UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS CARRERA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE GRADO PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA:

MÓDULO DIDÁCTICO DE ENTRENAMIENTO DE REDES NEURONALES PARA EL RECONOCIMIENTO DE PATRONES DE IMÁGENES Y VOZ CON RASPBERRY PI

AUTOR:
MARCELA BELÉN VALLEJOS CALDERÓN

DIRECTOR:
ING. COSME MEJÍA

Ibarra – Ecuador

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

La Universidad Técnica del Norte dentro del proyecto Repositorio Digital Institucional, determinó la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la Universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	100328996-2		
APELLIDOS Y NOMBRES:	VALLEJOS CALDERON MARCELA BELEN		
DIRECCIÓN:	GARCIA MORENO Y 12 DE FEBRERO ATUNTAQUI		
EMAIL:	mbvallejos@utn.edu.ec		
TELÉFONO FIJO:	062910168	**TELÉFONO MÓVIL:**	0991631332

Datos De La Obra	
Título:	Módulo Didáctico De Entrenamiento E Eredes Neuronales Para El Reconocimiento De Patrones De Imágenes Y Voz Con Raspberry Pi
AUTOR (ES):	Marcela Belén Vallejos Calderón
FECHA:	2018-03-08
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	■ **PREGRADO** □ **POSGRADO**
TITULO POR EL QUE OPTA:	**Ingeniería en Mecatronica**
ASESOR /DIRECTOR:	**Ing. Cosme Mejia E.**

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

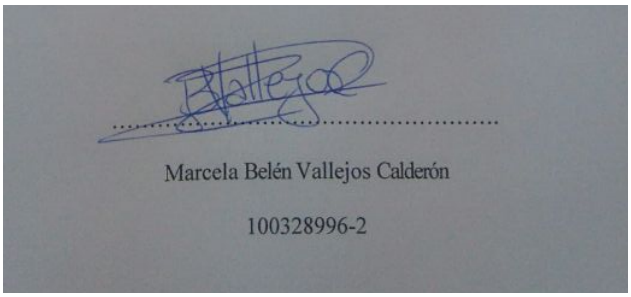
Yo, Marcela Belén Vallejos Calderón, con cédula de identidad Nro.1003289962, en calidad de autora y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en formato digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la Biblioteca de la Universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión; en concordancia con la Ley de Educación Superior Artículo 144.

3. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 28 días del mes de febrero del 2018

EL AUTORA:



Nombre: Belén Vallejos C.

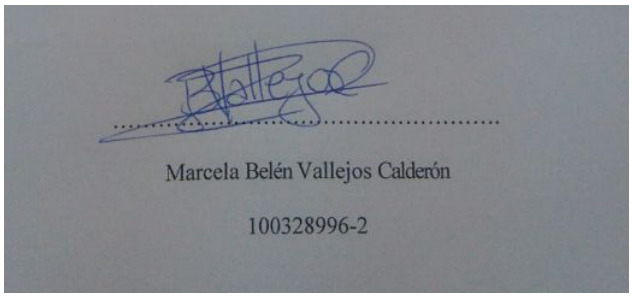
UNIVERSIDAD TÉCNICA DEL NORTE

CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

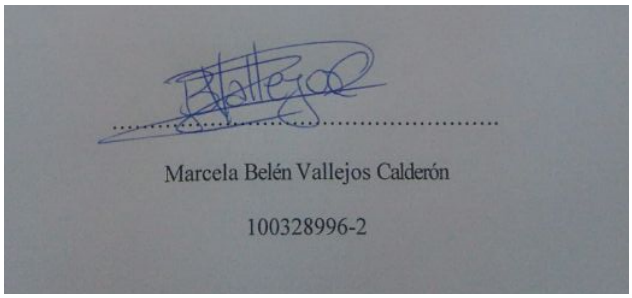
Yo, Marcela Belén Vallejos Calderón, con cédula de identidad Nro. 100328996-2, manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la Ley de Propiedad Intelectual del Ecuador, artículos 4, 5 y 6, en calidad de autor (es) de la obra o trabajo de grado titulado: "MÓDULO DIDÁCTICO DE ENTRENAMIENTO DE REDES NEURONALES PARA EL RECONOCIMIENTO DE PATRONES DE IMÁGENES Y VOZ CON RASPBERRY PI", que ha sido desarrollada para optar por el Título de Ingeniera en Mecatrónica en la Universidad Técnica del Norte, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En nuestra condición de autor nos reservamos los derechos morales de la obra antes citada. En concordancia suscribimos este documento en el momento que hacemos entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Técnica del Norte.

Ibarra, a los 28 días del mes de febrero del 2018



4. DECLARACIÓN

Yo, Marcela Belén Vallejos Calderón, declaro que este trabajo es de autoría propia, ya que no ha sido presentado para ningún trabajo de grado, y certificamos la veracidad de las fuentes bibliográficas que se incluyen en el presente trabajo.

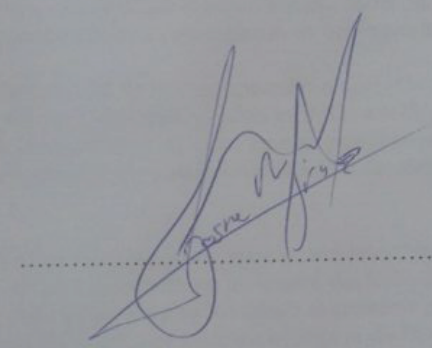


Marcela Belén Vallejos Calderón

100328996-2

CERTIFICACIÓN

Certifico que el presente trabajo, fue desarrollado por la Señorita Marcela Belén Vallejos Calderón, bajo la supervisión del que certifica



.....

Ing. Cosme Mejía

DIRECTOR DE TESIS

5. AGRADECIMIENTO

Aprender en la vida es un acto de valentía, es perder el miedo a lo desconocido y entrar en un mundo lleno de nuevas experiencias y oportunidades; en el camino del aprendizaje hay personas que te orientan, te ayudan y te guía para no desfallecer, son las personas que comparten conmigo este capítulo de mi vida llamado Universidad, por eso pienso que el mejor reconocimiento y acto de gratitud que todos pueden recibir son mis palabras expresadas en esta etapa importante de mi vida; por eso quiero agradecer a todos mis profesores, mis amigos de aula, mis amigos IEEE, el personal administrativo de la FICA por ser parte de esta hermosa etapa de mi vida; gracias por cada palabra de aliento, por apartar con un granito de arena para hacer de mí una mejor persona y sobre todo mejor ser humano, mi gratitud eterna a todos y todas por dejarme caer, por en mi paso por mi querida UTN se convirtieron en mi segunda familia.

Es parte esencial de la vida aprender a decir GRACIAS, pero no por compromiso, sino por Gratitud, gracias infinitas a todos quienes fueron parte de un capítulo de la historia de mi vida, gracias por dejarme ser parte de la suya, pero sobre todo gracias por enseñarme hacer un ser humano lleno de momentos e historias.

Y agradecer a Dios porque puso a personas maravillas en mi diario vivir, dentro de la Casona Universitaria, no quiere decir nombres por tener al olvido, solo me queda decir gracias a todos desde el conserje de mi facultad hasta el rector de mi Universidad, por dejarme el legado de llevar con orgullo el nombre de la Universidad Técnica del Norte., y enseñarme a volar Alto y Lejos.

Con cariño.

Belén

6. DEDICATORIA

"Coquetearle a la virtud, sin pretensiones de santidad"

La vida es una constante toma de decisiones, alguna fáciles otras difíciles, pero hoy tengo la decisión de dedicar un trabajo lleno de mucho esfuerzo y dedicación, de repente vienen a tu mente muchas momentos por los cuales pasaste para llegar a que esto sea posible, y te das cuenta que siempre existió una constante que se llama FAMILIA, quienes son tus raíces, la fuente de tus valores y principios, quienes son los que siempre vas a volver, pero también son los que te dan alas para volar y muy lejos, hoy quiero dedicar este trabajo a mis dos raíces, mi madre y mi padre, tan distintos que me enseñaron que en la diversidad está el verdadero sentido de la vida; dedicado a ti mami RUTH CALDERÓN, gracias por ser mi ejemplo de constancia, fortaleza y perseverancia, por jamás dejarme vencer, por educarme y confiar en mí, para ti papi GALO VALLEJOS, por ser el pilar de mi vida y el mayor ejemplo de amor puro; a ti Enana MAJO VALLEJOS porque eres el ángel que llegó a demostrarme que la verdadera felicidad esta en cada uno; porque sé que eres el milagro de mi vida; a todos mi tías, tíos , primos y primas por estar en siempre conmigo, con ustedes aprendí que la familia es la única que siempre va a estar ahí.

Y como olvidarme del que nunca me abandonó JESUS DE NAZARETH, eres el amigo que nunca falla, porque cada vez que sentía caer me recogiste en tus brazos, cada vez que fallaba me demostraste que todo lo puedo en Ti que me fortaleces, y como siempre lo dije aquí estoy para hacer tu voluntad; lo digo más convencida que nunca, dedicado a ti y a tu Madre la Virgen María.

Pero al final quiero dedicar este trabajo al cielo y las personas que no están aquí, que se convirtieron en ángeles permanentes de mi vida.

"Esperanza, Sabiduría y Voluntad"
ESV

Belén

7. CONTENIDO

8. INDICE DE FIGURAS

9. ÍNDICE DE TABLAS

10. RESUMEN

El trabajo que se detalla a continuación, se fundamenta en la necesidad de la enseñanza y aprendizaje práctico por parte del profesor o instructor guía y el alumno, para adquirir criterios en cuanto se refiere a diseño mecatrónico en un estudió dinámico sobre redes neuronales, y la aplicabilidad de este en diversos campos donde el control necesita una curva de aprendizaje progresiva sobre una red neuronal, el motivo por el cual el estudio y la aplicación de redes neuronales en sistemas dinámicos ha sido algo no explorado por los estudiantes, es por el largo proceso que requiere el realizar una red que aprenda y sea autosuficiente; entender el comportamiento de la misma, el proceso de aprendizaje y la implementación han sido un tabú dentro de muchos aplicaciones de control, es por eso que en la actualidad a pesar de que el proceso de crear o mejorar una red neuronal ya establecido, es por la facilidad de cambios inmediatos en la estructura de un proceso de control, además de tener una versatilidad dentro de las múltiples aplicaciones en las cuales se necesita un control inmediato y exacto,

El objetivo que persigue el proyecto propuesto es el desarrollo de un módulo didáctico que simplifique la creación de una red neuronal y la aplicación en el mundo real inmediatamente. Debido a ello el proyecto se ha dividido en dos partes fundamentales las cuales son: El Modelamiento de la red neuronal y el enlace con el medio externo. El presente proyecto también abarca el desarrollo de diferentes modelos de entrenamiento neuronal con los cuales el estudiante podrá realizar prácticas de reconocimiento de la función característica de los mismos, partiendo de la base teórica que origina el comportamiento de los sistemas físicos implementados.

11. ABSTRAC

The work that is detailed below is based on the need for teaching and practical learning by the teacher or instructor guide and the student to acquire criteria as they relate to mechatronics design in a dynamic study on neural networks, And the Applicability of this in various fields where the control needs a progressive learning curve on a neural network, the reason why the study and application of neural networks in dynamic systems has been something unexplored by the students, is the long process that requires the realization of a network that learns and is self-sufficient; Understand the behavior of the same, the learning process and implementation have been taboo within many control applications, that is why today despite the process of creating or improving an established neural network, it is by The ease of immediate changes in the structure of a control process, in addition to having a versatility within the multiple applications where immediate and accurate control is needed.

The objective of the proposed project is the development of a didactic module that simplifies the creation of a neural network and the application in the real world immediately. Due to this the project has been divided in two fundamental parts which are: The Modeling of the neural network and the connection with the external environment. The present project also covers the development of different models of neuronal training with which the student can perform practices of recognition of the characteristic function of the same, starting from the theoretical basis that originates the behavior of the physical systems implemented.

12. PRESENTACIÓN

El proyecto del módulo didáctico para el modelamiento de sistemas lineales con Matlab y tarjeta compatible USB, está estructurado en cinco capítulos: Análisis de la situación actual y objetivos del proyecto, marco teórico acerca de control neuronal aplicado a reconocimiento de patrones, Análisis del modelo matemático de la neurona, Implementación del algoritmo de entrenamiento y enlace con Raspberry pi, prácticas propuestas y conclusiones y recomendaciones.

El primer capítulo del presente trabajo se detalla por qué la realización del mismo, detallando la situación actual de los laboratorios y las practicas que se realizan en la materia de diseño mecatrónico, donde la implementación de este nuevo módulo servirá para que los estudiantes hagan prácticas en cada clase con diferentes modelos de redes neuronales y distintas aplicaciones.

El segundo capítulo, explica todo lo relacionado con las redes neuronales aplicadas en el reconocimiento de patrones de imágenes y voz, para tener claro los conceptos básicos sobre redes que se aplican en este tipo de entrenamiento.

El tercer capítulo, se detalla el modelo matemático del algoritmo de backpropagation, donde se detalla las ecuaciones las cuales tiene el algoritmo, además de la interpretación total de la funcionabilidad de dichas ecuaciones, para poder tener una fácil comprensión del modelo matemático.

En el cuarto capítulo del presente trabajo, se explica la funcionabilidad de las redes neuronales y de su aplicabilidad en temas de control, por eso se procesó al funcionamiento de la red neuronal en el software que se va a utilizar que es Matlab, además del enlace y configuración con el medio externo y su manejo remoto con Raspberry pi.

El quinto capítulo ya es la operatividad del módulo en la aplicación de 3 practicas donde se puede visualizar las redes ya realizadas y sus respectivas pruebas de funcionamiento.

El sexto capítulo consta de las conclusiones y recomendaciones del proyecto, a su vez la explicación del manual de funcionamiento y operatividad del módulo

13. CAPÍTULO 1

1. INTRODUCCIÓN

El cerebro humano es el sistema de control y cálculos más complejo que ha sido conocido por el hombre; es por eso que para un ser humano reconocer un rostro, una letra o un número es una tarea muy fácil de realizar, mientras que para un computador es una tarea muy difícil, mientras que los cálculos extensos son de suma facilidad para un computador y muy largo y tedioso para un ser humano. (Fausett., 2005)

Es por eso que la capacidad del cerebro de recordar, analizar y decidir ha llevado a todos los científicos a querer definir como es el proceso de aprendizaje del cerebro humano, tratando de imitar el modelo del mismo. De este arduo trabajo por imitar el modelo de aprendizaje del cerebro humano se está implementando una nueva tecnología llamada redes neuronales artificiales (RNA), que piensan ser implementadas en los procesos industriales que necesitan una mejor exactitud; es por eso que ahora se está desarrollando más este tipo de control. (Fausett., 2005)

El desarrollo del módulo didáctico para el control neuronal y el posterior entrenamiento de la neurona que permitirá el reconocimiento de patrones de imágenes y voz; lo que primero se debe definir es el método con el cual se podrá realizar el control neuronal, el método elegido es el de aprendizaje supervisado con el algoritmo de backpropagation (propagación de error hacia atrás), permite el entrenamiento de un perceptrón multicapa, se ha elegido este método ya que en la fase de entrenamiento requiere los datos de entrada y las variables de salida, es decir definir la topología de nuestra red neuronal; el modelo matemático del algoritmo de backpropagation ya está definido: además de la utilidad del toolbox de entrenamiento de redes neuronales que encontramos en Matlab..

En la realización del entrenamiento de la neurona para realizar el control neuronal que se va a realizar, se ha elegido el software Matlab, por la facilidad de escalabilidad del módulo, sabiendo que es un lenguaje de alto nivel y un entorno interactivo para el cálculo numérico, visualización y programación. Usando MATLAB se puede analizar los datos, implementar el algoritmo y crear modelos y aplicaciones; el lenguaje, las herramientas y funciones matemáticas integradas que permiten explorar múltiples enfoques y llegar a una solución más rápida con la integración de nuevas tecnologías que se han desarrollado para optimizar el control industrial.

En la parte del hardware de enlace con el medio externo, aprovechando los distintos protocolos de comunicación del software a utilizarse, se lo va a realizar con el hardware Raspberry PI, se realiza con este dispositivo por lo versátil, lo escalable y lo multifuncional para esta aplicación; además de la conexión directa entre Simulink de Matlab y Raspberry pi. Una de las principales ventajas es que este dispositivo puede ser portable, podemos grabar nuestro algoritmo en una memoria y utilizarlo sin necesidad de tener conectado al computador.

Las principales aplicaciones que se van a realizar son: reconocimientos de patrones de imágenes y voz., visualización y programación.

1.2 OBJETIVOS

1.2.1 OBJETIVO GENERAL

- Implementar el módulo de entrenamiento de redes neuronales en Simulink y Raspberry pi, para reconocer imágenes y voz

1.2.2 OBJETIVOS ESPECÍFICOS

- Implementar el algoritmo de backpropagation que permita el entrenamiento de la red neuronal.
- Enlazar Simulink con Raspberry pi, para desarrollar aplicaciones de redes neuronales que son reconocimientos de patrones de imágenes y voz.
- Diseñar tres prácticas modelo que ayude al mejor entendimiento del módulo didáctico, incluyendo los pasos a seguir para el entrenamiento y modelado, y la ejecución de prácticas de laboratorio.
- Elaborar un manual de usuario para el uso de estudiantes y una guía de prácticas de laboratorio

1.3 JUSTIFICACIÓN

De acuerdo a los nuevos métodos que se han venido implementado en las aulas de clase todo estudiante necesita de la implementación y estudio práctico para el refuerzo de lo aprendido, además en el área de sistemas de control, aplicado a las redes neuronales, al momento de realizan prácticas enfocadas en esta área de estudio no se cuenta con un módulo dedicado a dicho tema, por eso al ver necesario la utilización de tecnología nueva que sea aplicable a las redes neuronales ya que a un proceso largo que requiere de los materiales y equipos indicados para realizar satisfactoriamente la práctica.

El módulo de entrenamiento neuronal va a ampliar el área de conocimiento de los estudiantes, para que se pueda desarrollar aplicaciones reales, implementando en diferentes ámbitos de estudio; el modelo matemático que se va a ser aplicado para el control neuronal permite que el modulo sea adaptable en el tiempo, es decir no va quedar obsoleto, que ya está siendo implementado con nuevas tecnologías; además que aquí estamos utilizando un modelo matemático ya definido lo que nos permito cambiar dicho modelo y poder tener una apreciación diferente.

14. CAPITULO 2

2. INTRODUCCIÓN A CONTROL NEURONAL APLICADO AL RECONOCIMIENTO DE PATRONES

2.1 INTRODUCCIÓN

"Las redes neuronales artificiales (RNA), son modelos matemáticos que intentan reproducir el funcionamiento del sistema nervioso, así como todo modelo neuronal; también conocido como red con aprendizaje conexionista y procesamiento de distribución paralela; además realizan una simplificación del sistema real que simulan y toman las características principales para una mejor interpretación del mismo para la resolución de una tarea determinada y especifica.

En lo referente a la aplicación de redes neuronales existen dos categorías de problemas que se han resuelto con éxito usando esta metodología que son: el reconocimiento de patrones provenientes de diferentes datos o entradas la identificación de parámetros de sistemas y el control (conocido/desconocido) de sistemas dinámicos.

En la actualidad existe un creciente interesante en los sistemas de control que incluyen redes neuronales artificiales como elementos de control, por lo que se han implementado un importante número de soluciones. Las aplicaciones exitosas más relevantes influyen desde elementos que operan como identificadores hasta aquellos que trabajan como controladores de optimización utilizando modelos del proceso." (Barragán, 2008

2.2 EL MODELO BIOLÓGICO

"El cerebro es el elemento principal del sistema nervioso humano y está compuesto por un tipo especial de célula llamada neurona. "La neurona es considerada la unidad estructural y funcional fundamental del sistema nervioso. Esto quiere decir que las diferentes estructuras del sistema nervioso tienen como base grupos de neuronas. Además, la neurona es la unidad funcional porque puede aislarse como componente individual y puede llevar a cabo la función básica del sistema nervioso, esta es, la transmisión de información en la forma de impulsos nerviosos". (Marrero, 2005)

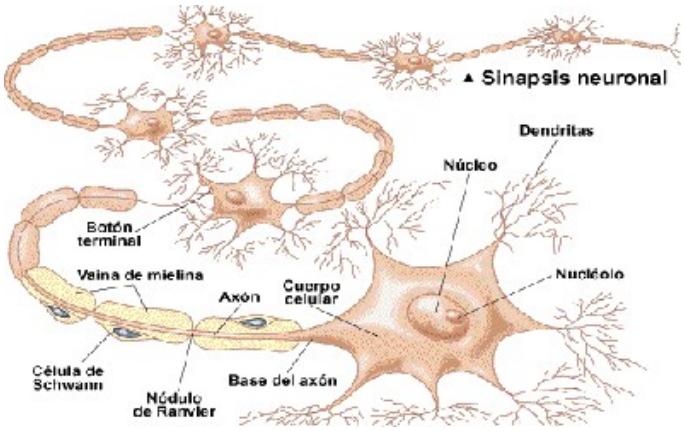


Figura 1. NEURONA BIOLÓGICA

"La neurona biológica está compuesta por un cuerpo celular o soma, del cual se desprende árbol de ramificaciones llamado árbol dendrítico, compuesto por las dendritas. Del soma también parte una fibra tubular, llamada axón, el cual suele ramificarse cerca de su extremo. Las dendritas actúan como un canal de entrada de señales provenientes desde el exterior hacia la neurona, mientras que el axón actúa como un canal de salida.

El espacio entre dos neuronas vecinas se denomina sinapsis. En el córtex cerebral se observa una organización horizontal en capas, así como también una organización

vertical en columnas de neuronas". (José R Hilera Martínez, 2000)

2.3 DEFINICIONES DE UNA RED NEURONAL.

Existen muchas formas de definir a las redes neuronales artificiales; desde las definiciones cortas hasta generales hasta las que intentan explicar más explícitamente qué son y que hacen las redes neuronales.

a) Una nueva forma de tecnología, inspirada en modelos biológicos.

b) Es un modelo matemático compuesto por un gran número de elementos procesales organizados en niveles y niveles ocultos.

c) Un sistema de control compuesto por un gran número de elementos simples, elementos de procesos muy interconectados, los cuales procesan información por medio de su estado dinámico como respuesta a entradas externas.

d) Redes neuronales artificiales son redes interconectadas masivamente en paralelo de elementos simples (usualmente adaptativos) y con organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico. (Fausett., 2005)

2.4 TIPOS DE APRENDIZAJE

"Se ha visto que los datos de entrada se procesan a través de la red neuronal con el propósito de lograr una salida. También se dice que las redes neuronales extraen generalizaciones desde un conjunto determinado de ejemplos anteriores de tales problemas de decisión. Una red neuronal debe aprender a calcular la salida correcta para cada constelación (arreglo o vector) de entrada en el conjunto de ejemplos. Este proceso de aprendizaje se denomina: proceso de entrenamiento o acondicionamiento. El conjunto de datos (o conjunto de ejemplos) sobre el cual este proceso se basa es, por ende, llamado: conjunto de datos de entrenamiento". (Cisterna, 2007).

Tabla 1. VENTAJAS DE LAS REDES NEURONALES

	Características
Aprendizaje Adaptativo	Capacidad de aprender tareas basadas en una experiencia inicial. Capacidadde autoajuste y adaptables. Siguen aprendiendo luego del entrenamiento. Capacidad de discriminar patrones de aprendizaje mal aprendidos
Auto - Organización	Representan la información que recibe por etapas de entrenamiento. Puede modificar la red neuronal completa para llegar al objetivo deseado. Puede organizar la información para poder obtener información adecuada.
Tolerancia a fallos	Las redes pueden aprender a reconocer patrones con ruido, distorsiones o incompletos. Pueden almacenar datos a pesar de los errores del sistema, porque existe redundancia de datos en las conexiones de la red. Almacenan información no localizada. Operación en tiempo real. Inserción dentro de tecnologías ya existentes.

(CARLOS ALBERTO RUIZ, MARTA SUSANA BASUALDO, 2001)

> "Si la topología de la red y las diferentes funciones de cada neurona (entrada, activación y salida) no pueden cambiar durante el aprendizaje, mientras que los pesos sobre cada una de las conexiones si pueden hacerlo; el aprendizaje de una red neuronal significa: adaptación de los pesos. (Cisterna, 2007)

En otras palabras, el aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante el mismo se reducen a la destrucción, modificación y creación de conexiones entre las neuronas. En los sistemas biológicos existe una continua destrucción y creación de conexiones entre las neuronas. En los modelos de redes neuronales artificiales, la creación de una nueva conexión implica que el peso de la misma pasa a tener un valor distinto de cero. De la misma manera, una conexión se destruye cuando su peso pasa a ser cero.

Durante el proceso de aprendizaje, los pesos de las conexiones de la red sufren modificaciones, por lo tanto, se puede afirmar que este proceso ha terminado (la red ha aprendido) cuando los valores de los pesos permanecen estables"= 0. (Matich, 2001)

"Un aspecto importante respecto al aprendizaje de las redes neuronales es el conocer cómo se modifican los valores de los pesos, es decir, cuáles son los criterios que se siguen para cambiar el valor asignado a las conexiones cuando se pretende que la red aprenda una nueva información.

Hay dos métodos de aprendizaje importantes que pueden distinguirse:

a) Aprendizaje supervisado.

b) Aprendizaje no supervisado.

Otro criterio que se puede utilizar para diferenciar las reglas de aprendizaje se

basa en considerar si la red puede aprender durante su funcionamiento habitual o si el aprendizaje supone la desconexión de la red, es decir, su inhabilitación hasta que el proceso termine. En el primer caso, se trataría de un aprendizaje on line, mientras que el segundo es lo que se conoce como off line.

"Cuando el aprendizaje es off line, se distingue entre una fase de aprendizaje o entrenamiento y una fase de operación o funcionamiento, existiendo un conjunto de datos de entrenamiento y un conjunto de datos de test o prueba, que serán utilizados en la correspondiente fase. Además, los pesos de las conexiones permanecen fijos después que termina la etapa de entrenamiento de la red. Debido precisamente a su carácter estático, estos sistemas no presentan problemas de estabilidad en su funcionamiento". (Fausett., 2005).

"Una generalización de la fórmula o regla para decir los cambios en los pesos es la siguiente:

Peso Nuevo = Peso Viejo + Cambio de Peso

Matemáticamente es:

$$W\ t+1\ = W\ t\ - \Delta W(t)$$ **ECUACIÓN 2-1**

Donde t hace referencia a la etapa de aprendizaje, wij (t+1) al peso nuevo y wij (t) al peso viejo". (José R Hilera Martínez, 2000)

2.4.1 APRENDIZAJE SUPERVISADO.

"El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada.

El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada". (Cisterna, 2007)

En este tipo de aprendizaje se suelen considerar, a su vez, tres formas de llevarlo a cabo, que dan lugar a los siguientes aprendizajes supervisados:

a) Aprendizaje por corrección de error.

Este tipo de aprendizaje consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos a la salida de la red, es decir, en función del error cometido en la salida.

"Un ejemplo de este tipo de algoritmos lo constituye la regla de aprendizaje del perceptrón, utilizada en el entrenamiento de la red del mismo nombre que desarrolló Rosenblatt en 1958". (Rosenblatt, 1958).

"Esta es una regla muy simple, para cada neurona en la capa de salida se le calcula la desviación a la salida objetivo como el error, δ. El cual luego se utiliza para cambiar los pesos sobre la conexión de la neurona precedente. El cambio de los pesos por medio de la regla de aprendizaje del Perceptrón se realiza según la siguiente regla":

$$\Delta w_{34} = \vartheta^{*} out_{4} * (a_{3} - out_{3}) \qquad \textbf{\textit{ECUACIÓN 2.2}}$$

Dónde: aquí es la salida deseada/objetivo de la neurona de salida Ni, δi = (aqi – outi) la desviación objetivo de la neurona Ni y σ el aprendizaje.

La salida de la neurona Nj (outj) se utiliza, porque este valor influye en la entrada global y, por ende, en la activación y luego en la salida de la neurona Ni. Esto es semejante a un "efecto en cadena".

Otro algoritmo muy conocido y que pertenece a esta clasificación es la regla de aprendizaje Delta o regla del mínimo error cuadrado (LMS Error: Least Mean Squared Error), que también utiliza la desviación a la salida objetivo, pero toma en consideración a todas las neuronas predecesoras que tiene la neurona de salida. Esto permite cuantificar el error global cometido en cualquier momento durante el proceso de entrenamiento de la red, lo cual es importante, ya que cuanta más información se tenga sobre el error cometido, más rápido se puede aprender. Luego el error calculado (δ) es igualmente repartido entre las conexiones de las neuronas predecesoras.

"Por último, se debe mencionar la regla de aprendizaje de propagación hacia atrás o de backpropagation, también conocido como regla LMS multicapa, la cual es una generalización de la regla de aprendizaje Delta. Esta es la primera regla de aprendizaje que permitió realizar cambios sobre los pesos en las conexiones de la capa oculta". (Marrero, 2005)

a) Aprendizaje por refuerzo.

Se trata de un aprendizaje supervisado, más lento que el anterior, que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada.

"En el aprendizaje por refuerzo la función del supervisor se reduce a indicar mediante una señal de refuerzo si la salida obtenida en la red se ajusta a la deseada (éxito = +1 o fracaso = -1), y en función de ello se ajustan los pesos basándose en un mecanismo de probabilidades. Se podría decir que en este tipo de aprendizaje la función del supervisor se asemeja más a la de un crítico (que opina sobre la respuesta de la red) que a la de un maestro (que indica a la red la respuesta concreta que debe generar), como ocurría en el caso de supervisión por corrección del error". (J.Rieta, F. Castells, C. Sanchez, V.Zarzoso, J.Millet, 2007)

a) Aprendizaje estocástico.

