UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

TESIS:

"DISEÑO DE UN SISTEMA DE COMPENSACIÓN DE ERROR DE PESAJE DE FAJA DE LLENADO DE MINERAL USANDO REDES NEURONALES"

PARA OBTENER EL GRADO ACADÉMICO DE MAESTRO EN CIENCIAS EN INGENIERÍA ELECTRÓNICA CON MENCIÓN EN AUTOMÁTICA E INSTRUMENTACIÓN

ELABORADO POR:

Ing. NELSON ESTEBAN CHAMBI QUIROZ

ASESOR:

M. Sc. STEVE NUÑEZ ESCOBAR

LIMA – PERÚ

2023

DEDICATORIA

A Dios, por toda la bendición que me brindo durante el desarrollo de esta tesis, a mi madre Olinda que este en el cielo que siempre me brindo su apoyo cuando estaba en vida sin ella no lo hubiera logrado. A mi Padre que día a día me brinda palabras de aliento y me motivan constantemente. A mis hijas Victoria y Micaela que son mi razón de levantarme y esforzarme cada día son mi principal motivación. A mis hermanos que día a día me impulsan a seguir adelante. A la memoria de mi abuelita Victoria.

AGRADECIMIENTOS

A mi asesor, Msc. Steve Núñez, por su apoyo y orientación, consejos y paciencia brindado durante el desarrollo de la elaboración de la presente tesis.

A mi alma mater la Universidad Nacional de Ingeniería por aceptar ser parte de ella y poder estudiar y culminar mi carrera. A los docentes de la maestría de la Facultad de Ingeniería eléctrica y electrónica por compartir sus conocimientos, tiempo y orientación.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

RESUMEN

En la presente tesis se desarrolla un diseño de un sistema de compensación de error usando redes neuronales para compensar el error de pesaje de faja de llenado de mineral, el cual está por encima del 10%.

Para el desarrollo del método de sistema de compensación de error, se usará redes neuronales, donde se va lograr calcular los pesos sinápticos, bias y la arquitectura de la red neuronal, estos valores se calculan con el algoritmo de back propagation durante el entrenamiento de le red neuronal, así mismo los parámetros de la red y luego serán validados para corroborar la compensación de error.

Para el entrenamiento de la red neuronal se plantea el modelo matemático, valores de entrada, salida y parámetros de la red neuronal para luego ser entrenados y analizar la compensación de error para cada entrenamiento con la finalidad de seleccionar la óptima red con pesos sinápticos y bias. Para lograr ello se obtuvo datos de campo que fueron recopilados de una faja de llenado de mineral el cual se usó el 70% de datos para el entrenamiento y el 30% para la validación, adicional a ello los datos se tomaron de manera aleatoria, para obtener los resultados se usa el software Matlab.

Para compensar el error se diseñó una red neuronal de 3 entradas, 1 salida y 2 capas ocultas con 5 neuronas en cada capa oculta, esta red neuronal se va a encargar del proceso de compensar el error de pesaje. Así mismo este modelo propuesto reduce el error a un 1.65%, para lograr ello se optimiza las capas ocultas y neuronas por cada capa.

La optimización del diseño de la red neuronal logra obtener buenos resultados que demostraron que la red diseñada realizo la compensación de error y por ende logra disminuir el margen de error en el peso final del mineral de 10% a 1.65%.

ABSTRACT

In the present tesis a design of an error compensation system is developed using Neural Networks to compensate for the weighing error of the mineral filling belt, which is above 10%.

For the development of the error compensation system method, the Neural Networks method will be used, where it will be possible to calculate the synaptic weights, bias and the architecture of the Neural Network, these values are calculated with the back propagation algorithm during the training of the Neural Networks, as well as the parameters of the Network and then they will be validated to corroborate the error compensation.

For the training of the Neural Network, the mathematical model, input and output values and parameters of the Neural Network are proposed to later be trained and analyze the error compensation for each training in order to select the optimal Network with synaptic weights and bias. To achieve this, field data was obtained that was collected from a mineral filling belt, which used 70% of the data for training and 30% for validation in addition to this the data was taken randomly and to obtain the results the Matlab software is used.

To compensate for the error, a Neural Network with 3 inputs, 1 output and 2 hidden layers with 5 neurons in each hidden layer was designed, this Neural Network will be in charge of the process of compensating for the weighing error. Likewise this proposed model reduces the error to 1.65% to achieve this the hidden layers and neurons for each layer are optimized.

The optimization of the design of the Neural Network manages to obtain good results that showed that the designed network made the error compensation and therefore manages to reduce the margin of error in the final weight of the mineral from 10% to 1.65%.

INTRODUCCIÓN

El uso de la red neuronal se ha convertido en una técnica útil para el desarrollo de predicción de datos en el campo industrial, dicha técnica permite encontrar solución al problema usando el algoritmo de back propagation en el entrenamiento de la red neuronal. Sin embargo, el campo de aplicación de la técnica de la red neuronal es muy amplia, así como también los diferentes tipos de redes neuronales

En el primer capítulo, se presenta la problemática con respecto a la necesidad de reducir el margen de error de pesaje, así mismo se presenta trabajos de investigación sobre el tema relacionados con el trabajo propuesto, la formulación del problema, así como también los objetivos de la tesis.

En el segundo capítulo, se detalla los conceptos teóricos relacionados con la presente tesis, así mismo se realiza la descripción de la dinámica del sistema de llenado de mineral y sus componentes.

En el tercer capítulo comprende el desarrollo de la metodología basada en redes neuronales, el cual consiste en calcular los pesos sinápticos óptimos y sesgos el cual son obtenidos del entrenamiento, así mismo se diseña la arquitectura de la red neuronal. La metodología propuesta es entrenada y validada en Matlab el cual se ingresan los valores de entrada y parámetros de la red neuronal tales como, función de activación, tasa de aprendizaje y numero de épocas y finalmente serán validados con valores de entrada.

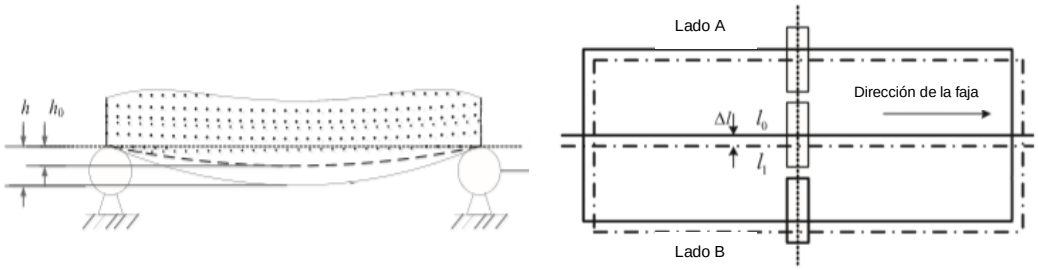
En el cuarto capítulo se realiza el análisis y comparación de resultados obtenidos durante el entrenamiento de la red neuronal, por ende, se compara y valida la metodología propuesta en el capítulo anterior. Para el diseño de arquitectura de la red neuronal se plantean diferentes arquitecturas con la finalidad de lograr la reducción de margen de error de pesaje.

CAPITULO I
ANTECEDENTES Y DESCRIPCIÓN DEL PROBLEMA

1.1. Antecedentes bibliográficos

En [1] los autores propusieron el método de Redes Neuronales para compensar el error de pesaje en una banda transportadora, planteando la técnica de backpropagation, la que sirve para saber cuál es el error de cada nodo, pudiendo ajustarlo para disminuirlo. Durante la selección de parámetros consideraron a la tensión o pandeo de la banda (Figura 1.1a) como la fuente principal de error de pesaje. Otro parámetro que consideran es la desviación de funcionamiento de la banda el cual lo define como el desplazamiento entre la línea central de la banda y línea central del rodillo (Figura 1.1b). La principal conclusión es que con el método propuesto se logra validar que el error de pesaje es inferior al 0.5%. Sin embargo, durante las pruebas no se considera la temperatura y humedad de la materia como influencia en el error de pesaje.

Este artículo se considera como referente porque los autores plantean el des alineamiento y pandeo de la faja para el cálculo del margen de error. Así mismo se va considerar un sensor de ángulo dado que es una faja con pendiente y variables como la humedad.



(a) Variación de Tensión de Banda (b) Desviación de funcionamiento banda

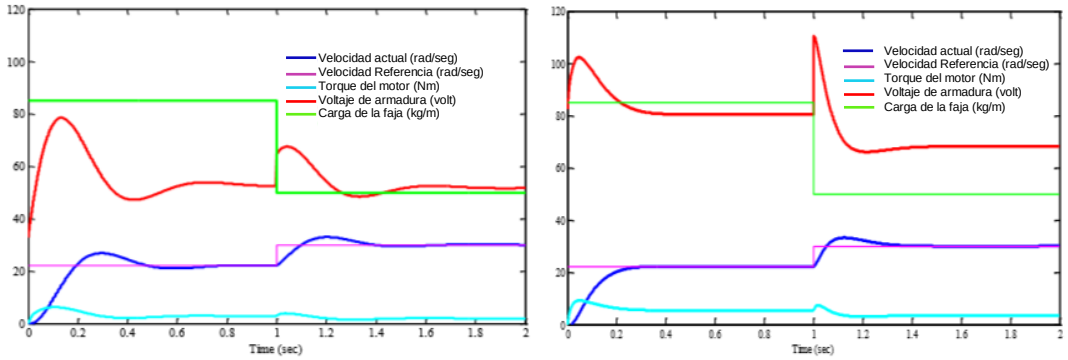
Figura 1.1 Parámetros que influyen en el pesaje

(Fuente: Referencia [1])

En [20] los autores propusieron el desarrollo de un control Lógico fuzzi PI, para controlar el pesaje de la faja transportadora. Plantean dos técnicas de control, una de ellas es el desarrollo de un control PI donde dicho control no puede controlar los parámetros de variación de perturbaciones de carga, velocidades y ello hace que dicho control no

proporciona un rendimiento óptimo. Otra técnica de control empleada es Lógica Fuzzi tipo PI en donde se puede ver que dicho control es más robusto a perturbaciones y variaciones del proceso. La conclusión más relevante es que al aplicar control tipo Lógico Fuzzi tipo PI el pesaje de la faja transportadora mejora en un 32.2% y la respuesta a la carga y velocidad es mucho más estable Figura 1.2

Este artículo es considerado importante dado los autores están haciendo el cálculo de un control para eliminar las perturbaciones, ello se tomará en cuenta para identificar las perturbaciones que hay en la faja de llenado de mineral.



a) Control PI b) Control Lógico Fuzzi con PI

Figura 1.2 Comparación de repuesta de carga de faja y velocidad

(Fuente: Referencia [20])

Por otra parte, en [16] los autores desarrollaron un sistema de pesaje dinámico usando el modelo FIR, respuesta infinita al pulso, de alta precisión para reducir el ruido y controlar la precisión pesaje. Plantean dos enfoques de pesaje dinámico el primero es basado en el modelo donde la señal medida se modela como una respuesta de un sistema FIR con parámetros desconocidos a una excitación tipo pulso. El segundo enfoque es el filtrado sin modelos para el pesaje dinámico, pero incluye filtros digitales diseñados para reducir el ruido y atenuar la parte oscilatoria, la ventaja del primer enfoque es que no necesita ajuste previo para diferentes condiciones de operación. Para obtener el orden del modelo los autores plantean tres criterios, criterio de información Akaike, criterio de información bayesiano y método de validación cruzada. El criterio de validación cruzada posee una ventaja sobre los otros 2 criterios debido a que este verifica el rendimiento real del modelo creado, mientras que los criterios de información predicen el orden. Como conclusión principal es que el filtrado basado en el modelo FIR garantizó una precisión de cuatro veces

mayor que otras soluciones. La desventaja es que, si se quiere usar para otro tipo de carga o material, se tendrá que calcular el orden del modelo.

Este artículo se considera importante porque se va considerar el análisis planteado por los autores para reducir el ruido dicha técnica se empleará para disminuir el ruido y otros factores.

Por su parte los autores de [5], realizaron el estudio de la distribución de la carga de pesaje, para disminuir el error de pesaje. Plantearon el análisis estático mediante el diagrama de fuerzas y el momento resultante para diferentes posiciones de ubicación en la balanza de la carga a pesar, así mismo se observó que la carga varia, mientras más lejos se ponga de la posición normal el margen de error aumenta. Para la prueba los autores [5] utilizaron dos tipos de balanzas de diferente diseño. La conclusión principal es la importancia de la posición de la carga mientras más centrada se encuentre con la celda de carga el margen de error disminuye. Dicha prueba servirá de apoyo para considerar la distribución de carga a pesar en la faja transportadora.

Este artículo es considerado muy importante dado que los autores realizan el cálculo para que la carga este distribuida correctamente dicho análisis será de base para el análisis cuando al se alimenta a la faja transportadora.

En [11] los autores plantearon un nuevo enfoque usando una red neuronal optima con un multiplicador de Lagrange para compensar el error no lineal en una celda de carga. Para entrenar la red neuronal se considera el principio de la propiedad de aumento monotónico de la función de entrada-salida de la celda de carga, seguidamente crear la función aumentada de Lagrange con un multiplicador y un factor de penalización. Mientras que el factor de penalización aumenta el rendimiento de la red neuronal y el error lineal disminuye. Como conclusión principal cuando se compara el error con el método Red Neuronal y Multiplicador Lagrange con el método de inducción de datos convencional el método Red Neuronal presenta menor error, pero cuando no hay la cantidad de entrenamientos el método convencional resulta ser mejor y con menos error. Cabe resaltar que dicho método presenta error alto cuando se requiere evaluar celdas de gran escala en donde se requiere equipo de calibración.

Se considera el artículo muy importante, ya que se tomará como base el modelo planteado por el autor para corregir la no linealidad de la celda de carga.

1.2. Descripción de la realidad problemática

Durante los años 2005 al 2014 según osinerming, 2017, la minería había logrado un incremento considerable de producción de cobre en un 4-5 % y ello ha generado que la extracción y exportación del mineral aumente a gran escala, tanto así que dicha exportación se realizaba en las bodegas de los buques graneleros de transporte marítimo cubriendo todo el volumen del mismo y el llenado se realizaba mediante fajas transportadoras. En los próximos años a raíz de la caída del precio del mineral en un 28% promedio anual, según [24] y [27], baja el nivel de exportación de mineral por ende el costo de exportar en bodegas se vuelve costoso. En algunos casos no se cubría el volumen completo de 30,000 TN y nace la necesidad de colocar mamparas o divisiones para llevar otros tipos de minerales y cubrir el volumen requerido aun así el costo de exportación era elevado. Ante esta necesidad se implementa la exportación de mineral en contenedor, cumpliendo el protocolo de identificación DIN ISO 6343.

Esta nueva forma de exportación resulta mucho más económica que la anterior, el llenado del mineral al contenedor se realizaba mediante equipos de menor capacidad el cual generaba perdidas en tiempo debido a que el ciclo de trabajo era muy largo, otra forma de llenado es con faja transportadora el cual lo hace en menor tiempo, pero no cuenta con la precisión de pesaje que exige el fabricante de contenedor y la norma ISO 668 y requiere del uso de equipo liviano y balanza externa para cumplir el peso adecuado.

Debido al incremento de la exportación de mineral en contenedores y la necesidad de disminuir el error de pesaje, el cual se encuentra en 12% al momento del llenado, se hace necesario disminuir el error de pesaje en 5%.

El enfoque de la presenta tesis es detectar el problema usando la técnica de Redes Neuronales para compensar el error de pesaje.

1.3. Formulación del problema

En la presente tesis se pretende disminuir el margen de error de pesaje con el diseño de un sistema de compensación, aplicando Redes Neuronales con la finalidad de disminuir el margen de error de pesaje en la faja de llenado de mineral.

1.4. Justificación e importancia de la investigación

La presente tesis surge de la necesidad de disminuir el margen de error teniendo en cuenta las variables del proceso y optimizar los recursos y tiempos durante el ciclo del proceso, el cual va generar beneficio al proceso.

La investigación desde el punto de vista técnico aporta una técnica de entrenamiento que va permitir compensar el error de pesaje de mineral, a mayor capacidad de entrenamiento será menor el error de pesaje.

Por otra parte, desde el punto de vista económico un mal llenado podría ocasionar daños irreparables en el contenedor, ocasionando derrames del mineral durante su traslado. Así mismo es de uso industrial y aplicado a empresas del rubro minero.

1.5. Objetivos

1.5.1. General

Diseñar un sistema de compensación de error usando Redes Neuronales para reducir el margen de error de pesaje

1.5.2. Específico

Los objetivos específicos de esta tesis son:

1.- Determinar el algoritmo para reducción del margen de error de pesaje por compensación empleando técnica de Redes Neuronales.

2.- Ejecutar los entrenamientos necesarios para la reducción de margen de error de pesaje por compensación.

3.- Simular dicho algoritmo mediante el programa Matlab.

1.6. Hipótesis

1.6.1 Principal:

La aplicación de la metodología de Redes Neuronales permitirá disminuir el error de pesaje en una faja transportadora de mineral lo cual se logrará con el entrenamiento de una red neuronal.

1.6.2. Secundaria:

La aplicación de la Red Neuronal permitirá generar un modelo óptimo para disminuir el error de pesaje en la faja de llenado de mineral.

A partir del entrenamiento de la Red Neuronal es posible disminuir el error de pesaje en la faja de llenado de mineral validando el peso actual con el requerido.

1.7. Variables e indicadores

1.7.1. Variables independientes

La variable independiente es el uso de la técnica de Red Neuronal, el cual me permitirá controlar el peso del mineral.

1.7.2. Variables dependientes

Reducción de margen de error

1.7.3. Variables intervinientes

Velocidad de faja de llenado

Temperatura y humedad del mineral

1.7.4. Indicadores

El indicador de la variable independiente es la cantidad de capas y numero de entrenamiento a realizar para en la red Neuronal, con la finalidad de reducir el margen de error.

El indicador de la variable dependiente es la variación del peso medido y el peso real del material.

1.8. Unidad de análisis

En la investigación se concentra en el análisis del sistema de pesaje para poder compensar el error de pesaje mediante el uso de red neuronal, se va considerar todos los parámetros que influyen en el entrenamiento.

1.9. Tipo y nivel de investigación

El presente trabajo de investigación plantea una metodología de nivel científico, dado que se considera una red neuronal basado en un entrenamiento óptimo y así lograr compensar el error de pesaje, se está considerando realizar el entrenamiento con valores reales. Aplicado a la industria así mismo el presente trabajo se realizó de manera experimental transaccional.

El presente trabajo corresponde a una tesis de maestría, debido a que aporta la solución al problema propuesto líneas arriba y utilizando el Software Matlab para realizar el entrenamiento de la red neuronal.

1.10. Periodo de análisis

La información que se usara como datos de entrada en el presente estudio, como velocidad, humedad del material y peso, serán obtenidos en un periodo continúo de 11 horas de operación y se considera 02 meses de operación continua. Sin embargo, ello no implica que se pueda prolongar el periodo de análisis, así mismo se considera operación continua sin paradas de mantenimiento preventivo.

Los valores obtenidos van servir de apoyo para el entrenamiento de la red neuronal y poder realizar el cálculo de numero de capas y para definir valores predominantes en el diseño del algoritmo.

1.11. Fuentes de información e instrumentos utilizados

Las fuentes de información para el presente trabajo son fuentes de investigación científica, artículos y tesis.

Por otro lado, los instrumentos que se utilizaran a nivel de Hardware una Laptop, a nivel de software el programa de Matlab R2020a.

1.12. Técnicas de recolección y procesamiento de datos

La información se realizará con la validación de datos y para generalizar se usará Excel, todo ello basado en pruebas experimentales y datos reales los cuales serán tomados en fajas transportadoras de empresas del rubro.

Con la información recolectada se procederá a realizar el entrenamiento adecuado y desarrollar la red neuronal, para luego verificar con Matlab R2020a y/o Python 3.10.

CAPITULO II
MARCO TEÓRICO Y MARCO CONCEPTUAL

2.1. Sistema de Faja de Llenado

2.1.1 Dinámica del circuito de llenado de mineral

El circuito de llenado se puede realizar de diferentes maneras, sin embargo, en la presenta tesis se realizará el estudio de llenado de mineral mediante una faja transportadora. La faja transportadora tiene como objetivo principal el traslado y pesaje de mineral. Así mismo el mineral es llenado a una tolva mediante un cargador frontal previa mezcla y/o regulación de ley de mineral. Dicho mineral es abastecido a la faja de manera uniforme para que sea trasladado por la faja hacia el contenedor, Por otra parte, el mineral que es trasladado pasa por una celda de pesaje que se encuentra en la parte intermedia de la faja de llenado de mineral, con la finalidad de contabilizar el pesaje final. Tal como se muestra en la Figura 2.1

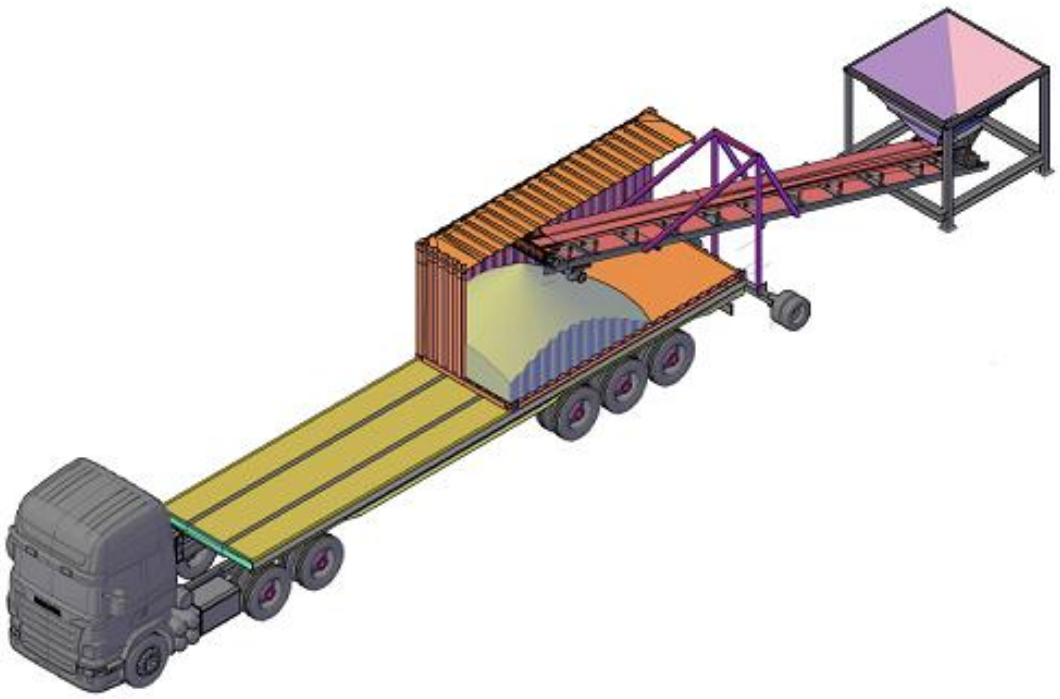


Figura 2.1 Circuito de llenado de mineral

2.1.2 Característica física del sistema de Faja de llenado de mineral

El sistema de faja de llenado de mineral consta de una tolva el cual dosifica el mineral a una velocidad constante, dicho mineral es pesado por una celda de carga que se encuentra en debajo de la faja y transportado por la faja de llenado hacia los contenedores el cual tiene una capacidad de 28TN. Sin embargo, se considera importante la altura de carga del mineral a transportador ya que esta debe mantenerse constante. La variación de la altura de mineral a transportar es considerada unas de las variables que genera un error de pesaje.

