, nos muestra los datos

a predecir y el error mínimo a obtener que es de **28,06%** según la Figura 8, se observa que los datos son muy dispersos y se obtiene un valor mínimo de error.

Figura 8. Análisis de datos con 5 iteraciones según algoritmo Multiple Linear Regression



ANÁLISIS ESPECÍFICO.

Al realizar el análisis del primer registro que es producto de las 1000 iteraciones con un valor real de 3698.29, obtenemos un valor de predicción de 3719,59, y un error de 21,30

Mientras que en el análisis del primer registro con 5 iteraciones obtenemos el valor real de 3698.29; valor de predicción de 3954,34; error de 256,05

Por lo tanto, el error en mayor iteraciones es menor, es decir todas estas filas son las respuestas de los 50 Patrones de prueba.

CONCLUSIÓN

- Al tener un menor número transiciones (epochs) los resultados no son buenos, es decir la red no aprende.
- Al ingresar una Tasa de aprendizaje



mínima, sugerido de 0,1 la red aprende mejor.

- Con un valor alto del número de neuronas de las capas ocultas y capas ocultas con una tasa de aprendizaje mínimo o alto los resultados no son buenos.
- Con los valores obtenidos en la Regresión lineal, podemos decir si tiende R a 1 es muy bueno y si tiende a 0 es malo.
- Se ha logrado obtener un error mínimo con un número de neuronas de la capa oculta bajo, bajo número de capas ocultas y la tasa de aprendizaje mínima.
- Existen limitantes en la capacidad de procesamiento del computador, ya que la funcionalidad del cerebro se hace de forma paralela mientras que la computacional está limitada a ser serial. El cerebro tiene la capacidad de tener un conjunto de percepciones al mismo tiempo, cosa que no ocurre con los computadores.
- Existen diferentes métodos de aprendizaje con redes neuronales, perceptrones, propagación posterior, aprendizaje bayesiano y los deductivos, que se diferencia de acuerdo con el método que utilizan para la obtención de los estados finales o de salida los mismos que podrían considerarse en estudios posteriores.

REFERENCIAS BIBLIOGRÁFICAS

Balbotín Noval, D., Martín Mateos, F., & Ruiz Reina, J. (2013). *Introducción a Redes Neuronales*. Madrid.

Borquez Ramírez, P., & Villanueva Ramos, J. (2006). *Predicción de signo a tres semanas de la acción caterpillar con redes neuronales*. Santiago de Chile.

Buendía-Rodríguez, E., Vargas-Pérez, E., Leyva-Ovalle, Á., & Terrazas-

Domínguez, S. (2012). *Aplicación de redes neuronales artificiales y técnicas SIG para la predicción de coberturas forestales*. México.

Castillo Ron, E., Cobo Ortega, Á., & Gutiérrez, J. (1999). *Introducción a la Redes Funcionales con Aplicaciones*. Paraninfo.

Galván, I., & Valls, J. (2010). *Redes de Neuronas Artificiales*. Madrid.

Gestal Pose, M. (2009). *Introducción a las Redes de Neuronas Artificiales*. La Coruña.

Hilera, J., & Martínez, V. (2000). *Redes Neuronales Artificiales*. Alfaomega-Rama.

Matich, D. (2010). *Redes Neuronales: Conceptos Básicos y*. Rosario.

Montero Granados, R. (2016). *Modelos de regresión lineal múltipl.* Granada.

Noriega Borge, M. (2017). *Fisiología Humana*. Cantabria.

Piloto Rodríguez, R. (2017). *Redes Neuronales Artificiales. Conceptos básicos y algunas aplicaciones en Energía*. La Habana.

Valencia Reyes, M., Yáñez Márquez, C., & Sánchez Fernán, L. (2006). *Algoritmo Backpropagation para Redes Neuronales: conceptos y aplicaciones*. México.