Consiste básicamente en realizar cambios aleatorios en los valores de los pesos

de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad.

"En el aprendizaje estocástico se suele hacer una analogía en términos termodinámicos, asociando a la red neuronal con un sólido físico que tiene cierto estado energético. En el caso de la red, la energía de la misma representaría el grado de estabilidad de la red, de tal forma que el estado de mínima energía correspondería a una situación en la que los pesos de las conexiones consiguen que su funcionamiento sea el que más se ajusta al objetivo deseado.

Según lo anterior, el aprendizaje consistiría en realizar un cambio aleatorio de los valores de los pesos y determinar la energía de la red (habitualmente la función energía es una función de Liapunov. Si la energía es menor después del cambio, es decir, si el comportamiento de la red se acerca al deseado, se acepta el cambio; si, por el contrario, la energía no es menor, se aceptaría el cambio en función de una determinada y preestablecida distribución de probabilidades". (B. Simon, L.Sornmo, P. Laguna, 2005)

2.4.2 APRENDIZAJE NO SUPERVISADO.

Las redes con aprendizaje no supervisado (también conocido como auto supervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta.

Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada.

Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado.

"En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las informaciones que se le han

mostrado hasta entonces (en el pasado). En otro caso, podría realizar una clusterización (clustering) o establecimiento de categorías, indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas." (M.T.M. Zi-qin, J-L Schiano, 2000)

En cuanto a los algoritmos de aprendizaje no supervisado, en general se suelen considerar dos tipos, que dan lugar a los siguientes aprendizajes:

a) Aprendizaje hebbiano.

"Esta regla de aprendizaje es la base de muchas otras, la cual pretende medir la familiaridad o extraer características de los datos de entrada. El fundamento es una suposición bastante simple: si dos neuronas Ni y Nj toman el mismo estado simultáneamente (ambas activas o ambas inactivas), el peso de la conexión entre ambas se incrementa.

Las entradas y salidas permitidas a la neurona son: {-1, 1} o {0, 1} (neuronas binarias). Esto puede explicarse porque la regla de aprendizaje de Hebb se originó a partir de la neurona biológica clásica, que solamente puede tener dos estados: activa o inactiva". (Novak, 2000)

b) Aprendizaje competitivo y comparativo

"Se orienta a la clusterización o clasificación de los datos de entrada. Como característica principal del aprendizaje competitivo se puede decir que, si un patrón nuevo se determina que pertenece a una clase reconocida previamente, entonces la inclusión de este nuevo patrón a esta clase matizará la representación de la misma. Si el patrón de entrada se determinó que no pertenece a ninguna de las clases reconocidas anteriormente, entonces la estructura y los pesos de la red neuronal serán ajustados" (Novak, 2000).

2.5 ESTRUCTURA DE UN SISTEMA NEURONAL ARTIFICIAL

Las redes neuronales son modelos matemáticos que intentan reproducir el comportamiento del cerebro humano. El principal objetivo de este modelo es la

construcción de sistemas capaces de presentar un cierto comportamiento inteligente. Esto implica la capacidad de aprender a realizar una determinada tarea.

Las características principales que reproducen las redes neuronales artificiales se pueden reducir a los siguientes tres conceptos: procesamiento paralelo, distribuido y adaptativo". [Del Brio y Sanz Molina, 2002].

"El verdadero secreto de funcionamento de este modelo radica en el procesamiento paralelo realizado por las neuronas artificiales. La neurona artificial es un elemento de procesamiento simple y constituye el elemento principal de un sistema neuronal artificial.

Estas neuronas artificiales se combinan en estructuras denominadas capas. Una red neuronal artificial está por compuesta por un conjunto de capas. De esta manera, la información se encuentre distribuida a lo largo de las sinapsis de la red, dándole a este sistema cierta tolerancia a fallos". (Bonifacio Martín del Rio, Alfredo Sanz Molina, 2001)

La neurona artificial es un elemento de procesamiento simple que a partir de un vector de entradas produce una única salida. En general podemos encontrar tres tipos de neuronas artificiales, donde cada una de las cuales tiene su contraparte en el sistema nervioso:

15. Las que reciben información directamente desde el exterior, a las cuales se las denomina neuronas de entrada.
16. Las que reciben información desde otras neuronas artificiales, a las cuales se las denomina neuronas ocultas. Es en estas neuronas, en particular en sus sinapsis, donde se realiza la representación de la información almacenada.
17. Las que reciben la información procesada y las devuelven al exterior. A estas neuronas se las denomina neuronas de salida.

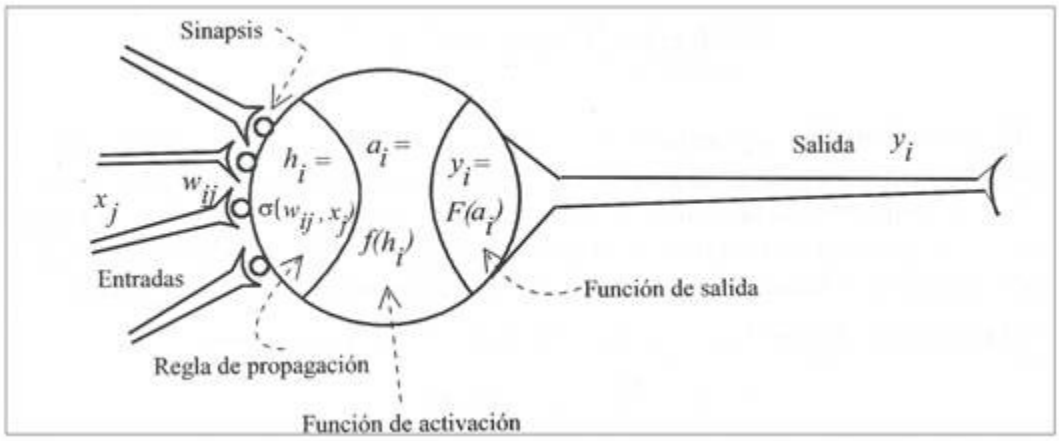


Figura 2. NEURONA ARTIFICIAL

- Conjunto de entradas, xj (t). Estas pueden ser provenientes del exterior o de otras neuronas artificiales.
- Pesos sinápticos, wij. Representan el grado de comunicación entre la neurona artificial j y la neurona artificial i. Pueden ser excitadores o inhibidores
- Regla de propagación, σi (wij, xj(t)). Integra la información proveniente de las distintas neuronas artificiales y proporciona el valor del potencial post sináptico de la neurona i.

Función de activación, fi (ai(t-1), hi(t)). Provee el estado desactivación actual de la neurona i.

- Función de salida, Fi (ai(t)). Representa la salida actual de la neurona i.

$$y_3\,t \;=\; F_3(f_3\,a_3\,t-1\;,\sigma_3(w_{34},x_4(t)) \qquad \textbf{ECUACIÓN 2- 3}$$

(K. Berkner, R. O.Wels, 1998)

De esta forma, la salida producida por una neurona i, para un determinado instante de tiempo t puede ser escrita en forma general de la siguiente manera:

(Aapo Hyyarinen, Erkki Oja, 2000)

2.5.1 ENTRADAS Y SALIDAS

"Las entradas y salidas de una neurona pueden ser clasificadas en dos grandes grupos, binarias o continuas. Las neuronas binarias (digitales) sólo admiten dos valores posibles. En general en este tipo de neurona se utilizan los siguientes dos alfabetos {0,1} o {-1,1}. Por su parte, las neuronas continuas (analógicas) admiten valores dentro de un determinado rango, que en general suele definirse como [-1, 1].

La selección del tipo de neurona a utilizar depende de la aplicación y del modelo a construir".

2.5.2 PESOS SINÁPTICOS

"El peso sináptico wij define la fuerza de una conexión sináptica entre dos neuronas, la neurona pre sináptica i y la neurona pos sináptica j. Los pesos sinápticos pueden tomar valores positivos, negativos o cero. En caso de una entrada positiva, un peso positivo actúa mientras que un peso negativo actúa como inhibidor. En caso de que el peso sea cero, no existe comunicación entre el par de neuronas. Mediante el ajuste de los pesos sinápticos la red es capaz de adaptarse a cualquier entorno y realizar una determinada tarea". (Claverl, 2007)

2.5.3 REGLA DE PROPAGACIÓN

Al mencionar este método para que una red neuronal aprendiera una asociación que existe entre sus patrones de entrada y las clases correspondientes.

"Este método es conocido como backpropagation, propagación del error hacia atrás o retro propagación, y está basado en la regla de aprendizaje que es posible aplicar solo a modelos de redes multicapa. Una característica importante de este algoritmo es

la representación interna del conocimiento que es capaz de organizar en la capa o capas intermedias, para conseguir cualquier correspondencia entre la entrada y la salida de la

$$h_i(t) = \sum_j w_{ij} * x_j(t)) \qquad \textit{ECUACIÓN 2- 5}$$

red". (Hugo Vega Huerta, Augusto Cortez Vásquez, Ana María Huayna, Luis Alarcón Loayza, Pablo Romero Naupari, 2009)

La regla de propagación determina el potencial resultante de la interacción de la neurona i con las N neuronas vecinas. El potencial resultante hi se puede expresar de la siguiente manera:

$$h_i(t) = \sigma_i(w_{ij}, x_j(t)) \qquad \textit{ECUACIÓN 2- 4}$$

La regla de propagación más simple y utilizada consiste en realizar una suma de las entradas ponderadas con sus pesos sinápticos correspondientes:

"Un punto importante en la red de retro propagación es su capacidad de auto adaptar los pesos de las neuronas de las capas intermedias para aprender la relación que existe entre un conjunto de patrones dados como ejemplo y sus salidas correspondientes. Y después utilizar esa misma relación a nuevos vectores de entrada con ruido o incompletos, dando una salida activa si la nueva entrada es parecida a las presentadas durante el aprendizaje". (José R Hilera Martínez, 2000)

2.5.4 FUNCIÓN DE ACTIVACIÓN

La función de activación; es el estado de activación que presenta actualmente la neurona en base al potencial resultante hi y al estado de activación anterior de la neurona ai (t-1). El estado de activación de la neurona para un determinado instante de tiempo t es expresado de la siguiente manera:

$$a_i(t) = f_i(a_i(t-1), h_j(t)) \qquad \textit{ECUACIÓN 2- 6}$$

Sin embargo, en la mayoría de los modelos se suele ignorar el estado anterior de la neurona, definiéndose el estado de activación en función del potencial resultante hi:

$$a_i(t) = f_i(h_j(t)) \qquad \textit{ECUACIÓN 2- 7}$$

Los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona. Por consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños. (Kasabov, 1998).

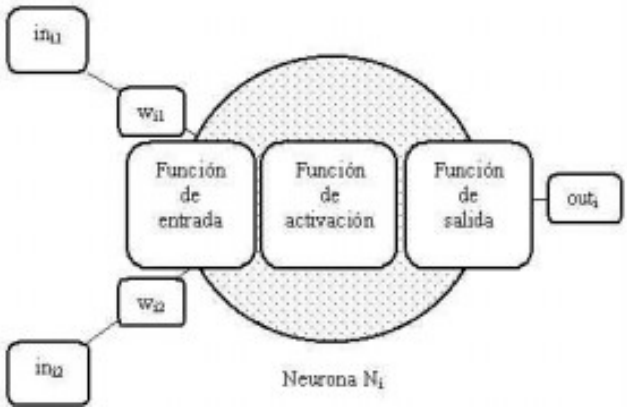


Figura 3 EJEMPLO DE UNA NEURONA CON 2 ENTRADAS Y 1 SALIDA

Para el entrenamiento de las neuronales se pueden utilizar las funciones de

activación de los distintos modelos de redes neuronales, en la siguiente tabla se detalla los distintos tipos de funciones:

Tabla 2 . FUNCIONES DE ACTIVACIÓN DE UNA RED NEURONAL

FUNCION	FORMULA	RANGO
IDENTIDAD	$y = x$	$[-\infty, \infty]$
ESCALÓN	$y = \begin{vmatrix} +1, & Si\ x \geq 0 \\ -1, & Si\ x < 0 \end{vmatrix}$	[0, 1]
	$y = \begin{vmatrix} \pm 1, & Si\ x \geq 0 \\ -1, & Si\ x < 0 \end{vmatrix}$	[-1, 1]
LINEAL A TRAMOS	$x, \quad Si -1 \leq x \leq 1$ $+1, \quad Si\ x > 1$ $-1, \quad Si\ x > -1$	[-1, 1]
SIGMOIDE	$y = \frac{1}{1 + e^{-x}}$	[0,1]
	$y = tanh(x)$	[-1, 1]
Sinusoidal	$y = sin(\omega x + \varphi)$	[-1, 1]

Fuente: (Dunnc, 2007)

Para explicar de mejor manera por qué se utilizan las funciones de activación se suele emplear la analogía a la aceleración de un automóvil; cuando un automóvil inicia su movimiento necesita una potencia elevada para comenzar a acelerar; pero al ir tomando velocidad, este demanda un menor incremento de dicha potencia para mantener la aceleración. Al llegar a altas velocidades, nuevamente un amplio incremento en la potencia es necesario para obtener una pequeña ganancia de velocidad.

En resumen, en ambos extremos del rango de aceleración de un automóvil se demanda una mayor potencia para la aceleración que en la mitad de dicho rango.

2.5.5 FUNCIÓN SALIDA

"El último componente que una neurona necesita es la función de salida que es el valor resultante de esta función es la salida de la neurona i (outi); donde la función de salida determina qué valor se transfiere a las neuronas vinculadas. Si la función de activación está por debajo de un umbral determinado, ninguna salida se pasa a la neurona subsiguiente. Normalmente, no cualquier valor es permitido como una entrada para una neurona, por lo tanto, los valores de salida están comprendidos en el rango [0, 1] o [-1, 1]. También pueden ser binarios {0, 1} o {- 1, 1}". (ben Krose, Patrick Van ser Smagt)

La función de salida proporciona el valor de la salida de la neurona, en base al estado de activación de la neurona. En la mayoría de los casos la identidad que es más utilizada es la identidad, la fórmula es:

$$\boldsymbol{y_i(t) = F_i(a_i(t) = a_j(t))} \qquad \textit{ECUACIÓN 2- 8}$$

2.6 ARQUITECTURA DE UNA RED NEURONAL

La arquitectura de una red neuronal, se define mediante una topología para su organización y disposición de las neuronas dentro de la red neuronal, esta viene dado por: el número de capas de la red, la cantidad de neuronas encada capa, el grado de conectividad y el tipo de conexión entre neuronas.

2.6.1 NIVELES O CAPAS DE UNA RED NEURONAL.

La distribución de neuronas dentro de la red se realiza formando niveles o capas, con un número determinado de dichas neuronas en cada una de ellas. A partir de su situación dentro de la red, se pueden distinguir tres tipos de capas:

1) De entrada: es la capa que recibe directamente la información proveniente de las fuentes externas de la red.

"Ocultas: son internas a la red y no tienen contacto directo con el entorno

exterior. El número de niveles ocultos puede estar entre cero y un número elevado. Las neuronas de las capas ocultas pueden estar interconectadas de distintas maneras, lo que determina, junto con su número, las distintas topologías de redes neuronales". (Hugo Vega Huerta, Augusto Cortez Vásquez, Ana María Huayna, Luis Alarcón Loayza, Pablo Romero Naupari, 2009)

De salidas: transfieren información de la red hacia el exterior.

En la Ilustración 4 podemos observar el ejemplo de la estructura de una red multicapa, en la que cada neurona únicamente está conectada con neuronas de un nivel superior; se puede observar que hay más conexiones que neuronas en sí; se dice que una red es totalmente conectada si todas las salidas desde un nivel llegan a todos y cada uno de las neuronas del nivel siguiente.

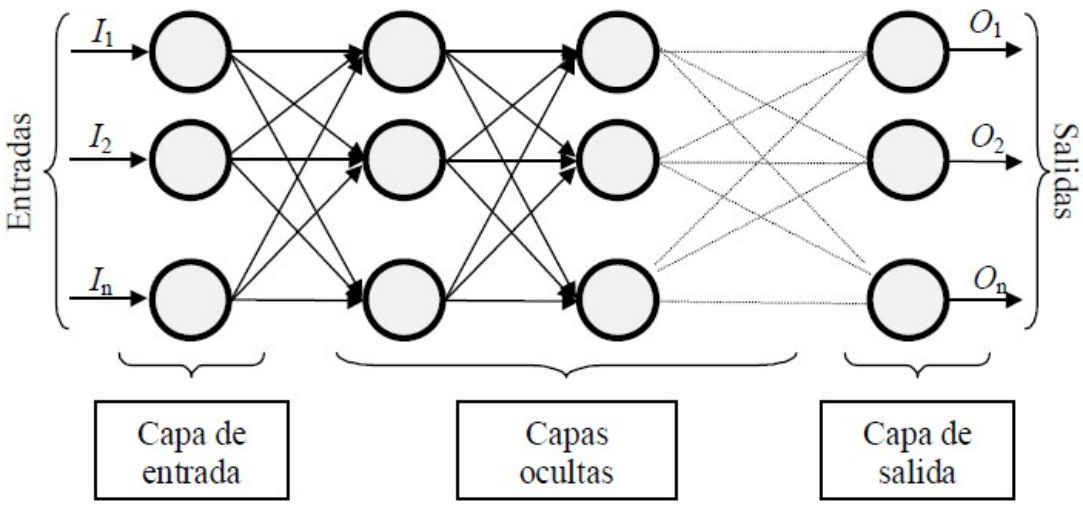


Figura 4 EJEMPLO DE UNA RED NEURONAL TOTALMENTE CONECTADA

2.6.2 TIPOS DE NEURONAS ARTIFICIALES.

Las neuronas artificiales se pueden clasificar de acuerdo a los valores que pueden tomar. Por ahora es suficiente distinguir entre dos tipos principales:

- Neuronas binarias.
- Neuronas reales.

Las neuronas binarias solamente pueden tomar valores dentro del intervalo {0,1} o {-1, 1}, mientras que las neuronas reales pueden hacerlo dentro del rango [0, 1] o [-1, 1].

"Los pesos normalmente no están restringidos a un cierto intervalo, aunque para aplicaciones específicas puede ser esto necesario". (Matich, 2001)

2.6.3 TÉCNICAS DE DECISIÓN.

De acuerdo a lo presentado el proceso de decisión puede ser caracterizado como se muestra en el diagrama de la Ilustración 6.

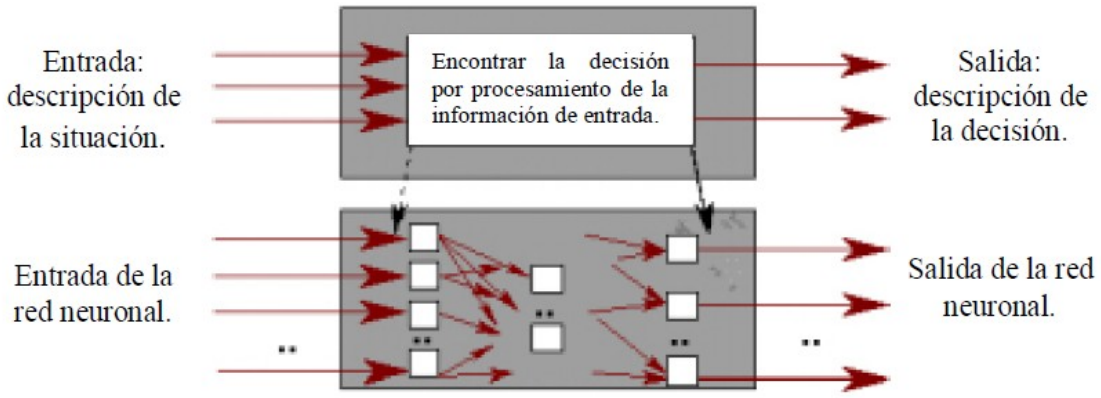


Figura 5 ESQUEMA DEL PROCESO DE DECISIÓN

(Oropeza, 2006)

2.6.4 ELECCIÓN DEL CONJUNTO INICIAL DE PESOS.

Antes de comenzar el proceso de entrenamiento se debe determinar un estado inicial, lo que significa: escoger un conjunto inicial de pesos para las diversas conexiones entre las neuronas de la red neuronal. Esto puede realizarse por varios criterios; por ejemplo, uno de "ellos es otorgar un peso aleatorio a cada conexión, encontrándose los mismos dentro de un cierto intervalo. Generalmente un intervalo del tipo [-n, n], donde n es un número natural positivo". (José R Hilera Martínez, 2000)

Cabe mencionar que durante el transcurso del entrenamiento los pesos no se encuentran restringidos a dicho intervalo.

2.7 TOPOLOGÍA DE LAS REDES NEURONALES.

Al hablar de redes neuronales lo que primero debemos definir es su topología o arquitectura de la red neuronal, que consiste en la organización y disposición de las neuronas, formando capas o subcapas de neuronas alejadas de la entrada y salida de dicha red. De acuerdo a esto los parámetros fundamentales de la red neuronal son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas.

2.7.1 REDES MONO CAPA.

"En las redes mono capa, se establecen conexiones entre las neuronas que pertenecen a la única capa que constituye la red. "Las redes mono capas se utilizan generalmente en tareas relacionadas con lo que se conoce como auto asociación (regenerar información de entrada que se presenta a la red de forma incompleta o distorsionada". (Lopez, 2003)

2.7.2 REDES MULTICAPA.

"Las redes multicapas son aquellas que disponen de un conjunto de neuronas agrupadas en varios (2, 3, etc.) niveles o capas. En estos casos, una forma para distinguir la capa a la que pertenece una neurona, consistiría en fijarse en el origen de las señales que recibe a la entrada y el destino de la señal de salida. Normalmente, todas las neuronas de una capa reciben señales de entrada desde otra capa anterior (la cual está más cerca a la entrada de la red), y envían señales de salida a una capa posterior (que está más cerca a la salida de la red). A estas conexiones se las denomina conexiones hacia adelante o feedforward". (José R Hilera Martínez, 2000)

Sin embargo, en un gran número de estas redes también existe la posibilidad de conectar la salida de las neuronas de capas posteriores a la entrada de capas anteriores; a estas conexiones se las denomina conexiones hacia atrás o feedback.

"Estas dos posibilidades permiten distinguir entre dos tipos de redes con múltiples capas: las redes con conexiones hacia adelante o redes feedforward, y las redes que disponen de

conexiones tanto hacia adelante como hacia atrás o redes feedforward/feedback". (Aapo Hyyarinen, Erkki Oja, 2000)

2.7.3 CONEXIÓN ENTRE NEURONAS.

"La conectividad entre los nodos de una red neuronal está relacionada con la forma en que las salidas de las neuronas están canalizadas para convertirse en entradas de otras neuronas. La señal de salida de un nodo puede ser una entrada de otro elemento de proceso, o incluso ser una entrada de sí mismo (conexión auto-recurrente)" (ben Krose, Patrick Van ser Smagt).

Cuando ninguna salida de las neuronas es entrada de neuronas del mismo nivel o de niveles precedentes, la red se describe como de conexión hacia delante

Cuando las salidas pueden ser conectadas como entradas de neuronas de niveles previos o del mismo nivel, incluyéndose ellas mismas, la red es de conexión hacia atrás.

"Las redes de propagación hacia atrás que tienen lazos cerrados son llamadas: sistemas recurrentes". (ben Krose, Patrick Van ser Smagt).

2.7.4 REDES DE PROPAGACIÓN HACIA ATRÁS (BACKPROPAGATION).

El nombre de backpropagation resulta de la forma en que el error es propagado hacia atrás a través de la red neuronal, en otras palabras el error se propaga hacia atrás desde la capa de salida. Esto permite que los pesos sobre las conexiones de las neuronas ubicadas en las capas ocultas cambien durante el entrenamiento. "El cambio de los pesos en las conexiones de las neuronas además de influir sobre la entrada global, influye en la activación y por consiguiente en la salida de una neurona. Por lo tanto, es de gran utilidad considerar las variaciones de la función activación al modificarse el valor de los pesos. Esto se llama sensibilidad de la función activación, de acuerdo al cambio en los pesos". (Veelenturf, 2008)

2.8 COMPARACION ENTRE PROGRAMACION NORMAL Y PROGRAMACION NEURONAL

Las técnicas tradicionales de programación que se emplean para la solución de un problema es la de crear un algoritmo que lo resuelva. "Se denomina algoritmo a un grupo finito de operaciones organizadas de manera lógica y ordenada que permite solucionar un determinado problema. Se trata de una serie de instrucciones o reglas establecidas que, por medio de una sucesión de pasos, permiten arribar a un resultado o solución" (RAE)

El diseño de un algoritmo para resolver un problema de contabilidad es sumamente sencillo, en cambio el crear un algoritmo para resolver diferentes problemas del mundo real es más complejo. La diferencia al desarrollar un algoritmo de RNA es que esta debe ser entrenada previamente, esto significa que en la capa de entrada de la red se muestra un ejemplo y la red se ajusta a lo que debe cumplir.

Las RNA's presentan una arquitectura totalmente distinta a las de un ordenador que por lo general es de arquitectura Von Neumann, que son procesos ejecutados en el orden que el algoritmo les dice, las operaciones que puede realizar son sumas y restas, etc., estas acciones son sincronizadas por un reloj.

En las RNA su capacidad de procesamiento se mide por la velocidad de interconexiones realizadas por segundo en el proceso de entrenamiento, además de que cuentan con sistema paralelo de ejecución de procesos.

Una de las características principales de las RNA es su capacidad de almacenamiento, la memoria está distribuida a lo largo de las conexiones ponderadas de la red. Algunas RNA presentan la característica de ser "asociativas" que significa que para una entrada parcial la red elegirá la entrada más parecida en memoria y generará una salida que corresponda a la entrada completa.

2.9 APLICACIONES DE LA REDES NEURONALES ARTIFICIALES (RNA).

Las principales características de los sistemas que emplean redes neuronales artificiales son los procesos en los cuales la exactitud es de prioridad alta.

La aplicación de las RNA provee de un acercamiento al reconocimiento y percepción humana de los métodos tradicionales de cálculos, las redes neuronales artificiales presentan resultados razonables en aplicaciones donde las entradas presentan ruidos; las principales aplicaciones son:

- Análisis y procesos de señales
- Control de procesos
- Robótica
- Diagnostico Médicos
- Reconocimientos de Imágenes
- Filtrada de Ruido
- Procesador del Lenguaje

(Olabe, Curso del Redes Neuronales Artificiales y sus Aplicaciones, 2007)

2.10 RASPBERRY PI

2.10.1 DEFINICIÓN

El Raspberry Pi es un bajo costo, de tarjetas de ordenador de tamaño que se conecta a un monitor de ordenador o un televisor, y utiliza un teclado y un ratón estándar. Es un dispositivo poco capaz que permite a las personas de todas las edades para explorar la computación, y para aprender a programar en lenguajes como arañazos y Python. Es capaz de hacer todo lo que espera de una computadora de escritorio que se puede hacer, desde navegar por Internet y reproducción de vídeo de alta definición, que hacen las hojas de cálculo, procesadores de texto, y jugar juegos. (Pi, 2011)

Raspberry Pi al ser un ordenador de placa reducida y de bajo coste, con amplias funciones de escalabilidad en proyectos innovadores, es el sistema embebido escogido para la interacción con el medio externo del algoritmo de entrenamiento.

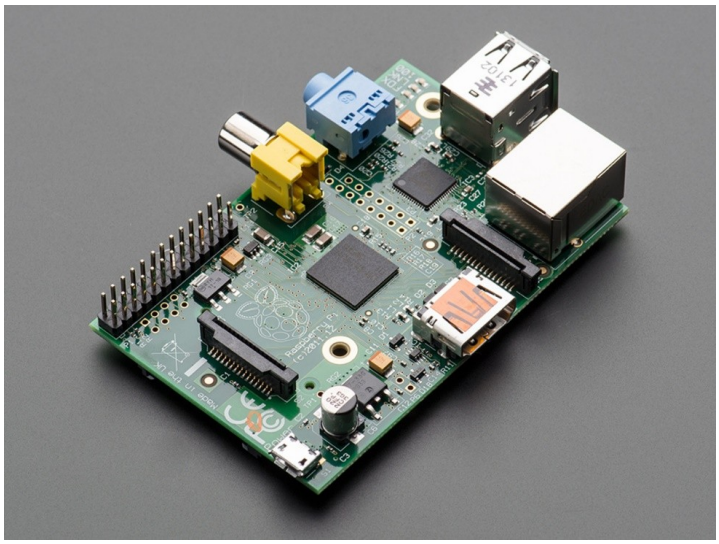


Figura 6 PLACA RASPBERRY PI

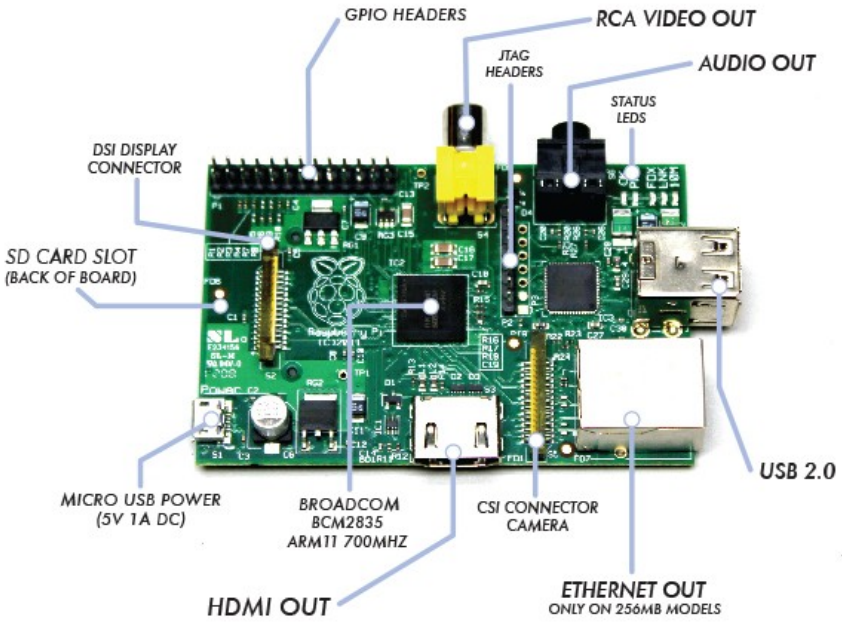


Figura 7 COMPONENTES DE LA PLACA RASPBERRY PI

2.10.2 CARACTERÍSTICAS Y ESPECIFICACIONES TÉCNICAS

Una de las principales características que se debe destacar de Raspberry pi es si funcionalidad en diferentes sistemas operativos; los diferentes modelos con los que actualmente cuenta Raspberry pi están diseñados a la optimización de sus proyectos y aplicaciones.