2.1.3 Altura de llenado de Mineral

Para poder lograr la altura de llenado del mineral dentro del contenedor (h), la faja debe tener una pendiente (α) entre 20° a 30°, tal como se puede apreciar en la figura 2.2. La pendiente de la faja va depender de la altura del contenedor. Se considera importante la altura de llenado para poder optimizar la capacidad de carga del contenedor.

Figura 2.2 Altura de llenado de mineral

(Fuente: Elaboración propia)

2.1.4 Sistema de Pesaje

De acuerdo con la referencia [32] el sistema de pesaje es instalado en las fajas transportadoras con la finalidad de realizar el pesaje de manera correcta. Sin embargo, dicho sistema está diseñado para soportar fuerzas verticales el cual es generada por el material y estas fuerzas son proporcionales a una señal que es procesado por el

controlador. Este método de pesaje optimiza el tiempo del proceso evitando paradas. Así mismo en [1] se define el cálculo del pesaje total mediante la integración (2.1). Es decir, cuando el material se transporta en un tiempo t a una velocidad constante y una distancia L, el valor de la integración de calcula con la siguiente ecuación.

$$I = \int_0^T \frac{W_L(t) * v(t)}{L} dt \quad (2.1)$$

Donde:

w: carga de material

v: velocidad.

L: longitud

El sistema de pesaje consta de 02 celdas de carga el cual van instaladas en forma paralela en una estructura tipo puente, ver figura 2.3. Dicha báscula realiza el pesaje de una cantidad de material que es transportado por la faja, el material ejerce una fuerza sobre las celdas de carga por medio de los rodillos el cual están unidos a la estructura del puente. El pesaje toma en cuenta la fuerza por peso sobre la plataforma para realizar el cálculo de la longitud efectiva.

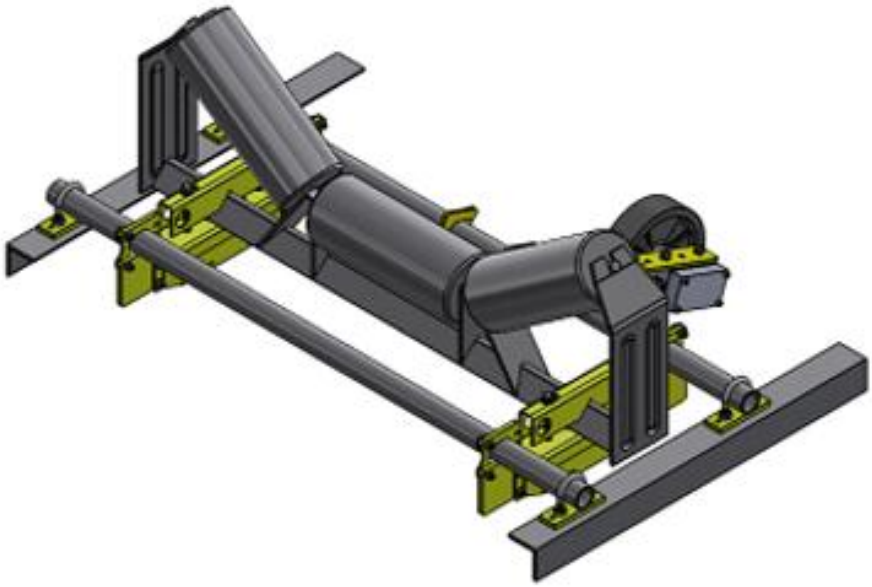


Figura 2.3 Báscula de Pesaje

2.1.5 Celda de carga

Considerando la referencia [6], se define a una celda de carga como un transductor que convierte la fuerza aplicada sobre ella en una señal eléctrica. Para lograr el cambio de fuerza a voltaje, las celdas de carga contienen internamente sensores de deformación, llamados galgas extensiométricas, el cual detectan los valores de deformación.

La variación de longitud, área de sección transversal y la resistividad generan un cambio en la resistencia en la galga extensiométrica. Así mismo si se requiere medir una fuerza o carga, el dispositivo normalmente lleva más de 02 galgas dependiendo de la precisión y carga que requiere medir, la propiedad y forma de los dispositivos varían según la necesidad y esfuerzo de tensión o compresión, en la figura 2.4 se puede observar un dispositivo para medir fuerza compuesto por 04 galgas el cual forman en puente Wheatstone [19].

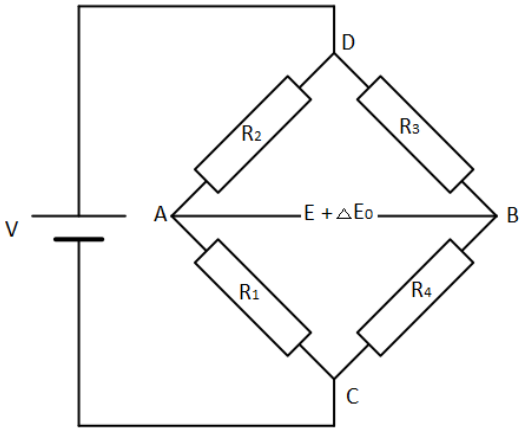


Figura 2.4 Puente Wheatstone

(Fuente: Referencia [19])

La báscula de pesaje lleva la celda tipo viga flexible el cual llevan 04 galgas el cual tiene la forma de viga recta apoyada a un extremo y el otro extremo va la carga, las galgas están distribuidas en la parte inferior y superior para medir fuerza de tensión y compresión.

2.1.6 Error de pesaje

El error de pesaje es la diferencia entre el peso Real y el peso indicado por el sistema de pesaje, previo a ello se debe realizar la tara del pesaje indicado por el sistema de pesaje, para ello se requiere que funcione sin carga.

2.1.7 Sensor de Velocidad

El sensor de velocidad que se usa es del tipo de rueda tal como se muestra en la figura 2.5. Así mismo posee un brazo de arrastre que se desplaza en un tramo de la faja de retorno. El sensor de rueda gira sobre el eje fijado en el brazo de arrastre. Internamente lleva un interruptor de proximidad que detecta la rotación de la rueda y genera una señal proporcional a la velocidad de la faja según [29], dicho sensor va instalado en el conjunto puente de la báscula de pesaje

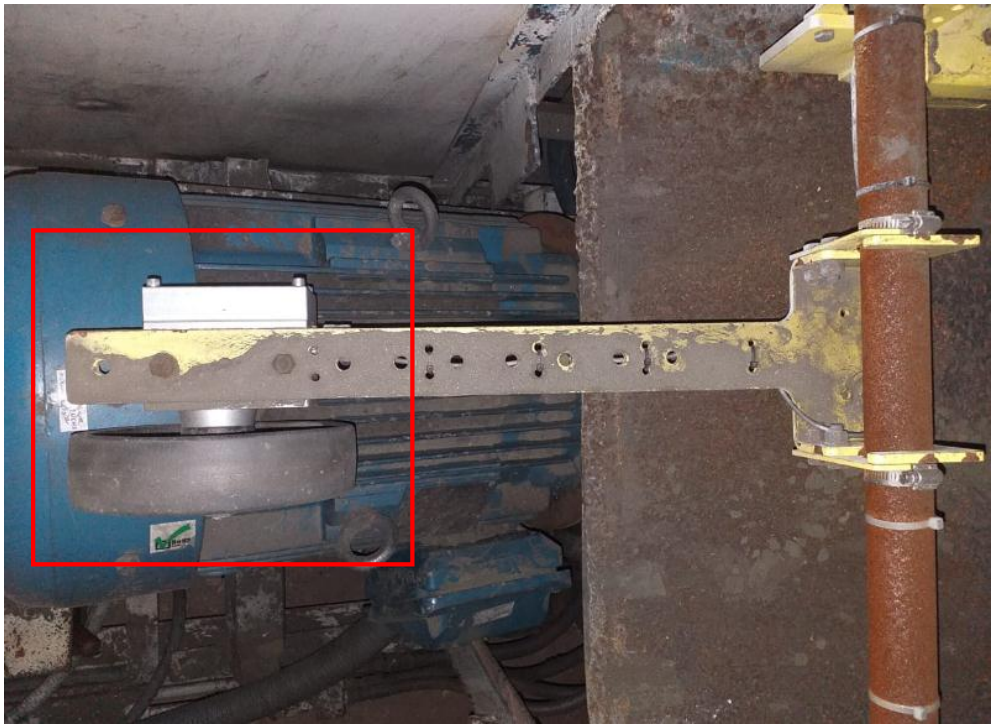


Figura 2.5 Sensor de rueda de velocidad

2.1.8 Sensor de inclinación

El sensor de inclinación permite cambiar el ángulo de inclinación para que pueda cubrir la altura de llenado del material dentro del contenedor, dicho sensor emita una señal eléctrica a medida que varía la inclinación de la faja transportadora.

2.2. Redes neuronales

2.2.1 Definición de red neuronal

La red neuronal se define de varias maneras según [17], se considera como un instrumento computacional de la ingeniería el cual tiene como base el funcionamiento del cerebro humano, por otra parte, la red neuronal tiene como objetivo principal el aprendizaje artificial, dicho aprendizaje es basado en la experiencia de varias pruebas con la finalidad de hacer lo mismo que hace el cerebro humano para aprender. También plantea que las redes neuronales están interconectadas en paralelo y con organización jerárquico con la finalidad de enlazar con los objetos de la vida real.

La red neuronal se basa en el procesamiento de la información que ocurre en las neuronas, estas señales se enlazan mediante la conexión entre neuronas, cada conexión

asociada tiene un peso. Para determinar la señal de salida esta es calculada por una función de activación el cual se aplica a la suma de sus pesos ponderados [15].

La neurona artificial tiene como objetivo superar las funciones básicas de la neurona biológica. Para ello se aplica varias entradas a la neurona, donde a cada entrada se le multiplica por un valor ponderado y estas se representan con una salida, Ver figura 2.6. La expresión vectorial que indica el funcionamiento de una neurona artificial se indica según la siguiente expresión [21]

$$NET = X * W \tag{2.2}$$

Donde NET es la señal de salida, X vector de entrada y W vector de pesos. Esta salida es procesada por una función de activación F para producir la salida OUT.

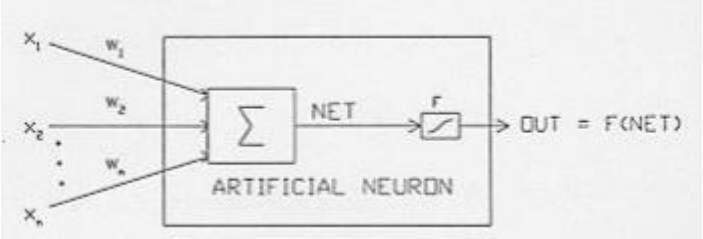


Figura 2.6 Modelo de Neurona Artificial
(Fuente: Referencia [21])

Según [21], considera que la red neuronal por elementos que tiene similitud a la neurona biológica cuyos elementos esta ordenados de manera similar a la neurona del cerebro humano. Así mismo la red neuronal presenta propiedades del cerebro humano y tiene la capacidad de aprender a través del estudio o experiencia, generalizar y abstraer las cualidades del objeto.

2.2.2 Característica de la red neurona artificial

Las características principales de la red neuronal son la de aprender con la experiencia y cambian su comportamiento según los valores de entradas que se le muestra, generaliza es decir amplia o extiende un dato frente a variaciones y brinda respuesta dentro del margen, otra característica es la abstracción en donde a un objeto le considera por separado sus cualidades, cuando aparentemente algunas entradas que no presentan aspectos comunes algunas redes neuronales abstrae la esencia del conjunto [21].

Según [15], considera la característica de la red según el patrón de conexión entre las neuronas, el tipo de entrenamiento o aprendizaje que usa para calcular los pesos de las

conexiones y su función de activación, así mismo considera a una red como una gran cantidad de elementos llamados neuronas, nodos en donde cada neurona está conectada y cada una con su peso respectivo, donde estos pesos ayudan a resolver el problema. La red neuronal puede almacenar, clasificar y recuperar patrones, realizar mapeos de patrón de entrada y salida para agrupar, y brindar solución los problemas.

2.2.3 Red neuronal de una capa y multicapa

La red neuronal de una capa es una red muy simple, debido a que solo cuenta con una capa las entradas está conectada a través de su peso correspondiente a cada neurona artificial. Ver figura 2.7

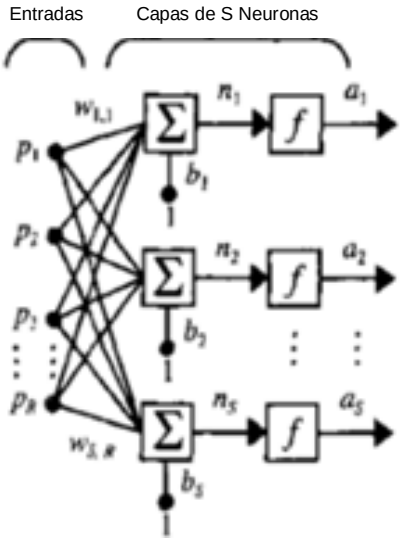


Figura 2.7 Red Neuronal de una Capa
(Fuente: Referencia [13])

La red de una capa puede tener varias entradas, ver figura 2.7, donde cada neurona tiene un peso, polarización o bías, función de activación y la salida, no siempre todas las salidas tienen la misma función de activación ello puede variar, cada peso es multiplicado por la entrada y luego se le suma el bías para que finalmente sea evaluado en la función de activación y da como salida el vector a [13] el cual se muestra en la ecuación 2.3

$$a = f(W * p + b) \tag{2.3}$$

La fila del índice de los pesos me indica la posición de la neurona y el índice de columna me indica la posición de la entrada y ello se puede mostrar como multiplicación de vectores ecuación 2.4

$$w = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ \vdots & \vdots & & \vdots \\ w_{S,1} & w_{S,2} & \ldots & w_{S,R} \end{bmatrix} \tag{2.4}$$

En cambio, una red de multicapa, consta de varias capas la cual están conectadas de manera parcial o total entre si la cual se le denomina etapa oculta. Esta red ofrece mejores cálculos y tienen mejor cualidad que la red de una capa. Ver figura 2.8

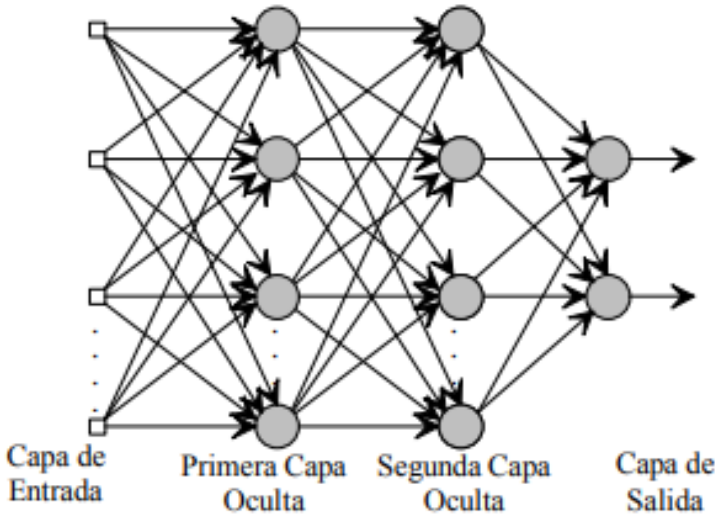


Figura 2.8 Red Neuronal Multicapa
(Fuente: Referencia [26])

Las redes multicapa pueden resolver problemas de mayor complejidad, así mismo la distribución se realiza con una cantidad determinada de neuronas en cada capa a continuación se detalla los tipos de capas.

- **Capa de entrada**, en esta capa se reciben la información proveniente de los datos de entrada
- **Capa Oculta**, esta capa es interna a la red y no tiene contacto con el exterior y el número de capas lo define el entrenamiento o complejidad del problema a resolver y pueden ser conectadas de diferentes maneras.
- **Capa de salida**, en esta capa están los valores de salida

A diferencia de las redes de una capa pueden tener diferente función de activación entre capas y pueden ser entrenados y aproximar la función de manera arbitraria. Para definir la cantidad de capas ocultas depende del entrenamiento, cabe resaltar que la capa de entrada está definida por las entradas a usar, así como los datos de salida.

En la figura 2.9 se muestra una red multicapa de tres capas, donde para cada capa, se calcula la salida en función de la función de activación [13] así mismo se muestra la salida general en la ecuación 2.5, 2.6 y 2.7.

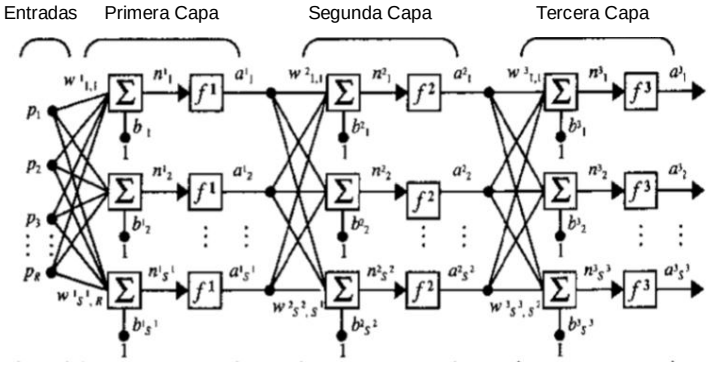


Figura 2.9 Red Neuronal de tres Capas

(Fuente: Referencia [13])

$$a^1 = f^1(w^1 * p + b^1) \quad (2.5)$$

$$a^2 = f^2(w^2 * a^1 + b^2) \quad (2.6)$$

$$a^3 = f^3(w^3 * a^2 + b^3) \quad (2.7)$$

Reemplazando las ecuaciones 2.5 y 2.6 en la ecuación 2.7 se obtiene la ecuación 2.9

$$a^3 = f^3(w^3 * f^2(w^2 * f^1(w^1 * p + b^1) + b^2) + b^3) \quad (2.8)$$

Dónde:

a^3 : salida de la capa 3.

w^1, w^2, w^3: pesos de la capa 1,2 y 3 respectivamente.

f^1, f^2, f^3 : función de activación de la capa 1,2 y 3 respectivamente

p : entrada de la red neuronal.

b^1, b^2, b^3 : bías de la capa 1,2 y 3 respectivamente.

2.2.4 Tipos de redes neuronales

Existen variedad de tipos de redes y estas dependen del tipo de entrenamiento, tipo de estructura, conexiones y algoritmo que emplean, a continuación, se detalla algunos tipos de redes.

2.2.4.1 Red forward propagation

Esta red presenta la forma básica en donde las conexiones son hacia adelante, no existe conexiones hacia atrás o lateral la información va hacia adelante, de ejemplo de estas redes tenemos el perceptrón, Adaline, Madaline y MLP.

2.2.4.2 Red back propagation

Esta red presenta capa de entrada, oculta, salida y se caracteriza por la forma de realizar el ajuste de sus pesos el cual va de adelante hacia atrás, los valores iniciales ingresan y pasan por las capas hasta obtener la salida, calcula el error y este valor es llevado hacia atrás partiendo de la capa de salida para realizar el ajuste de pesos

2.2.4.3 Red perceptrón multicapa

Según [30], el perceptrón multicapa se caracteriza por tener capa oculta ello es aparte de la capa de la entrada y salida, donde las capas están unidas entre sí a través de conexiones en donde esta conexión puede ser variable y aproxima funciones con precisión y puede representar una función booleana y resuelve problemas que no son linealmente separables.

2.2.4.4 Red recurrente

Según [26], las redes recurrentes permiten que las neuronas tengan lazos de retroalimentación y las capas conexiones hacia atrás entre su mismo nivel o posterior, estas redes presentan un retardo donde la salida es la entrada retrasada en un tiempo mínimo el cual permite modelar comportamientos dinámicos algo que no se podía con los otros tipos de redes. La forma que guarda información es con la retroalimentación que realiza el cual le permite retener los valores del primer paso para usarlo en el siguiente paso, dado estas características dichas redes son usadas para procesos de series temporales y problemas de control.