A continuación un cuadro donde se detalla las especificaciones de los modelos de Raspberry pi.

Tabla 3. CARACTERÍSTICAS DE LOS MODELOS DE RASPBERRY PI

Características	Modelo A	Modelo B	Modelo B+
Sistema en el Chip (SoC)	Broadcom BCM2835 (CPU+GPU+DSP+SDRAM+PuertoUSB)		

CPU	ARM 1176JZF-S a 700 MHz (Familia (ARM 11)		
Juego de Instrucciones	RISC de 32 Bits		
GPU	Broadcom VideoCore IV OpenGL ES2.0,MPEG-2 y VC-1 (con licencia), 1080p30 H.264/MPEG-4 AVC		
Memoria (SDRAM)	256MiB (Compartidos con la GPU	512 MiB (compartidos con la GPU)	
Puertos USB 2.0	1 Puerto	2 Puertos(via hub USB integrado)	4 Puertos
Entradas de Video	Conector MIPI CSI que permite instalar un módulo de cámara desarrollado por la RPF		

Salidas de Video	Conector RCA (PAL y NTSC), HDMI (Rev1.3 y 1.4) Interfaz DSI para el panel LCD.	
Salidas de Audio	Conector de 3.5mm, HDMI	
Almacenamiento integrado	SD / MMC / Ranura para SDIO	Micro SD
Conectividad de red	Ninguna	10/100 Ethernet (RJ-45) Via hub USB
Periféricos de bajo nivel	8 x GPIO, SPI, I2C, UART	

Reloj en tiempo real	Ninguno		
Consumo Energético	500 mA (2.5 V)	700 mA (3.5 V)	600 mA (3.0 V)
Fuente de alimentación	5 V Vía MicroUSB o GPIO header		
Dimensiones	85.60mm x 53.98mm		
Sistemas Operativos Soportados	GNU/Linux: Debian (Raspbian), Fedora (Pidora), Arch Linux (Arch Linux ARM), Slackware Linux, RISC OS.		

2.11 PROTOCOLO TCP/IP

En los años 70 y 80 se produjo un cambio entre los campos de los computadores y las comunicaciones que ha desencadenado un cambio drástico en las tecnologías, productos y en las empresas que se dedican al mundo de las comunicaciones. La revolución que se menciona ha producido los siguientes cambios significativos:

- No hay grandes diferencias entre el procesamiento de datos y las comunicaciones de datos.
- No hay diferencia fundamental entre la transmisión de datos, de voz o de video.
- Las fronteras entre computadores, monoprocesador o multiprocesadores; así como entre las redes local, regional y nacional son cada vez más difusas.

Como sabemos el objetivo primordial de todo sistema de comunicaciones es intercambiar información ente dos entidades, el ejemplo más claro que podemos tener en entre una estación de trabajo y un servidor a través de una red telefónica.

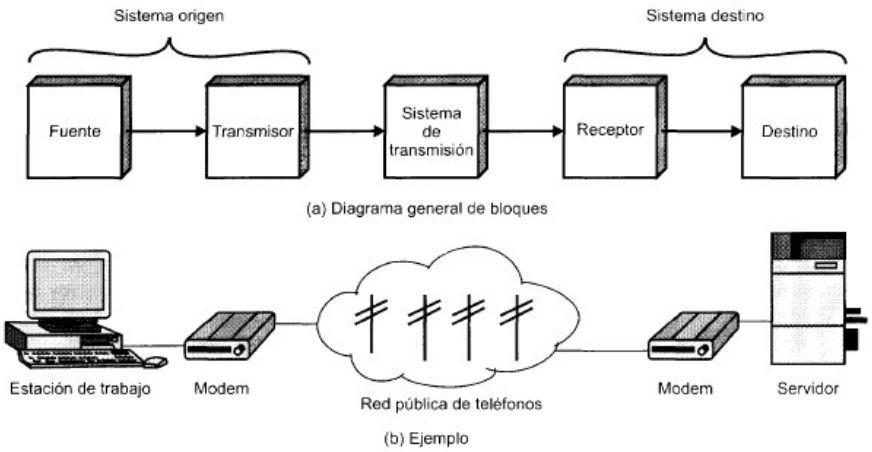


Figura 8 MODELO SIMPLIFICADO PARA LAS COMUNICACIONES

Stallings William. Comunicaciones y redes de computadoras.

Aunque el este esquema se puede observar un sistema de comunicación sencillo, implica una gran complejidad, para que se pueda entender mejor las tareas que se deben realizar en el sistema de comunicaciones en la siguiente tabla se detalla todos los elementos que intervienen.

Tabla 4. ELEMENTOS EN LOS SISTEMAS DE COMUNICACIONES.

- Utilización del sistema de transmisión.
- Implementación de la interfaz.
- Generación de la señal
- Sincronización
- Gestión del intercambio
- Detección y corrección de errores.
- Control de flujo.
- Direccionamiento
- Encaminamiento
- Recuperación
- Formato de mensajes
- Seguridad
- Gestión de red.

(Stallings William Comunicaciones Y Redes De Computadoras, 2000)

El tipo de comunicación que se va a utilizar en este proyecto es punto a punto, con el protocolo TCP/IP, por la facilidad de configuración entre Matlab y Raspberry pi.

2.11.1 PROTOCOLO TCP/IP

Dentro del mundo de los protocolos de comunicaciones han existido dos arquitecturas básicas y determinantes en el desarrollo de los estándares de comunicación; el conjunto de protocolos TCP/IP; que es la arquitectura más adoptada para la interconexión de sistemas.

TCP/IP es el resultado de la investigación y desarrollo llevados a cabo en la red experimental de comunicaciones de paquetes ARPANET, y se denomina globalmente como la familia de protocolos TCP/IP, esta familia consiste en una extensa colección de protocolos que se han erigido como estándares de internet.

Dentro de los protocolos estándares que se han desarrollado, todas las tareas involucradas en la comunicación este se puede organizar en cinco capas relativamente independientes:

- **Capa de aplicación.-** Esta capa contiene la lógica necesaria para posibilitar las distintas aplicaciones de usuario. Para tipo de aplicación, como por ejemplo la trasferencia de ficheros, se necesita un módulo bien diferenciado.
- **Capa de origen-destino o de transporte.-** Independientemente de la naturaleza de las aplicaciones que están intercambiando datos, es usual requerir que los datos se intercambian de forma segura. Esto sería deseable asegurar que todos los datos llegan a la aplicación destino y en el mismo orden que fueron enviados. Los procedimientos que garantizan una transmisión segura están localizados en esta capa. El protocolo TCP (Transmission Control Protocol), es el as utilizad para proporcionar esta funcionalidad.
- **Capa internet.-** la capa de acceso a la red está relacionada con el acceso y encaminamiento de los datos a través de la red. En situaciones en las que dos dispositivos estén conectados a redes diferentes, se necesitaran una serie de procedimientos que permitan que los datos atraviesen las distintas redes interconectadas, esta es la función de la capa internet. El protocolo internet (IP, internet protocol) se utiliza en esta capa para ofrecer el servicio de encaminamiento a través de varias redes. Este protocolo se implementa tanto en los sistemas finales como en los routers intermedios
- **Capa de acceso a la red.-** es el responsable del intercambio de datos entre el sistema final y la red a la cual se está conectando. El emisor debe proporcionar a la red la

dirección del destino, de tal manera que la red pueda encaminar los datos hasta el destino apropiado. El emisor puede requerir ciertos servicios, como por ejemplo solicitar una determinada prioridad que pueden ser proporcionados por el nivel de red. El software en particular que se use en esta etapa dependerá del tipo de red que se disponga.

- **Capa física.-** se define como la interfaz física entre el dispositivo de transmisión de datos y el medio de transmisión o red. Esta capa se encarga de la especificación de las características del medio de transmisión, la naturaleza de las señales, la velocidad de datos y cuestiones afines.

En el siguiente grafico se puede ver como se aplica cada uno de las capas en un sistema de comunicación:

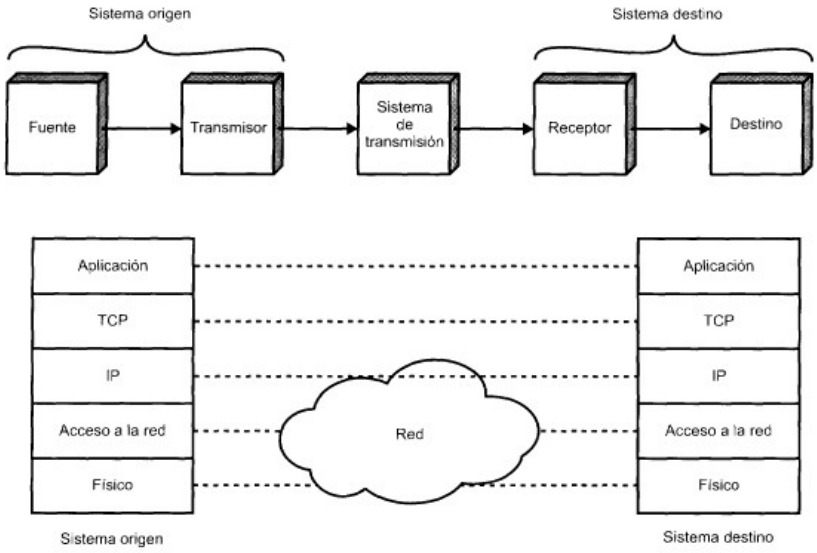


Figura 9 MODELO DE ARQUITECTURA DE PROTOCOLO.

(Comunicaciones y redes de computadores, Stallings Willians, 2000)

2.11.2 ARQUITECTURA TCP/IP

El conjunto de protocolos TCP/IP reconoce la tarea de la comunicación es la suficientemente compleja y diversa como para realizarse en una única unidad.

Consecuentemente, la tarea se descompone en diversos módulos o entidades que se pueden comunicar con sus entidades pares del sistema remoto. Una entidad dentro de un sistema proporciona servicios a otras entidades y, a su vez, utiliza los servicios de otras entidades. Las reglas de diseño del software de calidad dictan que estas entidades se deben agrupar de forma jerárquica. (William S. ,2000).

Para conectar un computador a una subred se utiliza algún tipo de protocolo de acceso como por ejemple el Ethernet; este protocolo permite al computador enviar datos a través de la subred a otro computador o, en caso de que el destino final este en otra subred, a un dispositivo de encaminamiento IP se implementara en todos los sistemas finales y dispositivos de encaminamiento.

Como ya se detalló anteriormente las fases de un comunicación del protocolo TCP/IP son:

- Capa de Aplicación
- Capa de transporte o extremo a extremo
- Capa internet
- Capa de acceso a la red
- Capa física

Para tener éxito en la transmisión entre dos dispositivos, cada uno debe tener en su sistema global una única dirección, en la siguiente grafica se puede observar cómo actúa el protocolo en la comunicación entre dos dispositivos.

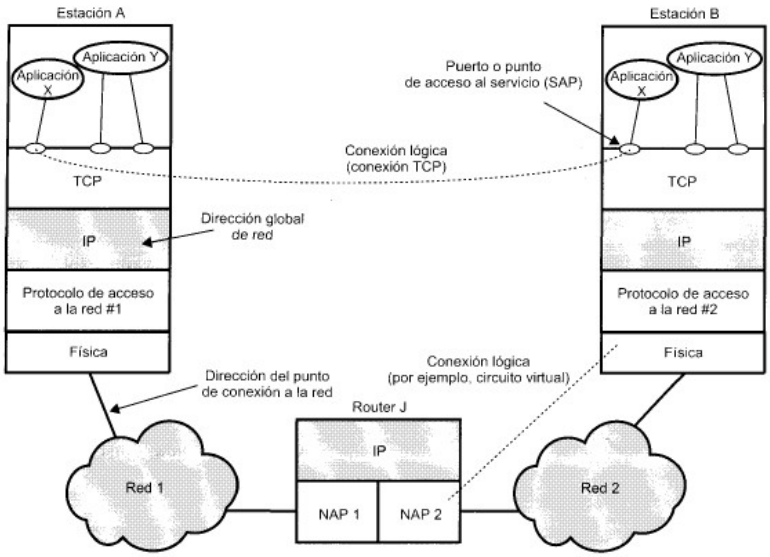


Figura 10. CONCEPTOS DE DIRECCIONAMIENTO.

Para tener una idea clara de la arquitectura del protocolo TCP/IP que se muestra en la ilustración 14, vamos a suponer que un proceso está asociado al Puerto 1 del computador A, y se desea enviar un mensaje a otro proceso, que sería el puerto 2 del computador B; el TCP pasa el mensaje al IP con instrucciones de que envié al computador B, no es necesario comunicarle al IP la identidad el puerto destino. Todo lo que se necesite saber son los datos que van dirigidos al computador B, lo que se hace el IP es pasar el mensaje a la capa de acceso de la red, que será lo lógica la Ethernet, con la orden de enviarlo al dispositivo de encaminamiento X.

Al realizar esta operación se debe transmitir información de control junto con los datos de usuario.

Los datos que se deben tomar en cuenta para la transmisión son los siguientes:

- Puerto destino.- cuando la entidad TCP en B recibe el segmento, debe conocer a quien no se la deben entregar los datos.

- Numero de Secuencia.- TCP numera secuencialmente los segmentos que envía a un puerto de destino que si llegan desordenados la entidad TCP en B pueda reordenarlos.
- Suma de comprobación.- la entidad emisora TCP incluye un código calculado en función del resto del segmento. La entidad receptora TCP realiza el mismos calculo y compara el resultado con el código secreto. Si se observa alguna discrepancia implicara que habido algún error en la transmisión.

Luego de todo esto TCP pasa cada segmento al IP con las instrucciones para que los transmita a B; estos segmentos se transmitirán por una o varias subredes y serán transmitidas en uno o más dispositivos de encaminamiento intermedios.

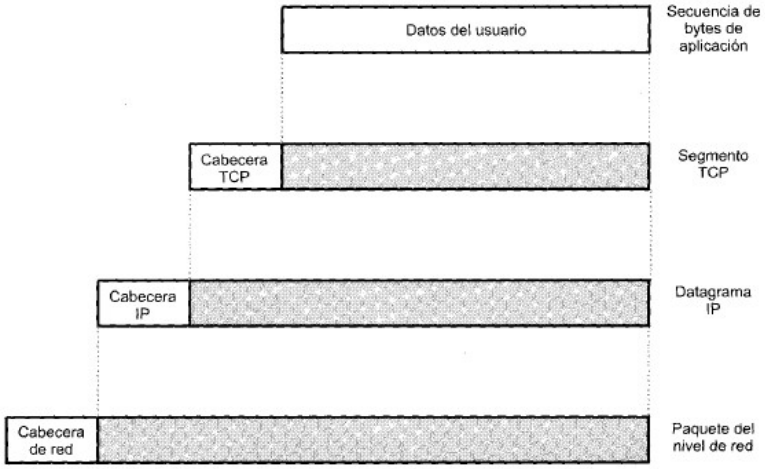


Figura 11 UNIDADES DE DATOS DE PROTOCOLO EN LA ARQUITECTURA TCP/IP

Para finalizar cada datagrama IP se pasa a la capa de acceso a la red para que se envié a través de la primera subred. La capa de acceso a la red añade su propia cabecera creando una trama o un paquete, este se transmite a través de la red al dispositivo de encaminamiento J.

Para este proyecto la comunicación entre Matlab y Raspberry la configuración que nos da puede ser la de crear una red local entre el modem que nos provee internet o una comunicación directa entre computador y la Raspberry.

18. CAPITULO 3

19. 3. ANÁLISIS DEL MODELO MATEMÁTICO DE LA NEURONA

3.1 INTRODUCCIÓN

En este capítulo se va a detallar como se obtiene el modelo matemático de la neurona; la manera en la que se va a realizar el algoritmo la selección de los pesos y el bias; además de las distintas maneras en las que se puede entrenar un algoritmo de backpropagation y por qué se haelegido este método para entrenar la neurona en la parte de reconocimientos de patrones de imágenes y voz.

3.2 SELECCIÓN DE LA RED NEURONALES ARTIFICIAL

El desarrollo de este capítulo va a estar enfocado netamente en el método y procedimiento a utilizar para el entrenamiento de la red neuronal; se va a especificar por qué se ha escogido el algoritmo de back propagación para el desarrollo y la implementación de este proyecto; además de las comparaciones de los diferentes métodos de entrenamientos de redes.

En la siguiente tabla tenemos los diferentes tipos de algoritmos que se pueden utilizar para el entrenamiento y así como el tipo de red a la que pertenece:

Tabla 5. MÉTODOS DE ENTRENAMIENTO DE UNA RED NEURONAL

MÉTODOS DE ENTRENAMIENTO	DISEÑADOR	CARACTERISITICAS	AÑO	TIPO
Adeline y Madaline	Bernad Widrw	Técnicas de adaptación para el reconocimiento de patrones	1960	Predicción
Adaptive Resonance Theory Networks (ART)	Carpenter, Gossberg	Reconocimiento de patrones y Modelo del Sistema Neuronal, Concepto de Resonancia Adaptativa	1960 - 1986	Conceptualización
Back - propagation	Rumelhart y Parker	Solución a las limitaciones de su red predecesor Perceptrón	1985	Clasificación
Bi-Directional Associative Memory (BAM) Memory	Bart Kosko	Inspirada en la red ART	1987	Asociación
The Boltzmann Machine	Ackley, Hinton y Sejnowski	Similar a la red Hopfield.	1985	Asociación
Brain Stateina a Boz	James Anderson	Red Asociativa Lineal	1970 1986	Asociación
Cascade-Correlation-Networks	Fahhman y Lebiere	Adición de nuevas capas ocultas en cascada	1990	Asociación
Counter-Propagation	Hecht-Nielsen	Clasificación adaptativa de patrones	1987	Clasificación
Delta-bar.delta (DBD) Networks	Jacob	Métodos Heurísticos para acelerar la convergencia	1988	Clasificación
Digital neural Network Architecture (DNNA) Networks	Neural Semiconductor Inc.	Implementación Hardware de la función Sigmoid	1990	Predicción
DirectedRandom Search (DRS) Networks	Maytas y Solis	Técnica de valores random en el mecanismos de ajuste de pesos	1965-1981	Clasificación

Functional-Link Networks (FLN)	Pao	Versión Mejorada de la red Back propagación	1989	Clasificación
Hamming Networks	Lippman	Clasificador de vectores binarios utilizando la distancia Hamming	1987	Asociación
Hopfield Networks	Hopfielp	Concepto de la red en términos de energía	1982	Optimización
Learning Vector Quantization (LVQ) Networks	Kohomen	Red clasificadora	1988	Clasificación
Perceptron Networks	Rosenblatt	Primer modelo de sistema neuronal artificial	1950	Predicción
Probabilistic Neural Network (PNN)	Spetcht	Clasificación de patrones utilizando métodos estadísticos	1988	Asociación

Recirculation Networks	Hinton y McClelland	Alternaiva de la red backpropagación	1988	Filtrado
Self- Organizing Map (SOM)	Kahonen	Aprendizaje sin supervisión	1979-1982	Conceptualización
Spatio- temporal-Pattern Reconnition (SPR)	Grossberg	Red clasificadora Invariante en el espacio y tiempo	1960 1970	Asociación

Luego de un investigación y análisis comparativo con todos métodos de entrenamiento de redes detallados y en las diferentes áreas en las que pueden ser implementados, se ha decidido implementar un algoritmo de backpropagation en el desarrollo de este proyecto; las razones por la cuales se decide utilizar este tipo de entrenamiento es porque el software en el cual se realiza el entrenamiento nos brinda la herramienta necesaria.

3.3 ALGORITMO DE BACKPROPAGATION

3.3.1 INTRODUCCIÓN

En la búsqueda por desarrollar mejores algoritmos de entrenamiento de redes multicapa en los últimos años no se ha tenido los resultados esperados; ya que después de la comprobación de los sistemas de una sola capa nadie se dedicó a desarrollar los sistemas multicapas.

Uno de los grandes avances logrados con el algoritmo Backpropagation que es esta red aprovecha la naturaleza paralela de las redes neuronales para reducir el tiempo requerido por un procesador secuencial para determinar la correspondencia entre unos patrones dados. Además el tiempo de desarrollo de cualquier sistema que se esté tratando de analizar se puede reducir como consecuencia de que la red puede aprender el algoritmo correcto sin que alguien tenga que deducir por anticipado el algoritmo en cuestión.

3.3.2 ARQUITECTURA DE LA RED DE BACK PROPAGACIÓN

La estructura de la unidad de procesamiento de una red back propagación está formada por las entradas que se encuentran en el lado izquierdo, y a la derecha se encuentran la salida de la red, como se muestra en la figura.

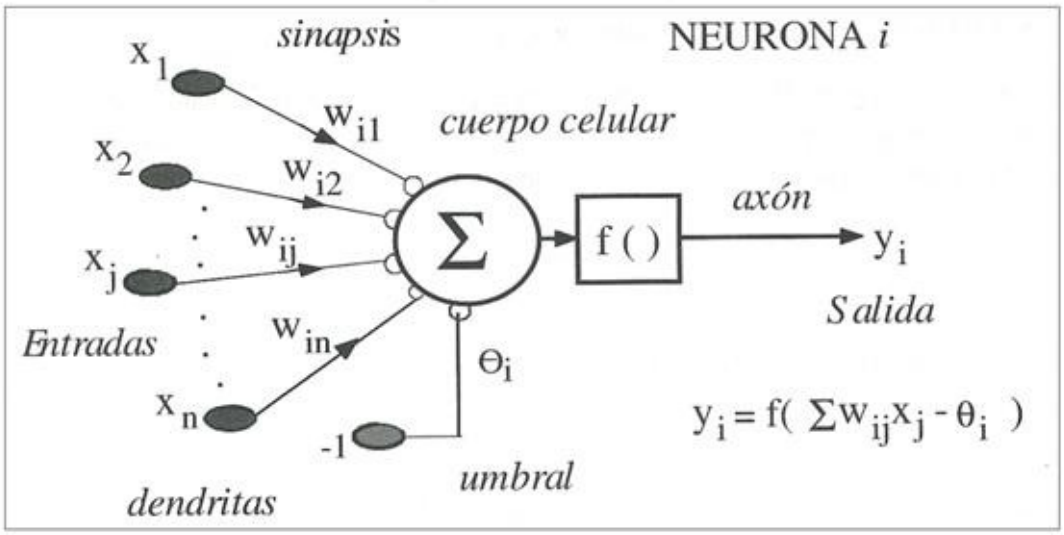


Figura 12 UNIDAD PROCESADORA BÁSICA BACKPROPAGATION

El procesamiento de la red de backpropagation se caracteriza por la suma ponderada de las entradas X_4, presenta una salida y_3 y tener un valor θ_3, asociado que se utilizara en el proceso de ajuste de los pesos. El peso está asociado a la conexión desde la unidad j a la unidad i que se representa por w_{34}, y es modificado durante el proceso de aprendizaje.

La topología de la red de backpropagation utiliza tres o más capas de unidades de procesamiento, en donde la capa inferior es la capa de entrada y se caracteriza por ser la única capa cuyas unidades de procesamiento reciben entradas desde el exterior, sirven como distribuidores esta capa no realiza ningún calculo. Las unidades procesadoras de las demás capas procesan las señales; la segunda capa es la capa oculta y todas sus unidades están interconectadas con la capa de entrada y la capa de salida; la capa finales la salida y la respuesta de la red.

El algoritmo Backpropagation emplea un ciclo propagación – adaptación de dos fases. Una vez que se ha aplicado un patrón a la entrada de la red como estímulo, este se propaga desde la primera capa a través de las capas superiores de la red, hasta generar una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas.

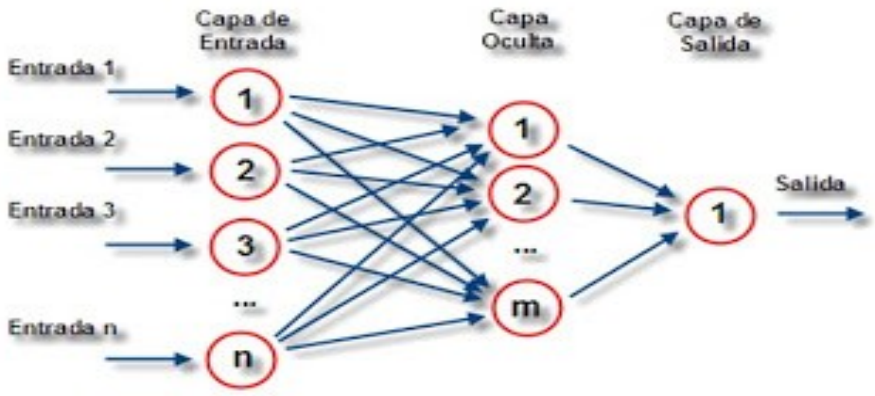


Figura 12 RED BACKPROPAGATION COMPLETAMENTE INTERCONECTADA.

3.3.3 ECUACIONES DEL ENTRENAMIENTO DEL ALGORITMO DE BACKPROPAGATION

"Es un método de aprendizaje supervisado de gradiente descendente, en el que se distinguen claramente dos fases: primero se aplica un patrón de entrada, el cual se propaga por las distintas capas que componen la red hasta producir la salida de la misma. Esta salida se compara con la salida deseada y se calcula el error cometido por cada neurona de salida. Estos errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de las capas intermedias". (Fritsch, 1996)

El método de entrenamiento que emplean las redes backpropagation es un entrenamiento supervisado, a la red se le presentan patrones, es decir un patrón de entrada emparejado con un patrón de salida a obtener. Por cada entrenamiento los pesos sinápticos son ajustados en forma que disminuyen el error entre la salida que deseamos y la respuesta de la red.

Cada neurona recibe un error proporcional a su contribución sobre el error total de la red, basándose en el error recibido se ajustan los pesos sinápticos de cada neurona.

$$\boldsymbol{W(t+1) = W(t) - \Delta W(t)} \qquad \textit{ECUACIÓN 3-1}$$

"El algoritmo propone una actualización iterativa de los pesos de la siguiente manera:

$$\boldsymbol{W(t+1) = W(t) - \alpha \Delta E[W(t)]} \qquad \textit{ECUACIÓN 3-2}$$

Si tomamos una variación proporcional al gradiente de una función de error E(W) tenemos que:

Como ya se lo digo antes el primer paso de este algoritmo consiste en propagar hacia adelante un patrón de entrada Xp y obtener una salida de la red Yp.

La salida de la neurona i viene dada según su estado de activación si consideraos la función identidad tenemos que:

$$\boldsymbol{y_i(t) = F_i\big(a_i(t)\big) = a_i(t)} \qquad \textit{ECUACIÓN 3-3}$$

Siendo

$$\boldsymbol{a_i(t) = f_i\big(h_i(t)\big)} \qquad \textit{ECUACIÓN 3-4}$$

La regla de propagación más simple y utilizada consiste en realizar una suma de las entradas ponderas con sus pesos sinápticos correspondientes.

$$\boldsymbol{h_i(t) = \sum_j w_{ij} * x_i\big((t)\big)} \qquad \textit{ECUACIÓN 3-5}$$

Se compara la salida obtenida Yp con la salida deseada Dp, obteniendo un error que viene dada por:

$$\boldsymbol{e_p = \frac{1}{2}\sum_{k-1}^{M}(d_{pk} - y_{pk})} \qquad \textit{ECUACIÓN 3-6}$$

Donde k es el índice de neurona para las neuronas de la última capa, y M el total de neuronas de la misma.

El error total de la red está dado por :

$$e = \frac{\sum_{p-1}^{p} e_p}{P} \qquad \textit{ECUACIÓN 3-7}$$

Siendo p el índice de ejemplo, y P el número de total de ejemplos.

De acuerdo a la ecuación (3- 2) la variación de los pesos sinápticos será proporcional al gradiente de la función error:

$$\Delta w_{ij} = -\alpha \frac{\delta_{ep}}{\delta w_{ji}} \qquad \textit{ECUACIÓN 3-8}$$

Si aplicamos la regla de cadena a la ecuación (3-8) obtenemos que:

$$\frac{\delta_{ep}}{\delta w_{ji}} = \frac{\delta_{ep}}{\delta h_{ji}} \frac{\delta h_j}{dw_{ji}} \qquad \textit{ECUACIÓN 3-9}$$

La ecuación (3-9) nos representa la derivada del error en función de dos derivadas. La derivada del error respecto al potencial resultante h_4 indica como varía el error al variar la entrada de la neurona j; mientras que la derivada con respecto al peso sináptico w_{43} indica como varia la entra de la neurona j al variar el peso de la conexión que va desde la neurona i hasta la neurona j.

El segundo término de la expresión (3-9) lo podemos expresar a partir de la ecuación (3-5) de la siguiente manera:

$$\frac{\delta h_j}{\delta w_{ji}} = \frac{\delta \Sigma_i w_{ij} y_{pi}}{\delta w_{ji}} = y_{pi} \qquad \textit{ECUACIÓN 3-10}$$

Si escribimos al primer término de la ecuación (3-9) como:

$$\frac{\delta e_p}{\delta h_j} = -\delta_{pi} \qquad \textit{ECUACIÓN 3-11}$$

Tenemos que:

$$\frac{\delta e_p}{\delta w_{ji}} = -\delta p_i y_{pi} \qquad \textit{ECUACIÓN 3-12}$$

Y por lo tanto la ecuación (3.8) queda expresada de la siguiente manera:

$$\Delta w_{ji} = -\alpha \delta_{pi} y_{pj} \qquad \textit{ECUACIÓN 3-13}$$

Para calcular el valor de delta se vuelve a aplicar la regla de la cadena:

$$\delta_{pi} = \frac{\delta_{ep}}{\delta h_j} = \left[\frac{\delta_{ep}}{\delta y_{pj}} \frac{\delta y_{pj}}{\delta h_j}\right] \qquad \textit{ECUACIÓN 3-14}$$

El cálculo del segundo término de la ecuación (3-14) es simple si observamos las ecuaciones (3-3) y (3.4).

$$\frac{\delta y_{pj}}{\delta h_j} = \frac{\delta f_j(h_j)}{\delta h_j} = f_j(h_j) \qquad \textit{ECUACIÓN 3.15}$$

Ecuaciones tomadas de (Fausett., 2005)

Al definir todas las ecuaciones básicas del algoritmo de backpropagation podemos iniciar con la programación básico del algoritmo que se va a utilizar en la aplicación de reconocimiento de patrones de imágenes y voz.

3.4 ENTRENAMIENTO DE LA RED NEURONAL PARA EL RECONOCIMIENTOS DE PATRONES DE IMAGENES

Para poder entrenar a la neurona, se va aplicar un algoritmo en el cual se podrá ingresar la matriz que se obtiene de la cámara web, al ya tener el tipo de matriz de las imágenes la neurona reconoce, procese a la comparación entre matrices y ahí es donde se ajustan los pesos, el bias y la cantidad de neuronas ocultas para tener un sistema más eficiente.

Las ecuaciones que se van aplicar a nuestro algoritmo de entrenamiento para el reconocimiento de patrones son las siguientes:

Ecuación de activación de entrada (input activation)

$$\boldsymbol{a(x) = b + \sum_i w_i x_i = b + w^T x} \qquad \textit{ECUACIÓN 3.16}$$

Ecuación de activación de la salida de la neurona (Output Activation)

$$\boldsymbol{h(x) = g\big(a(x)\big) = g(b + \sum_i w_i x_i)} \qquad \textit{ECUACIÓN 3.17}$$

En donde:

- **W :** Son los pesos de conexión
- **b:** es el sesgo de la neurona (Bias)
- **g(.):** Función de activación

Durante el proceso de entrenamiento del algoritmo se determinaron los valores con los cuales se va a ejecutar el proyecto, estos valores pueden variar dependiendo del tiempo y de exactitud con lo que se reconozcan los patrones de imágenes, mientras más se acerque a la matriz final. En el manual de funcionamiento del módulo didáctico se especificara los pesos que se arrancan para el entrenamiento de la neurona.

Para una mejor explicación sobre la función de activación, que es la que hace que la neurona se pueda entrenar, vamos a detallar como se hace matemáticamente este proceso, debemos mencionar que las funciones de activación del algoritmo de backpropagation pueden ser: lineal, Sigmoidea, tangente hiperbólica.

En el siguiente cuadro se detalla las características de cada una de estas funciones con sus respectivas ecuaciones.

Tabla 6. FUNCIONES MATEMÁTICAS DEL ALGORITMO DE BACKPROPAGATION

Función de Activación	Ecuación g(a)	Caracteristicas
Lineal	$g(a) = reclin(a) = \max(0.a)$	Es una de las funciones en las cuales no se puede detectar un aprendizaje. La curva de aprendizaje

		siempre será la misma Los valores parten de 0 y van en ascenso. No tiene valores negativos.
Sigmoide	$g(a) = sigm(a) = \frac{1}{a + \exp(-a)}$	Los valores en la neurona están entre 0 y 1. Siempre son positivos. Delimitada Estrictamente Creciente.
Tangente Hiperbólica	$g(a) = tanh(a) = \frac{\exp(a) - \exp(-a)}{\exp(a) + \exp(-a)} = \frac{\exp(2a) - 1}{\exp(2a) + 1}$	Los valores de entrada de la neurona están entre -1 y 1. Pueden ser positivos o negativos Delimitida Estrictamente creciente.

En este cuadro podemos observar las ventajas de un algoritmo de backpropagation en el momento de elegir la ecuación de entrenamiento, podemos elegir la opción más eficiente en el tiempo.

3.5 ENTRENAMIENTO DE LA RED NEURONAL PARA EL RECONOCIMIENTO DE VOZ

3.5.1 ALGORITMO DE ENTRENAMIENTO DE LA RED PARA VOZ

El presente trabajo al estar enfocado en dos áreas distintas de reconocimiento de patrones como son los de imágenes y voz; el proceso para la voz es totalmente distinto al que se realizado en imágenes por la diferencia de señal que recibimos, pero esto no cambia el método de entrenamiento de la RNA sigue siendo backprogation.

$$y_{pi} = f_j^h\left(N_{pj}^h \sum_{i=1}^{m} w_{ji}^h x_{pi} + b_i^h\right) \qquad \textit{ECUACIÓN 3.19}$$

A continuación esta la explicación del entrenamiento de la red.

1. Inicializar los pesos de la red (w) con valores aleatorios pequeños.
2. Mientras la condición de paro sea falsa realizar los pasos (3-6).
3. Se presenta un patrón de entrada, $(x_{p1}, x_{p2}, \ldots, x_{pi})$ y se especifica la

salida deseada que debe generar la red ($d_{p1}, d_{p2,\dots,} d_{px}$).

4. Se calcula la salida actual de la red, para ello se presentan las entradas a la red y se va calculando la salida que presenta cada capa hasta llegar a la cada de salida ($y_1, y_{2,\dots,} y_x$. Los pasos son los siguientes:

a) Se determinan las entradas netas para las neuronas ocultas procedentes de las neuronas de entrada.

$$N_{pk}^{o} = \sum_{i=1}^{m} w_{ji}^{o}\, p_j b_k^{o} \qquad \textit{ECUACIÓN 3.20}$$

$$y_{pk} = f_k^0\left(\sum_{i=1}^{m} w_{ji}^{o}\, y_{ij} + b_k^{o}\right) = N_{pk}^{o} \qquad \textit{ECUACIÓN 3.21}$$

b) Se aplica la funcion de activacion a cada una de las entradas de la neurona ocualta para oabtener su respectiva salida.

$$e = (d_{pk} - y_{pk}) \qquad \textit{ECUACIÓN 3.22}$$

c) Se realizan los mismo cálculos para ontener las respectivas salidas de las neuronas de la capa de salida.

5. Determinacion de los términos de error para todas las neuronas:

$$\delta_{pk}^{0} = e * f_k^0(N_{pk}^{0}) \qquad \textit{ECUACIÓN 3.23}$$

a) Cálculo del error (Salida deseado- salida obtenida).

$$N_{pj}^{h} = \sum_{i=1}^{m} w_{ji}^{h}\, x_{pi} + b_i^{h} \qquad \textit{ECUACIÓN 3.18}$$

b) Obtención de la delta (producto del error con al derivada de la funcion de

activación con respecto a los pesos de la red).

6. Actualización de los pesos. Se emplea el algoritmo recursivo del gradiente descendente,

$$\boldsymbol{w_{kj}^{0h}(t+1) = w_{ij}^{h}(t) + \Delta w_{kj}^{h}(t+1);} \qquad \textit{ECUACIÓN 3.25}$$

comenzando por las neurnoas de la salida y trabajnado hacia atrás hasta llegar a la capa de entrada.

$$\boldsymbol{\Delta w_{kj}^{0}(t+1) = miu\delta_{pk}^{0} y_{pj}}$$

a) Para los pesos de las neuronas de la capa de salida:

b) Para los pesos de las neuronas de la capa oculta;

7. Se cumple la condición de paro (error minimo o número de iteraciones alcando logrado). (Luis. A. Cruz-Beltrán1 and Marco. A. Acevedo-Mosqueda, 2016).

3.6 SOFTWARE DE ENTRENAMIENTO

3.6.1 INTRODUCCIÓN

En este capítulo se detallará la parte principal del proyecto, la aplicación de todas las ecuaciones; la elaboración del algoritmo de backpropagation, la implementación del Toolbox y la comunicación entre Matlab y Raspberry pi; además en el desarrollo de este capítulo se explicara el por qué se ha elegido el software ya indicado, las diferentes opciones de entrenamiento de una red neuronal con backpropagation.

Es importante destacar la escalabilidad de este proyecto al tener una placa que nos brinda distintas aplicaciones en muchas áreas del control avanzado, en este proyecto se aplica en la comunicación con Matlab y la implementación en la parte remota del

$$\boldsymbol{w_{kj}^{0}(t+1) = w_{kj}^{0}(t) + \Delta w_{kj}^{0}(t+1);} \qquad \textit{ECUACIÓN 3.24}$$

control neuronal; en la escalabilidad que tenemos que pensar al implementar este proyecto podemos detallar todas la ventajas que nos brinda Raspberry pi.

3.6.2 MATLAB

En muchos casos para el entrenamiento de redes neuronales se hace necesario una interfaz gráfica la cual ayude en el procesamiento de los datos, para ello se ha elegido a MATLAB por su gran desempeño y aplicabilidad en varios ámbitos, exclusivamente en el área de sistemas de control para el procesamiento de señales y comunicaciones. En este proyecto se va aplicar en la versión R2015b, ya que cuenta con el la librería de Raspberry pi.

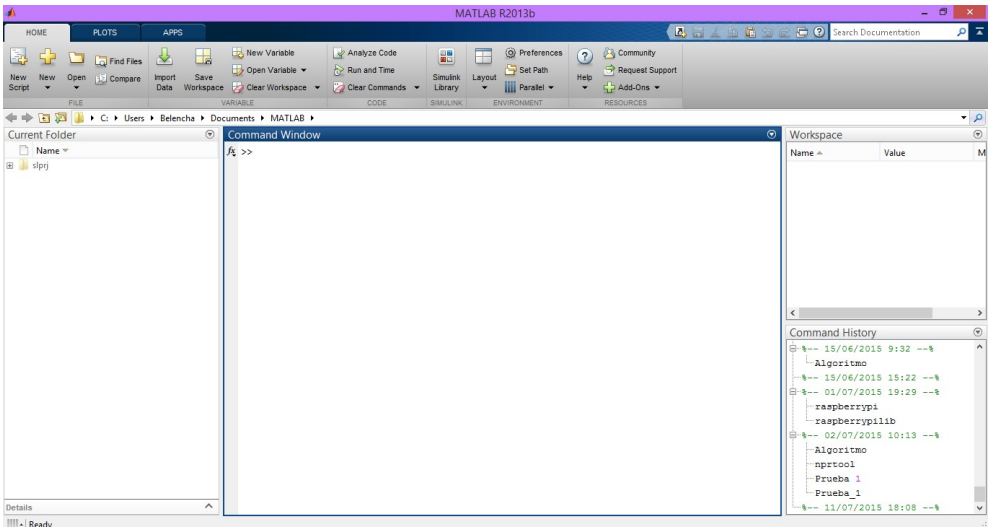


Figura 13. Escritorio de MATLAB

3.6.3 NEURAL NETWORK START

El Toolbox es herramientas de entrenamiento redes neuronales ofrece funciones y aplicaciones para el modelado de sistemas complejos no lineales que no son fácilmente representado por una ecuación de forma cerrada. El Toolbox para redes neuronales apoya el aprendizaje supervisado con alimentación directa, base radial y redes dinámicas. También es compatible con aprendizaje no supervisado, con mapas auto-organizados y

capas competitivos. Con las herramientas que usted puede diseñar, entrenar, visualizar y simular redes neuronales. También es compatible con aprendizaje no supervisado con los mapas de auto-organización y capas competitivos. Con el toolbox que usted puede diseñar, entrenar, visualizar y simular redes neuronales. Puede utilizar Neural Network Toolbox para aplicaciones tales como datos de ajuste, reconocimiento de patrones, la agrupación, la predicción de series de tiempo, y modelado de sistemas dinámicos y control. (Mathwork, 2015)

Los principales comandos que se utilizan en el Toolbox son:

Tabla 7. COMANDOS PRINCIPALES DEL NEURAL NETWORK START

COMANDO	DEFINICIÓN
nnstart	Red neuronal empieza la interfaz del gui
nprtool	Herramienta de reconocimiento de patrones
nctool	Abre la gui neural agrupación
nftool	Abre la herramienta neuronales guie, montaje de redes.
ntstool	Abre la herramienta neuronal series de tiempo de la red y te lleva a través de la solución de un problema de conexión mediante una red de alimentación directa de dos capas.

El programa de entrenamiento de la red neuronal emplea un algoritmo backpropagation que está siendo ejecutado en código .m: donde podemos evaluar la función de activación, el bias y los pesos sinápticos de la neurona.

Pero para un mejor enlace entre Matlab y Raspberry Pi se procede a entrenar la

neurona directamente en Simulink, ya que esta herramienta cuenta con la librería que habilita la cámara web de Raspberry.

Para el modelamiento de la red en Simulink se debe tener en cuenta el enlace de control que vamos a utilizar.

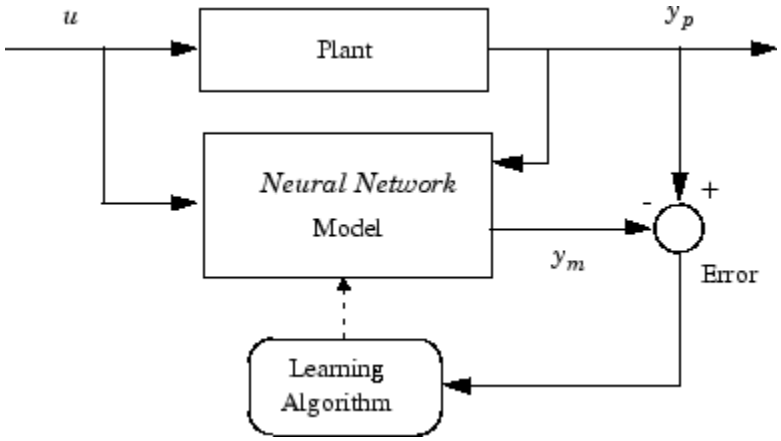


Figura 14. Modelo de Control de la Red neuronal

Al ser un algoritmo de backpropagation el modelo que se está implementado en el entrenamiento; este debe tomar en cuenta los valores anteriores de la red, para así poder predecir los valores futuros que obtendrá, al hacer la comparación entre los datos anteriores y los datos nuevos; es por eso que el modelo de la red queda de la siguiente manera:

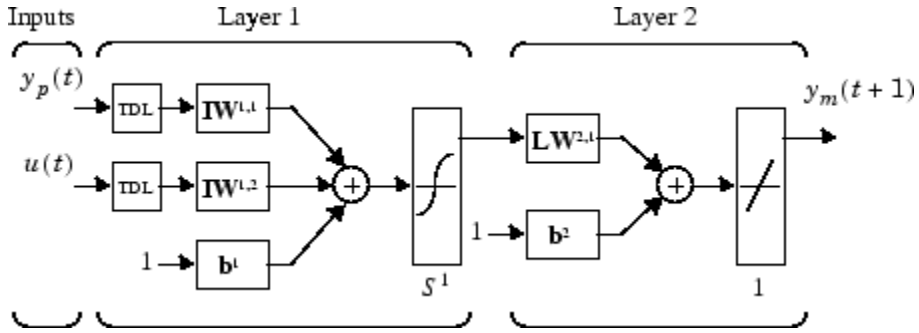


Figura 15 Entrenamiento de la Red Neuronal

Los pasos para la implementación del algoritmo de backpropagation en

Matlab son los siguientes:

- Recolectar Datos
- Crear la Red
- Configurar la Red (Entradas y salidas)
- Inicializar los pesos y el bias
- Capacidad de la red – conceptos de formación
- Validar la red
- Utilizar la red

En la parte de la recolección de datos, se debe tomar en cuenta en que se va a emplear la red neuronal; en nuestro caso es el reconocimientos de patrones de imágenes y voz, para lo cual en las imágenes los datos de entradas van hacer la captura de la cámara de Raspberry pi, la cual será convertida en matriz; el mismo procedimiento será para la voz la cual será un filtro el que nos identifique la matriz.

Luego que se ha creado la red neuronal y se ha configurado las entradas y las salidas, se procede a su entrenamiento para que sea solución al problema a resolver, según la definición de los datos de la muestra. Después de que la red se ha configurado, los parámetros de la red que son modificables son el peso sináptico y el bias, que serán los que optimicen el rendimientos y el tiempo de entrenamientos de la red; debemos saber que las matrices de salida serán fijas, mientras que las matrices de entrada dependerán de la imagen que nos envíe la cámara.

20. CAPITULO 4

21. IMPLEMENTACIÓN DEL ALGORITMO DE ENTRENAMIENTO Y ENLACE CON RASPBERRY PI

4.1 INTRODUCCIÓN

En este capítulo se explicará la forma en la que se debe realizar el algoritmo de entrenamiento, la función que va a desempeñar el Toolbox de entrenamiento, como obtener la matriz de la entrada, la configuración de los pesos de la neurona; además de la forma en la que se enlazara Raspberry con Matlab y Simulink, la parte importa de la comunicación Ethernet entre estas dos plataformas.

Una de los detalles más relevantes en el desarrollo e implementación de este proyecto es la parte de enlazar dos softwares completamente diferentes en software, y como el protocolo Ethernet nos brinda esa comunicación.

Para el proceso de la realización del algoritmo de entrenamiento se debe tomar en cuenta aplicación a la cual está destinada que es el reconocimiento de patrones de imágenes y voz, y la aplicación que va a tener después de su desarrollo completo del proyecto, el cual puede ser implementado en muchas áreas donde el reconocimiento de imágenes sea importante.

4.2 ALGORITMO DE BACKPROPAGATION

Backpropagation o retro propagación es la generalización de la regla de aprendizajes Widrow- Hoff a redes de múltiples capas y funciones de transferencia no lineales diferenciables. Entrada vector y los vectores de destino correspondientes se

utilizan para entrenar a una red hasta se puede aproximar una función, de entrada asociado vectores con salida específica vectores, o clasificar los vectores de entrada de una manera adecuada tal como se define por ti. Redes con sesgos, una capa sigmoide, y una capa de salida lineal son capaces de aproximar cualquier función con un número finito de discontinuidades. (Neural_Neural_Toolbox, 2015)

Las redes backpropagation adecuadamente entrenadas suelen dar respuestas razonables al problema para el cual están siendo entrenadas. Por lo general, una nueva entrada conduce a una salida similar a la salida correcta para los vectores de entrada utilizados en formación que son similar a la nueva entrada que esta siendo presentado.

El objetivo principal de este capítulo es explicar cómo utilizar las funciones de entrenamiento del algoritmo de backpropagation en el toolbox; los pasos que define el manual son los siguientes:

- Crear los datos de entrenamiento (Matriz X, T)
- Crear el objeto de la red.
- Entrenar a la red
- Simular la respuesta de la red a las nuevas entradas.

Para una mejor comprensión de los comandos de la creación del algoritmo, en el siguiente cuando se detallarán cada uno de ellos con su explicación.

Tabla 8. COMANDOS DEL ALGORITMO DE BACKPROPAGATION

COMANDOS	DEFINICIÓN
load house_dataset	Carga Los Valores De Los Vectores Predeterminados
houseInputs	Matriz de entrada contiene 506 vectores columna con 13 variables de raíces de propiedades diferentes.

newff	Crea Una Red De Dos Capas Con 20 Neuronas Ocultas
houseTargets	Matriz Que Carga Los Datos De Las Neuronas Ocultas

Para la realización del algoritmo de entrenamiento de este proyecto se va tomar en cuenta como referencia el manual de funcionamiento del Toolbox de Neural Network Start; en el cual se detalla todos los comandos y procedimientos del ingreso de los datos de entrada, selección de pesos sinápticos, bias de la neurona y función de activación.

En el script lo primero que debe realizar es definir una matriz X con 26 columnas, una por cada letra del alfabeto: cada columna tiene 35 valores que pueden ser 1 o 0; cada columna de 35 valores define un mapa de bits de 5x7. La matriz T es la matriz Identidad 26x26 que mapea el vector 26 de entrada de las clases.

En el caso de nosotros el rango de la matriz X va a cambiar, ya que va hacer la cámara web quien nos envié la imagen que debemos convertirla en matriz.

Como se explicara en el capítulo 5 de este proyecto, como es el proceso de creación del algoritmo y la parte de enlace con el Toolbox y luego en Simulink.

La siguiente fase en la parte de entrenamiento dentro del Toolbox es la parte del ingreso de los datos de entrenamientos que son:

Training (Entrenamiento).- éstos se presentan a la red durante el entrenamiento, y la red se ajusta en función de su error.

Validation (Validación).- estos se utilizan para medir la generalización de la red,

y para detener la formación cuando se detiene la mejora de generalización

Testing (Pruebas).- éstos no tienen efecto sobre la formación y así proporcionan una medida independiente de rendimiento de la red durante y después del entrenamiento.

Los valores que se despliegan serán por configuración los que estén implementados.

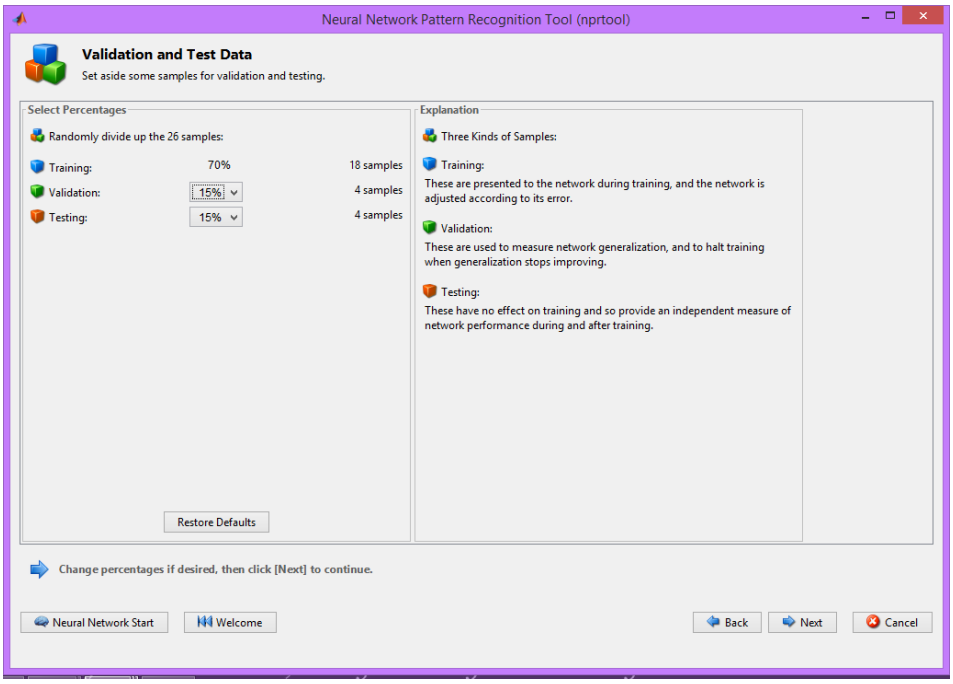


Figura 16. VALORES DE ENTRENAMIENTO DE LA RED

4.3 DIAGRAMA EN SIMULINK

4.3.1 INTRODUCCIÓN

Luego de un entrenamiento de la red neuronal como se detalla en el capítulo 4 sección 4.2; mediante el algoritmo de backpropagation y la utilización del Toolbox para una comprobación, en este parte se explicara la utilización de Simulink en el enlace con el medio externo y la comunicación con Raspberry.

Simulink es una aplicación que permite construir y simular modelos de sistemas físicos y sistemas de control mediante diagramas de bloques. El comportamiento de dichos sistemas se define mediante funciones de transferencia, operaciones matemáticas, elementos de Matlab y señales predefinidas de todo tipo. Simulink dispone de una serie de utilidades que facilitan la visualización, análisis y guardado de los resultados de simulación. Simulink se emplea profusamente en ingeniería de control. (Javier Aracil y Fabio Gómez–Estern, 2007).

El razón por la cual el entrenamiento de la red no se hace directamente en Simulink es porque se debe tomar en cuenta que existen otros algoritmos de entrenamiento de redes neuronales.

4.3.2 LIBRERÍA DE NEURAL NEURAL START EN SIMULINK

Para una poder entender cómo se elabora el diagrama de bloques de la red neuronal en Simulink debemos tener presente todas las funciones que pueden utilizar, por eso se detallara todas las funciones que existen en la librería, se debe tomar en cuenta que en la elaboración de este proyecto todos las funciones que se van a utilizar son las que estén de acorde al algoritmo de backpropagation.

La librería de Simulink está dividida de la siguiente forma:

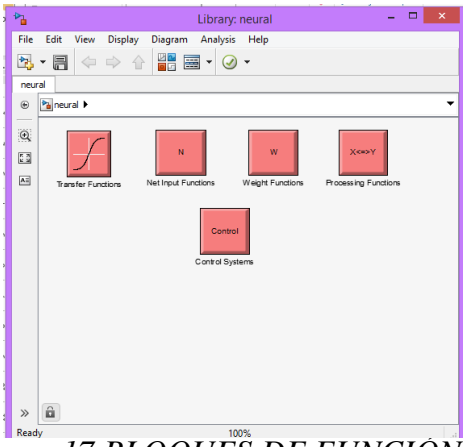


Figura 17 BLOQUES DE FUNCIÓN DE TRANSFERENCIA

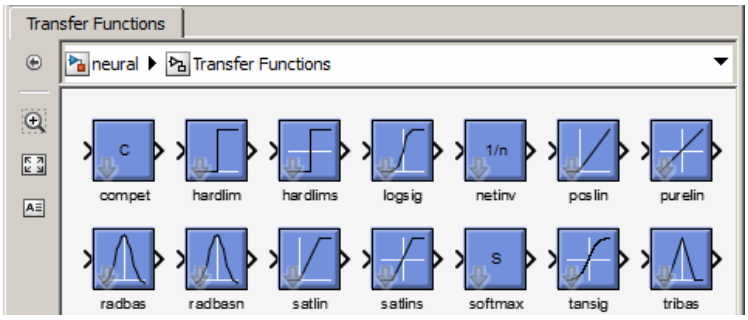


Figura 18 BLOQUES DE ENTRADA NETOS.

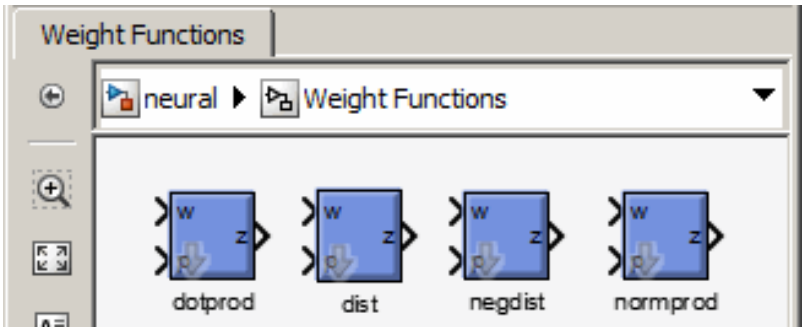


Figura 20 BLOQUES DE PESO.

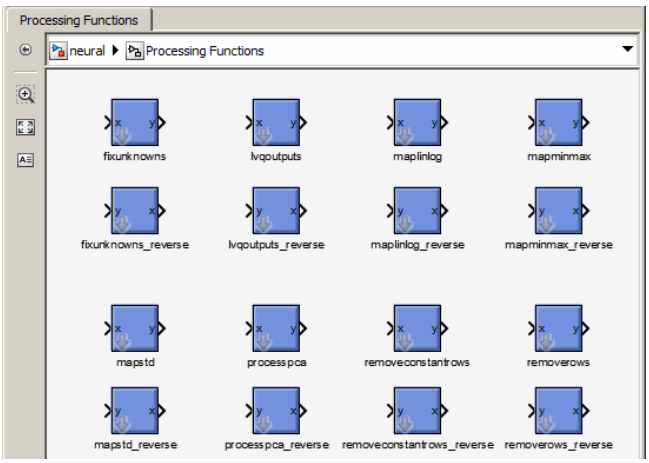


Figura 19 BLOQUES DE CONTROL.

4.3.2 CONFIGURACIÓN DE LA RED NEURONAL EN SIMULINK

Al conocer la forma en la que se elabora los diagramas en Simulink y al ya contar con una estructura de la red ya creado lo que debemos hacer es configurar nuestros parámetros para poder enlazar.

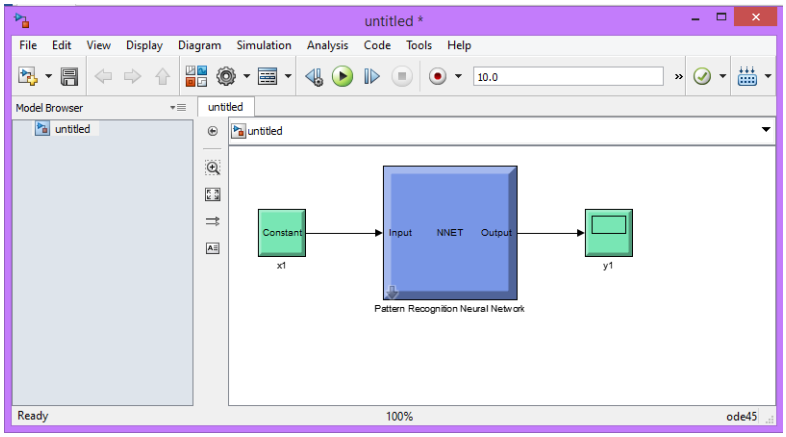


Figura 21 RED NEURONAL EN SIMULINK

4.4 PROTOCOLO DE COMUNICACIÓN ETHERNET ENTRE RASPBERRY PI Y SIMULINK

4.4.1 INTRODUCCIÓN

En el mundo actual estamos inversos en el mundo de las comunicaciones y de los desarrollos de ellas; en esta parte se va explicar como uno de los protocolos de comunicación más utilizado en el mundo el TCP/IP se aplica en el desarrollo de este proyecto.