2.2.5 Algoritmo

En la referencia [22] se define algoritmo como un conjunto de instrucciones secuenciales, figura 2.10, que manipula la información con la finalidad de encontrar la solución a un problema planteado, en el contexto matemático se define algoritmo como una serie de normas específicas que cumple la ejecución de una actividad sin generar incertidumbre al momento de realizar el trabajo. Así mismo se define los siguientes algoritmos:

- Lenguaje natural
- Lenguaje de Programación
- Pseudocódigo
- Diagrama de Flujo

Figura 2.10 Representación Básica de un Algoritmo
(Fuente: Referencia [22])

Para que un algoritmo sea optimo debe ser eficaz, pueda adaptarse a diferentes problemas, debe ser robusto, el razonamiento empleado debe ser entendible y poseer una alta capacidad de generalización [26].

2.5.1 Algoritmo de Retro Propagación

Según [12] el algoritmo de Retro Propagación abarca cualquier método de aprendizaje supervisado, el cual se apoya en el cálculo del vector gradiente o de la matriz Jacobiana y por consiguiente la matriz Hessiana. Ambos algoritmos tienen como principal funcionamiento el aprendizaje de la función subyacente a una serie de datos entregados. En resumen, el algoritmo va de adelante hacia atrás para calcular el gradiente necesario para ajustar los parámetros de la Red Neuronal.

Según [31] considera la red de propagación hacia atrás en tres capas, la primera capa de entrada donde se tiene N unidades de proceso, la segunda es la capa de salida en donde tiene M unidades de proceso y ultimo la capa oculta donde tiene L neuronas, ver figura 2.11 al aplicar un vector de entrada se calcula la salida y luego se compara la salida deseada con el error.

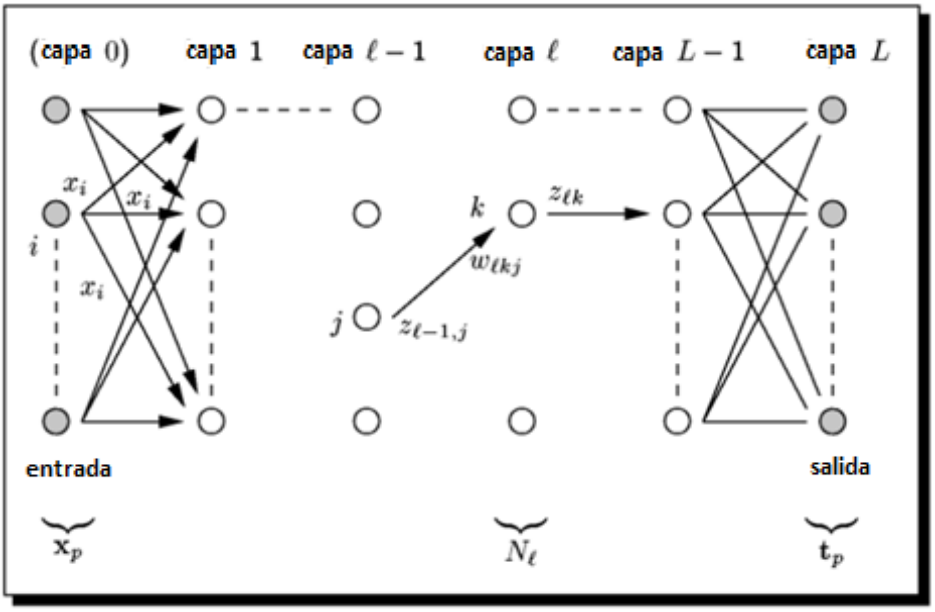


Figura 2.11 Estructura de una Red de Propagación hacia atrás
(Fuente: Referencia [7])

Este algoritmo no aplica para redes de una capa, dado que para actualizar el peso deriva el error y este es cero por ende los pesos no se actualizan. Se debe tener cuidado al momento de escoger los pesos iniciales si el valor es alto puede provocar que la red se sature es importante brindar el valor adecuado dado que el algoritmo puede encontrar el mínimo local y no el global. El factor de aprendizaje que se debe tomar al momento de realizar el entrenamiento no debe ser muy elevado debido a que esto originaria inestabilidad y si es demasiado pequeño genera demoras en el entrenamiento. Los datos para el entrenamiento deben tener lógica con el problema a resolver y se debe realizar en 2 etapas, en la primera etapa se realiza el entrenamiento y en la segunda etapa se realiza la comprobación del aprendizaje, con ello se puede evitar que la red sobre aprenda. Para definir las capas ocultas se realiza la prueba de ensayo y error [26].

Según [9] si ocurre que ningún patrón de entrenamiento haga que ningún nodo alcance la salida próxima a los valores extremos entonces dicho nodo alcanzo la saturación prematura para ello es posible que dicho nodo pase por el proceso de des aprendizaje antes de continuar con el entrenamiento. El algoritmo inicializa con dar valores aleatorios pequeños a los pesos iniciales a continuación se brindan los siguientes pasos para el algoritmo:

1. Presentar el patrón en la capa de entrada.
2. Las capas ocultas evalúan la salida usando el patrón ingresado.
3. La capa de salida evalúa la salida.

4. Se realiza la comparación con la salida deseada.
5. Se calcula la derivada en los nodos de salida.
6. Entrena cada nodo usando el descenso de gradiente.
7. Para cada nodo oculto, se calcula la derivada.
8. Para cada nodo oculto se usa la derivada calculada en 7 para entrenar de acuerdo con el descenso gradiente.

Los pasos del 1 al 3 la red calcula los pesos y la salida con los pesos iniciales aleatorios, del paso 4 al 8 se realiza el método del algoritmo hacia atrás o retro propagación. En el paso 7 se propaga la derivada desde los nodos de salida hacia atrás con la finalidad de volver a ajustar los pesos y hacer mínimo el error a la salida.

Según [7] se realizan los siguientes cálculos para realizar el algoritmo de retro propagación, de la figura 2.11 se calcula la salida la neurona "k" de la capa oculta "l" ecuación 2.9

$$Z_{LK} = f\left(\sum_{j=1}^{N_{\ell-1}} w_{\ell,j,k} * z_{\ell-1,j}\right) \tag{2.9}$$

Los pesos de la red se pueden representar de manera matricial, en la ecuación 2.10 se calcula los pesos de la capa l.

$$w_l = \begin{pmatrix} w_{\ell,1,1} & \cdots\cdots & w_{\ell,1,N_{\ell-1}} \\ \vdots & \ddots & \vdots \\ w_{\ell,N_l,1} & \cdots\cdots & w_{\ell,N_\ell,N_{\ell-1}} \end{pmatrix} \tag{2.10}$$

Se calcula la salida para la capa oculta l, se tiene la ecuación 2.11

$$z_\ell^T = f\left(a_\ell^T\right) = \left(f\left(a_{\ell,1}\right), \ldots\ldots\ldots\ldots\ldots f\left(a_{\ell,N_l}\right)\right) \tag{2.11}$$

Dónde:

$$a_\ell = w_\ell * z_{\ell-1} \tag{2.12}$$

Se requiere realizar el ajuste de los pesos para minimizar el error y para ello se define el error cuadrático medio, Este error es comparado entre la salida deseada y lo calculado, según la ecuación 2.13, 2.14 y 2.15.

$$E = (z_{\ell,q} - t_q) \tag{2.13}$$

$$E(w) = \frac{1}{2} * \sum_{q=1}^{N}\left[z_{\ell,q}(x) - t_q(x)\right]^2 \tag{2.14}$$

$$E_{tot}(w) = \frac{1}{2} * \sum_{p=1}^{P} E(w) = \sum_{p=1}^{P}\sum_{q=1}^{N_L}\left[z_{Lq}(x_p) - t_q(x_p)\right]^2 \tag{2.15}$$

Donde

$t_q(x)$: Salida deseada

$z_{\ell,q}(x)$: Salida de la Neurona q

$E(w)$: Error cuadrático medio

E : Función error de salida

$E_{tot}(w)$: Suma de error total, solo depende de los pesos de E

Como el error está en función de los pesos y lo que se quiere es calcular el peso para realizar el ajuste. Para ello usamos el vector gradiente [7], es decir la derivada parcial con respecto al peso que se quiere realizar el ajuste, ecuación 2.16 y 2.17.

$$\nabla E = \left\{\frac{\partial E(w)}{\partial w_{\ell,j,i}}\right\} \tag{2.16}$$

$$w_{\ell,j,i}(\mathrm{t}+1) = w_{\ell,j,i}(t) - \mu * \left.\frac{\partial E(w)}{\partial w_{\ell,j,i}}\right|_{w(t)} = w_{\ell,j,i}(t) - \mu * \sum_{p=1}^{P} \left.\frac{\partial E(w)}{\partial w_{\ell,j,i}}\right|_{w(t)} \tag{2.17}$$

Donde

μ : Constante de Aprendizaje y es mayor que cero

$w_{l,j,i}(t)$: Peso anterior

$w_{l,j,i}(t+1)$: Peso actualizado

∇E : Gradiente de error respecto a los pesos

Tener en cuenta que el valor de u es positivo y debe ser un valor intermedio, es decir si es muy grande el aprendizaje no va encontrar el error mínimo local y va generar oscilaciones cuando se aproxime al error mínimo y si es demasiado pequeño el aprendizaje se realiza lento [7].

El gradiente de error ∇E se calcula en función del peso que se requiere ajustar y la derivada de la función de activación se calculan de manera recursiva, ecuación 2.18 y 2.19.

$$\nabla_{z,\ell} E = w_{\ell+1}^{T}.\left[\nabla_{z,\ell+1} E \odot f^{l}(a_{\ell+1})\right] \tag{2.18}$$

$$(\nabla E)_{\ell} = \left[\nabla_{z,\ell} E \odot f^{l}(a_{\ell})\right].z_{\ell-1}^{T} \tag{2.19}$$

Donde

$\odot$: Operador producto de matriz

El error $E(w)$ depende de los pesos $\mathrm{w}_{\ell,j,i}$ para la salida de la neurona (j, i) y se calcula con la ecuación 2.24.

$$\frac{\partial E}{\partial w_{\ell,j,i}} = \frac{\partial E}{\partial z_{\ell,j}} \frac{\partial z_{\ell,j}}{\partial w_{\ell,j,i}} \tag{2.20}$$

$$\frac{\partial z_{\ell,j}}{\partial w_{\ell,j,i}} = \frac{\partial}{\partial w_{\ell,j,i}} \left[f\left(\sum_{m=1}^{N_{\ell-1}} w_{\ell,j,m} . z_{\ell-1,m}\right)\right] =$$

$$f'\left(\sum_{m=1}^{N_{\ell-1}} w_{\ell,j,m} . z_{\ell-1,m}\right) . \frac{\partial}{\partial w_{\ell,j,i}} \left[\sum_{i=0}^{N_{\ell-1}} w_{\ell,j,m} . z_{\ell-1,m}\right]$$

$$\frac{\partial z_{\ell,j}}{\partial w_{\ell,j,i}} = f'\left(\sum_{m=1}^{N_{\ell-1}} w_{\ell,j,m} . z_{\ell-1,m}\right) . z_{\ell-1,m} \Rightarrow \frac{\partial z_{\ell,j}}{\partial w_{\ell,j,i}} = f'\left(a_{\ell,j}\right) . z_{\ell-1,i} \tag{2.21}$$

$$\frac{\partial E}{\partial z_{\ell,j}} = \sum_{m=1}^{N_{\ell+1}} \frac{\partial E}{\partial z_{\ell+1,m}} \frac{\partial z_{\ell+1,m}}{\partial z_{\ell,j}} \tag{2.22}$$

$$\frac{\partial E}{\partial z_{\ell,j}} = \sum_{m=1}^{N_{\ell+1}} \frac{\partial E}{\partial z_{\ell+1,m}} \frac{\partial}{\partial z_{\ell,j}} \left[f\left(\sum_{q=1}^{N_{\ell}} w_{\ell+1,m,q} . z_{\ell,q}\right)\right]$$

$$\frac{\partial E}{\partial z_{\ell,j}} = \sum_{m=1}^{N_{\ell+1}} \frac{\partial E}{\partial z_{\ell+1,m}} . f'\left(\sum_{q=1}^{N_{\ell}} w_{\ell+1,m,q} . z_{\ell,q}\right) \frac{\partial}{\partial z_{\ell,j}} \left[\sum_{q=1}^{N_{\ell}} w_{\ell+1,m,q} . z_{\ell,q}\right]$$

$$\frac{\partial E}{\partial z_{\ell,j}} = \sum_{m=1}^{N_{\ell+1}} \frac{\partial E}{\partial z_{\ell+1,m}} . f'\left(\sum_{q=1}^{N_{\ell}} w_{\ell+1,m,q} . z_{\ell,q}\right) \frac{\partial}{\partial z_{\ell,j}} \left[\sum_{q=1}^{N_{\ell}} w_{\ell+1,m,q} . z_{\ell,q}\right]$$

$$\frac{\partial E}{\partial z_{\ell,j}} = \sum_{m=1}^{N_{\ell+1}} \frac{\partial E}{\partial z_{\ell+1,m}} . f'\left(a_{\ell+1,m}\right) . w_{\ell+1,m,j} \tag{2.23}$$

Reemplazando las ecuaciones 2.20 y 2.22 en 2.19 se tiene la función error, la ecuación 2.22 realiza el cálculo del error desde la capa L-1 hasta la capa 1

$$\frac{\partial E}{\partial w_{\ell,j,i}} = \frac{\partial E}{\partial z_{\ell,j}} \frac{\partial z_{\ell,j}}{\partial w_{\ell,j,i}} = \left[\sum_{m=1}^{N_{\ell+1}} \frac{\partial E}{\partial z_{\ell+1,m}} . f'\left(a_{\ell+1,m}\right) . w_{\ell+1,m,j}\right] f'\left(a_{\ell,j}\right) . z_{\ell-1,i} \tag{2.24}$$

2.2.6 Entrenamiento

En referencia [2] para realizar el entrenamiento de una red primero se debe definir la cantidad de neuronas, capas y que función de activación se va usar luego se debe ajustar parámetros internos del modelo neuronal, para adaptarse a variables del entrenamiento. Sin embargo, los valores de ajuste de peso se realizan mediante la regla de aprendizaje, esta regla va evaluar el rendimiento de la red.

Según [21] considera que el objetivo de una red neuronal es lograr la aplicación para las entradas que generan la salida deseada, así mismo considera 2 grupos de entrenamiento.

- Entrenamiento Supervisado: En entrenamiento consiste en presentar un vector de entrada a la red, calcular la salida de la red y con el resultado del

error realimenta la red y cambia el peso, ello se aplica de manera consecutiva.

- Entrenamiento No supervisado: Este entrenamiento no requiere de un vector de salida deseado, por ende, no realizan comparaciones entre salidas deseadas y reales, por lo cual el algoritmo de modifica los pesos de la red para generar vector de salida consistente.

2.2.6.1. Modos de entrenamiento

El entrenamiento de redes neuronales requiere de una señal de error para analizar el comportamiento del error con los parámetros de la red, con la finalidad de ajustar los parámetros de la red logrando así disminuir el error observado, la siguiente ecuación 2.25 traduce el cálculo del error a cada uno de los parámetros de la red [2]

$$\Delta w_{ij}^{k} = -n\frac{\partial E}{\partial w_{ij}^{k}} \tag{2.25}$$

- Entrenamiento por Lotes: Recorre todo el conjunto de entrenamiento para acumular error y modular los pesos en cada época.
- Entrenamiento online: En este entrenamiento los pesos se ajustan en la red.
- Entrenamiento por mini lotes: Para una muestra pequeña de entrenamiento el peso se ajusta a partir del error.

2.2.7 Aprendizaje

La red Neuronal tiene la capacidad de aprender mediante un entrenamiento, el cual parte de un conjunto de patrones, ejemplos. Así mismo las Redes neuronales pueden modificar su comportamiento ante una respuesta y se ajustan. Cabe resaltar que existen varios algoritmos de aprendizaje que se pueden aplicar a red neuronal.

2.2.7.1 Supervisado

Según [25] en este aprendizaje a la red Neuronal se muestra unos patrones junto con la salida deseada. Donde la red de manera repetitiva ajusta sus pesos hasta aproximarse a la salida deseada. Los pesos varían de acuerdo al error producido entre la salida real y de la red ver figura 2.12, esta variación es proporcional.

Aprendizaje supervisado es el proceso que se lleva a cabo bajo la tutela de un maestro, en donde compara la salida actual con la deseada, además puede realizar el ajuste de

pesos de acuerdo a su patrón de entrenamiento, sin embargo, existe un criterio de parada según la medida del error [30].

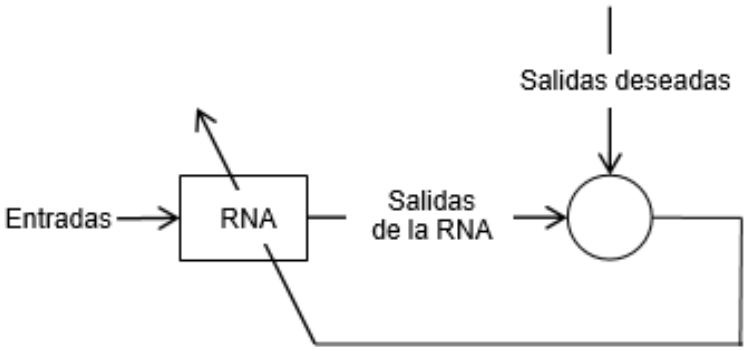


Figura 2.12 Aprendizaje supervisado
(Fuente: Referencia [25])

2.2.8 Arquitectura de red neuronal

La arquitectura de una red neuronal se basa en la organización de las neuronas en la red donde estas se agrupan formando capas para tener diferentes características, A continuación [8] describe diferentes arquitecturas de redes neuronales, figura 2.13.

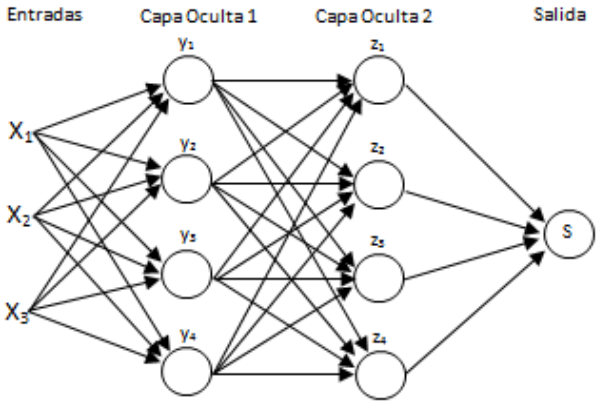


Figura 2.13 Arquitectura de una Red Neuronal
(Fuente: Referencia [8])

La propiedad de una red es la igualdad que existe entre las entradas y salidas de diferentes neuronas, otra propiedad también es la existencia de las conexiones laterales entre capas de las neuronas, en la arquitectura de conexión cruzada se muestra dos capas con el coeficiente de ponderación regulable, sin embargo, en la arquitectura es limitada.

La arquitectura de la red neuronal presenta principales características tales como tipo de señal, operación, topología de conexión, velocidad de operación, método de

modificación de coeficiente de ponderación y otras adicionales que van a depender de la solución al problema. La conexión directa presenta una limitada estructura, la conexión cruzada presenta el coeficiente de ponderación ajustable el cual es determinado por la señal de salida de la primera capa. La conexión ordenada hacia atrás, las conexiones laterales entre capas y las neuronas es una característica de la estructura, así como también la cantidad de entradas, salidas de cada capa de la red [9].

2.2.9 Función de Activación de una red Neuronal

Según [2] la función de activación para una Red Neuronal para estimular a la neurona con la finalidad de la salida de la neurona, estas funciones se clasifican de la siguiente manera:

- Función Discreta: Esta función solo presenta valor 0 y 1 y en neuronas bipolares es -1 o 1
- Función de activación Lineal: Es la función de la identidad
- Función continua: Determina la probabilidad con la que se activa la neurona.
- Función no Lineal: Se aplica cuando la red presenta varias capas
- Función sigmoide: Transforma valores introducidos a una escala (0,1), donde los valores altos tienen como limite a 1 y los valores bajos a 0.

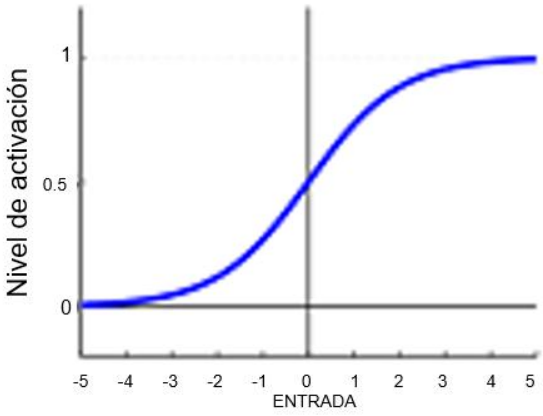


Figura 2.14 Función de Activación Sigmoidal
(Fuente: Referencia [2])

$$y = f_{logistic}(z) = \sigma(z) = \frac{1}{1+e^{z}} \tag{2.26}$$

2.3. Herramienta de trabajo

Para realizar el entrenamiento de la Red Neuronal vamos a utilizar el software Matlab el cual es un programa de cálculo matemático. La versatilidad de este programa se da en

el tratamiento de datos con matrices y la creación de subrutinas para una variedad de tareas el cual se va a utilizar para el entrenamiento y validación de la red neuronal. Así mismo se pueden crear funciones el cual va a ser ejecutados durante el entrenamiento y proporciona un gran número de funciones matemáticas el cual va a ser usadas en los códigos a implementar.

El software contiene estructuras de redes neuronales que se pueden adecuar a la necesidad del usuario en este caso una vez obtenida los parámetros de la red neuronal con las funciones propias de redes neuronales del Matlab podemos simular y visualizar la red neuronal diseñada.