4.4.2 RASPBERRY PI EN MATLAB

Para poder tener un comunicación entre estas dos plataformas en necesario descargar la Liberia que nos permita manejar la Raspberry pi, como ya e indico antes el tipo de comunicación que se va a tener entre Matlab y Raspberry es vía Ethernet.

Es necesario aclarar que la librería *raspberrypilib*, esta disponle desde la versión 2015 de Matlab.

El procedimiento para instalar la librería, se detalla en la manual de usuario e instalación del módulo que se encuentra como anexo.

22. CAPITULO 5

23. PRACTICAS DE LABORATORIO DE IMPLEMENTACIÓN DEL MÓDULO DE ENTRENAMIENO DE REDES NEURONALES CON BACKPROPAGATION

5.1 INTRODUCCIÓN

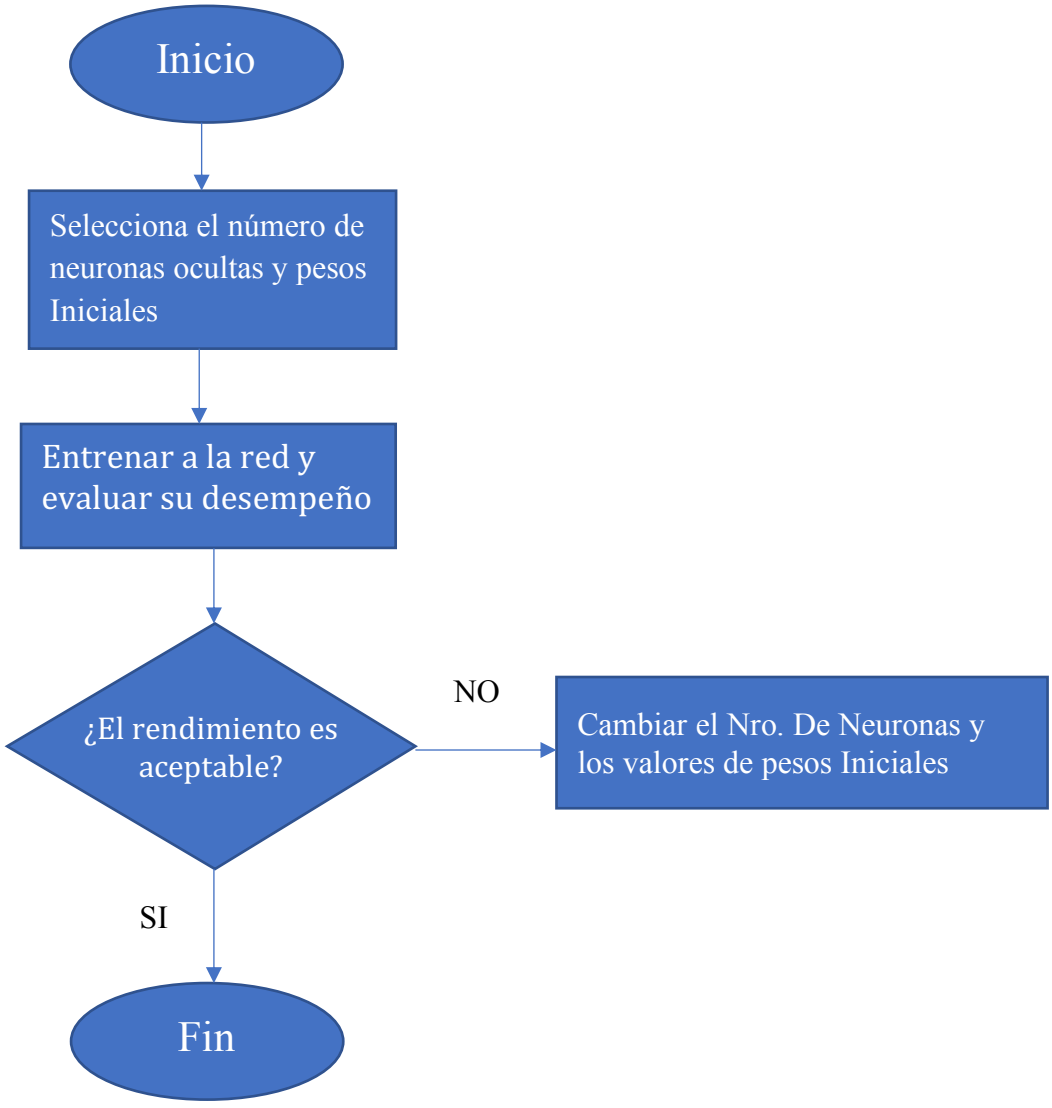
En este capítulo se describen algunas prácticas de ejemplo para las personas que van a utilizar el proyecto aquí descrito, iniciando con una breve introducción a las utilidades que ofrece el módulo, hasta la realización de varias prácticas para el entrenamiento de redes neuronales con los diferentes algoritmos descritos en este proyecto. En el Anexo 2 se proporciona una guía de prácticas para el refuerzo del estudiante en la realización de las mismas.

También se procederá a explicar el proceso de diseño e implementación de algoritmo de backpropagation con Raspberry, que se va a utilizar para la realización de cada práctica, describiendo las características del módulo a implementar y entrenar a la red para diferentes aplicaciones.

5.2 PRACTICA 1. RECONOCIMIENTOS DE IMÁGENES

En esta práctica que ya lo que se pide hacer es un algoritmo de backpropagation, que sea capaz de reconocer caracteres de letras de las dimensiones 7x6 cm, las imágenes serán obtenidas por la cámara web que debe estar conectada a la Raspberry pi; el entrenamiento de la neurona debe hacer en código m, para poder obtener las matrices de todas las letras del alfabeto.

A continuación tenemos el algoritmo de backpropagation con las matrices que nos dejan ver como se entrena nuestra neurona para el reconocimiento de patrones de imágenes, explicado en el diagrama siguiente:



Este es el algoritmo que nos garantiza que nuestras matrices están creadas y las vamos a poder utilizar en el Toolbox, las matrices de vamos a obtener están representados en mapa de bits de la siguiente manera.

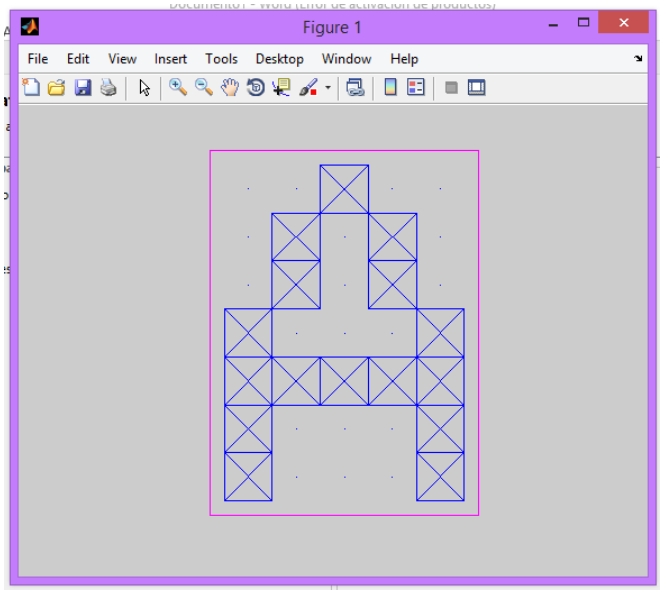


Figura 22 MATRIZ DE LA LETRA A

Cuando se empieza el entrenamiento de la neurona con el mapa de bits de la letra que ya tenemos se deben desplegar las siguientes ventanas:

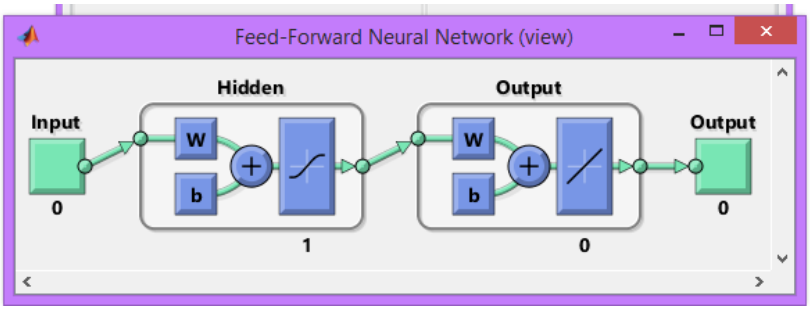


Figura 23 ECUACIÓN DE LA RED NEURONAL

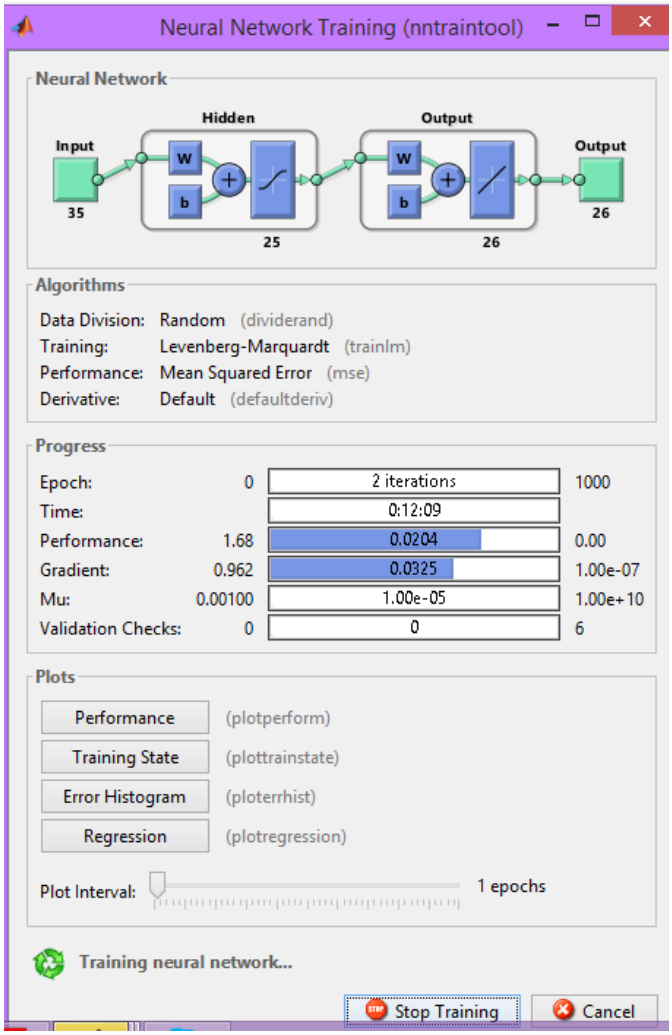


Figura 24 COMIENZO DEL ENTRENAMIENTO DE LA RED

Al finalizar el proceso de entrenamiento con la validación, el tiempo que se demora en entrenar a la red y el mapa de bits de la imagen final las ventanas queda así:

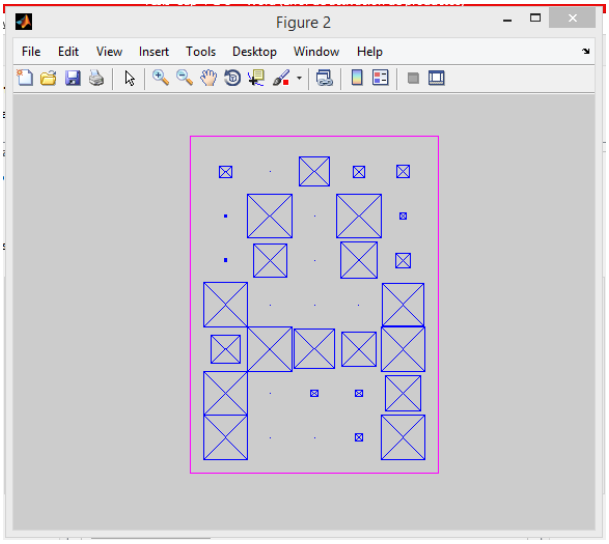


Figura 25 MAPA DE BITS DE LA RED ENTRENADA

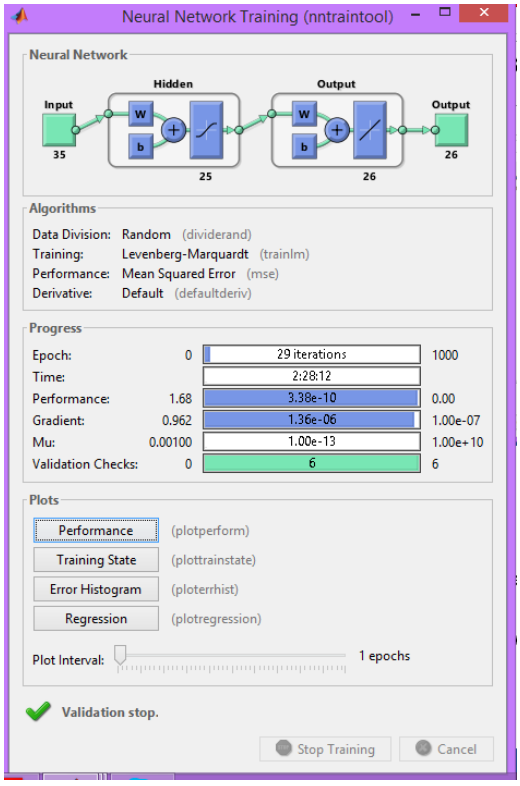


Figura 26 VALORES FINALES DEL ENTRENAMIENTO DE LA RED

Para podemos obtener un mejor análisis del entrenamiento de la red neuronal y saber que valores se deben cambiar para un mejor entrenamiento y validación las gráficas que se obtienen son las siguientes:

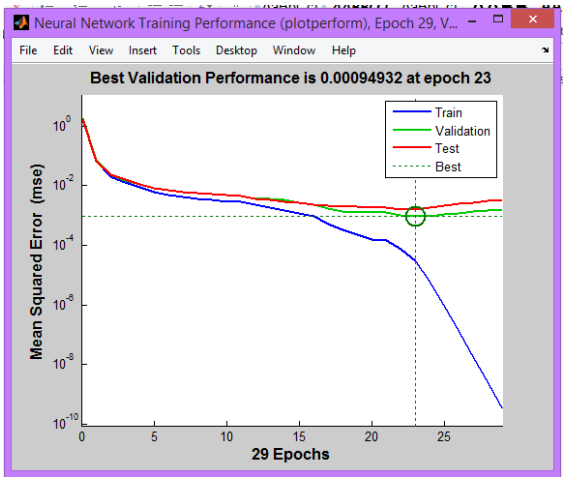


Figura 27 PERFOMANCE DIAGRAM

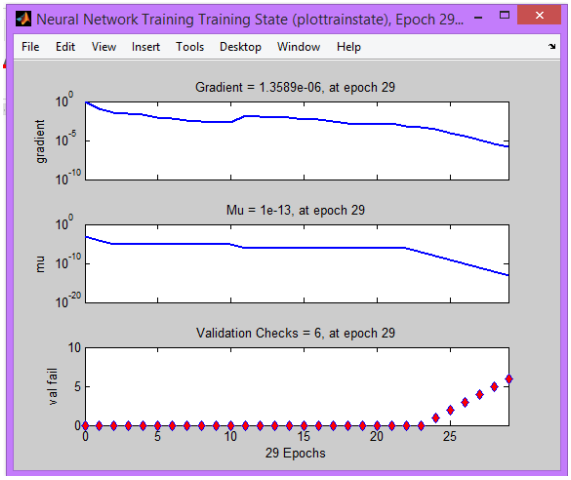


Figura 28. NEURAL NETWORK TRAINING

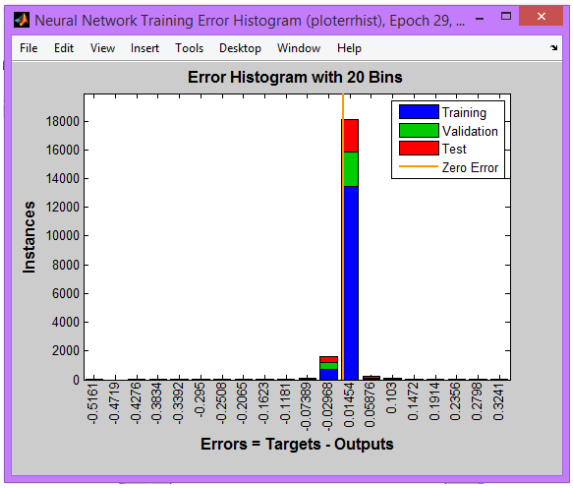


Figura 29 NEURAL NETWORK TRAINING ERROR HISTOGRAM

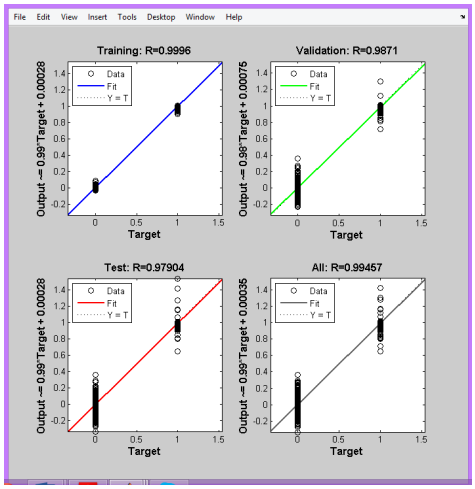


Figura 30 NEURAL NETWORK TRAINING REGRESSION

Las 4 graficas que se observan sirven de análisis de la red neuronal, los comandos que se utilizan en el código.

Luego de diseñado el algoritmo de entrenamiento procedemos a ingresar al Toolbox, para eso en la pantalla principal de Matlab ingresamos el comando *nnstart* y nos aparece la siguiente pantalla:

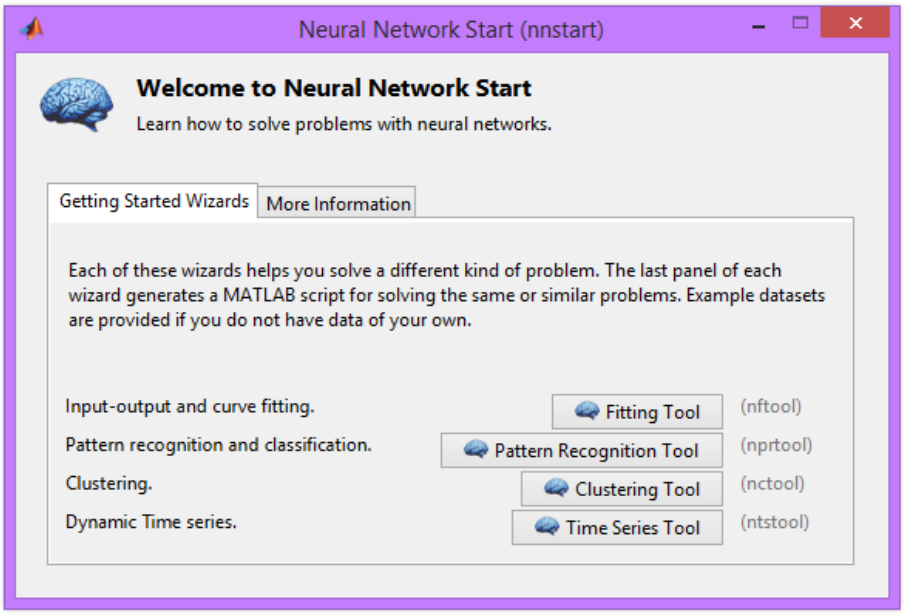


Figura 31 PANTALLA DEL TOOLBOOX

Al visualizar la explicación del tipo del algoritmo con el cual se entrena y ver el esquema hacemos click en next, la pantalla que se despliega es la de configuración de las entradas y salidas de la neurona, para la cual debemos tener abierto el algoritmo que se menciona antes porque las matrices que se van a ingresar son X y T.

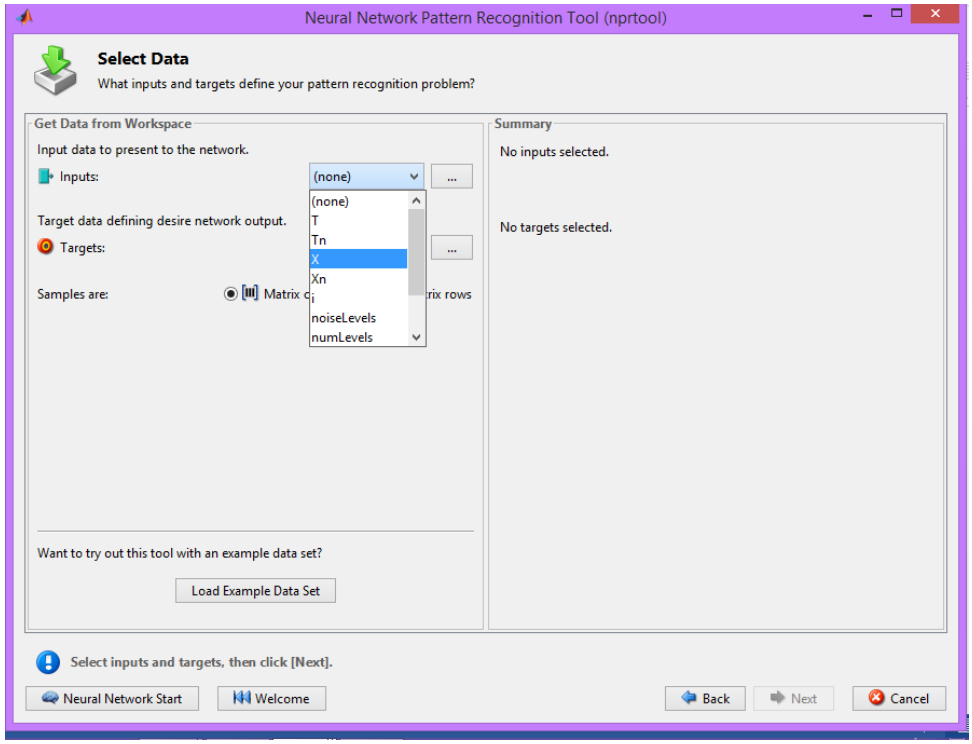


Figura 32 SELECCIÓN MATRIZ X (ENTRADA)

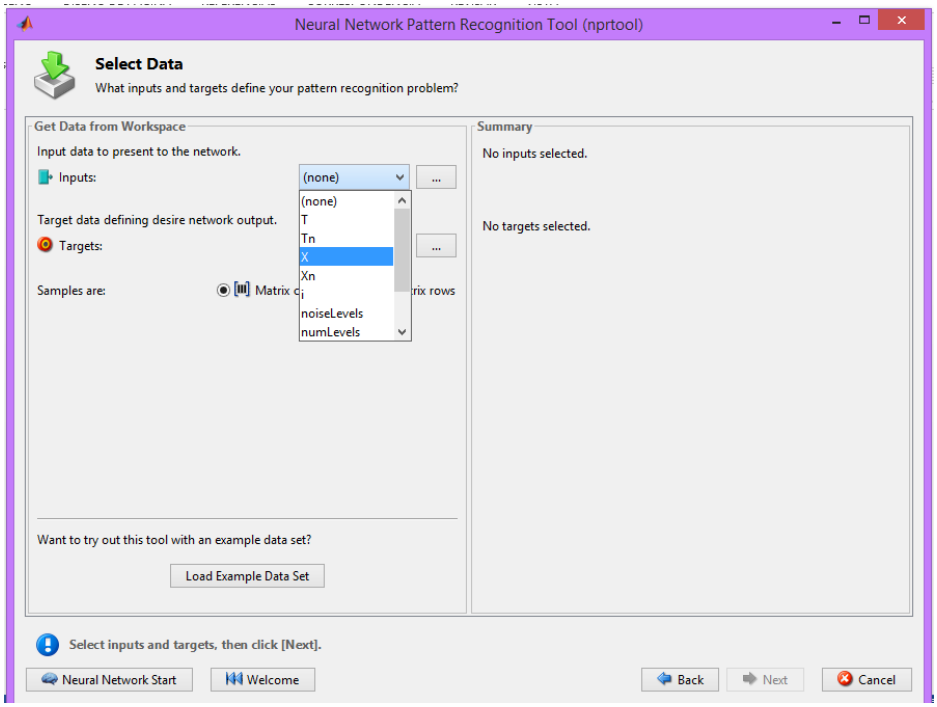


Figura 33 SELECCIÓN DE LA MATRIZ T (SALIDA)

Una de las ventajas de utilizar el Toolbox sobre el algoritmo es que este nos facilita las gráficas de análisis en una sola ventana.

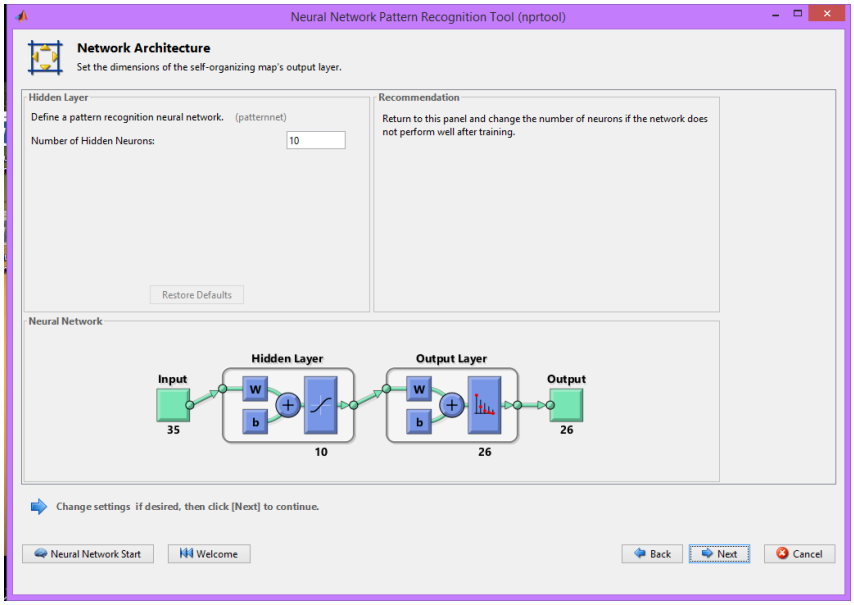


Figura 34 INGRESO DE LAS CAPAS OCULTAS

En la siguiente imagen ya se configura cuantas capas ocultas va a tener nuestra red.

El entrenamiento de la red empieza de la siguiente manera:

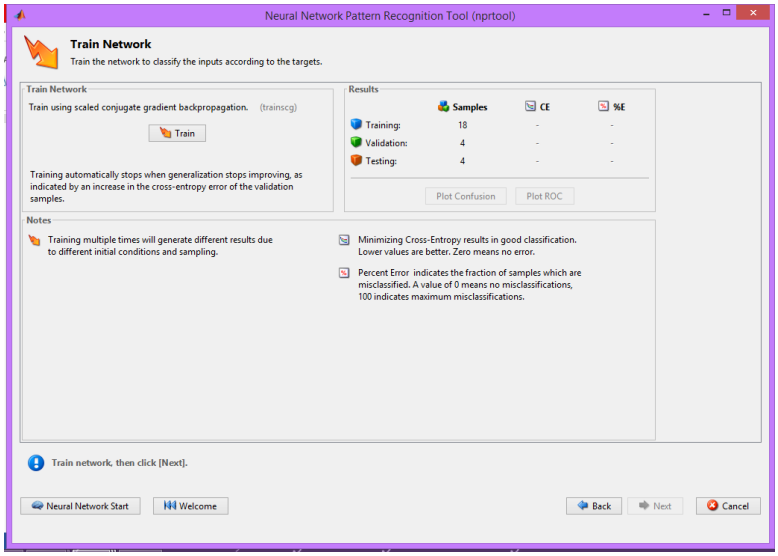


Figura 35 . INICIO DEL ENTRENAMIENTO

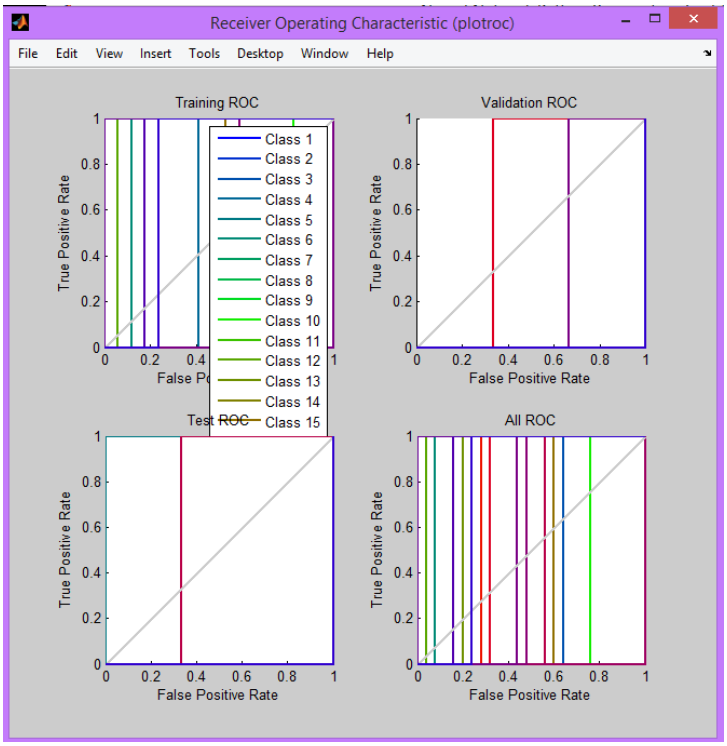


Figura 36 GRAFICAS DE ANALISIS LUEGO DEL ENTRENAMIENTO

Para finalizar ya con la parte del entrenamiento de la red neuronal y pasar a la parte de configuración en Simulink la última ventana que nos debe salir es la siguiente

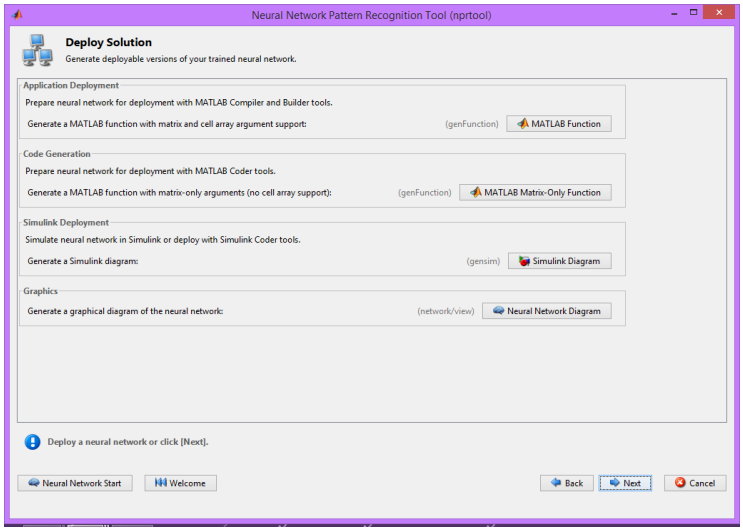


Figura 37 VENTANA FINAL DEL TOOLBOX

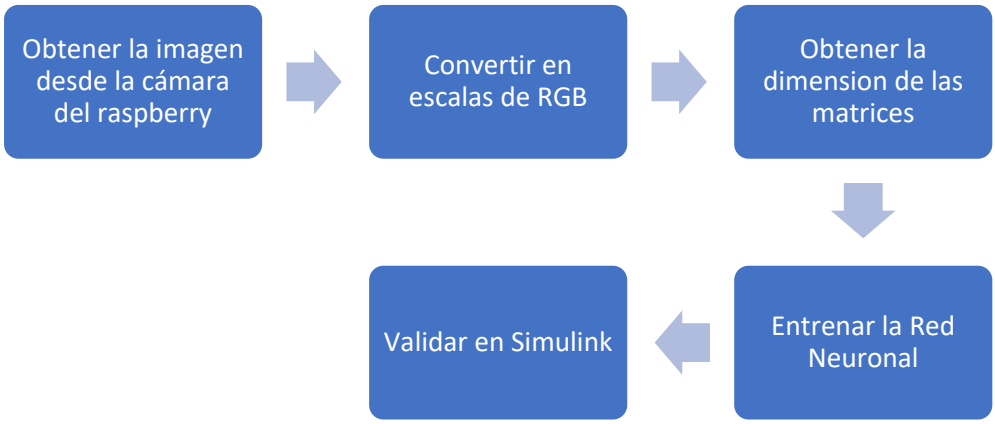
Con esto concluye la fase de entrenamiento de la red neuronal para el reconocimiento de patrones de imágenes, utilizando el Toolbox de Matlab para la red Neuronal, con un pequeño código para establecer las dimensiones de la matriz.