Existen variedad de software para el entrenamiento de una red neuronal tales como el Python, Tensor flow, Keras, Java y otros más

2.4. Normalización de datos

La normalización de datos es considerada una herramienta común dado que esta se utilizada cuando se realiza el entrenamiento de las redes neuronales. El objetivo de la normalización es que las entradas se encuentran dentro de un rango aceptable el cual es el mismo rango de la función de activación. La función de activación Sigmoide está acotada entre los valores [0 1] y si los datos de entrada no se encuentran en este rango se realiza el proceso de normalización [23]. El proceso de normalización es el siguiente:

1. Se normaliza los datos de entrada usar formula 2.26
2. Se realiza el entrenamiento con los datos normalizados
3. Se realiza la des normalización para los datos de salida

CAPÍTULO III
DESARROLLO DEL TRABAJO DE LA TESIS

En el presente capítulo, se explica cómo se desarrolla el objetivo de la presente tesis, el cual consiste en diseñar un sistema de compensación de error usando redes neuronales para reducir el margen de error de pesaje.

3.1. Metodología

La metodología a usar consta de dos etapas el cual va de manera secuencial, la primera etapa se desarrolla en 02 subetapas, donde la primera subetapa consiste en la toma de datos y para la toma de datos se realizarán pruebas en campo esta subetapa se considera muy importante ya que con ello vamos a lograr un entrenamiento adecuado, la segunda subetapa consta de la selección de variables de entrada y salida de la red neuronal.

La segunda etapa se define todas las características de la red neuronal, arquitectura, algoritmo y entrenamiento con la finalidad de optimizar el modelo a usar, asimismo se describe la selección del diseño de la red y tipo de algoritmo a usar, en la figura 3.1 se puede visualizar el proceso de la metodología.

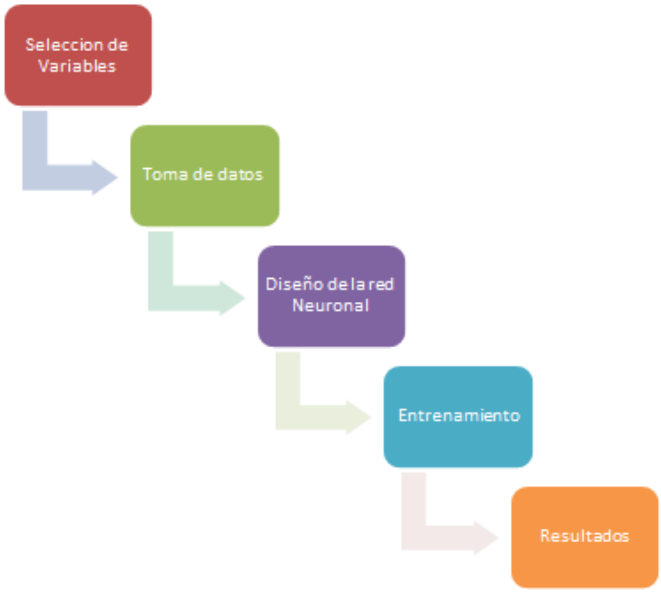


Figura 3.1 Diagrama de Metodología

La metodología que se utilizó para las diversas evaluaciones de entrenamiento y validación, se dan en las siguientes etapas:

Etapa 1: se realiza la selección de la data para entrenamiento y validación mediante el método de validación cruzada. El cual consiste en dividir la data en dos partes una para realizar el entrenamiento y la otra para la validación de la red neuronal.

Etapa 2: se define los parámetros de la red neuronal. En este caso los pesos y bias, el cual se logran mediante el método de descenso gradiente, este método consiste en la búsqueda del mínimo error deseado, donde se define el vector gradiente y luego se resta el error, ello se hace hasta encontrar el mínimo error o hasta que complete el número de iteraciones.

Etapa 3: se define los hiperparametros de la red neuronal, las capas ocultas de la red, según [3], indica que no existe una regla, ni se puede conocer a priori para un problema dado, así mismo en [2], menciona que las reglas genéricas que hayan funcionado en problemas similares o buscar una mejor opción mediante la prueba y error es una forma de definir los hiperparametros de la red neuronal, en tal sentido esta investigación inicia los hiperparametros desde valores muy bajos y luego los incrementa hasta lograr el valor óptimo de numero de capas ocultas, lo cual ha sido aplicado en casos similares como lo realizado en la investigación de [14] y también en el trabajo de investigación de [34].

Etapa 4: se define el número de las neuronas de cada capa oculta y para ello se tomará como referencia la secuencia de Hirose, para la búsqueda optima de numero de neuronas de cada capa oculta en base al error deseado.

Etapa 5: En esta etapa se aplicó el algoritmo de backpropagation, tomando en cuenta los valores arrojados por el método de validación cruzada. El mencionado algoritmo consiste en aplicar un valor de peso a la entrada de la red y este se propaga hacia adelante desde la primera capa a través de las capas seguidas de la red, hasta generar una señal salida, esta señal se compara con la salida deseada y se calcula una señal de error para cada una de las salidas. Por lo cual estas salidas de error se propagan hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa oculta que contribuyen directamente a la salida.

3.2. Diagrama de bloque funcional

Para la segunda etapa de la metodología se empleará el programa de Matlab, para la etapa de entrenamiento y validación, en la figura 3.2 se muestra el diagrama de bloque general con cada etapa de la metodología y en la figura 3.3 se visualiza el diagrama de

bloques detallado, donde se puede visualizar cada proceso del desarrollo de la red neuronal.

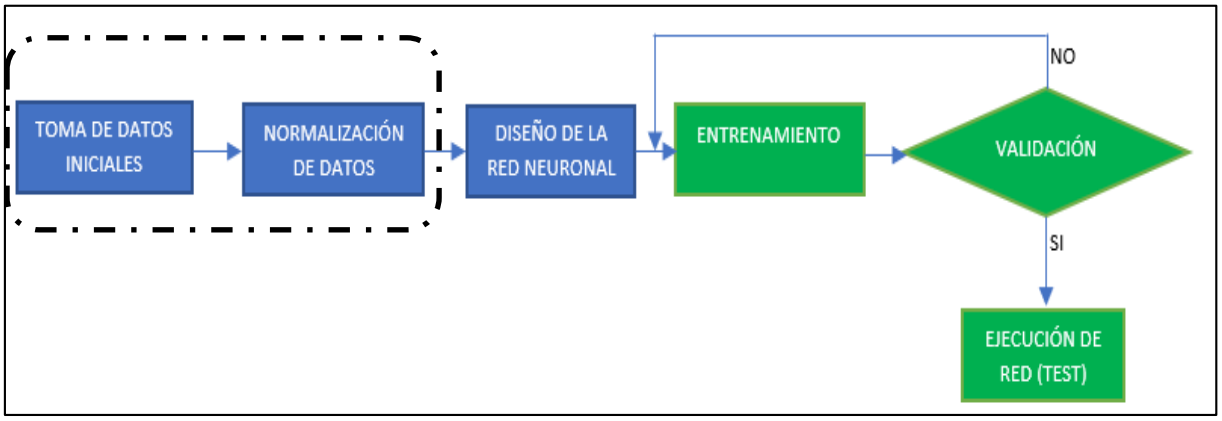


Figura 3.2 Diagrama de Bloque

3.2.1. Explicación en detalle de cada etapa o bloque

A continuación, se describe el cada proceso del bloque funcional:

3.2.2 Toma de Datos

Los datos de peso y voltaje para el entrenamiento se registran de un trabajo en campo. Para la toma de datos se utilizó un multímetro fluke, balanza Estática para verificar los pesos indicados por el módulo de control de pesaje de la faja transportadora. De los datos que fueron tomados se tomara el 70% para el entrenamiento de la red neuronal y con el 30% de datos se realiza la prueba de la red neuronal.

Durante la toma de datos se consideró un flujo de mineral de 324 Tn/h de mineral, ángulo de inclinación de faja de llenado de 25º. Por ende, se tomaron 320 valores de peso de los cuales 224 datos fueron tomados para la etapa de entrenamiento y 96 datos de pesos fueron para la etapa de validación, mencionar también que los pesos fueron tomados de manera aleatoria tanto para la etapa de aprendizaje y validación. Ver tabla 3.1

Tabla 3.1 Valores de peso de mineral

Ítem	Sensor (mV)	VELOCIDAD (m/s)	ÁNGULO (grados)	PESAJE (Kg)
1	3.62	4.1	25	98
2	3.45	4.0	25	95
3	3.55	4.2	25	96
4	3.60	4.3	25	97

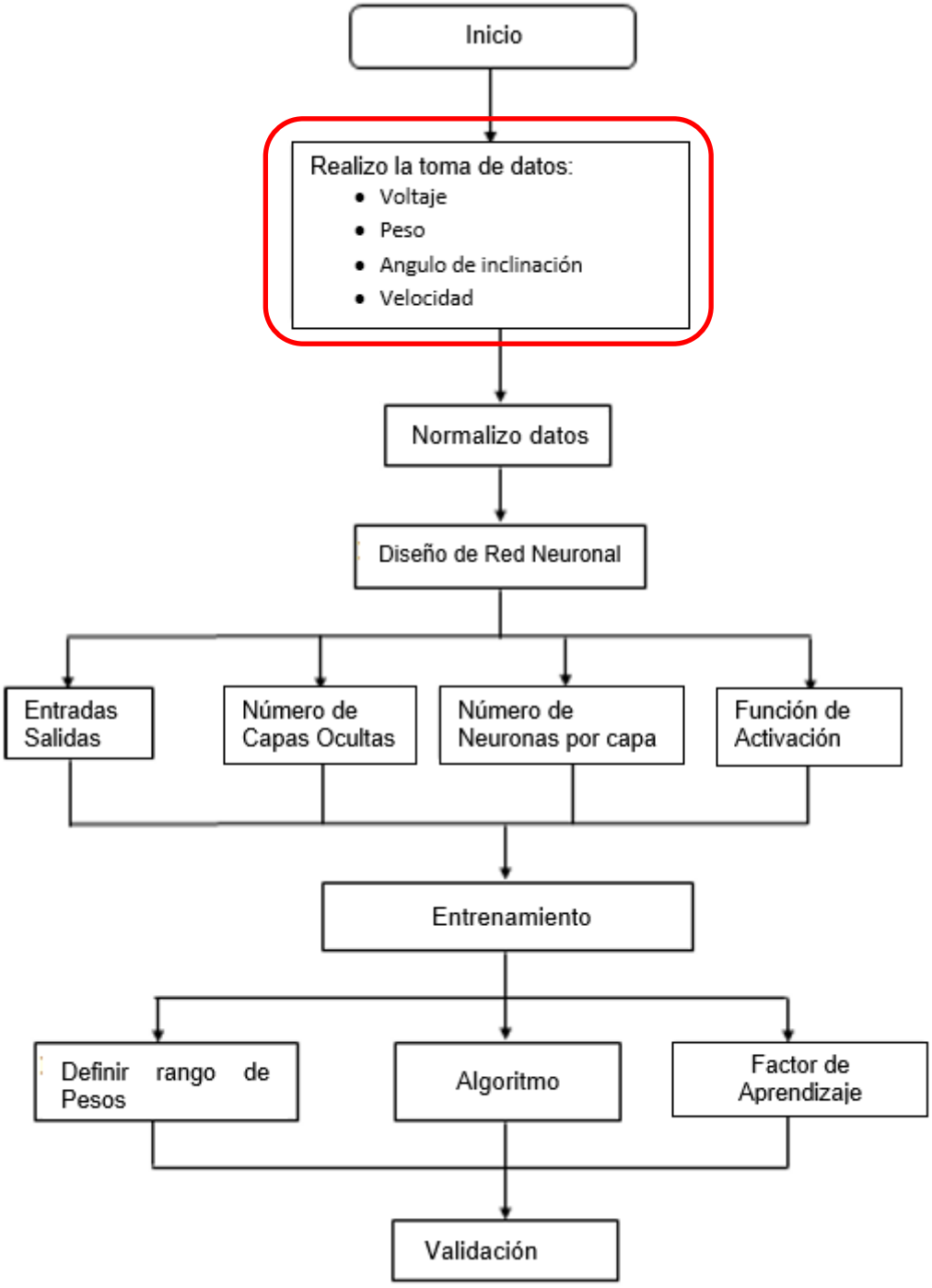


Figura 3.3 Diagrama de Bloque detallado

Para la adquisición de datos se realizan mediante los siguientes pasos:

Paso 1. Se realiza la adquisición de datos en campo y se define los valores de entrado y de salida.

Paso 2. Se realiza la partición de datos, para la parte entrenamiento y validación, para ello se usa la partición de 70-30, donde 70% de datos son para el entrenamiento y el 30% son para la validación de la red neuronal.

Paso 3. Se realiza la normalización de los datos de entrenamiento.

Paso 4. Se realiza el entrenamiento de la red neuronal.

Paso 5. Se realiza la validación de la red neuronal

La adquisición de datos fue tomada en campo, una vez culminada el proceso de la toma de datos, estos fueron particionados mediante el método de Cross validation, usando la técnica de descomposición de manera aleatoria, esta técnica divide los datos en 70% - 30% donde los datos que corresponden al 70% sirven para el entrenamiento de la red neuronal y la segunda parte de datos el 30% servirá para la validación. Cabe resaltar que la toma de datos de manera aleatoria para cada entrenamiento permite a disminuir el sesgo.

3.2.3 Normalización de Datos

En esta etapa se procede con la normalización de datos en el intervalo de [0,1]. La finalidad de este proceso es tener las variables de entradas de la red neuronal estén dentro del rango de los límites de la función de activación y puedan recibir la misma atención en el proceso de entrenamiento [10]. Se plantea la ecuación 3.35 para la normalización de datos, sin embargo, a la salida se debe realizar el proceso de la desnormalización el cual se puede ver en la ecuación 3.36

$$x_{norm} = \frac{2*(dato - Val_{min})}{(Val_{max} - Val_{min})} - 1 \tag{3.35}$$

$$x_{desnorm} = \frac{(1 + x_{norm})*(Val_{max} - Val_{min})}{2} - Val_{min} \tag{3.36}$$

Dónde:

Val_{min}: Valor mínimo del rango de trabajo

Val_{max}: Valor maximo del rango de trabajo

$dato$: Valor a Normalizar

$x_{desnorm}$: Valor desnormalizado

x_{norm} : Valor Normalizado

Al momento de realizar el programa de entrenamiento de la red en Matlab se va considerar 2 algoritmos para la normalización para los datos de entrada y des normalización para los datos de salida. Ver Anexo 1.

3.2.4 Diseño de la red neuronal

En este bloque se procede con el diseño de la red neuronal para ello primero se define las entradas y salidas de la red neuronal, ver tabla 3.2.

Tabla 3.2 Valores de entrada y salida de la red neuronal

Variable de entrada	**Rangos de Trabajo**	**Valores de Entrada**	**Valor de Salida**
Sensor de pesaje	0-150 KG	9-15 VDC	0-5 mV
Sensor de Velocidad	25-350 RPM	10-30 VDC	0-200 mA
Sensor de Inclinación	0° - 360°	10-30 VDC	4-20 mA
Variable de Salida			
Pesaje Total	25000- 30000 Kg		

Una vez definido las variables de entrada y salida se proceden con el diseño de la arquitectura de la red, tipo de red a usar y numero de capas ocultas.

Sin embargo, para determinar el número de capas ocultas usaremos el procedimiento de ensayo y error, previo a ello definimos como punto de partida 02 capas oculta con 04 neuronas por capa.

Tabla 3.3 Arquitectura de la red neuronal

Arquitectura de la Red	**Valores**
Número de entradas	03
Número de Salidas	01
Numero de Capas Ocultas	02
Numero de Neurona de capa oculta	04
Función de activación	Tangente Sigmoidal

3.2.5 Entrenamiento

En esta etapa se realiza el entrenamiento con el algoritmo Back Propagation, con la finalidad de encontrar los pesos óptimos y hacer nuestro modelo eficiente. Así mismo los pesos iniciales serán considerados de manera aleatoria, con la finalidad de ir calculando los siguientes pesos de las otras redes hasta llegar a la neurona de la salida, donde el valor final se comparará con el valor deseado y dicha diferencia de error ayudará a recalcular los pesos usando el método de Back-Propagation, hasta obtener el error mínimo y por ende el cálculo de los pesos ajustados. Para el entrenamiento usara el 70% de la data obtenida.

El algoritmo de entrenamiento realiza el ajuste de pesos con la finalidad de minimizar el error entre el valor calculado por la red y el valor deseado y este finaliza cuando haya alcanzado el mínimo error. Para el entrenamiento y cálculo de pesos se usará el software Matlab.

3.2.6. Validación

Para verificar el funcionamiento de la red neuronal, se utilizará el método de validación cruzada el cual consiste en dividir los datos muestreados, para nuestro caso se va considerar el 30% de datos para realizar la prueba de validación o test. Considerar que los datos que son usados para la validación no deben ser tomados de los datos de entrenamiento

3.3. Pseudocódigo y diagrama de flujo del programa

El proceso antes descrito también se explica mediante el diagrama de flujo tal como se muestra en la figura 3.4. Los pasos se visualizan en el Anexo 2.

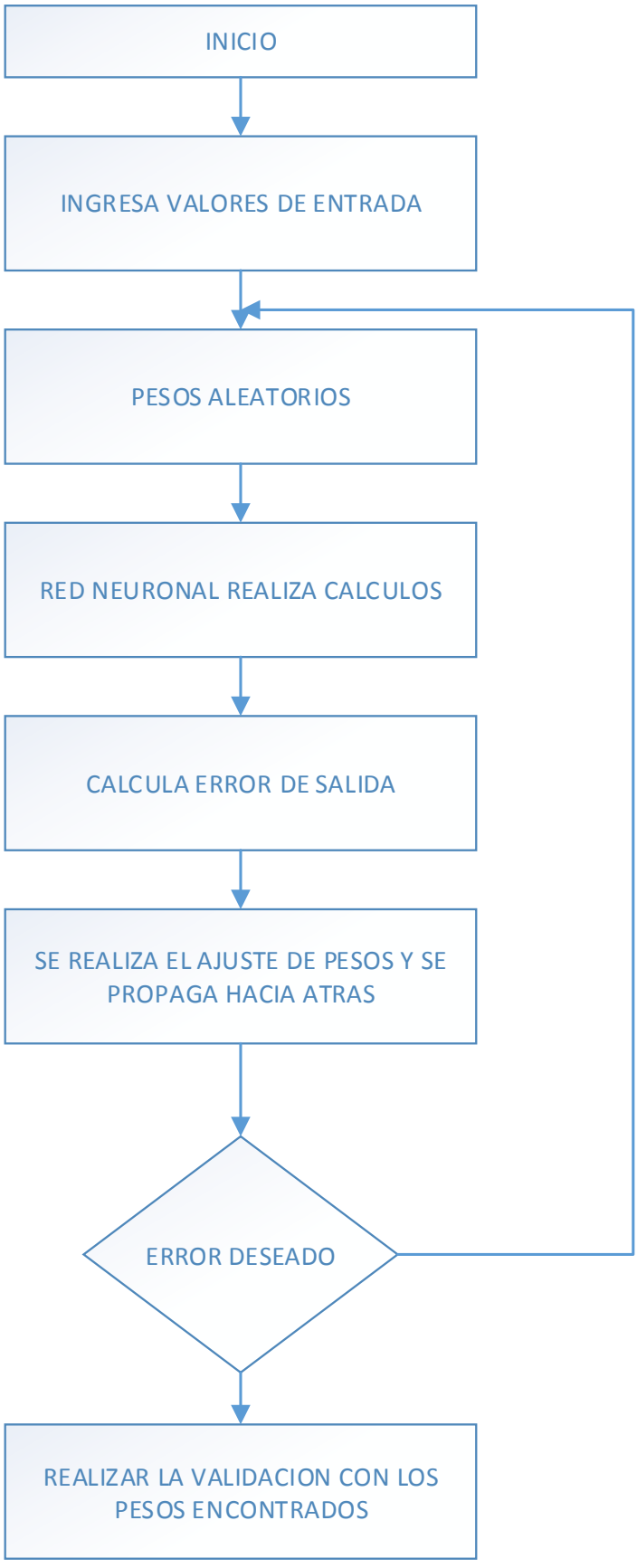


Figura 3.4 Diagrama de Flujo de programa

3.4. Diseño del experimento

Para el entrenamiento de la red neuronal se considera los siguientes parámetros el cual va a permitir que la red sea óptima.

3.4.1 Selección de neuronas de capas ocultas

Se debe tener en cuenta que para la selección de neuronas de la capa oculta se tomara como modelo el método de Hirose [36], Ver figura 3.5, para la comparación en donde este método plantea un algoritmo de podado de neurona. Cabe resaltar que una red muy pequeña no podrá lograr aprender el problema y una red muy grande puede generar el sobre aprendizaje. Así mismo se usará también el método de ensayo y error.

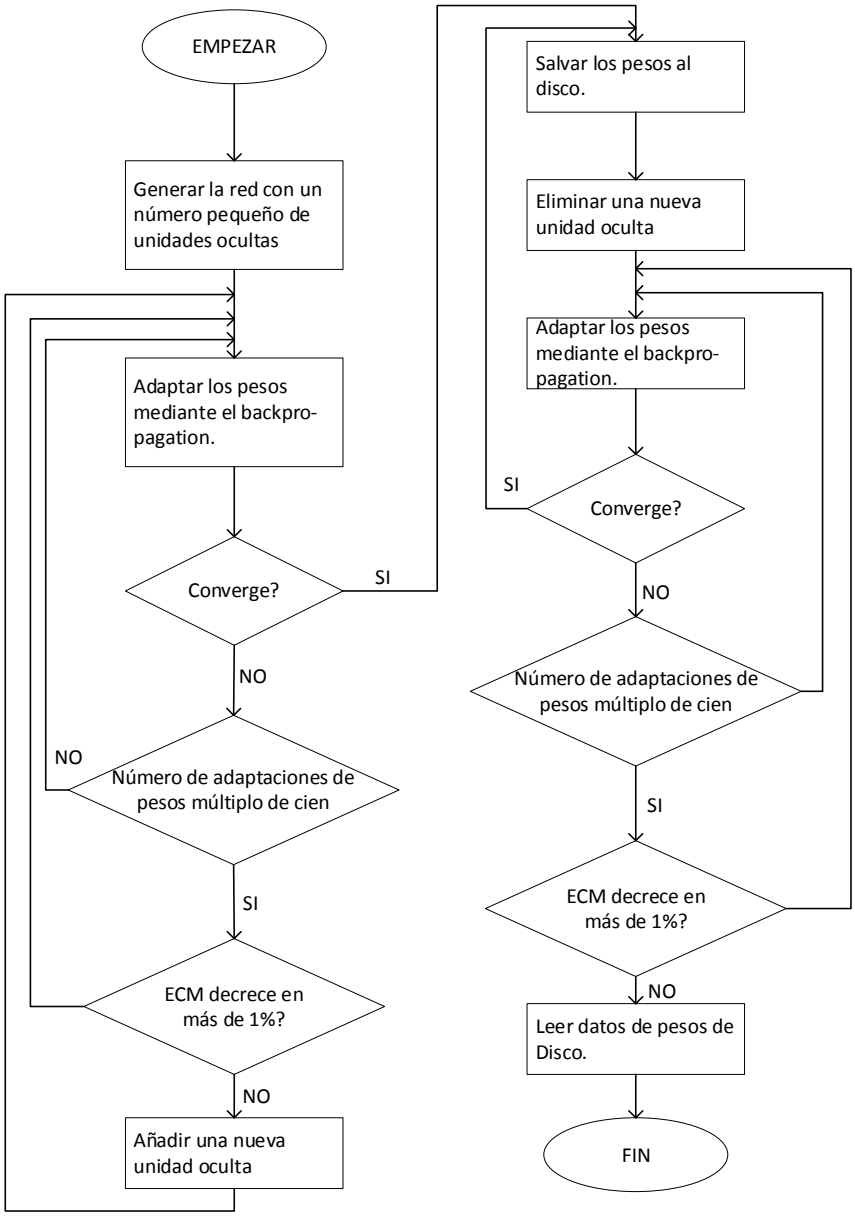


Figura 3.5 Algoritmo Hirose
(Fuente: Referencia [36])

3.4.2 Selección de capas ocultas

Para la selección de las capas ocultas, se toma en cuenta el cálculo del error mínimo local en [35] comparar redes neuronales con diferentes capas ocultas en donde concluye que una red de 02 capas ocultas presenta mayor posibilidad de lograr alcanzar el error mínimo local. Así mismo la sensibilidad de la red aumenta con el número de capas ocultas es decir a mayores capas ocultas la red puede lograr un aprendizaje inestable. Para ello vamos a usar la metodología prueba error

Tabla 3.4 Número de capas ocultas

Número de Capas	1 capas oculta	2 capas ocultas	3 capas ocultas
Número de neuronas	02 neuronas	02 neuronas	02 neuronas
	05 neuronas	05 neuronas	05 neuronas

Para el diseño de la topología de la red neuronal, estas van a diferir del grado de complejidad del problema a solucionar, número de entradas y salidas, así como también en muchos casos estas se relacionan con la cantidad de neuronas en cada capa oculta según [4]. Sin embargo, en [33], plantea una relación entre el número de neuronas ocultas y número de entradas y salidas, tal como se muestra en la siguiente ecuación:

$$k = \sqrt{m+n} + a \quad \text{..... (3.1)}$$

Donde:

m y n, representa el número de entradas y salidas.

a= parámetro ajustable que va desde 0 al 10.

Por ello para la presente tesis se considera una topología de 2 capas ocultas y 5 neuronas en cada capa oculta, tomando como referencia lo mencionado en [4] y [33].

Para el caso de la cantidad de muestras, ello va depender mucho del error deseado del problema, es decir en caso no se llegue al error estimado, se debe aumentar la cantidad de datos, tal es el caso de [28], [14] y [33], donde muestran 75, 101 y 400 datos respectivamente para su entrenamiento y validación de la red neuronal. Asimismo, el número de muestras de la data cuenta con 278 valores de los cuales 195 se utilizaron en la etapa de entrenamiento y 83 datos para la validación.

3.4.3 Selección de pesos aleatorios

La inicialización de pesos se considera un factor importante al momento de realizar el entrenamiento de la red, dado que la correcta selección influye en la velocidad de convergencia de la red neuronal, es decir puede que logre alcanzar de manera rápida el

mínimo local, también influye en que la probabilidad de la red aprenda y mucho va depender de la cantidad de épocas que va desarrollar la red y por último influye en que tan eficiente es la red. La clásica forma de iniciar los pesos parte de un intervalo de [-0.5, 0.5].

En la tabla 3.5 se presenta un resumen de los parámetros a considerar en la red, así como también se detalla el número de capas ocultas, neuronas de capa oculta. Las neuronas de las condiciones iniciales se eligen mediante el criterio prueba y error.

Tabla 3.5 Parámetros iniciales de la red neuronal

Parámetro	Condición Inicial	Criterio de elección
Número de neuronas de la Capas Ocultas	2-30	Hirose / Prueba y Error
Número de Capas Ocultas	1-3	Prueba y Error
Pesos Iniciales	[-.5 .5]	Mínimo error de validación
Factor de Aprendizaje	0-0.5	

3.5. Modelamiento matemático

3.5.1 Cálculo de las funciones de salida de las capas de salida

Para realizar el cálculo de las funciones de la capa oculta vamos a usar las ecuaciones 2.2, 2.3 y 2,9, para una red neuronal con 02 capas ocultas, ver figura 3.3. Realizando los cálculos se obtiene las siguientes ecuaciones:

Capa Oculta 1

$$\boldsymbol{y_1 = X_1 * w_{11} + X_2 * w_{21} + X_3 * w_{31} + b_1} \quad (3.2)$$

$$\boldsymbol{y_2 = X_1 * w_{12} + X_2 * w_{22} + X_3 * w_{32} + b_2} \quad (3.3)$$

$$\boldsymbol{y_3 = X_1 * w_{13} + X_2 * w_{23} + X_3 * w_{33} + b_3} \quad (3.4)$$

$$\boldsymbol{y_4 = X_1 * w_{14} + X_2 * w_{24} + X_3 * w_{34} + b_4} \quad (3.5)$$

Reemplazando en la ecuación 2.3 obtendremos la función final de salida el cual será la entrada para la capa oculta 2.

$$\boldsymbol{Y_1 = f(y_1)\,;\ Y_2 = f(y_2)\ ;\ Y_3 = f(y_3);\ Y_4 = f(y_4)} \quad (3.6)$$

Capa Oculta 2

$$\boldsymbol{z_1 = Y_1 * w_{Y1Z1} + Y_2 * w_{Y2Z1} + Y_3 * w_{Y3Z1} + Y_4 * w_{Y4Z1} + b_5} \quad (3.7)$$

$$\boldsymbol{z_2 = Y_1 * w_{Y1Z2} + Y_2 * w_{Y2Z2} + Y_3 * w_{Y3Z2} + Y_4 * w_{Y4Z2} + b_6} \quad (3.8)$$

$$\boldsymbol{z_3 = Y_1 * w_{Y1Z3} + Y_2 * w_{Y2Z3} + Y_3 * w_{Y3Z3} + Y_4 * w_{Y4Z3} + b_7} \quad (3.9)$$

$$\boldsymbol{z_4 = Y_1 * w_{Y1Z4} + Y_2 * w_{Y2Z4} + Y_3 * w_{Y3Z4} + Y_4 * w_{Y4Z4} + b_8} \quad (3.10)$$

Reemplazando en la ecuación 2.3 obtendremos la función final de salida el cual será la entrada para la capa de salida.

$$\boldsymbol{Z_1 = f(z_1)\ ;\ Z_2 = f(z_2)\ \ ;\ \ Z_3 = f(z_3);\ \ Z_4 = f(z_4)} \tag{3.11}$$

Salida de la Red

$$\boldsymbol{s = Z_1 * w_{Z1s} + Z_2 * w_{Z2s} + Z_3 * w_{Z3s} + Z_4 * w_{Z4s} + b_9 \ \rightarrow\ S = f(s)} \tag{3.12}$$

La función de activación que se usará la red neurona será la función sigmoidea ecuación 2.25. Así mismo los pesos iniciales serán considerados de manera aleatoria

3.5.2 Cálculo del error de salida y de capas ocultas

Para el cálculo del error de la red se usará la ecuación 2.2 y 2.15.

Error de capa de salida

$$\boldsymbol{E_o = (S - S_d) * f(s) * (1 - f(s))} \tag{3.13}$$

Error de capa oculta 2

$$\boldsymbol{E_{Z1} = E_o * w_{Y1Z1} * f(z_1)\ * (1 - f(z_1)\ )} \tag{3.14}$$

$$\boldsymbol{E_{Z2} = E_o * w_{Y2Z1} * f(z_2)\ * (1 - f(z_2)\ )} \tag{3.15}$$

$$\boldsymbol{E_{Z3} = E_o * w_{Y3Z1} * f(z_3)\ * (1 - f(z_3)\ )} \tag{3.16}$$

$$\boldsymbol{E_{Z4} = E_o * w_{Y4Z1} * f(z_4)\ \ * (1 - f(z_4)\ \ )} \tag{3.17}$$

Error de capa oculta 1

$$\boldsymbol{E_{Y1} = (E_{Z1} * w_{Y1Z1} + E_{Z2} * w_{Y2Z1} + E_{Z3} * w_{Y3Z1} + E_{Z4} * w_{Y4Z1}) * f(y_1)\ * (1 - f(y_1))} \tag{3.18}$$

$$\boldsymbol{E_{Y2} = (E_{Z1} * w_{Y1Z2} + E_{Z2} * w_{Y2Z2} + E_{Z3} * w_{Y3Z2} + E_{Z4} * w_{Y4Z2}) * f(y_2)\ * (1 - f(y_2))} \tag{3.19}$$

$$\boldsymbol{E_{Y3} = (E_{Z1} * w_{Y1Z3} + E_{Z2} * w_{Y2Z3} + E_{Z3} * w_{Y3Z3} + E_{Z4} * w_{Y4Z3}) * f(y_3)\ * (1 - f(y_3))} \tag{3.20}$$

$$\boldsymbol{E_{Y4} = (E_{Z1} * w_{Y1Z4} + E_{Z2} * w_{Y2Z4} + E_{Z3} * w_{Y3Z4} + E_{Z4} * w_{Y4Z4}) * f(y_4)\ * (1 - f(y_4))} \tag{3.21}$$

3.5.3 Cálculo y ajuste de pesos de capas ocultas y capa de salida

Para el cálculo del error de la red se usará la ecuación 2.17.

Pesos de capa de salida

$$\boldsymbol{w_{Z1s} = w_{Z1s} + \mu * E_o * z_1} \tag{3.22}$$

$$\boldsymbol{w_{Z2s} = w_{Z2s} + \mu * E_o * z_2} \tag{3.23}$$

$$w_{Z3s} = w_{Z3s} + \mu * E_o * z_3 \quad (3.24)$$

$$w_{Z4s} = w_{Z4s} + \mu * E_o * z_4 \quad (3.25)$$

Pesos de capa de oculta 2

$$w_{Y1Z1} = w_{Y1Z1} + \mu * E_{Z1} * y_1 \quad (3.26)$$

$$w_{Y1Z2} = w_{Y1Z2} + \mu * E_{Z2} * y_2 \quad (3.27)$$

$$w_{Y1Z3} = w_{Y1Z3} + \mu * E_{Z3} * y_3 \quad (3.28)$$

$$w_{Y1Z4} = w_{Y1Z4} + \mu * E_{Z4} * y_4 \quad (3.29)$$

Pesos de capa de oculta 1

$$w_{11} = w_{11} + \mu * E_{Y1} * x_1 \quad ; w_{12} = w_{12} + \mu * E_{Y2} * x_1 \quad (3.30)$$

$$w_{13} = w_{13} + \mu * E_{Y3} * x_1 \quad ; w_{14} = w_{14} + \mu * E_{Y4} * x_1 \quad (3.31)$$

$$w_{21} = w_{21} + \mu * E_{Y1} * x_2 \quad ; w_{22} = w_{22} + \mu * E_{Y2} * x_2 \quad (3.32)$$

$$w_{23} = w_{23} + \mu * E_{Y3} * x_2 \quad ; w_{24} = w_{24} + \mu * E_{Y4} * x_2 \quad (3.33)$$

$$w_{31} = w_{31} + \mu * E_{Y1} * x_3 \quad ; w_{32} = w_{32} + \mu * E_{Y2} * x_3 \quad (3.34)$$

$$w_{33} = w_{33} + \mu * E_{Y3} * x_3 \quad ; w_{34} = w_{34} + \mu * E_{Y4} * x_3 \quad (3.35)$$

El factor de aprendizaje para el reajuste de los pesos se considera entre 0 y 1.

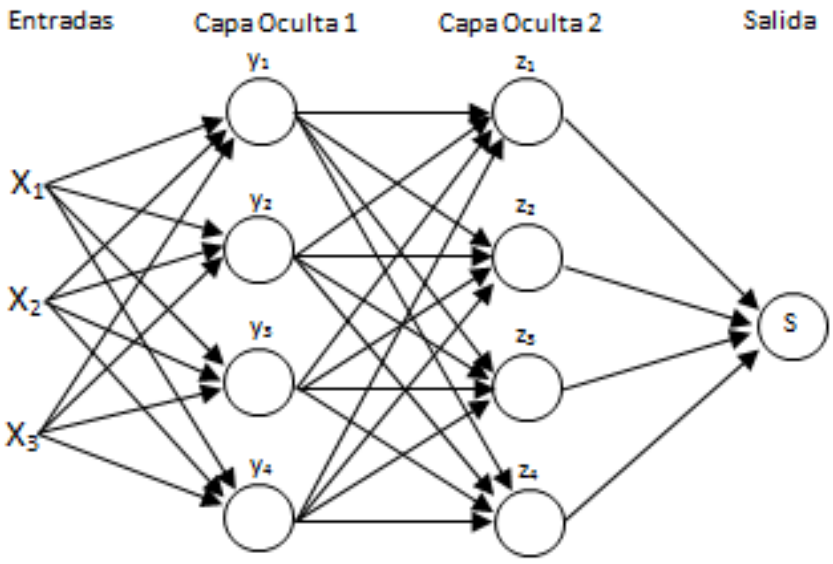


Figura 3.6 Red Neuronal

CAPÍTULO IV
ANALISIS Y RESULTADOS

En este último capítulo se presentan los resultados del entrenamiento y del desarrollo de la tesis, a continuación de detalla el orden del desarrollo del presente capitulo:

- En el punto 4.1, se presenta el análisis del entrenamiento de la red neuronal, tanto para 01 y 02 capas ocultas, con diferentes valores de factor de aprendizaje y numero de neuronas en cada capa mencionada.
- En el punto 4.2, se presenta los resultados del entrenamiento de la red neuronal, comparativos d1 parámetros de la red y la estructura final de la red neuronal.
- En el punto 4.3, se presentan, la contrastación de las hipótesis primaria y secundarias.

4.1. Análisis del entrenamiento de la red neuronal

A continuación, se presentan los resultados obtenidos mediante simulación para ello se va tomar diferentes capas ocultas y numero de neuronas por cada capa.

A continuación, se presenta el caso de 01 capa oculta y tasa de aprendizaje 0.001

Tabla 4.1 Parámetros de la Red Neuronal – Entrenamiento 1

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	20
Número de Capas Ocultas	1
Número de Épocas	1000
Factor de Aprendizaje	0.001
Error	3.01%

Para este caso se está considerando un entrenamiento de 1000 épocas y una tasa de aprendizaje de 0.001 en donde se puede visualizar en la figura 4.1 que la red se aproxima a la salida real, así mismo en la figura 4.2 se puede visualizar el error de la red del 3.01%.

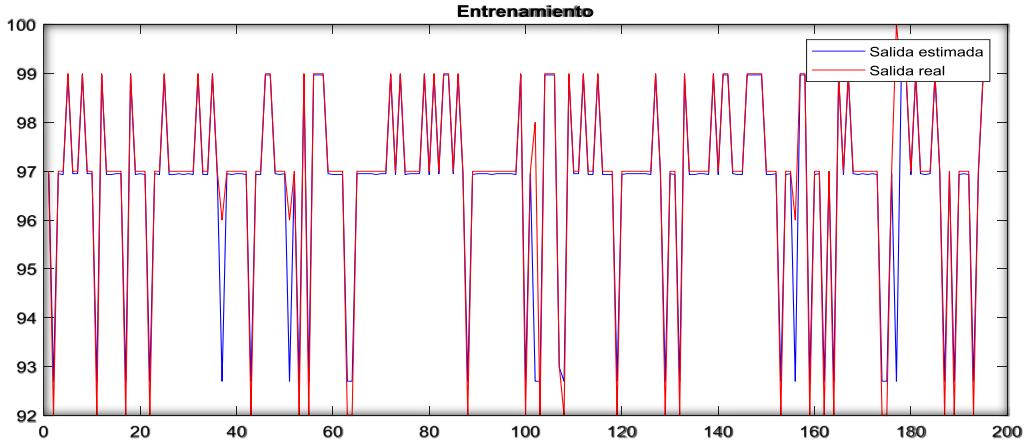


Figura 4.1 Comparación del entrenamiento de la Red Neuronal

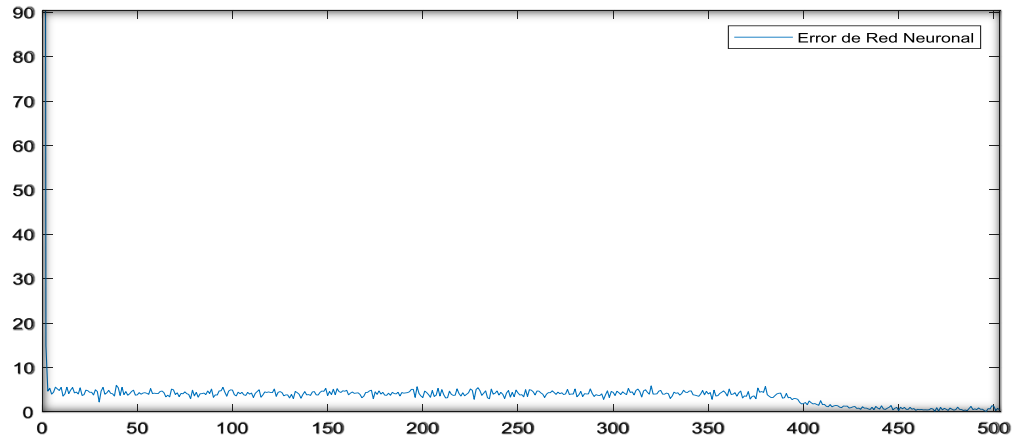


Figura 4.2 Error de la Red Neuronal – Entrenamiento1

Al graficar la salida real y la estimado podemos observar que el valor estimado presenta un error mayor, ello podemos visualizarlo en la figura 4.1, este entrenamiento es para una capa oculta con 20 neuronas. En la figura 4.2 se visualiza el error de la red en donde a medida que aumenta el número de épocas mayor a 400 el error tiende a cero.

A continuación, se presenta el caso de 01 capa oculta y tasa de aprendizaje 0.005

Tabla 4.2 Parámetros de la Red Neuronal – Entrenamiento 2

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	30
Número de Capas Ocultas	1
Número de Épocas	1000
Factor de Aprendizaje	0.005
Error	4.69%

Para este caso se está considerando un entrenamiento de 1000 épocas y una tasa de aprendizaje de 0.005 en donde se puede visualizar en la figura 4.3 que la red se aproxima a la salida real, así mismo en la figura 4.4 se puede visualizar el error de la red del 4.69%.

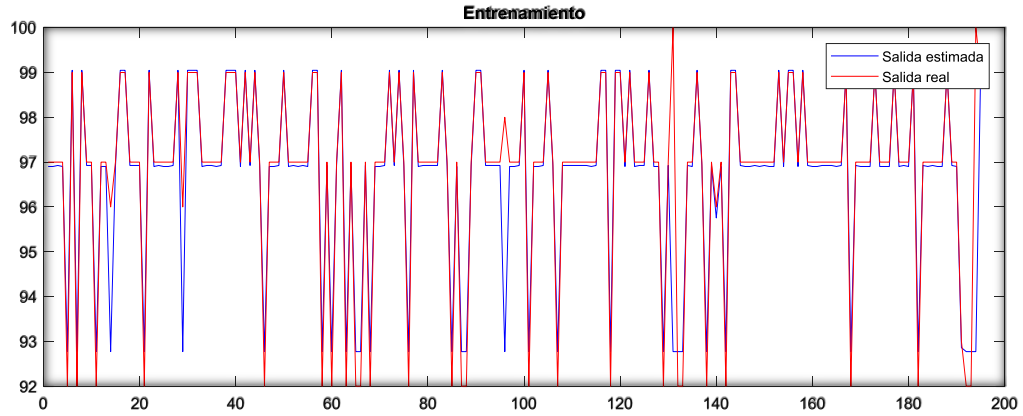


Figura 4.3 Comparación del entrenamiento de la Red Neuronal

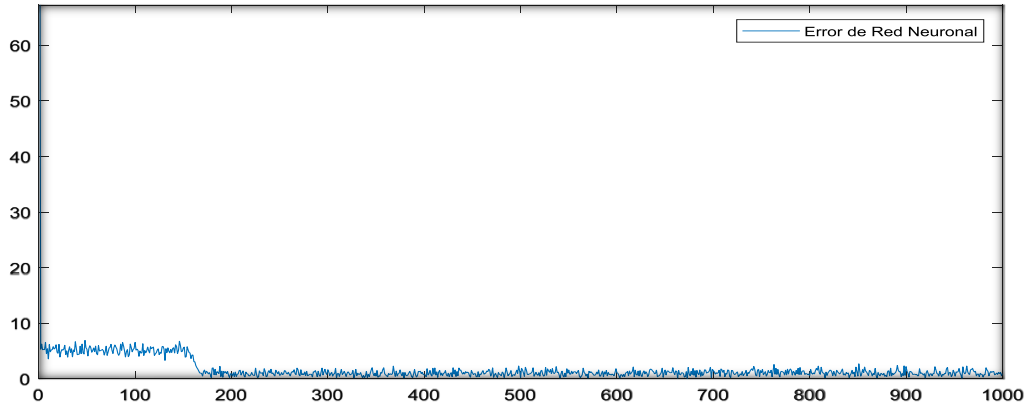


Figura 4.4 Error de la Red Neuronal – Entrenamiento 2

De la figura 4.3 podemos observar que si aumentamos el número de neuronas de capa oculta y el factor de aprendizaje el error aumenta.