En el manual de Funcionamiento y Practicas se encuentra desarrolladas dos diferentes practicas con la obtención de una imagen desde la cámara del Raspberry Pi.

5.2.1 RECONOCIMIENTO Y ENLACE SIMULINK Y RASPBERRY PI

Al completar la fase de entrenamiento y funcionalidad de la red ya entrenado proceso a nuestra prueba de funcionamiento en Simulink, aquí es donde la imagen que obtenemos de la cámara va hacer analizada de acuerdo a nuestro entrenamiento.

El procedimiento para la obtención de nuestra imagen es binarizar la imagen y transformarle a escalas de grises con el siguiente diagrama de bloques:



Es importante recordar que todo el proceso de la obtención de la imagen y su conversión a escala de grises esta detalla en el anexo llamado manual de funcionamiento y prácticas.

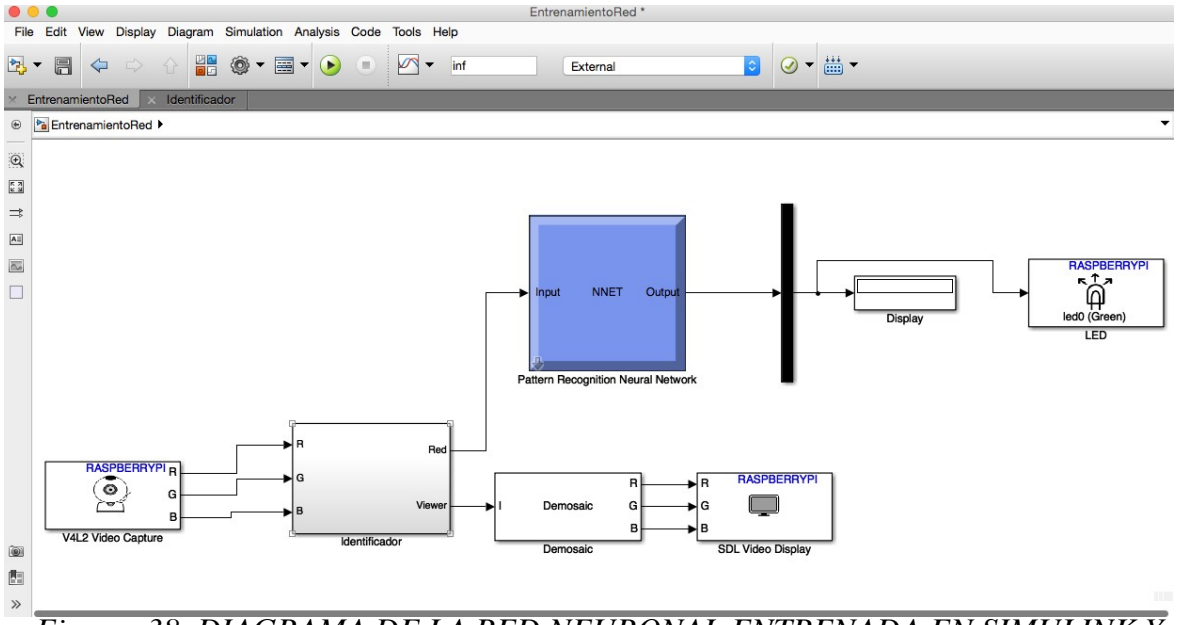


Figura 38. DIAGRAMA DE LA RED NEURONAL ENTRENADA EN SIMULINK Y CON ENLACE A RASPBERRY PI.

5.2 PRACTICA 2. RECONOCIMIENTO DE VOZ

La verificación o identificación de personas en un canal telefónico, empleando su patrón de voz, es la base para la realización de esta práctica en el cual se propone una aplicación diferente del algoritmo de backpropagation que emplea las características del patrón de voz y Redes Neuronales Artificiales.

En esta práctica ya no tenemos una matriz establecida por una imagen sino que tenemos un archivo *wav que es de sonido, por su versatilidad de manejo con el software Matlab, cada uno de los 25 archivos tienen las característica.

El objetivo de esta etapa es acondicionar la señal de entrada para que esta pueda ser procesada por la RNA, primero acotamos la señal de voz eliminando la parte inicial y final de la misma, que solo representan ruido para obtener la señal de voz a la cual le aplicaremos las Wavelets, se toma la subseñal a[n] correspondiente a las bajas frecuencias de la señal de voz donde se localiza la mayor cantidad de energía de la misma, despreciándose la subseñal b[n] que corresponde a las altas frecuencias ya que es donde se encuentra la mayor cantidad de ruido de la señal (ruido ambiental y el ruido del canal telefónico). Obteniendo así una señal de voz compacta y filtrada con respecto a la original. Posteriormente se normaliza la señal de voz resultante, para finalmente extraer los coeficientes LPC de la señal, que servirán para el diseño de los patrones de entrenamiento de la RNA. (Luis A. Cruz-Beltrán, Marco A. Acevedo- Mosqueda, 2008).

Las ecuaciones y el procedimiento de entrenamiento es similar al de imágenes, las ecuaciones no cambian ya que el algoritmo de entrenamiento sigue siendo el de backpropagation, el Toolbox, el diagrama en Simulink y el enlace con Raspberry pi se mantiene igual.

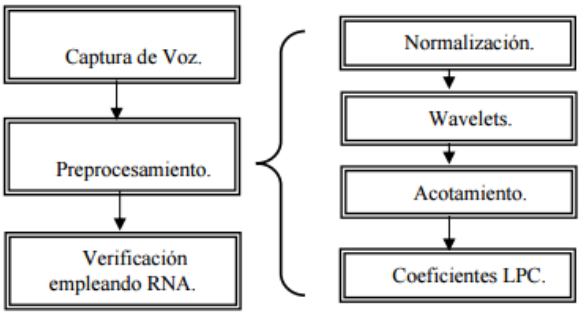


Figura 39 SISTEMA PROPUESTO PARA EL ALGORITMO DE RECONOCIMIENTO *DE VOZ*

EL procedimiento de entrenamiento del patrón de voz en el toolkit es simular al de imagen, lo que cambia es la forma en que obtenemos la señal, esta debemos grabar y vectorizar para que obtener las matrices de la voz, palabra o

sonido a reconocer.

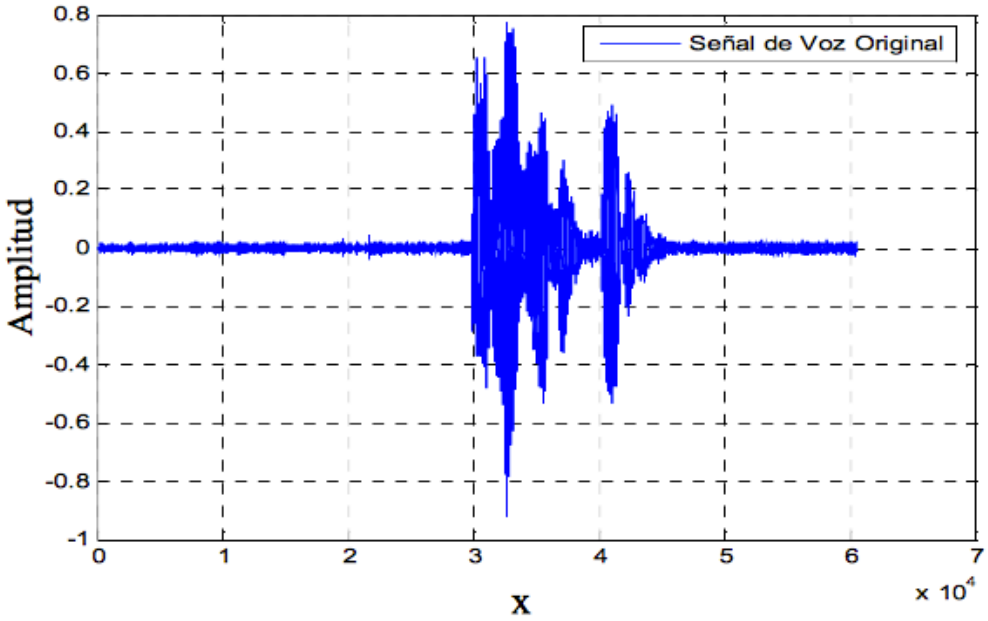


Figura 40. SEÑAL DE LA VOZ GRABADA

Para poder obtener un señal nítida y sin ruido es necesario filtrar la señal, el proceso se realizó 5 veces en la grabación del sonido a identificar, obteniendo así 5 archivos de audio que fueron convertidos en formato *.Wav por su versatilidad de manejo con el software Matlab.

Cabe resaltar que se empleó una velocidad de muestreo de sonido de 11 Khz. Con la finalidad de cumplir con el criterio de Nyquist que es mayor ó igual a 2 veces la frecuencia de muestreo, que para nuestro caso pertenece al canal telefónico que es aproximadamente 4 Khz.

24. CAPÍTULO 6

6.1 CONCLUSIONES

- En el desarrollo de este proyecto para el entrenamiento de las redes neuronales se debió considerar, todos los metodos matemáticos con los que cuenta el entrenamiento de RNA, en la elección del método de adecuado del entrenamiento de la red neuronal se analizó cual es el método más eficiente, con cual se obtiene más posibilidades de cambiar de un algoritmo de backpropagation por otro tipo de modelo matematico.
- Al disponer de un software como Matlab la posibilidades de diseño y ejecución son infinitas, pero se penso en otro elemento como lo es Raspberry PI, para la ejecución de la red neuronal, la concexión entre raspberry y Simulink de matlab se logro exitosamente, en la parte de reconocimento de imágenes.
- El trabajo muestra un aceptable comportamiento de la red neuronal de backpropagation frente a la tarea de reconocimiento de un patrón tan estocástico como lo es la voz. Aunque su tasa de acierto dista aún mucho de las obtenidas mediante otros métodos, existen muchas variables en la red neuronal que pueden ser ajustadas para lograr mejores comportamientos de la misma; entre ellas, se pueden considerar:
    - Número de neuronas de la red.
    - Dimensión de los patrones y de dichas neuronas.
    - Cantidad de locutores y de archivos sonoros utilizados en el entrenamiento.
    - Método de codificación o mapeo elegido para la aplicación, uso adecuado del filtro para la decoficiación.
    - Valores de los coeficientes de aprendizaje y sintonización, y forma de decaimiento en el tiempo.

- Al diseñar las 3 practicas modelos de entrenamiento de una red neuronal, se diseña diferentes metolodogías de identificación y entrenar la RNA, asi como la facilidad de poder implementar diferentes modelos dentro de una misma RNS, para pruebas posteriores.

6.2 RECOMENDACIONES

- El módulo debe ser probado por los estudiantes de la carrera de Ingeniería en Mecatrónica en la asignatura de diseño mecatrónico II, para poder ver la funcionalidad y la escalabidad de este módulo, además de la implementación de otras prácticas dentro de la materia y además de ver la funcionalidad de la implementación de un sistema nuevo en las aulas.
- Se debe rescatar que la comunicación que poseemos, nos brinda una amplia posibilidad de control remoto a nuestra placa, podemos contar con un gran número de aplicaciones
- Actualmente en el estudio de RNA, se manejan el término Machine Learning, por lo que recomiendo, emplear de mejor forma el uso de Raspberry PI, para el entrenamiento de una RNA directamente inicializa en la placa.
- La implementación de RNA en el uso de proyecto de clasificación y de robótica, abre la mente de otro tipo de control aplicado a la industria, se recomiendo incentivar el estudio dentro de las estudiantes de Mecatrónica.
- En cuanto a la seguridad, el cual no es el tema principal a tratar en este trabajo, hay que tener cuidado con algunos aspectos en cuanto al uso de la Raspberry Pi, puesto que no es un dispositivo pensado para ser seguro, y es bastante susceptible a sufrir un ataque, principalmente debido a que el sistema operativo Raspbian tiene una contraseña por defecto que siempre es la misma y debido al servidor SSH que se emplea para trabajar desde el ordenador, por lo que es importante tener unos conocimientos básicos de Linux para configurar adecuandamenta la lista blanca y la lista negra.

25. BIBLIOGRAFÍA

Olabe, X. B. (2008). Redes Neuronales Artificiales y sus Aplicaciones. Obtenido de http://cvb.ehu.es/open_course_ware/castellano/tecnicas/redes_neuro/Course_listing.html

Pi, R. (2011). Foundation Raspberry Pi. Obtenido de Foundation Raspberry Pi: https://www.raspberrypi.org/help/what-is-a-raspberry-pi/

Mathwork. (2015). Mathworks. Obtenido de www.mathwork.com

Neural_Neural_Toolbox. (2015). Obtenido de www.mathworks.com William, S. (2000). Comunicaciones y Redes de Computadoras. Printice Hall.

Javier Aracil y Fabio Gómez–Estern. (2007). Introducción a Matlab y Simukink. Obtenido de http://www.esi2.us.es/~fabio/apuntes_matlab.pdf

Luis A. Cruz-Beltrán, Marco A. Acevedo-Mosqueda. (2008). Reconocimiento de Voz usando Redes Neuronales Artificiales. Obtenido de http://campusv.uaem.mx/cicos//cicos2008/imagenes/memorias/6tocicos2008/Articulos/Ponen cia%207.pdf

José R Hilera Martínez. (2000). “Redes Neuronales Artificiales, Fundamentos, Modelos y.

Mexico: Alfa Omega. Simon Haykin. (1999). “Neural Networks”,. New Jersey: Prentice -Hall.

Sadaoki Furui. (2001). “Digital Speech Processing, Synthesis, and Recognition". Cambridge: Press.

Fausett., L. (2005). “Fundamentals Neuronal Network, architectures, algorithms, and. New Jersey: Prentice.

Oropeza, J. L. (2006). “Algorithms and Methods for the Automatic Speech Recognition in.

Barragán, D. (5 de Agosto de 2008). Manual de Interfaz Grafica de Usuario con Matlab. Obtenido de http://www.matpic.com

William, S. (2008). Comunicaciones y Redes de Computadoras. New Jersey: Prentice Hall. Mathworks. (s.f.). Mathworks. Obtenido de www.matheorks.com

Matich, D. J. (marzo de 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) .

Universidad Politécnica de Madrid, E. (s.f.). Universidad Politécnica de Madrid, Españ. (T. d. Neuronales., Productor) Obtenido de www.gc.ssr.upm.es/inves/neuronal/ann2/anntuto.htm)

J.Rieta, F. Castells, C. Sanchez, V.Zarzoso, J.Millet. (2007). Atrial activity extraction for atrial fibrillation analysis usgin blind sorce separation. (IEEE, Ed.) (8), 1530-1533.

F.Castells, P. Laguna, L.Sornmo, A.Bollman, J.M. Roig,. (2007). Principal component analysis in ecg signal processing,. EURASIP journal on Advances in Signal Processing , 21.

C. Vasquez, A. Hernandez, F. Mora, G. Carreault, G. Passariello. (2001). Atrial activity enhancement by wiener filtering using an artifical neural network. IEEE Biomedical Engineering , 940-944.

B. Simon, L.Sornmo, P. Laguna. (2005). Mejora en el alinamiento de registros usando interpolacion. XIV Congreso Anual de la Sociedad Española de Ingenieria Biomedica , 49-51.

M.T.M. Zi-qin, J.L. Schiano. (2000). LMS Learning algorithms: Misconceptions and new Results on convergence. IEEE Neural Networks , 47-56.

Novak, D. (2000). Processing og ECG Signals wavelets. Praga: Czech Technical University.

Bonifacio Martín del Rio, Alfredo Sanz Molina. (2001). "Redes Neuronales y Sistemas Borrosos". Madrid: Ra-Ma.

K. Berkner, R. O.Wels. (1998). Wavelet Transforms and denosoing algorithms. En R. O. K. Berkner. Signals, Systems and Computers.

Aapo Hyyarinen, Erkki Oja. (2000). Independen componen analysis. Algorithms and applications neural network. 411-430. Finlandia.

Claverl, O. (2007). Modelado y simulacion de un Sustema de Detección de

Intrusos utilizando redes Neuronales recurrente. Departamento de Computacion, Electronica y Mecatronica.

Hugo Vega Huerta, Augusto Cortez Vásquez, Ana María Huayna, Luis Alarcón Loayza, Pablo Romero Naupari. (Julio de 2009). Facultad de Ingeniería de Sistemas e Informática Universidad Nacional Mayor de San Marcos,. Recuperado el 25 de 03 de 2015, de Facultad de Ingeniería de Sistemas e Informática Universidad Nacional Mayor de San Marcos,: http://docplayer.es/5579958-Reconocimiento-de-patrones-mediante-redes-neuronales- artificiales.html

Kasabov, N. K. (1998). Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering. Press Cambridge, MassachusettsLondon, London: England Page iv Second printing.

Dunnc, R. A. (2007). A Statistical Approach to Neural Netqwoks for Patter Recognition (Vol. 1). (A. W. Sous, Ed.) North Ryde, New Nu¡out Walks, Australia: Wiley Intersciencie.

ben Krose, Patrick Van ser Smagt. An instroduction Neural Networks (Vol. Octava Edicion). Germany, 1996.

Veelenturf, L. (2008). Analysis and applications of artificial neural networks. (F. Ital, Ed.)

Olabe, X. B. (2007). Curso del Redes Neuronales Artificiales y sus Aplicaciones d. (X. B. Olabe, Productor) Recuperado el 24 de Mayo de 2015, de Escuela Superior de Ingeniería de Bilbao: http://www.ciberesquina.una.edu.ve:8080/2014_2/350_E.pdf

Luis. A. Cruz-Beltrán1 and Marco. A. Acevedo-Mosqueda. (Marzo de 2016).

Reconocimiento de Voz usando Redes Neuronales Artificiales Backpropagation y Coeficientes LPC. Recuperado el 1 de Agosto de 2016, de Repositorio Digital UEAM Mexico:http://campusv.uaem.mx/cicos/cicos2009/imagenes/memorias/6tocicos2008/Articulos/Ponenc ia%207.pdf

Carlos Alberto Ruiz, Marta Susana Basualdo. (1 de Marzo de 2001). Repositorio Digital Universidad Tecnologica Nacional. Recuperado el 30 de Enero de 2015, de Facultad Regional de Rosario: http://docplayer.es/8032554-Catedra-informatica-aplicada-a-la-ingenieria-de- procesos-orientacion-i-tema-redes-neuronales-conceptos-basicos-y-aplicaciones.html

Lopez, B. (12 de Septiembre de 2003). Instituto Tecnologico de Nuevo Loredo. Recuperado el 3 de Agostode 2015, de Repositorio digital: http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_ alumnos/RNA/Redes%20Neuronales.pdf

ANEXOS

MANUAL DE INSTALACIÓN , PRÁCTICAS Y FUNCIONAMIENTO DEL MÓDULO DE ENTRENAMIENTO DE REDES NEURONALES

1. INTRODUCCIÓN

El objetivo primordial de este manual es dar a los estudiante una guía de como entrenar una red neuronal de diferentes mecanismos utilizando el mismo principio, de backpropagation, empezar por una red entrenada por código, ara luego utilizar el Toolbox de Neural Network Start, y realizar el enlace entre Raspberry pi y Simulink.

El entrenamiento de la red neuronal, cuenta de dos formas de realizar, la primera es mediante código m, y la otra forma en utilizando el Toolbox de Matlab, para el reconocimiento de imágenes y voz.

En lo que respecta a la parte de comunicación con el medio externo, el dispositivo que vamos a utilizar es Raspberry pi, por las bondades tecnológicas que nos brinda, el protocolo de comunicación que vamos a utilizar es el TCP/IP Ethernet, que nos permite tener una conexión remota hacia nuestro proyecto.

26.

2. INSTALACION DE MATLAB

Lo que primero debemos hacer es tener instalado el software elegido para el entrenamiento que es Matlab 2016b, se ha elegido esta versión porque desde esta versión ya existe la librería de Raspberry pi, que se detallara a continuación como se instala.

En la ventana principal de Matlab que se muestra en la ilustración 1. Debemos hacer clic donde dice *new;* luego en *Script,* y se nos abre la nueva ventana que se muestra en la ilustración 2; aquí empezamos a ingresar la configuración del algoritmo de entrenamiento.

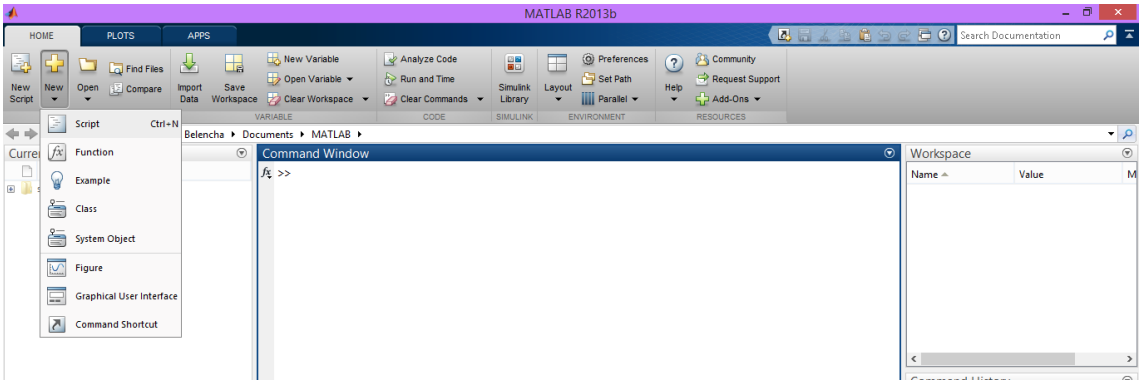


Ilustración 1. Pantalla de Inicio de Matlab

En el ingreso de los datos al *Script,* se debe considerar el ingreso de las matrices de las letras, para tener un diagrama ya elaborado de dichas matrices vamos a utilizar el comando *prprob,* que es el que inicializa las matrices de ingreso.

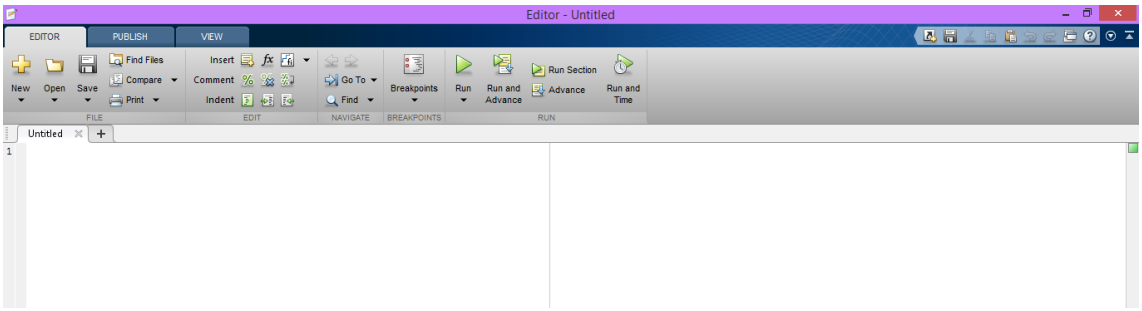


Ilustración 2. Pantalla Script

3. Instalación de la Librería de Raspberry Pi en Matlab

Para el proceso de instalación de la librería de Raspberry, debemos irnos a la pantalla principal de Matlab en la parte superior derecha.

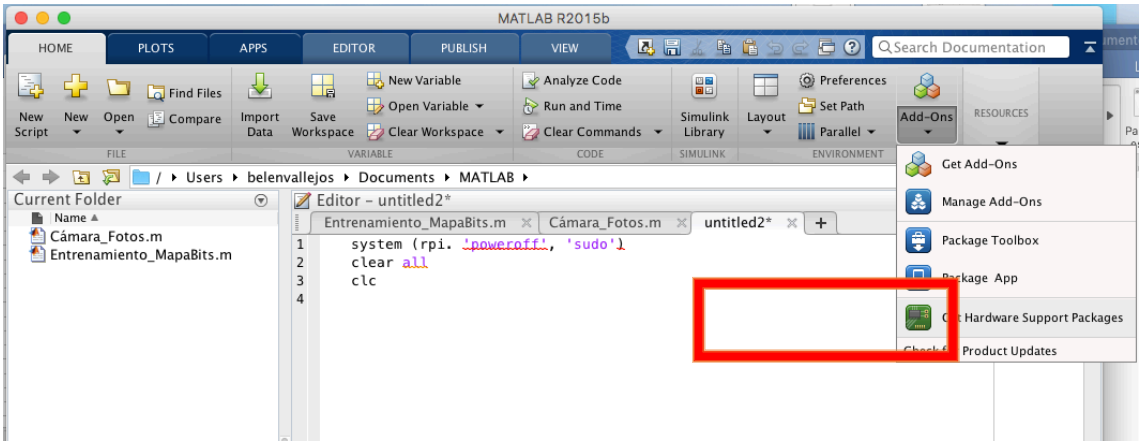


Ilustración 3. Instalación de la Librería.

Luego de hacer clic en donde dice ***Get Hardware Support Packages,*** la ventana que se nos tiene que abrir es la siguiente.