A continuación, se presenta el caso de 01 capa oculta y tasa de aprendizaje 0.01

Tabla 4.3 Parámetros de la Red Neuronal – Entrenamiento 3

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	30
Número de Capas Ocultas	1
Número de Épocas	1000
Factor de Aprendizaje	0.01
Error	2.69%

Para este caso se está considerando un entrenamiento de 1000 épocas, con 30 neuronas en la capa oculta y una tasa de aprendizaje de 0.01 en donde se puede visualizar en la figura 4.5 que la red se aproxima a la salida real, así mismo en la figura 4.6 se puede visualizar el error de la red del 2.69%.

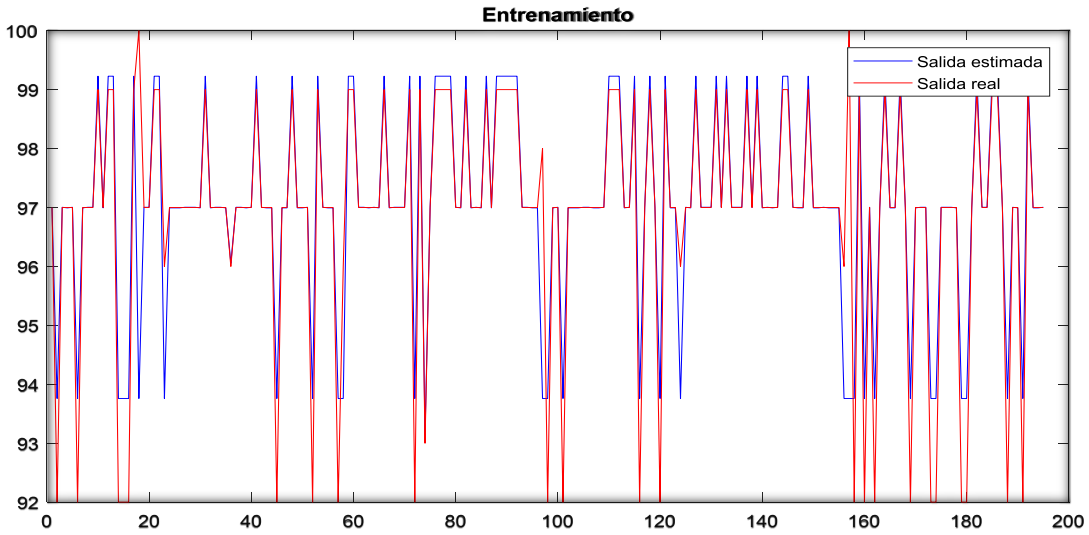


Figura 4.5 Comparación del entrenamiento de la Red Neuronal

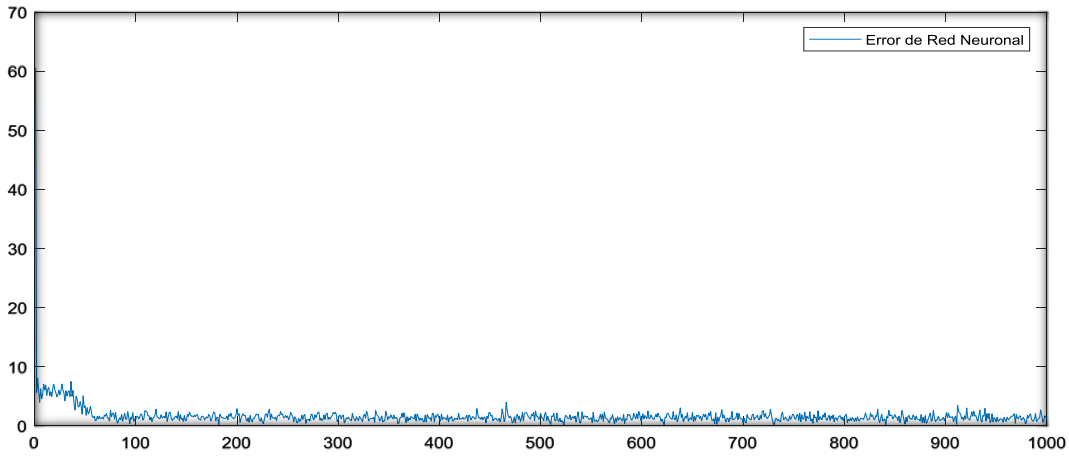


Figura 4.6 Error de la Red Neuronal – Entrenamiento3

El aumentar el factor de aprendizaje a 0.01 podemos ver que el error tiende a disminuir, pero se puede notar que hay varios valores distorsionados ello se puede notar en la figura 4.5, sin embargo, en la figura 4.6 se puede notar que el error disminuye de manera rápida es decir en la época 100 ya presenta una tendencia al valor de 2.69%.

A continuación, se presenta el caso de 01 capa oculta y tasa de aprendizaje 0.1

Tabla 4.4 Parámetros de la Red Neuronal – Entrenamiento 4

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	30
Número de Capas Ocultas	1
Número de Épocas	1000
Factor de Aprendizaje	0.1
Error	NA

Para el siguiente caso se está considerando un entrenamiento de 1000 épocas, 30 neuronas en la capa oculta y una tasa de aprendizaje de 0.1.

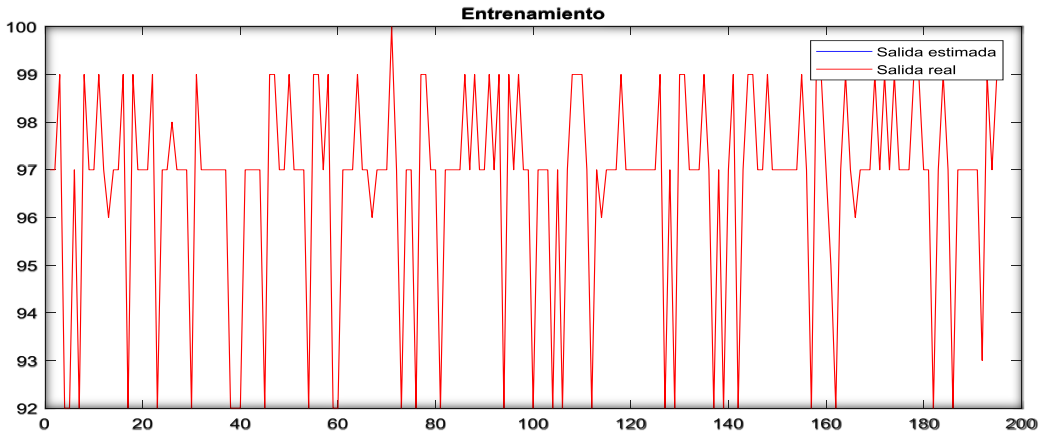


Figura 4.7 Comparación del entrenamiento de la Red Neuronal

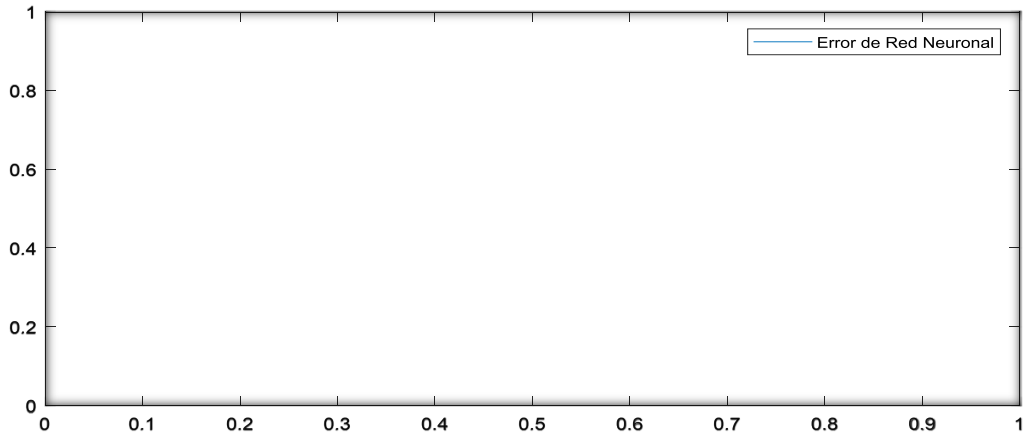


Figura 4.8 Error de la Red Neuronal – Entrenamiento4

De la figura 4.7 podemos notar que si seguimos aumentando el factor de aprendizaje el valor del error tiene a subir es decir la red neuronal no logra el aprendizaje deseado de

la salida real y en este caso esta des aprendiendo, ello podemos visualizar en la figura 4.8, el error es indeterminado.

Ahora analizamos el caso de 01 capa oculta con 5 neuronas en cada capa oculta y tasa de aprendizaje 0.005

Tabla 4.5 Parámetros de la Red Neuronal – Entrenamiento 5

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	5
Número de Capas Ocultas	1
Número de Épocas	1000
Factor de Aprendizaje	0.005
Error	4.37 %

Para el entrenamiento 05 se está considerando un entrenamiento de 1000 épocas, con 05 neuronas en la capa oculta y una tasa de aprendizaje de 0.005.

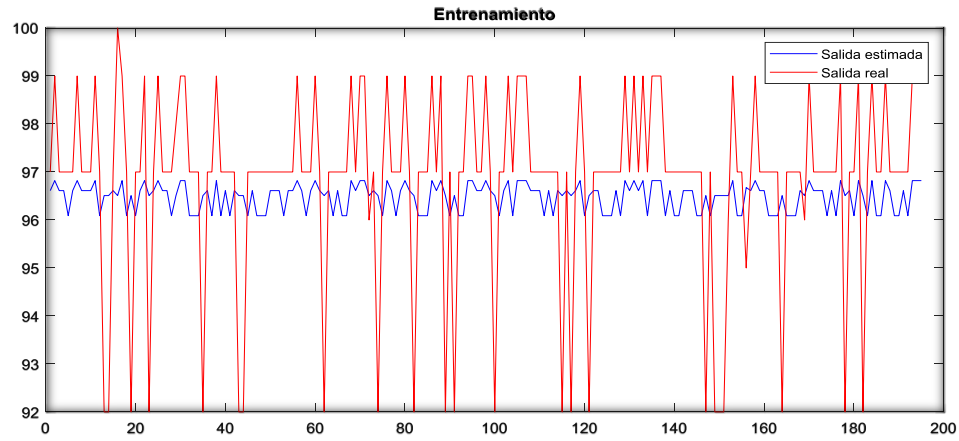


Figura 4.9 Comparación del entrenamiento de la Red Neuronal

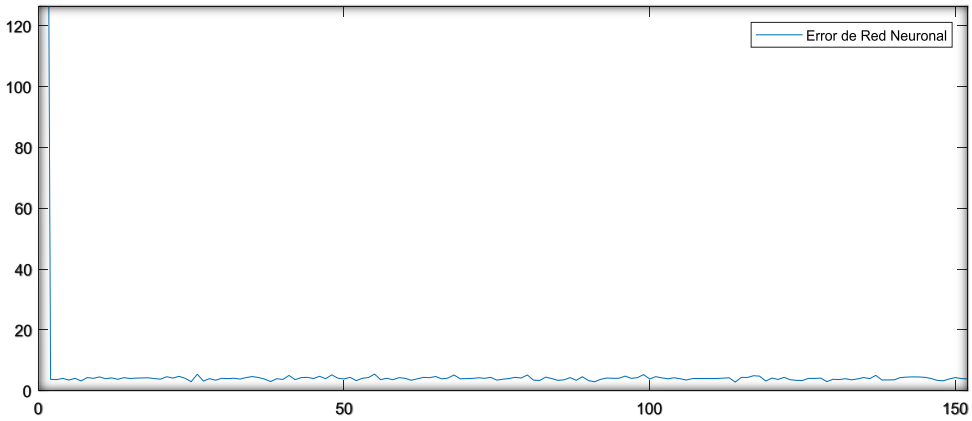


Figura 4.10 Error de la Red Neuronal – Entrenamiento 5

Al reducir el número de capas ocultas se logra un error de 4.37 y la red se aproxima a la salida real, si continuamos con el ajuste podemos lograr una mejor un menor error.

Ahora analizamos el caso de 02 capa oculta con 5 neuronas en cada capa oculta y tasa de aprendizaje 0.001

Tabla 4.6 Parámetros de la Red Neuronal – Entrenamiento 6

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	5
Número de Capas Ocultas	2
Número de Épocas	1000
Factor de Aprendizaje	0.001
Error	1.65 %

Para el este entrenamiento se está considerando, 5 neuronas en cada capa oculta y una tasa de aprendizaje de 0.001 y 1000 épocas de entrenamiento.

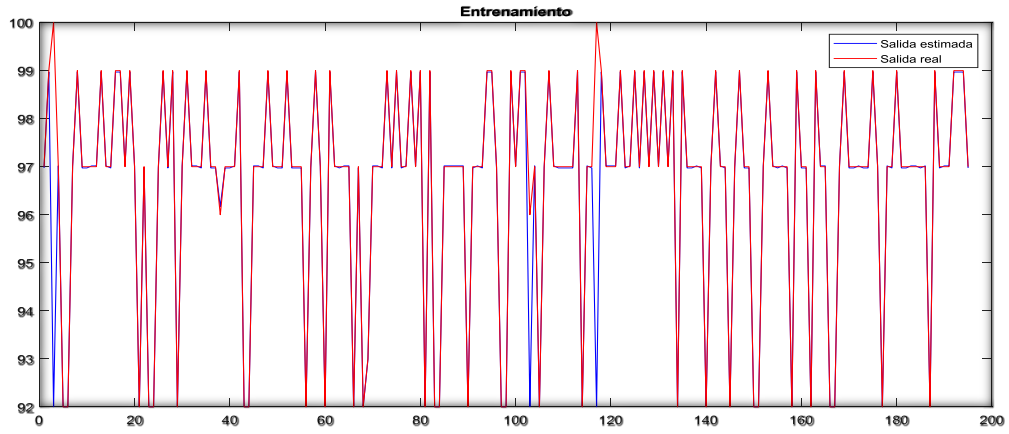


Figura 4.11 Comparación del entrenamiento de la Red Neuronal

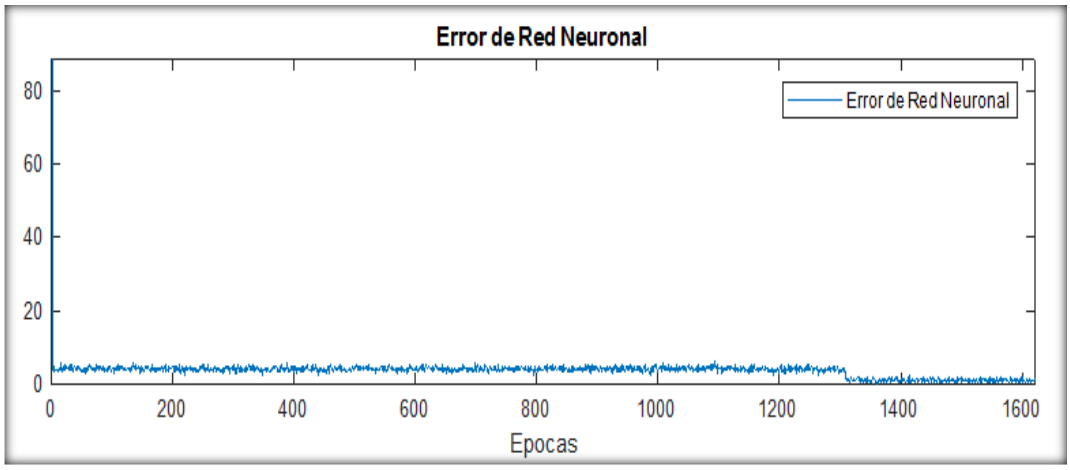


Figura 4.12 Error de la Red Neuronal – Entrenamiento 6

En figura 4.11 se puede visualizar que la red se aproxima a la salida real, así mismo en la figura 4.12 se puede visualizar el error de la red del 1.65%, el cual es el adecuado.

Se tiene el siguiente caso de 02 capa oculta con 3 neuronas en cada capa oculta y tasa de aprendizaje 0.001

Tabla 4.7 Parámetros de la Red Neuronal – Entrenamiento 7

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	3
Número de Capas Ocultas	2
Número de Épocas	1000
Factor de Aprendizaje	0.001
Error	4.26 %

Para este caso se está considerando un entrenamiento de 1000 épocas, con 03 neuronas por cada capa oculta y una tasa de aprendizaje de 0.001.

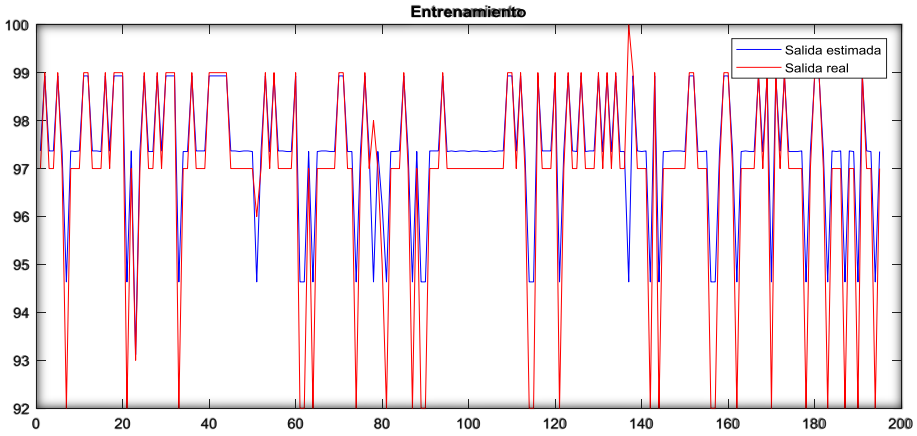


Figura 4.13 Comparación del entrenamiento de la Red Neuronal

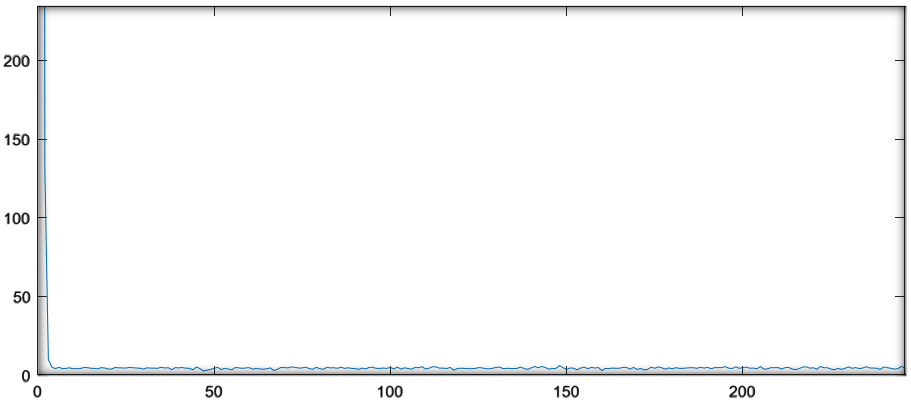


Figura 4.14 Error de la Red Neuronal – Entrenamiento 7

En la figura 4.13 se puede visualizar que la red no se aproxima a la salida real, así mismo en la figura 4.14 se visualizar el error de la red del 4.26% y ello se logra en una menor cantidad de épocas y este se mantiene constante.

A continuación, analizamos el caso de 02 capas ocultas con 7 neuronas en cada capa oculta y tasa de aprendizaje 0.001

Tabla 4.8 Parámetros de la Red Neuronal – Entrenamiento 8

Parámetro	Condición Inicial
Número de neuronas de la Capas Ocultas	7
Número de Capas Ocultas	2
Número de Épocas	1000
Factor de Aprendizaje	0.001
Error	2.25 %

Para este caso se está considerando un entrenamiento de 1000 épocas, con 07 neuronas por cada capa oculta y una tasa de aprendizaje de 0.001.

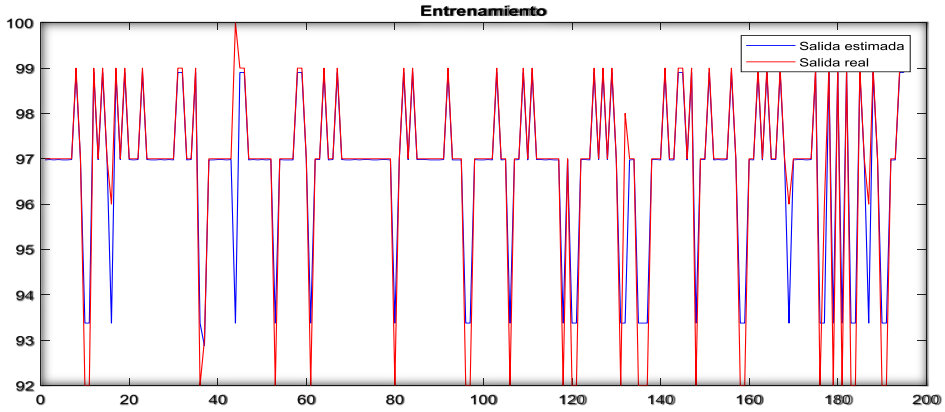


Figura 4.15 Comparación del entrenamiento de la Red Neuronal

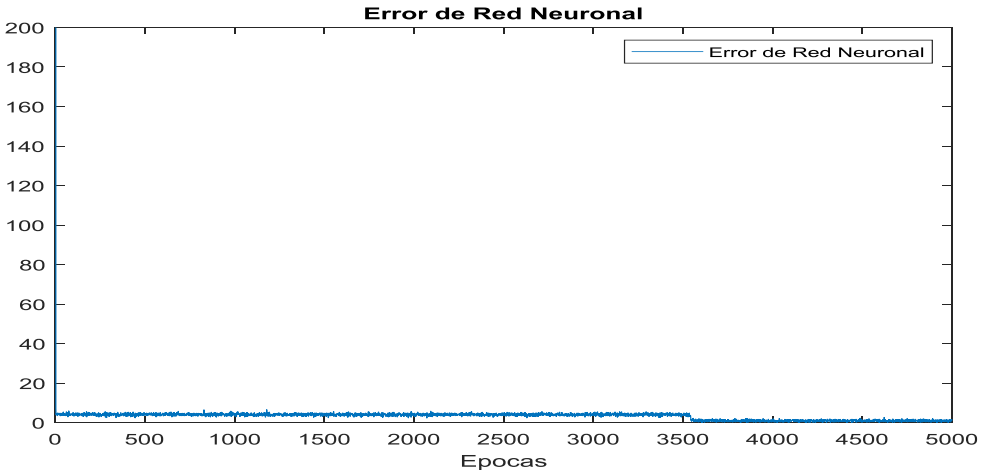


Figura 4.16 Error de la Red Neuronal – Entrenamiento 8

Para este caso se está considerando un entrenamiento de 1000 épocas, con 07 neuronas en cada capa oculta y una tasa de aprendizaje de 0.001 en donde se puede visualizar en la figura 4.15 que la red se aproxima a la salida real, así mismo en la figura 4.16 se puede visualizar el error de la red del 2.25%. Se está considerando un entrenamiento para una red de 2 capas ocultas

4.1.1. Validaciones de las redes neuronales entrenadas

Para la validación de la red neuronal se tomaron las siguientes consideraciones:

- Se implementa la función de divide dato esta función va permitir escoger de manera aleatoria el 30 % de datos de entrada con ello vamos a poder validar de manera correcta, cabe resaltar que dicha metodología va permitir realizar una mejor validación.
- Se va considerar los parámetros considerados en el entrenamiento.
- Se va considerar los pesos calculados de la red entrenada el cual cumple la condición de menor porcentaje de error.
- Se está considerando la misma función de activación usada en el entrenamiento de la red neuronal.

4.1.1.1 Red Neuronal 02 capas ocultas

03 neuronas en cada capa oculta

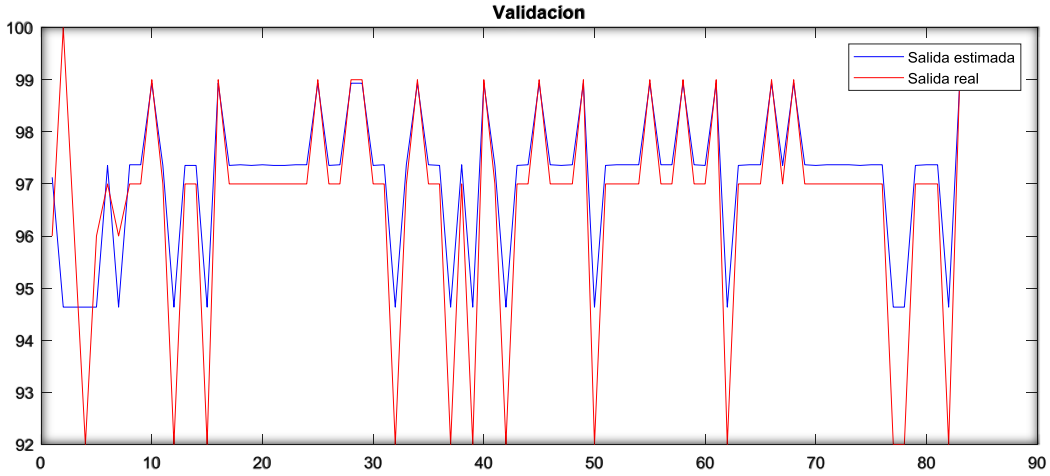


Figura 4.17 Validación de la Red Neuronal, con 3 neuronas en cada capa oculta y con un error de 4.26%

Al realizar el test de esta reda notamos que la salida estimada no es la deseada, se logran valores menores al valor real y se obtiene un mayor error de peso. En algunos casos la red sobrepasa al valor deseado, tal como se visualiza en la figura 4.17.

A continuación, vemos la gráfica de 05 neuronas en cada capa oculta

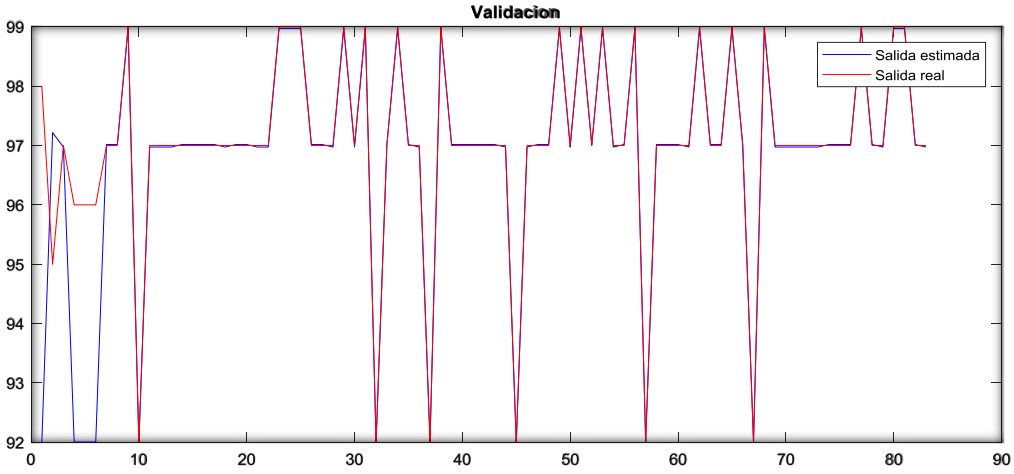


Figura 4.18 Validación de la Red Neuronal, con 5 neuronas en cada capa oculta y con un error de 1.65%

En la figura 4.18 se logra un menor error de peso, la red neuronal logra aproximarse al valor deseado, esto es comprobado con la data para validar los parámetros hallados en el entrenamiento de la red.

A continuación, presentamos la gráfica de 07 neuronas en cada capa oculta

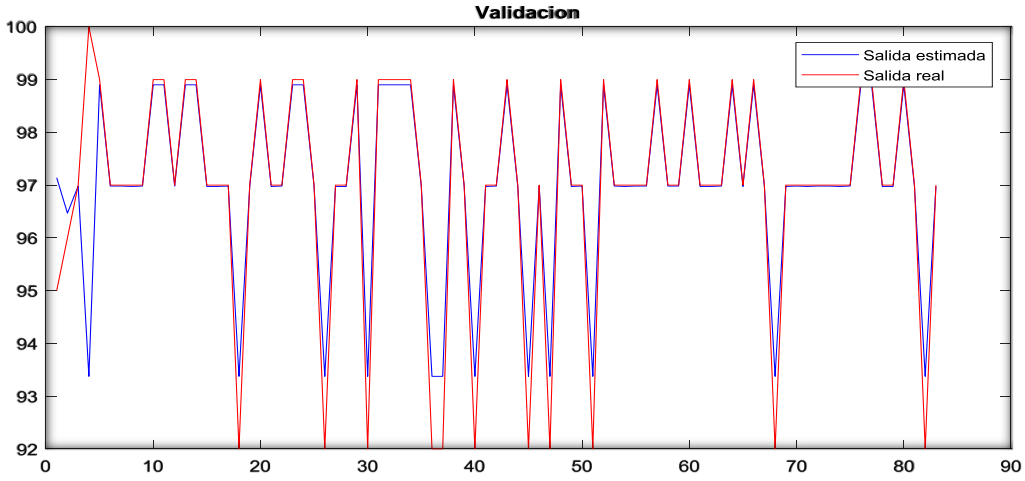


Figura 4.19 Validación de la Red Neuronal, con 7 neuronas en cada capa oculta y con un error de 2.25%.

Por último, validamos la red de 7 neuronas en la capa oculta y se logra una gráfica similar a la deseada, pero con un error de 2.25% el cual está considerado dentro de lo esperado. Cabe resaltar que los pesos de la red fueron los hallados en el entrenamiento.

4.1.1.2 Red Neuronal 01 capa oculta

A continuación, presentamos la gráfica de 05 neuronas en cada capa oculta

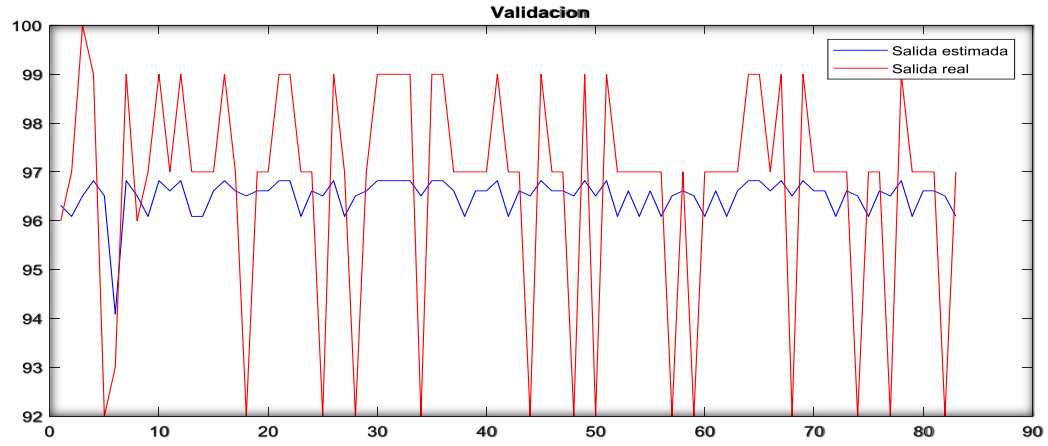


Figura 4.20 Red Neuronal, con 5 neuronas en cada capa oculta, error de 4.37%

De la figura 4.20 podemos notar que la salida real no es la deseada, se logran valores con un error de 4.37% lo cual indica que faltaría peso, dado que en todos os valores hallados el peso es menor que el peso real.

A continuación, presentamos la gráfica de 30 neuronas en cada capa oculta y factor de aprendizaje de 0.1

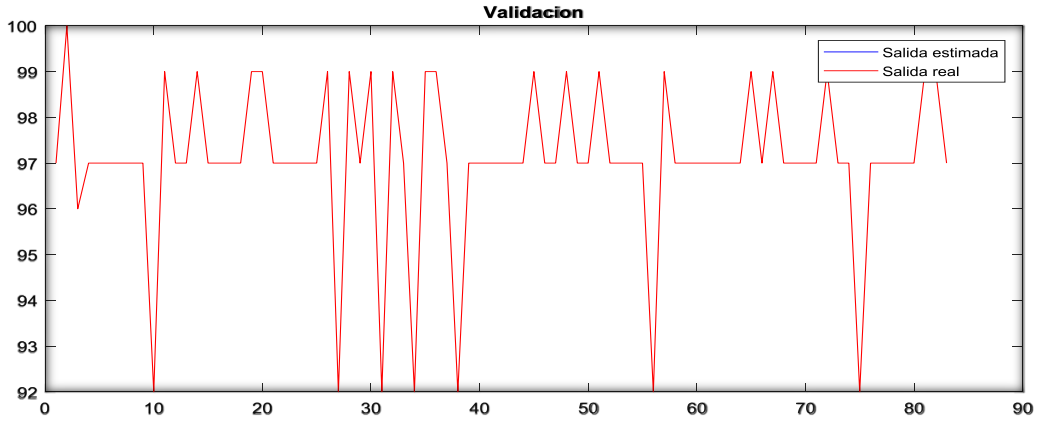


Figura 4.21 Red Neuronal, con 30 neuronas en cada capa oculta y con un error de fuera del rango

De la figura 4.20 notamos que la red neuronal no llega a realizar el aprendizaje adecuado no se puede visualizar o calcular la salida deseada, este modelo de red no es el adecuado.

A continuación, presentamos la gráfica de 30 neuronas en cada capa oculta y factor de aprendizaje de 0.01

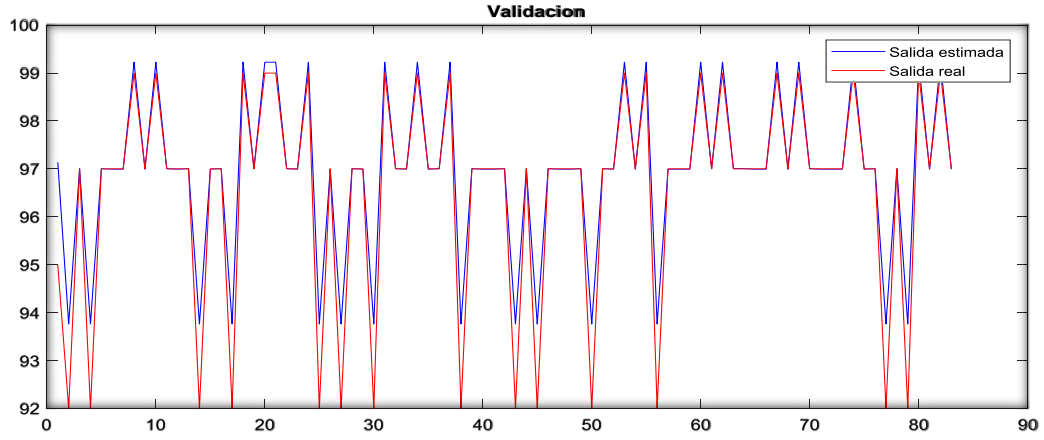


Figura 4.22 Red Neuronal, con 30 neuronas en cada capa oculta y con un error de 2.69%

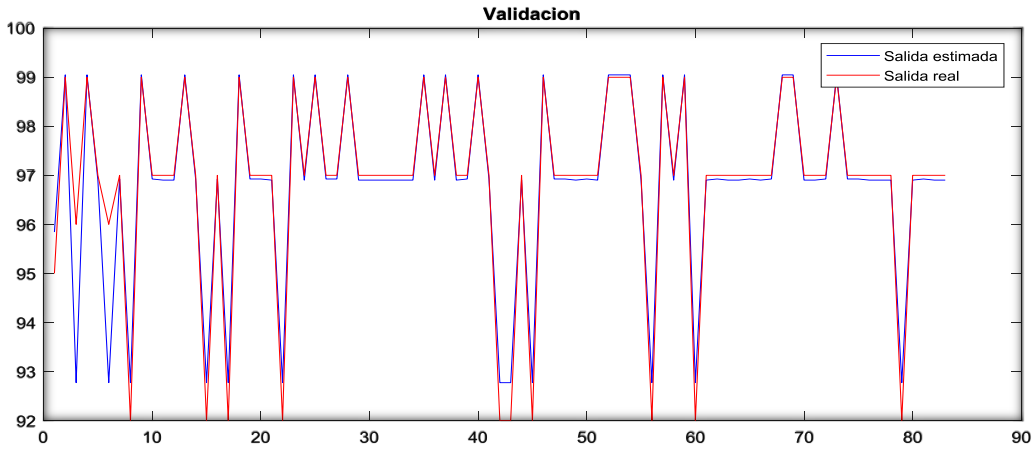


Figura 4.23 Validación de la Red Neuronal, con 30 neuronas en cada capa oculta, factor de aprendizaje de 0.005 y con un error de 4.69%

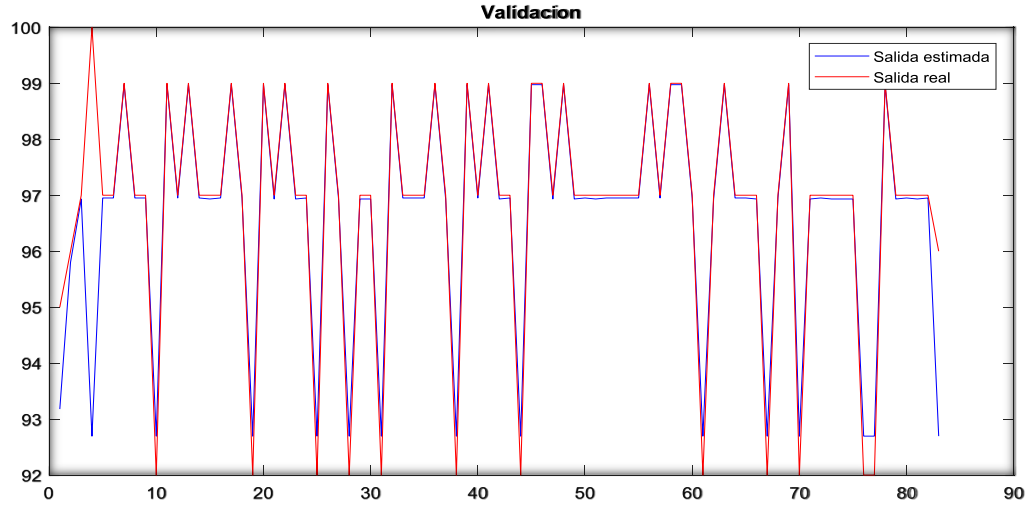


Figura 4.24 Validación de la Red Neuronal, con 30 neuronas en cada capa oculta, factor de aprendizaje e 0.001 y con un error de 3.01%

Al validar la red neuronal de 1 capa con diferentes neuronas de capa oculta y diferente factor de aprendizaje se encontró 02 valores de error mínimo uno de ellos es de 2.69% pero la gráfica deseada difiere en varios puntos, ver figura 4.22.

Por ende, la otra red con las mismas neuronas, pero con un factor de aprendizaje de 0.001, el cual es mucho menor a la anterior se logra una mejor salida deseada en donde solo difiere en 3 puntos, ver figura 4.24, en caso se use una red neuronal de 01 capa oculta se pude considerar esta red como optimo, dado que el error es de 3.01,

4.2. Resultados del entrenamiento de la red neuronal

De los datos de entrenados se tomaron diferentes parámetros en base a las metodologías mencionadas, del total de datos se tomaron el 70% para el entrenamiento y el 30% para la validación de la red, a continuación, se muestra el resumen de los resultados.

Tabla 4.9 Resultados de entrenamientos para una red neuronal con 01 capa oculta y 1000 épocas

Red Neuronal	**Entrenamiento 1**	**Entrenamiento 2**	**Entrenamiento 3**	**Entrenamiento 4**	**Entrenamiento 5**
Número de capas ocultas	1	1	1	1	1
Número de Neuronas en capa oculta	5	20	30	30	30
Número de Épocas	1000	1000	1000	1000	1000
Tiempo (Segundos)	12.59	9.85	9.14	11.39	9.35
Factor de Aprendizaje	0.005	0.001	0.005	0.01	0.1
Error de Red	4.37 %	3.01 %	4.69%	2.69 %	No entrena

De la tabla 4.9 se puede observar que la red óptima para la red neuronal de 01 capa oculta es la que cuenta con 30 neuronas en dicha capa, cabe resaltar que al tener una cantidad elevada de neuronas por capa hace que la red neuronal sea muy lenta al momento de realizar el cálculo. Así mismo en el entrenamiento 5 la red no aprende debido a que el ajuste del factor de aprendizaje fue mayor a 0.01, mientras que el número de neuronas y capas ocultas es la misma, por lo cual la red no aprende de manera eficiente.

Tabla 4.10 Resultados de entrenamientos para una red neuronal con 02 capa oculta y 1000 épocas

Red Neuronal	**Entrenamiento 1**	**Entrenamiento 2**	**Entrenamiento 3**
Número de capas ocultas	2	2	2
Número de Neuronas en capa oculta	3	5	7
Número de Épocas	1000	1000	1000
Tiempo (Segundos)	45.25	55.24	67.54
Factor de Aprendizaje	0.001	0.001	0.001
Error de Red	4.26%	1.65%	2.25%

De la tabla 4.10 se puede observar que la red óptima para una red neuronal de 02 capas ocultas es la que cuenta con 05 neuronas en cada capa oculta, no se considera la red neurona de 03 capas dado que el error es muy alto.

4.2.1. Comparativo de parámetros de red neuronal

Para la elección de los parámetros de la red se usaron la metodología planteada en el capítulo 3 para el cálculo de capas ocultas y numero de neuronas ocultas.

Tabla 4.11 Resultados de entrenamientos

Red Neuronal	**Método Hirose**	**Método ensayo Error**	**Método Hirose**	**Método ensayo Error**
Número de capas ocultas	1	1	2	2
Número de Neuronas en capa oculta	20	30	7	5
Número de Épocas	1000	1000	1000	1000

4.2.1. Comparativo de resultados

En la tabla 4.11 se puede apreciar los resultados de algunos trabajos recientes encontrados en la literatura.

Tabla 4.12 Resultados de entrenamiento con otros trabajos

Autor	**Metodología**	**Sistema**	**Error**
Bingying Li [1]	Red Neuronal	Pesaje Dinámico	1.0 %
Mahajan [20]	Logica Fuzzi	Pesaje Dinámico	4.5 %
Propuesto	Red Neuronal	Pesaje Dinámico	1.65 %

En [1] Bingying usa redes neuronales para la reducción de error donde la red que plantea cuanta con 01 sola capa oculta y con 8 neuronas en dicha capa oculta el cual logra reducir el error en un 80% del error encontrado antes del uso de la red neuronal. En [20] para mejorar el margen de error de pesaje usa un controlador PI lógico fuzzi, pero dicho control es para la velocidad de la faja y con ello se logra una alimentación uniforme a la banda transportadora logrando una reducción de error del 40% del error inicial.

De los casos que se analizan se observa que la red neuronal de 02 capas ocultas y con 05 neuronas logra obtener un menor error que el de una red neurona de 07 neuronas en cada capa, ello se puede visualizar en tabla 4.10, sim embargo una red neuronal con 01 capa oculta para lograr un error menor al 3% requiere 30 neuronas en la capa oculta. Así mismo se considera mejor en comparación a los otros métodos propuestos, debido a la pocas neuronas que tiene por cada capa oculta por lo cual su procesamiento de datos

es menor en comparación a los demás y no necesita de un hardware muy sofisticado para poder implementarlo.