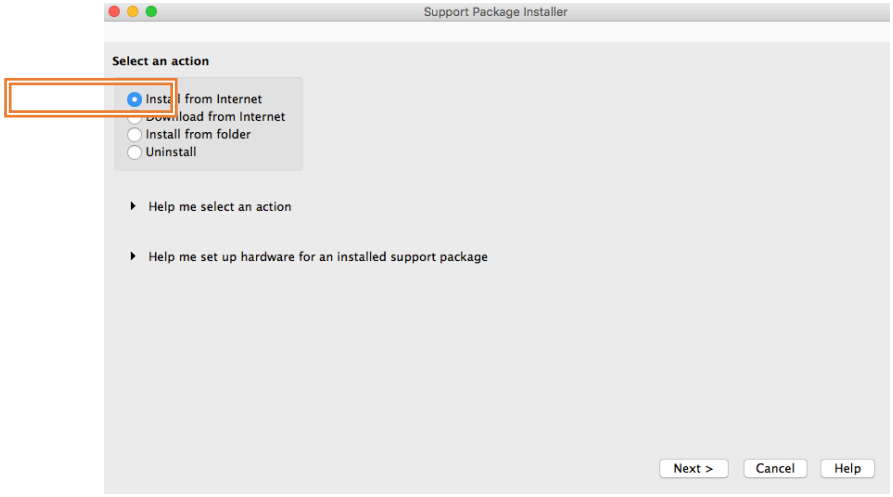


Ilustración 4. Ventana de Instalación de la Librería

Vamos a descargar el software de desde internet para lo que hacemos clic en *next,* y la pantalla es la ilustración 5, donde seleccionamos Raspberry pi, y hacemos clic en *next.*

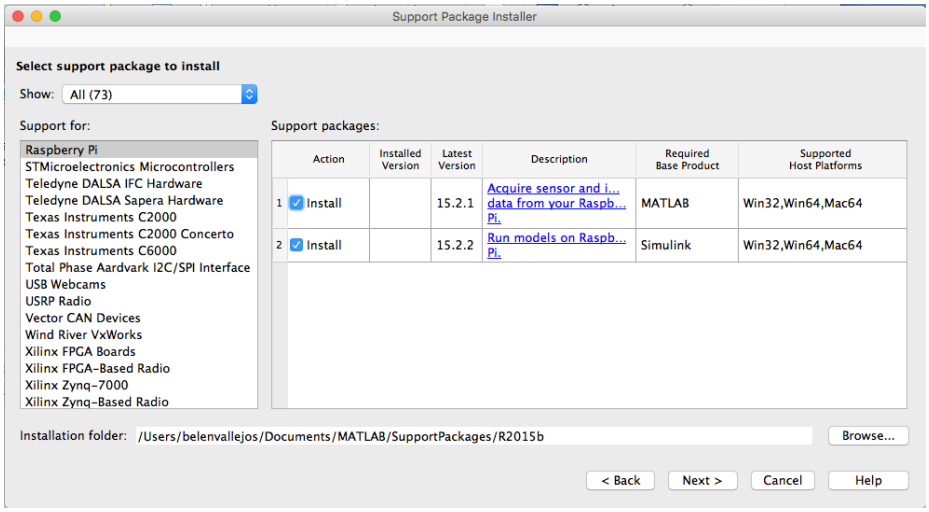


Ilustración 5. Elección del paquete a instalar

Luego aceptamos los términos en condiciones hasta que la ventana de instalación nos quede así, donde hacemos clic en *Install.*

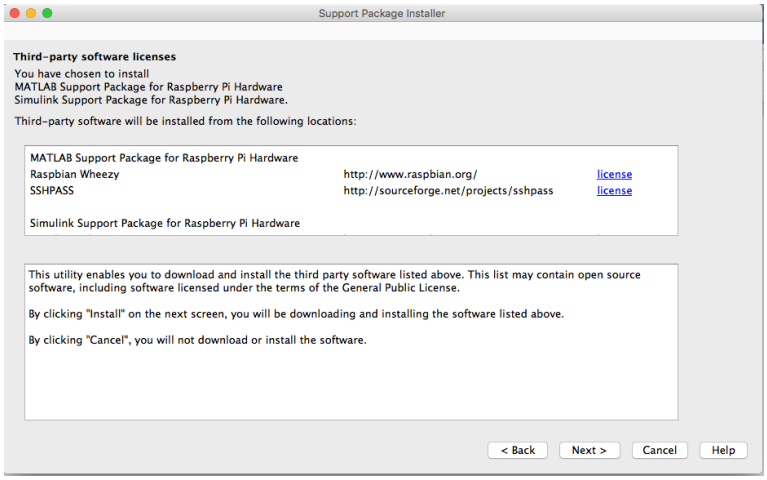


Ilustración 6. . Los tres paquetes que se van a instalar

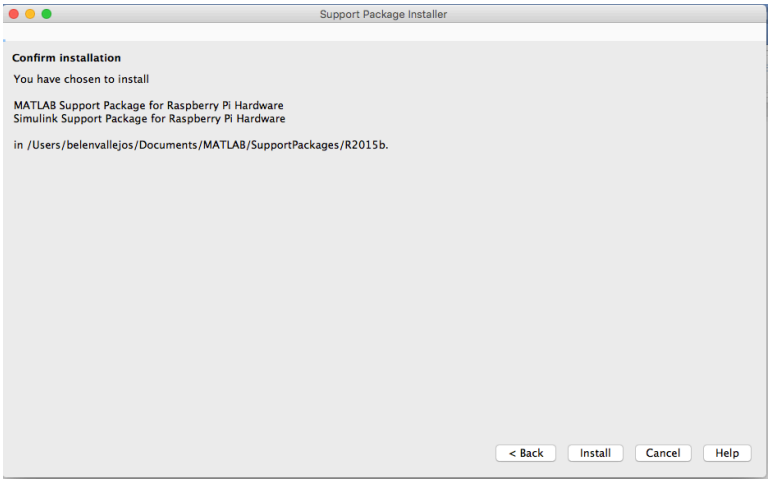


Ilustración 7. Pantalla donde se muestra el lugar donde se instalara la librería.

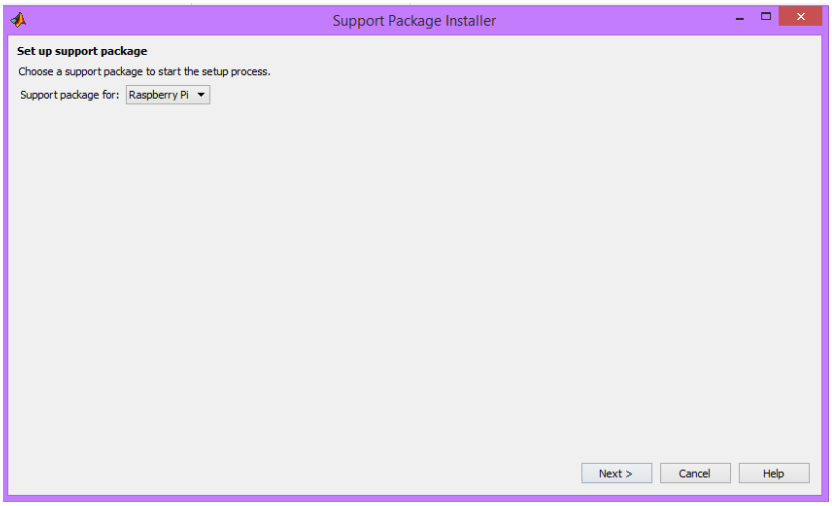


Ilustración 8. Elección del paquete a instalarse

En la ilustración 9 se muestra la forma de configuración de la comunicación Ethernet, como se puede observar existe dos maneras de configurar pero en esta ocasión vamos a utilizar la ***Local área home network.***

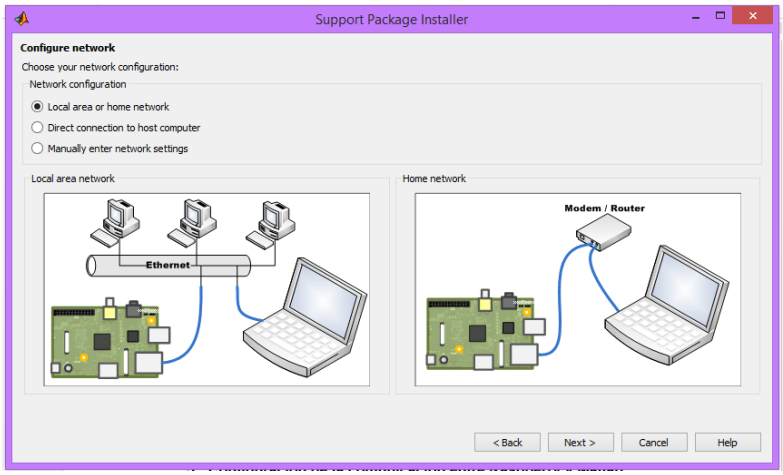


Ilustración 9, Tipos de comunicación

Luego de ya elegir el tipo de comunicación, Matlab procede a instalar el firmware en la memoria SD que se encuentra en el Raspberry pi.

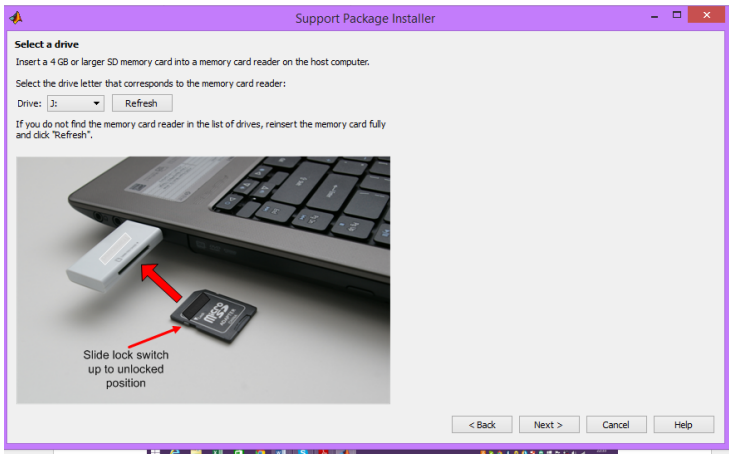


Ilustración 10. Instalación del Firmware

Este es el paso final a continuación se debe configurar la comunicación entre estos dos dispositivos que se detalla en el punto 5.

4. Configuración Raspberry Pi

Primero que debemos hacer es descargar la imagen que debeos grabar en la memoria SD que se inserta en el Raspberry pi.

El sistema operativo que debemos instalar es el ***Raspian whezzy (debían),***

este imagen debemos descargarla directamente de la página oficial de Raspberry pi. Al usar Raspbian, lo recomendable es descargar la versión Lite, que es más pequeña y tardará menos. Es recomendable tener una memoria superior a 4GB.

Para instalar la Imagen en la tarjeta de memoria, el procedimiento es el siguiente:

Windows	IOS
Formatear la tarjeta SD	Formatear la tarjeta SD
Instalar WinDisk 32	Instalar Pi Baker
Elegir la imagen de Raspian	Elegir la Imagen de Raspian.
Acceder al Raspberry.	Acceder a Raspberry.

Luego de ya tener instalada la imagen en nuestra memoria SD, debemos configuración la placa Raspberry pi.

Para la configuración de todos los puertos necesitaremos:

- Modem con conexión a internet
- Computador portátil
- Cable de alimentación micro USB 5V.
- Cable Ethernet
- Raspberry pi.

En el proceso de habilitación vamos a necesitar dos programas adicionales que debemos instalar estos programas son:

- **Tera Term.-** nos va ayudar para acceder a la Raspberry utilizando una comunicación mediante ssh.
- **IPScan.-** el otro para poder buscar la IP del Raspberry pi.

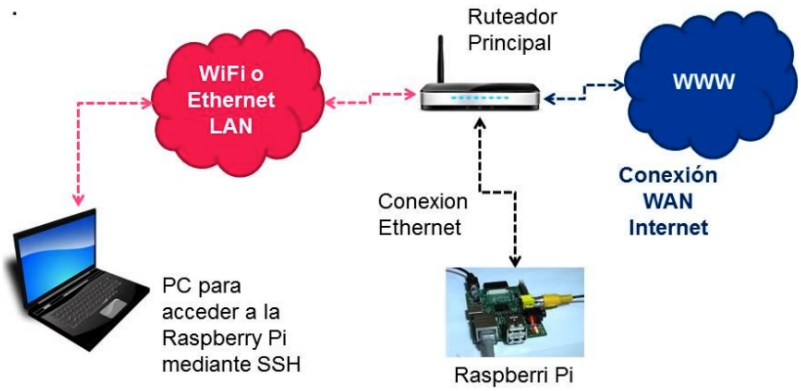


Ilustración 11. Inicio de sesión con SSH

Al ya tener conectado nuestra placa Raspberry pi a nuestro ordenador, debemos ir al programa IPScan para visualizar la dirección de nuestro Raspberry pi.

En mi caso el modem que me está proveyendo de internet es de la empresa estatal CNT, sabemos que la asignación de las IP a cada uno de sus equipos es dinámica, quiere decir que cada vez que asigna una dirección diferente, por eso cada vez que conectemos nuestra Raspberry pi a la red debemos buscar la IP. Para poder visualizar si está conectada a Matlab.

El rango de direcciones en el que vamos a buscar la IP del Raspberry debe ser 192.168.1.1 : 192.168.1.10, porque nuestro modem solo permite 10 equipos.

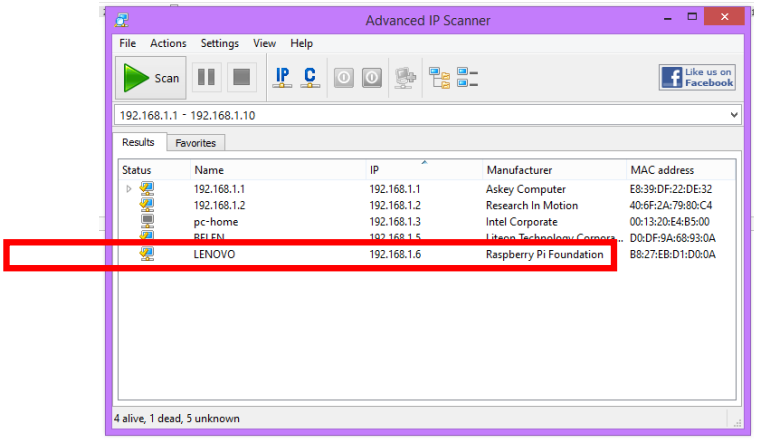


Ilustración 12. Pantalla de IP Scanner

Como podemos observar la IP asignada a nuestro Raspberry es 192.168.1.6.

Ahora para poder configurar el SSH debemos abrir el Tera Term, el cual nos

permitirá acceder remotamente a nuestro Raspberry pi.

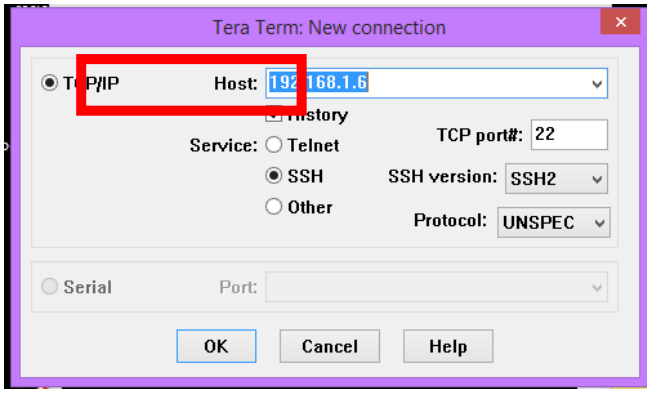


Ilustración 13. Pantalla Principal del Tera Term

En este programa debemos ingresar la IP, que obtuvimos al scanner nuestro modem.

Al hacer clic en ok nos deben aparecer las siguientes pantallas:

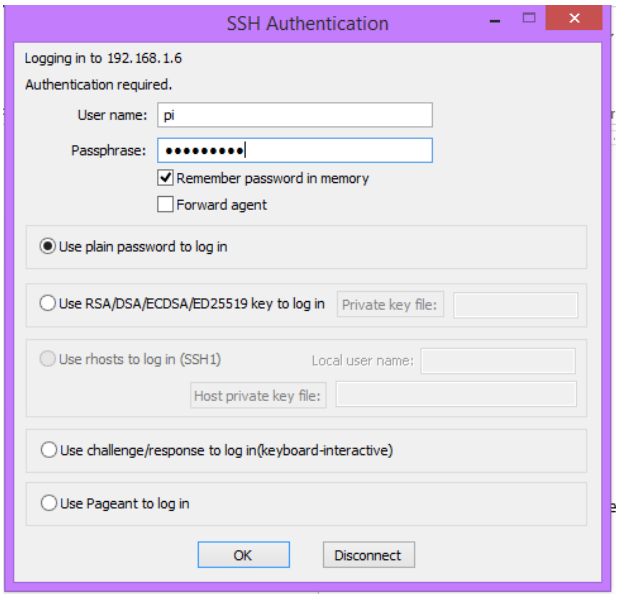


Ilustración 14. Pantalla Tera term

En la ilustración 15, se puede observar que nos pide el *user name y pasword* de nuestra placa, por defecto estos son:

User_Name: pi

Password: raspberry

La siguiente pantalla que nos sale es:

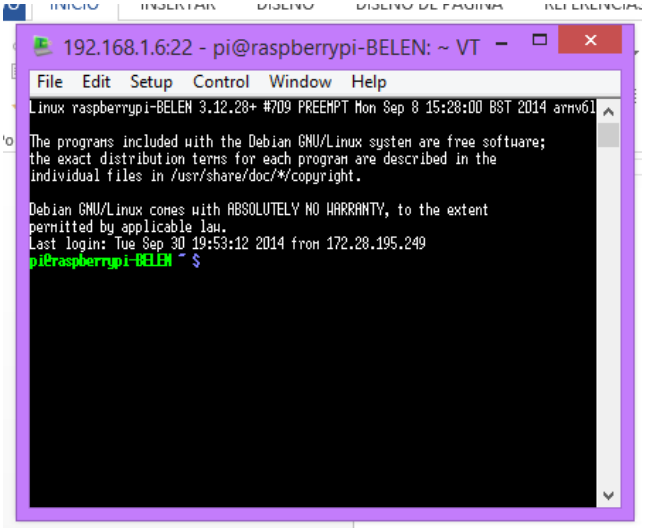


Ilustración 15. Pantalla SSH del Raspberry Pi

Los comandos para la configuración principal son:

Ssh -l [Direccion IP]	Para conectar por ssh desde el terminal
sudo raspi-config	Comando de acceso a la configuración general de Raspberry.
sudo apt-get update	Comando que actualiza en forma general el raspberry
sudo apt-get upgrade	Actualiza todos los paquetes que tienes instalados en el sistema.
Sudo clear	limpia la ventana del terminal.
Sudo date	Muestra la fecha actual.
sudo find /-name prueba.txt	Busca en todo el sistema el archivo prueba.txt y genera una lista de todos los directorios que contienen el archivo.
Sudo nano prueba.txt	Abre el archivo prueba.txt en "Nano", el editor de texto de Linux.
Sudo poweroff:	Apaga el sistema de forma inmediata.
Sudo raspi-	Abre el menú de configuración.

config	
Sudo reboot	Reinicia el sistema de forma inmediata.
Sudo shutdown -h now	Apaga el sistema de forma inmediata.
Sudo shutdown -h 18:34	Apaga el sistema a las 18:34.
Sudo startx:	Abre el interfaz gráfico

Estos tres comandos nos permiten habilitar los puertos de comunicación con Matlab, para verificar que nuestra configuración nos está bien realizada debemos dirigirnos a la pantalla principal de Matlab y escribir este comando ***!ping 192.168.1.6,*** esto nos permite ver si existe o no comunicación entre nuestro computador y raspberry pi.

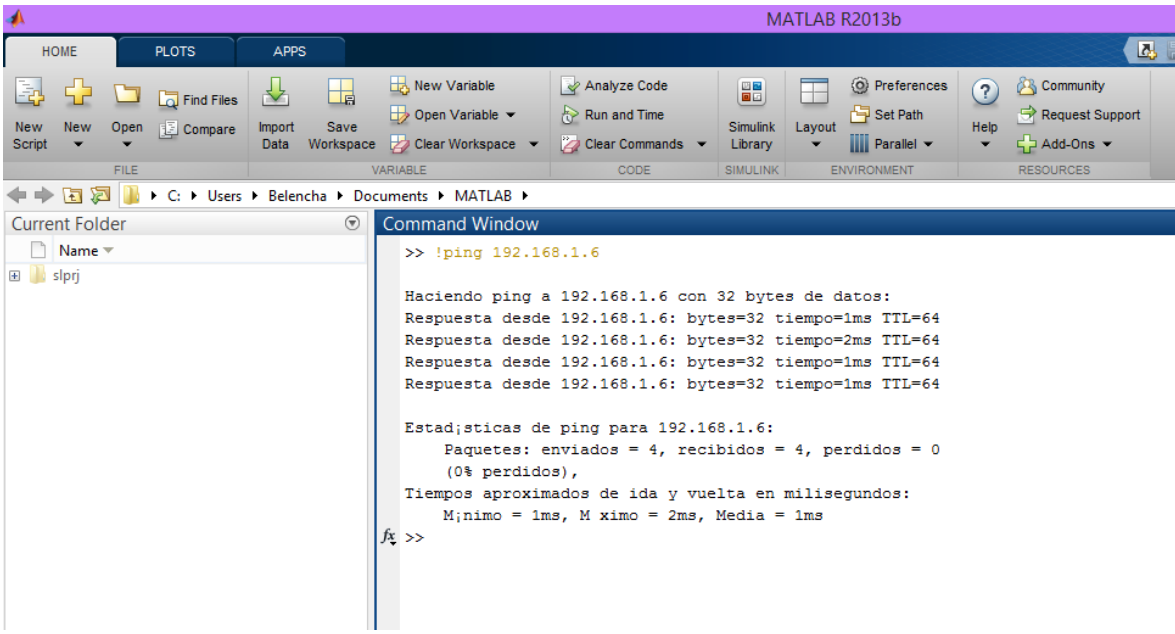


Ilustración 16. Comunicación exitosa

Realizado ya la comunicación exitosa entre Matlab y Raspberry. Podemos realizar los códigos de entrenamiento de imágenes y voz, para luego evaluar en Simulink.

Por recomendación y par aun manejo más rápido se le asigna una IP estática a la Raspberry; a continuación se explica la manera en la que se hace este procedimiento.

27.

28. Configuración para poner IP estática en el Raspberry

Una de las cosas más cómodas a la hora de manejar la Raspberry como enlace con el exterior y para aplicaciones que se utilice como servidor es tener la dirección IP fija.

Lo normal es que tanto con Raspbian como con los otros sistemas operativos, la IP sea dinámica, y se la asigne el router dentro de un rango determinado.

Para hacer esto hay que conocer y tener en cuenta una serie de cuestiones:

- Lo más fácil es tener la IP fija por conexión por cable, tiene más sentido tener la Raspberry junto al router.
- Necesitamos conocer la dirección IP de nuestro router y la de la Raspberry antes de hacer los cambios. Aunque podemos hacerlo incluso sin conexión a red si le conectamos teclado, ratón y monitor para ello.
- Necesitamos ver la configuración del router para saber si tiene algún rango reservado para utilizarlo como IP fijas o no. Si esto es muy complicado, lo que podemos hacer es elegir una dirección muy diferente de la que tenga ahora la Raspberry. De esta forma estaremos lejos de las que el router va poniendo de forma automática y evitaremos problemas de que haya dos dispositivos (PC, móvil, etc) utilizando la misma dirección que solo provocaría problemas. Por ejemplo si la Raspberry Pi tiene antes del cambio una IP parecida a 192.168.1.4 … le ponemos una IP fija como 192.168.1.204 o al revés, si tiene una de tipo 192.168.1.104 le ponemos una como 192.168.1.14; lo importante es que el número del final sea mayor que 2 y menor que 254 para saber el rango de direcciones que utilice el router, utilizamos el programa IPScan.
- El procedimiento para configurar la IP estatica es la Siguiente:

Comandos	Función
sudo nano/etc/dhcpcd.conf	Acceso a los ficheros donde esta la dirección IP .
sudo nano/etc/network/interfaces	Acceso al fichero de IP
ifconfig	Sirve para comprobar el estado de la conexión inalámbrica que estamos utilizando, por ejemplo para ver si wlan0 tiene asignada o no una dirección IP.
iwconfig	Para comprobar a qué red estamos conectados de forma inalámbrica.
iwlist wlan0 scan	Muestra una lista con todas las redes inalámbricas disponibles.
iwlist wlan0 scan | grep ESSID	Si a la orden anterior añadimos | grep junto con el nombre de un campo, el sistema nos mostrará por pantalla tan sólo el campo que necestiamos. Utilizando la orden de ejemplo tan sólo se listará el campo ESSID.
nmap	Escanea tu red y lista los dispositivos conectados, el número de puerto, el protocolo, el sistema operativo, las direcciones MAC
ping	Prueba la conectividad entre dos dispositivos conectados a una misma red. Por ejemplo, ping 10.0.0.32 enviará un

	paquete al dispositivo con IP 10.0.0.32 y esperará una respuesta. También funciona con las direcciones de sitios web lo que nos puede ayudar a saber si tenemos conexión a la red o no utilizando, por ejemplo, ping www.google.es
wget http://www.miweb.com/prueba.txt	Descarga el archivo prueba.txt desde la página web www.miweb.com y lo guarda en el directorio actual.

Instalación de la cámara en la Raspberry Pi

Con las características analizadas de la cámara se llego a la conlusión que es perfecta para proyectos que requieran una cámara pequeña y estable.

La instalación de la cámara en la Raspberry Pi, la cual está contenida en una bolsa anti estática, ya que la placa es sensible a la electricidad estática:

Ilustración 17. Camara dentro de una bolsa antiestatica

La cámara va a ir conectada a cable plano adjunto a la cámara y donde

el cable no se puede doblar no hay que hacer dobleces . Se extrae el plástico protector de la cámara:

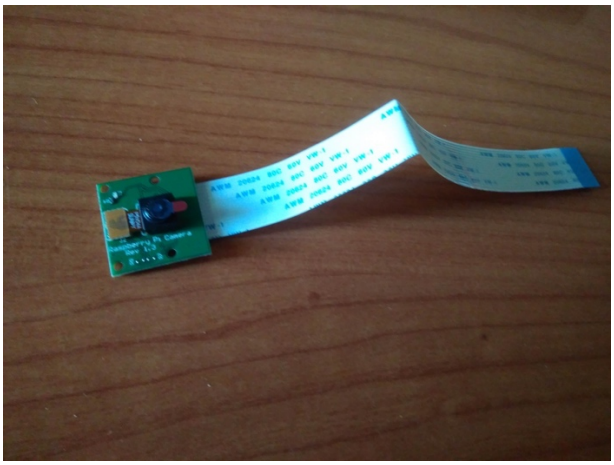


Ilustración 18. Cámara Raspberry

Ya se puede conectar a la Raspberry Pi. Para hacer esto es recomendable extraer la carcasa de la misma (si tiene carcasa protectora), algo que lo hace bastante más fácil.

El puerto que va a utilizarse para conectar el cable plano de la cámara se encuentra al lado del puerto de Ethernet. Utilizando dos dedos, simplemente se presiona ambos lados del plástico y se levanta hacia arriba:

Posteriormente se conecta el cable plano al puerto mencionado anteriormente, asegurándose de que los conectores brillantes estén mirando hacia afuera del puerto de Ethernet, y se cierra el plástico:

Una vez conectada se procede a instalar el software necesario y configurarlo una vez conectada la Raspberry Pi e instalados el SSH y FTP.

Comandos para la instalación de la Cámara

sudo apt−get install python−pip	**Instalación de la librería de la cámara.**

29.

5. Practica 1. ENTRENAMIENTO DE LA RED PARA EL RECONOCIMIENTO DE IMÁGENES CON MAPA DE BITS Y MEDIANTE CODIGO M, SE IDENTIFICAN LAS MATRICES.

OBJETIVO:

Identificar las dimensiones de las matrices de una imagen pixelada, para tener claro los datos que debemos ingresar al Toolbox, para entrenar la red neuronal.

El código que vamos a utilizar para obtener las matrices es el siguiente donde las dimensiones de la matrices es [35,26], representados por la letra [X,T], para iniciar el entrenamiento y luego evaluar a la red, empezamos con una número de neuronas ocultas de 30 ; un error porcentual de 0.05.

30. Código de Entrenamiento.

Luego de ya escrito todo este algoritmo al enviar a correr el programa la primera imagen que se nos va a visualizar es la de la matriz de la letra A, que es por defecto será la A. en la línea de comando *plotchar* es donde se puede elegir la letra que queremos obtener las matrices, dando un valor numérico a cada letra.