Un factor de aprendizaje mayor a 0.09 hace que la red no logre un entrenamiento adecuado es decir no cumple una propiedad fundamental de la red neuronal la de generalizar, ello se puede aprecia en la figura 4.8 y 4.21.

4.2.2. Estructura de la red Neuronal

A continuación, se presenta los parámetros de la red neuronal.

Tabla 4.13 Parámetros de la Red Neuronal

Parámetro	Estructura
Número de neuronas de la Capas Ocultas	5
Número de Capas Ocultas	2
Número de Épocas	1000
Factor de Aprendizaje	0.001
Error	1.65 %

Una vez realizado el entrenamiento de la red neuronal el cumple con la condición planteada, se procede a realizar la validación el cual, para verificar el margen de error, ver figura 4.18. A continuación se detalla los parámetros y estructura de la red neuronal con sus respectivos pesos.

Tabla 4.14 Pesos y bías de la Red Neuronal entrenada

Capa entrada		**Capa Oculta 1**		**Capa oculta 2**		**Capa Salida**
Capa Entrada	Entrada1	1.0818	4.2303	-4.2669	0.0324	0.8366
	Entrada2	-0.1189	0.4572	-4.5168	0.1881	1.9396
	Bías	-0.2479	0.5840	-1.7065	1.5815	-0.7779
Capa Oculta	Neurona1	0.1722	-0.4346	-0.0599	0.3534	0.1733
	Neurona2	0.2439	0.4618	0.3564	0.1987	0.2345
	Neurona3	1.7825	1.6337	1.7697	1.7945	1.6636
	Neurona4	0.3024	-0.0533	-0.4781	-0.1392	0.4014
	Neurona5	1.3504	1.7947	1.5801	1.6658	1.9892
	Bías	1.4258	1.9597	2.2	2.0077	1.5646
Capa Salida	Salida	16.5734	16.7096	16.1049	16.2147	16.6600
	Bías	16.58				

De la tabla 4.14 se tiene los pesos calculados para la red neuronal entrenada, así como también el bías, con estos valores se puede realizar el cálculo para otros valores de pesajes que se requieran. La red propuesta se está considerando de 02 capas ocultas con 05 neuronas en cada capa oculta con ello se va lograr reducir el error de pesaje menor al 3%.

Tabla 4.15 Pesos Real y Peso calculado con la Red Neuronal

N°	Peso Real	Peso Entrenado	Error	N°	Peso Real	Peso Entrenado	Error
1	98.00	93.37156	4.628	43	99.00	98.95121	0.049
2	95.00	97.13663	-2.137	44	99.00	98.95121	0.049
3	97.00	97.14219	-0.142	45	92.00	93.37156	-1.372
4	92.00	93.37156	-1.372	46	97.00	97.14219	-0.142
5	93.00	97.98944	-4.989	47	99.00	98.95121	0.049
6	96.00	93.37156	2.628	48	97.00	97.17638	-0.176
7	97.00	97.14219	-0.142	49	97.00	97.17638	-0.176
8	100.00	93.37156	6.628	50	97.00	97.14219	-0.142
9	97.00	97.14219	-0.142	51	97.00	97.14219	-0.142
10	97.00	97.17638	-0.176	52	97.00	97.14219	-0.142
11	97.00	97.14219	-0.142	53	97.00	97.17638	-0.176
12	92.00	93.37156	-1.372	54	97.00	97.17638	-0.176
13	99.00	98.95121	0.049	55	99.00	98.95121	0.049
14	97.00	97.17638	-0.176	56	97.00	97.14219	-0.142
15	97.00	97.17638	-0.176	57	97.00	97.17638	-0.176
16	97.00	97.14219	-0.142	58	97.00	97.14219	-0.142
17	92.00	93.37156	-1.372	59	92.00	93.37156	-1.372
18	97.00	97.17638	-0.176	60	99.00	98.95121	0.049
19	97.00	97.14219	-0.142	61	99.00	98.95121	0.049
20	97.00	97.17638	-0.176	62	97.00	97.14219	-0.142
21	97.00	97.17638	-0.176	63	97.00	97.14219	-0.142
22	92.00	93.37156	-1.372	64	92.00	93.37156	-1.372
23	99.00	98.95121	0.049	65	99.00	98.95121	0.049
24	97.00	97.14219	-0.142	66	99.00	98.95121	0.049
25	99.00	98.95121	0.049	67	92.00	93.37156	-1.372
26	97.00	97.14219	-0.142	68	97.00	97.14219	-0.142
27	97.00	97.14219	-0.142	69	97.00	97.17638	-0.176
28	99.00	98.95121	0.049	70	97.00	97.14219	-0.142
29	97.00	97.14219	-0.142	71	92.00	93.37156	-1.372
30	99.00	98.95121	0.049	72	97.00	97.14219	-0.142
31	97.00	97.14219	-0.142	73	97.00	97.17638	-0.176
32	97.00	97.17638	-0.176	74	92.00	93.37156	-1.372
33	92.00	93.37156	-1.372	75	97.00	97.14219	-0.142
34	97.00	97.14219	-0.142	76	92.00	93.37156	-1.372
35	97.00	97.17638	-0.176	77	99.00	98.95121	0.049
36	97.00	97.17638	-0.176	78	97.00	97.14219	-0.142
37	97.00	97.17638	-0.176	79	97.00	97.14219	-0.142
38	97.00	97.14219	-0.142	80	99.00	98.95121	0.049
39	92.00	93.37156	-1.372	81	97.00	97.14219	-0.142
40	97.00	97.14219	-0.142	82	97.00	97.17638	-0.176
41	99.00	98.95121	0.049	83	97.00	97.17638	-0.176
42	97.00	97.17638	-0.176		3,962.00	3,974.83	-12.833
	4,053.00	4,058.108	-5.11	Peso Total		8,015.00	
				Peso de la Red		8,032.94	

De la tabla 4.15 se puede observar la diferencia de peso entre el peso real y el peso hallado con la red neuronal una vez que esta fue entrenada. Así mismo se puede verificar dicha red neuronal con una mayor cantidad de pesos.

4.3. Contrastación de la hipótesis

La hipótesis principal plantea que la metodología de redes neuronales permitirá disminuir el error de pesaje en una faja transportadora de mineral lo cual se está logrando con los resultados mostrados en la sección 4.1 donde se muestra que es posible lograr reducir el error de pesaje menor al 2% mediante el uso de red neuronal, por lo que esto valida la hipótesis planteada.

La primera hipótesis secundaria plantea que la aplicación de la Red Neuronal permitirá generar un modelo óptimo para disminuir el error de pesaje en la faja de llenado de mineral, los modelos generados para la red neuronal se plantean en la sección 4.1.1 donde se plantea 02 modelos de red neuronal con un margen de error menor al 3% por lo que hace que el modelo de red neuronal sea óptimo para la reducción de margen de error. Por ende, ello valida la hipótesis planteada.

La segunda hipótesis secundaria plantea que a partir del entrenamiento de la red neuronal es posible disminuir el error de pesaje en la faja de llenado de mineral validando el peso actual con el requerido. Los resultados se muestran en la sección 4.2.3 donde se muestra que se consiguió un error menor al 2%, así mismo se plantea los parámetros de la red neuronal los cuales fueron hallados durante el entrenamiento y posteriormente validado. Por ende, se valida la hipótesis planteada.

CONCLUSIONES

1. Se determinó que el algoritmo hallado logra compensar el error de pesaje, para lo cual el modelo y arquitectura de la red neuronal encontrado cumple con la compensación de error, el cual es de 1.65%. No obstante, se obtuvo otra red neuronal con 7 neuronas y con un error de 2.25%.

2. Mediante la ejecución de los 1000 entrenamientos la red neuronal logra calcular los pesos sinápticos que van a lograr disminuir el margen de error de pesaje. Dichos pesos sinápticos son validados por valores de entrada aleatoria. Así mismo se encontró dos modelos de red neuronal de 02 y 01 capa oculta, con 5 neuronas en cada capa oculta, donde la red neuronal de 2 capas ocultas logra una reducción de error de pesaje de 1.65%.

3. Con el uso del Software Matlab se calculó los pesos sinápticos de la red optima, luego se comprobó que la red neuronal mejora la reducción de error de pesaje verificando la reducción del margen de error de pesaje en la validación de la red neuronal, lo cual se ha validado con el 30% de datos iniciales el cual fueron tomados de manera aleatoria.

4. Los resultados obtenidos en el escenario de una red de una capa oculta aportan una importante información sobre la variación del factor de aprendizaje dado que al variar el factor de aprendizaje se logra una reducción mínima de error de pesaje mínima de 2.6% sin embargo, al aumentar el valor de factor de aprendizaje la red neuronal no aprende.

RECOMENDACIONES

1. Se recomienda a los futuros investigadores que para un funcionamiento continuo de la faja se debe realizar pruebas con otro tipo de red neuronal, tal como la red neuronal recurrente, esta red posea un algoritmo secuencial que almacena la información anterior por ende puede mejorar la reducción de error de pesaje en este tipo de funcionamiento continuo.

2. Se recomienda migrar al código Python o código C para poder lograr el funcionamiento en tiempo real con sus respectivos filtros y amplificadores y con interfaz para PC. Así mismo dichos softwares cuentan con librerías más sofisticadas y por ende hacen más rápida y versátil el entrenamiento e incluso de uso comercial.

3. Se recomienda a los usuarios de este sistema de pesaje que la alimentación del mineral hacia la faja de llenado debe ser de manera constante para lograr un mejor pesaje. Con ello se va lograr optimizar el tiempo y mantener una carga de pesaje distribuida.

BIBLIOGRAFÍA

[1] **Bingying L., Yongxin L., Haita W., Yuming M., Qiang H., Fangli G**. (2018), Compensation of automatic weighing error of belt weigher based on BP neural network, School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing, China, paper Publicacion IEEE, Measurement

[2] **Berzal F.** (2018), Título del Libro REDES NEURONALES & DEEP LEARNING, Universidad de Granada, España 2018.

[3] **Cólas J.** (2019). Aplicación de modelos de redes neuronales a la predicción de la fiebre. Tesis Maestría, Universidad Politécnica de Madrid. España.

[4] **De Villiers J., Barnard E.** (2019). Backpropagation Neural Nets with One and Two Hidden Layers. IEEE transactions on Neural Networks, Vol. 4, No. 1, January 1992. South Africa, paper Publicación IEEE.

[5] **Erie M., Zainal A., Arief H., Mulyadi B.** (2018). The Effect of Object Position on Reading Error on Weighing Scale, Dept. of Mechanical Engineering Institute Technology Bandung, Indonesia, Editorial IEEE.

[6] **Espinoza M.** (1995) Diseño y construcción de una celda de carga, México Tesis Maestría, Universidad Autónoma de Nueva León, México.

[7] **Fukunaga K.** (1990) Computer Science and Scientific Computing Series - The artificial neural network book-Academic Press, Canada.

[8] **Galushkin I.,** (2007), Título del Libro Neural Network Theory-Springer, Moscow Institute of Physics & Technology, Rusia

[9] **Gurney K.** (2004) Título del Libro An introduction to neural networks This edition published in the Taylor & Francis e-Library, 2004, New York, EE UU.

[10] **Gestal M.** Introducción a las redes neuronales artificiales, Dpto. Tecnológico de la información y las comunicaciones, Universidad la Coruña, España.

[11] **Haijun L., Lucai W., Jingrong Y., Zhaosheng T., Houde D.** (2015). Nonlinear Error Compensation for Load Cells Based on the Optimal Neural Network with an Augmented Lagrange Multiplier., Hunan University, Changsha, China, Editorial IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT

[12] **Hernández J., Rodríguez J.** (2018), Algoritmos de Retro propagación con restricciones para la estimación de parámetros de curvas de titulación, Universidad de los Andes, Ciencia e Ingeniería, vol. 39, núm. 1. URL disponible en https://www.redalyc.org/jatsRepo/5075/507555109002/html/index.html

[13] **Hagan M., Demuth H., Beale M.** (2002), Libro Neural Network Design 2nd Edition. China.

[14] **Hemmat M., Toghraie D., Amoozad F.** (2023). Prediction of viscosity of MWCNT-Al2O3 (20:80)/SAE40 nano-lubricant using multi-layer artificial neural network (MLP-ANN) modeling. Engineering Applications of Artificial Intelligence 121 (2023) 105948. Irán, paper Publicación Elsevier.

[15] **Laurene F.** (1994), Libro Fundamental of Neural Networks, Architectures algorithms and applications, EE UU.

[16] **Maciej N., Przemysław P.** (2017), High-Precision FIR-Model-Based Dynamic Weighing System, IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT

[17] **Matich D., Ruiz C., Basuald M.** (2001), Redes Neuronales: Conceptos Básicos y Aplicaciones, Universidad Tecnológica Nacional – Facultad Regional Rosario Departamento de Ingeniería Química Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ), Argentina.

[18] **Monar M., William L.** (2014), Aplicación de las Redes Neuronales al Reconocimiento de Objetos en robot manipulador, Escuela Politécnica Nacional, Ecuador. Tesis Maestría.

[19] **Murray M., Miller W.** (1992) The bonded electrical resistance strain gage _ an introduction (Oxford University Press).

[20] **Nayana M., Dr. Sumant K., Dr. Sadanand B.** (2017), Design of PI Fuzzy Logic Controller to Overcome Belt Weigh Feeder System Nonlinearity, Yeshwantrao Chavan College of Engineering (YCCOE) RTM University, Nagpur, India, paper Publicación ICECA

[21] **Olabe X**. Redes Neuronales Artificiales y sus aplicaciones, Escuela Superior de Ingeniería Bilbao, España.

[22] **Olarte L.** Conogasi.org, (2018). Algoritmo, URL disponible en http://conogasi.org/articulos/algoritmo/.

[23] **Polanco F., Álvarez D., Moreno V.** (2015), Implementación mediante hardware de una Red Neuronal Artificial para reconocimiento de caracteres, RIELAC, Cuba.

[24] **Revista** del Banco Central de Reserva del Perú. Memoria 2017. URL disponible en https://www.bcrp.gob.pe/publicaciones/memoria-anual/memoria-2017.html

[25] **Rubiolo M.** (2014) Desarrollo de nuevos modelos y algoritmos basados en redes neuronales para tareas de minería de datos, Tesis Doctorado, Universidad Tecnológica Nacional, Argentina.

[26] **Serrano A., Soria J., Martin J.** (2009-2010) Libro, Redes Neuronales Artificiales, Escuela Técnica superior Ingeniería, Departamento de Ingeniería electrónica, España.

[27] **Servicio** de estudios BBVA Research. URL disponible en https://www.bbvaresearch.com/publicaciones/situacion-peru-cuarto-trimestre-2017/

[28] **Sharif Y., Hosseinpour M., Moghbeli A., Sharif H.** (2019). Lateral Torsional Buckling Capacity Assessment of Castellated Steel Beams Using Artifcial Neural Networks. International Journal of Steel Structures. Department of Computer Engineering, Vali-e-Asr University of Rafsanjan, Irán, paper Publicación Springer.

[29] **Siemens** (2019) Belt scales Militonics MSI and MMI, operating Instructions. Division Process Industries and Drives. URL disponible en https://support.industry.siemens.com/cs/attachments/109764407/

[30] **Stephen G.** (1993) Libro Neural Network learning and expert systems, EE UU.

[31] **Ugalde J.** (1997) Predicción de caudal y consumo en acueducto mediante Redes Neuronales de propagación hacia atrás, Revista de la Universidad de Costa Rica.

[32] **Vassaux L.** (2007) Instalación de Pesaje en Banda Transportadora para Totalizar la Producción de Productos terminados, Tesis Maestría, Nicaragua.

[33] **Wang Z., Chen Q., Wang Z., Xiong J.** (2022). The investigation into the failure criteria of concrete based on the BP neural network. Engineering Fracture Mechanics 275 (2022) 108835. China, paper Publicación SciendeDirect.

[34] **Wu Y., Cheng S., Li Y., Lv R., Min F**. (2023). STWD-SFNN: Sequential three-way decisions with a single hidden layer feedforward neural network. Information Sciences 632 (2023) 299–323. China, paper Publicación SciendeDirect.

[35] **Xiao** – Hu Yu Can Backpropagation error Surface not have local mínima, IEEE Transactions on neural networks Vol 3.

[36] **Yoshio H., Koichi Y., Shimpei H.** (1991) Back-Propagation Algorithm Which Varies the Number of Hidden Units, Fujitsu Laboratories Ltd. Neural networks Vol 4.

ANEXOS

ANEXO 1:

A continuación, se detallan los archivos m, realizados en MATLAB, para el entrenamiento y validación de la red neuronal

```
clc
clear
close all
% Se procede a realizar la carga de datos
load Dataset.mat
percentage = 0.7; % Se elige el 70% de datos de entrenamiento
[TrainInput,TrainOutput,TestInput,TestOutput]  = divideData
(P,T,percentage);
% Se procede a normalizar los datos de entrada
TrainInput(1,:)=2*(TrainInput(1,:)-min(P(1,:)))/(max(P(1,:))-
min(P(1,:)))-1; % Datos de Entrada
TrainInput(2,:)=2*(TrainInput(2,:)-min(P(2,:)))/(max(P(2,:))-
min(P(2,:)))-1; % Datos de Entrada
% Se procede a normalizar los datos para la validación
TestInput(1,:)=2*(TestInput(1,:)-min(P(1,:)))/(max(P(1,:))-
min(P(1,:)))-1; % Datos de Entrada de validación
TestInput(2,:)=2*(TestInput(2,:)-min(P(2,:)))/(max(P(2,:))-
min(P(2,:)))-1; % Datos de Entrada de validación
% Ingresamos los parámetros de la red neuronal
alfa = 0.001; % Valor de la Tasa de aprendizaje
nodeHidden=7; % Se ingresa el número de neuronas de capa oculta 1
nodeHidden2=7; % Se ingresa el número de neuronas de capa oculta 2
fhidden='tansig'; % Se elige función de activación de capas ocultas
foutput='linear'; %Se elige función de activación de capa de salida

%% Se procede a realizar El Entrenamiento de la red neuronal
[W1,b1,W2,b2,W3,b3,emedio]=neuralTrain2(TrainInput,TrainOutput,node
Hidden,nodeHidden2,alfa);
figure
plot(emedio)
xlabel('Epocas')
axis ([0 5000 0 200])
legend("Error de Red Neuronal")
title("Error de Red Neuronal")
fprintf('Error = %f\n',mean(emedio))

%% Se realiza la Validación de la red neuronal
pesoTotal = 0;
Q=size(TrainInput,2);
for q = 1:Q
a1 = neuralPredict(W1,TrainInput(:,q),b1,fhidden);
a2 = neuralPredict(W2,a1,b2,fhidden);
a3(q) = neuralPredict(W3,a2,b3,foutput);
```

```
pesoTotal = pesoTotal+a3(q);
end
figure
plot(a3,'b'),hold on,plot(TrainOutput,'r')
xlabel('Epocas')
ylabel('Peso')
legend("Salida estimada","Salida real")
title("Entrenamiento de la Red Neuronal")
%Se muestra el peso total
fprintf('Peso Total = %f\n',pesoTotal)
Q=size(TestInput,2);
for q = 1:Q
a1 = neuralPredict(W1,TestInput(:,q),b1,fhidden);
a2 = neuralPredict(W2,a1,b2,fhidden);
a3T(q) = neuralPredict(W3,a2,b3,foutput);
end
figure
plot(a3T,'b'),hold on,plot(TestOutput,'r')
legend("Salida estimada","Salida real")
title("Validación de la Red Neuronal")
```

ANEXO 2:

A continuación, se detalla los pasos del proceso de reducción de margen de error

Paso 1. Condiciones iniciales. - Se ingresa las entradas para el proceso de entrenamiento de la red neuronal esto incluye número de neuronas y capas ocultas.

Paso 2. Se realiza la normalización de datos de entrada para que puedan ser procesados en la red neuronal y se procede a seleccionar el 70% de datos para el entrenamiento y el 30% para la validación.

Paso 3. Se selecciona los pesos iniciales el cual se considera valores aleatorios el cual está entre 0 y 1 y se selecciona la función de activación para cada capa. En este paso para lograr un mejor funcionamiento de la red no es recomendable elegir pesos iguales ya que ello puede retardo en la optimización de la red.

Paso 4. Se calcula los valores de salida de cada capa oculta.

Paso 5. Se calcula los valores de ingreso para cada neurona de capa oculta.

Paso 6. Se calcula los términos de error para la capa de salida y se compara con la salida deseada.

Paso 7 Mientras que la condición del error sea falsa se procede a realizar el algoritmo de Back Propagation para realizar el ajuste de pesos caso contrario pasa al paso 9

Paso 7.1 Calculo de los pesos hacia atrás de la capa de salida

Paso 7.2 Calculo de los pesos hacia atrás de la capa oculta

Paso 7.3 Calculo de los pesos hacia atrás de la capa de entrada

Paso 7.4 Se actualiza pesos de capa de salida y capa oculta

Paso 7.5 Se calcula valores de salida de capa oculta y capa de salida

Paso 7.6 Se compara salida deseada con calculada y se verifica si cumple las condiciones de error:

Si error no cumple condición repite pasos del 7.1 al 7.5

Si error cumple la condición avanza a paso 10

Si error no converge avanza a paso 8

Paso 8 Se agrega 01 neurona a la capa oculta y repite pasos del 1-7

Paso 9. Se realiza el ajuste del factor de aprendizaje y se repite los pasos del 1-7

Paso 10. Se realiza la selección de pesos calculados cuando cumple la condición de error

Paso 11. Se realiza la desnormalización de los datos de salida

Paso 12. Se realiza el entrenamiento de la red