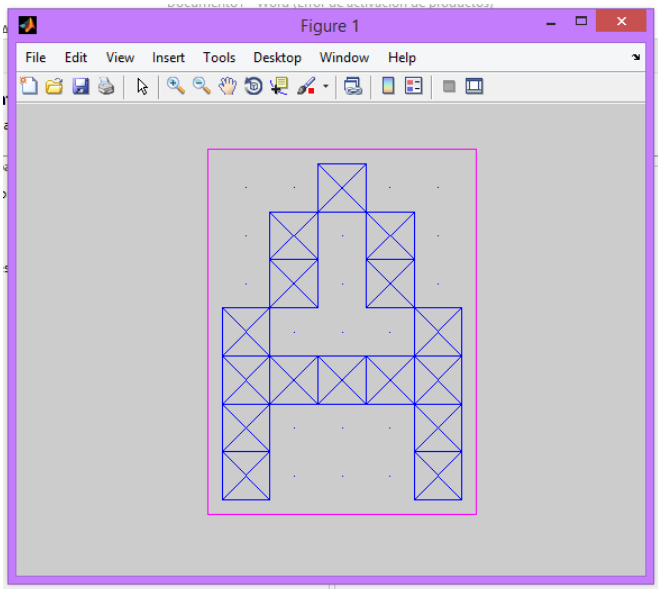


Figura 1. Matriz de la Letra A

Cuando se empieza el entrenamiento de la neurona con el mapa de bits de la letra que ya tenemos se deben desplegar las siguientes ventanas:

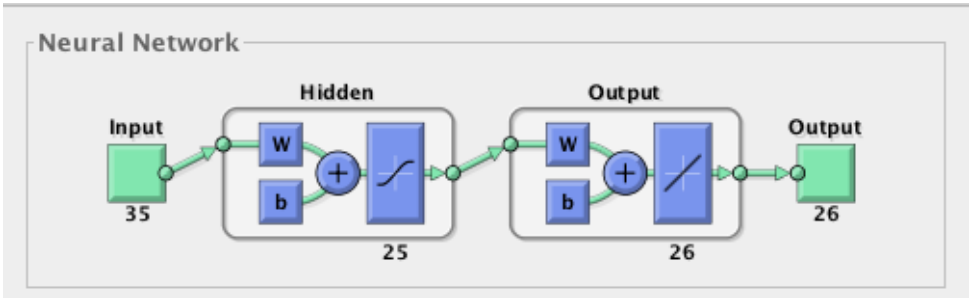


Figura 2. Ecuación de la Red Neuronal

En la figura 4, lo que se observa el número de neuronas ocultas que se ingreso al igual que el número de neuronas de entrada y salida

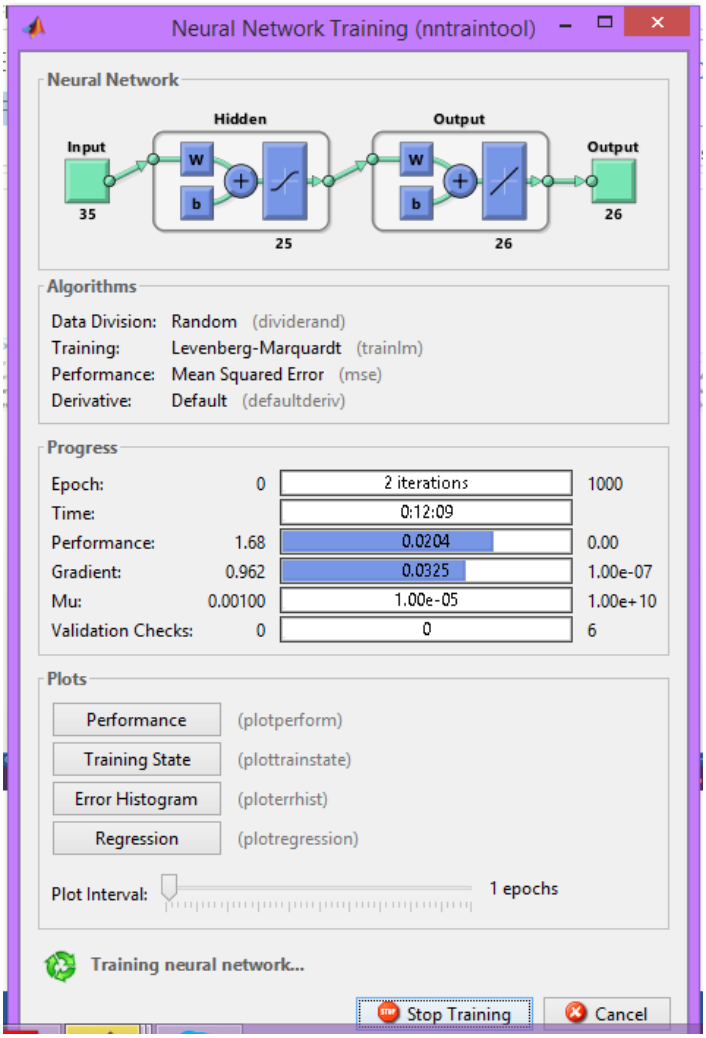


Figura 3. Comienzo del Entrenamiento de la Red

Al finalizar el proceso de entrenamiento con la validación, el tiempo que se demora en entrenar a la red y el mapa de bits de la imagen final las ventanas queda así:

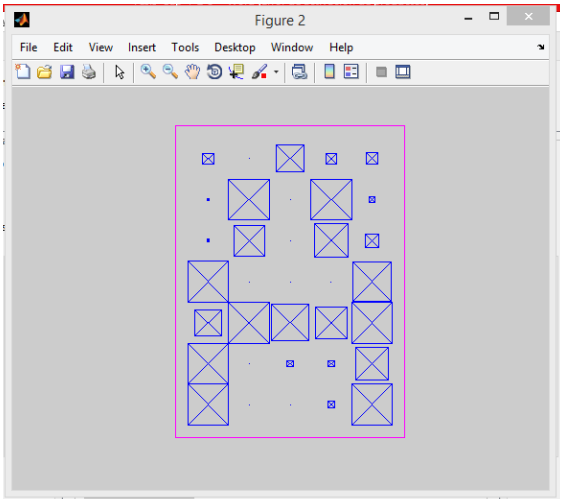


Ilustración 19. Mapa de Bits de la Red Entrenada

En esta parte de entrenamiento de la red lo único que hemos ingresado por código es la creación de las matrices de las letras del alfabeto que se deberán ir cambiando en la siguiente línea del código:

net1 = feedforwardnet(1);

En donde el número es el que representa la letra del alfabeto, partiendo desde 1 es A.

Para finalizar esta parte del algoritmo de backpropagation, pasamos a la parte del Toolbox donde se puede observar donde se ingresa el valor de las matrices creados desde el código.

Para poder tener una idea clara de lo que entrenamos podemos de la Ilustración 7, podemos obtener las 4 graficas del resultado del entrenamiento.

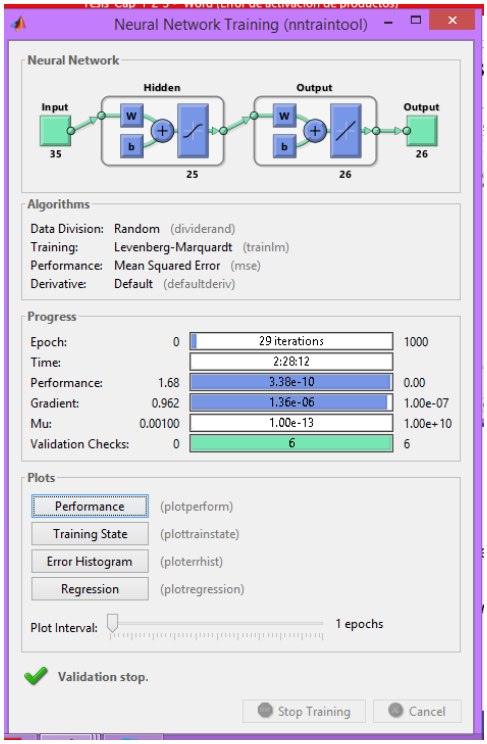


Ilustración 20. Valores Finales del entrenamiento de la red

Para obtener un mejor análisis del entrenamiento de la red neuronal y saber que valores se deben cambiar para un mejor entrenamiento y validación las gráficas que se obtienen son las siguientes:

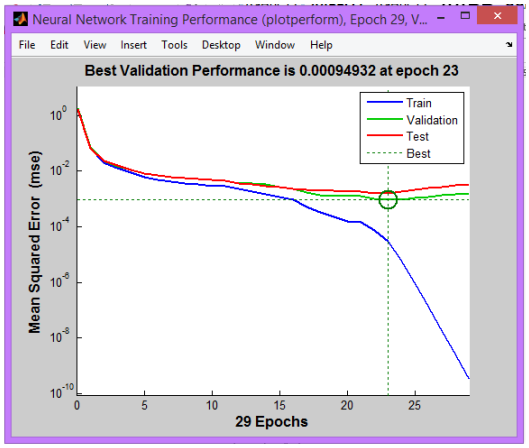


Ilustración 21. Perfomance Diagram

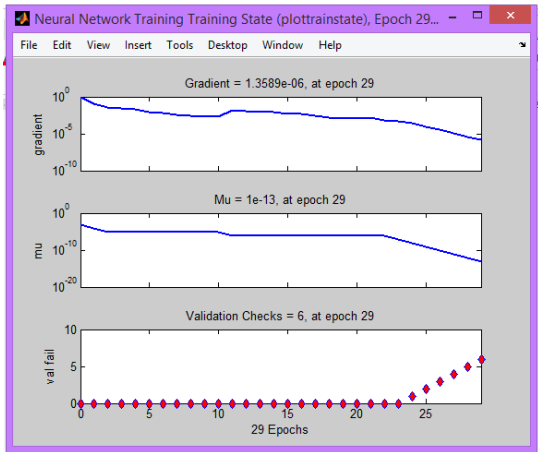


Ilustración 22. Neural Network Training

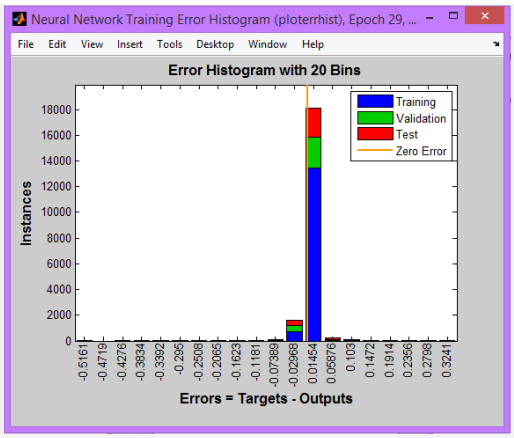


Ilustración 23. Neural Network Training Error Histogram

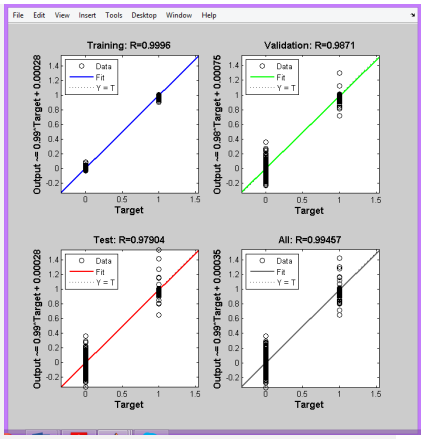


Ilustración 24. Neural Network Training Regression

6. Neural Network Start

Luego de diseñado el algoritmo de entrenamiento procedemos a ingresar al toolbox, para eso en la pantalla principal de Matlab ingresamos el comando *nnstart* y nos aparece la siguiente pantalla:

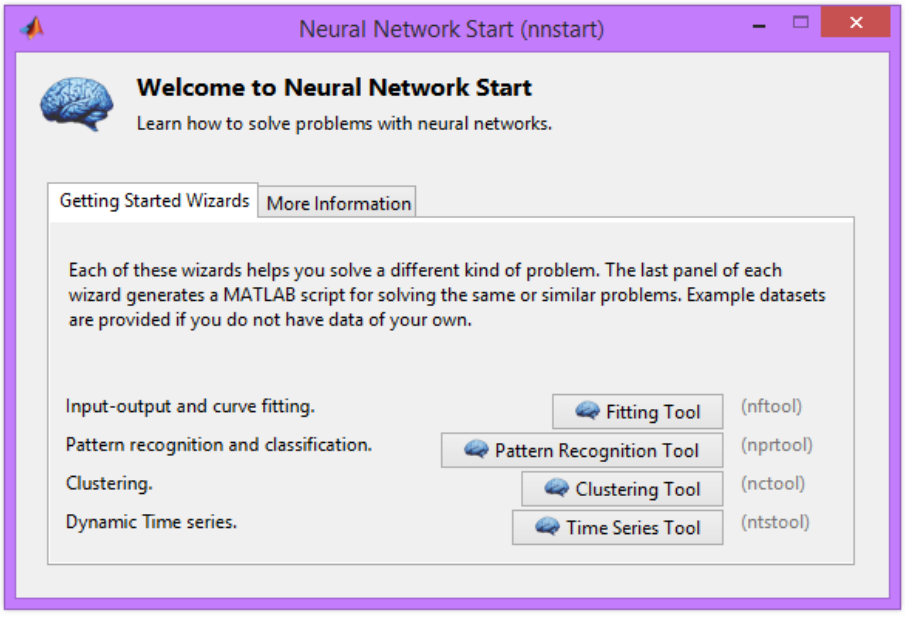


Ilustración 25. Pantalla del Toolbox

Luego seleccionamos el ítem de entrenamiento que será ***Pattern Recognition Tool***, que es lo adecuado para el desarrollo de este proyecto, luego de eso la ventana que nos va a salir es la siguiente.

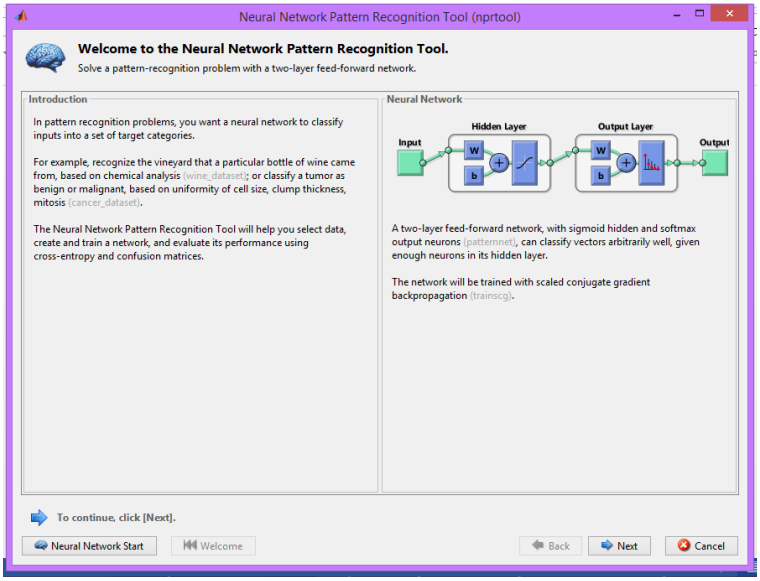


Ilustración 26- Ventana de explicación de la red neuronal

Al visualizar la explicación del tipo del algoritmo con el cual se va a entrenar y ver el esquema hacemos click en *next, la pantalla que se despliega es la de configuración de las entradas y salidas de la neurona, para la cual debemos tener abierto el algoritmo que se menciona antes porque las matrices que se van a ingresar son X y T.*

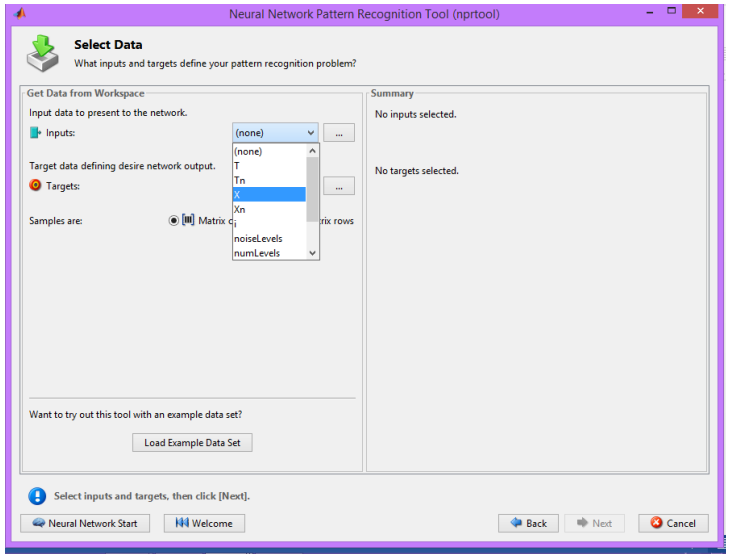


Ilustración 27. Selección matriz X (entrada)

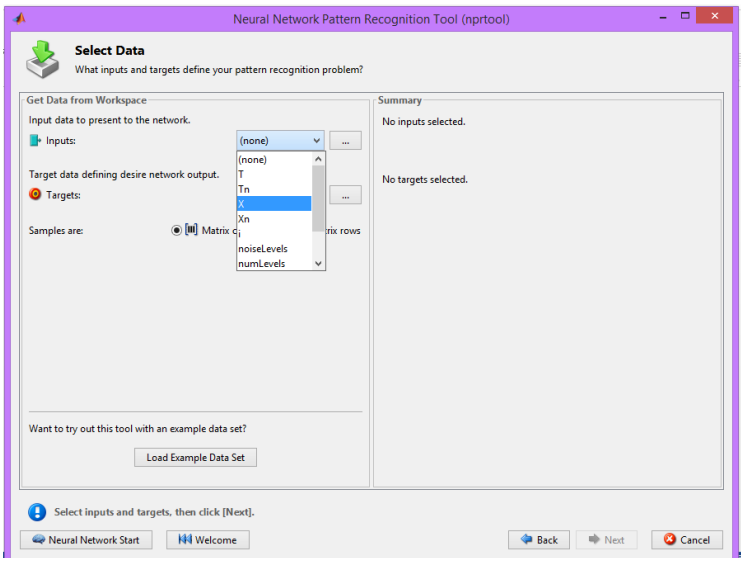


Ilustración 28. Selección de la Matriz T (salida)

En la siguiente imagen ya se configura cuantas capas ocultas va a tener nuestra red.

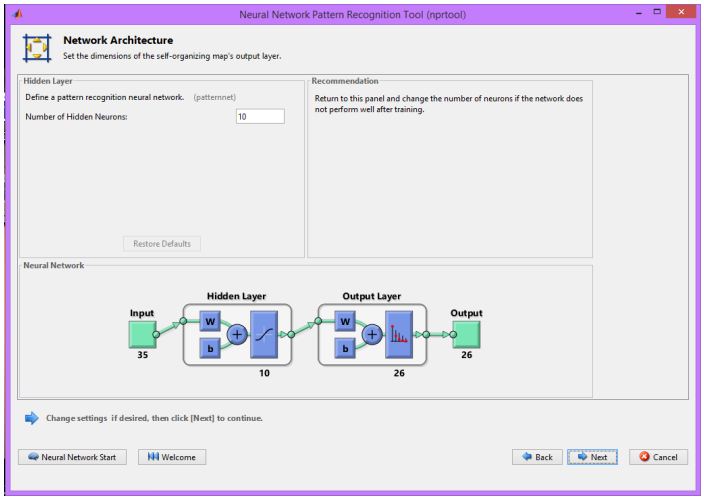


Ilustración 29. Ingreso de las Capas Ocultas

El entrenamiento de la red empieza de la siguiente manera:

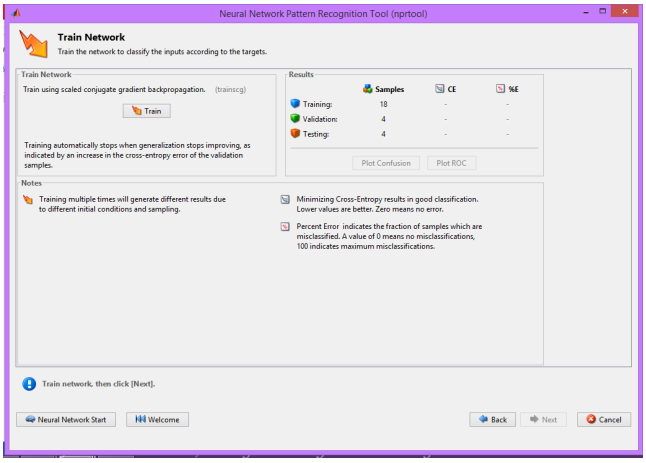


Ilustración 30. Inicio del Entrenamiento

Una de las ventajas de utilizar el toolbox sobre el algoritmo es que este nos facilita las gráficas de análisis en una sola ventana.

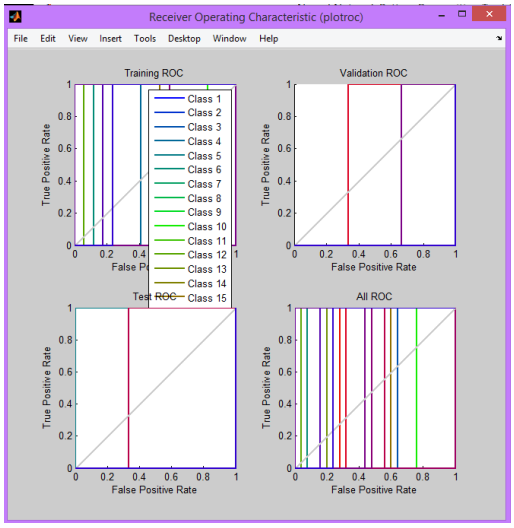


Ilustración 31. Graficas de Analisis luego del Entrenamiento

Para finalizar ya con la parte del entrenamiento de la red neuronal y pasar a la parte de configuración en simulink la última ventana que nos debe salir es la siguiente.

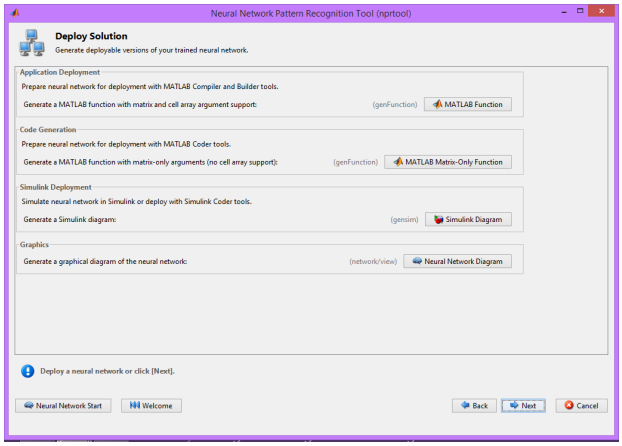


Ilustración 32. Ventana final del Toolbox

Los valores obtenidos al finalizar el entreamiento pueden ser cambiados, como son el numero de neuronas ocultas, el valor de los pesos iniciales, la dimensión de las matrices.

6. Practica 2. ENTRENAMIENTO DE LA RED PARA EL RECONOCIMIENTO DE LETRAS OBTENIDAS POR LA CAMARA DEL RASPBERRY PI.

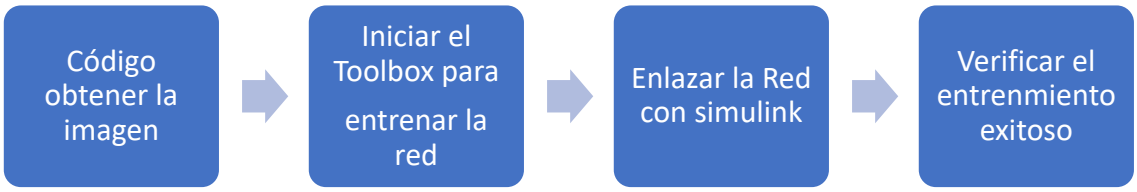
OBJETIVO:

Crear un código que identifique la dimensión de la matriz de la imagen que se obtiene de la cámara del Raspberry pi; para poder obtener la dimensión de la matrices y poder entrenar la red neuronal en el Toolbox , luego enlazar con Simulink.

31. Código de Entrenamiento.

El primer paso para crear nuestro código de entrenamiento, es poder obtener una imagen que sea tomada por la cámara de Raspberry pi;

A continuación el diagrama de bloques que de lo que vamos hacer.



La imagen que se obtiene desde el Raspberry debe ser llamada con el siguiente programa.

```
import time
import sys
from subprocess import call
def envia_foto (foto):
choose BCM or BOARD
```

```
      if len (foto) > 0:
        ejecutar = ['fbi','-d','/dev/fb1','-T','1','-noverbose','-
a',foto]
        call (ejecutar)
        else:
          print "No hay foto"
      def capturar ():
        camera = picamera.PiCamera ()
        camera.resolution = (320, 240)
      %tama?o por defecto de la imagen
        name = str (camera.resolution)
        name = str
('/home/pi/rodrigo_proyecto/2016/Pi2Deep/source/img'+name+'.jpg')
      %asignacion del nombre de la imagen
        camera.capture (name)            %captura
        envia_foto (name)
        camera.close ()
        return name
      def resultado (fichero):
        ejecutar = ['sort','-n','-k2','-r',fichero,'-
o','/home/pi/Pictures/2016/Pi2Deep/source/log2.txt']
      % ordenamos el resultado para que salga la mayor probabilidad
primero
        call (ejecutar)
        res = open
('/home/pi/pictures/2016/Pi2Deep/source/log2.txt','r')
        %abrimos fichero resultado
        linea = res.readline ()  %leemos la primera linea
        linea = linea.split ()   %separamos la linea en una lista
de palabras

          ejecutar =
['/home/pi/picture/2016/Pi2Deep/source/jpcnn','-i',foto,'-
n','/home/pi/rodrigo_proyecto/2016/Pi2Deep/networks/jetpac.ntwk','-
t','-
      m','s','-d']outfile =
open('/home/pi/rodrigo_proyecto/2016/Pi2Deep/source/log.txt', 'w')
      errfile = open ('/dev/null', 'w')
          call (ejecutar,stdout = outfile,stderr = errfile)
      %ejecucion del reconocimiento
          outfile.close ()
          errfile.close ()

        envia_foto ('Fin del programa')
        ejecutar = ['rm','-
r',foto,'/home/pi/picture/2016/Pi2Deep/source/log.txt','/home/pi/pi
cture/2016/Pi2Deep/source/log2.txt']
      28

      def op1 ():
        envia_foto (capturar ())
        time.sleep (3)
      def op2 ():
        envia_foto (fotos[10])
        time.sleep (4)
      def op3 ():
        re =
```

```
'/home/pi/rodrigo_proyecto/2016/Pi2Deep/source/img(320, 240).jpg'
        reconoce (re)
        envia_foto (fotos[10])
      def op0 ():
        salir (fotos[10])
        call ('poweroff')
        exit (0)
```

Luego de obtener la imegen desde los ficheros del raspberry, lo que debmos ahcer es binarizar la imagen con el siguiente código:

Código de obtención de matrices de la Imagen

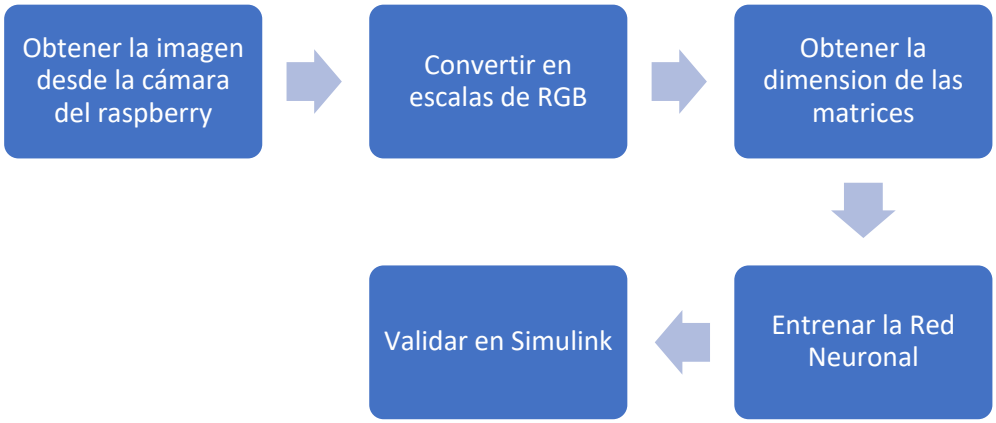
```
1.    function y = Binarizacion(imagen,umbral)
2.    im=imread(imagen);
3.    imD=double(im);
4.    [f,c]=size(imD);
5.    for i=1:f
6.        for j=1:c
7.            if imD(i,j)<=umbral
8.                nuevaI(i,j) = 0;
9.            else
10.               nuevaI(i,j) = 255;
11.           end
12.       end
13.   end
14.   imB = uint8(nuevaI);
15.   imshow(imB);
16.   y = nuevaI;
```

Como en la práctica Nro. 1 ya se reconoce letras en esta práctica vamos a reconocer los números del 0 al 9, individualmente.

Luego de ya obtener las matrices se repité el procedimiento de entrenamiento de la red en el Toolbox ya que este código nos genera las matrices, como esta en la práctica anterior.

Luego de ya tener la red entrenada debemos enlazar con Simulink y

raspberry de la siguiente forma.



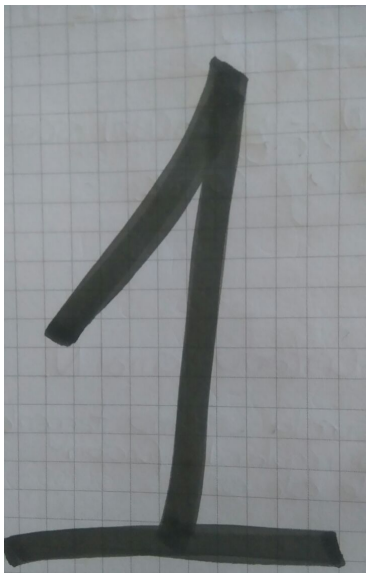


Ilustración 33 .Imagen Obtenida por la cámara de Raspberry.

El esquema de bloques en Simulink nos queda de la siguiente manera:

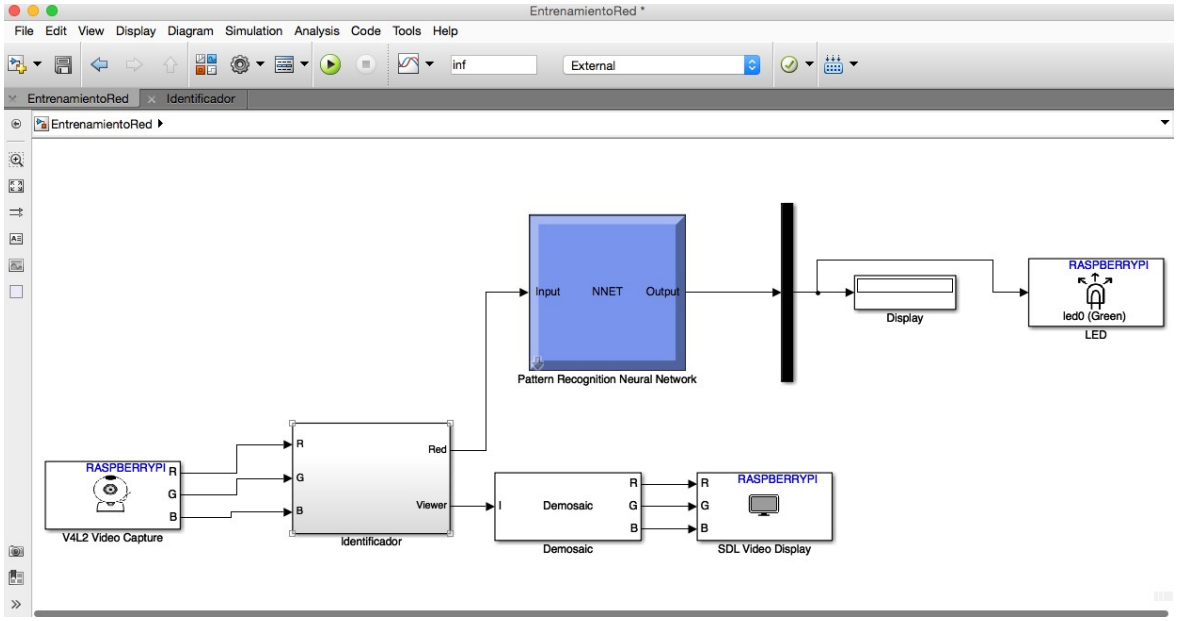


Ilustración 34.Esquema final del enlace entre Simulink y Raspberry PI

7. Practica 3. ENTRENAMIENTO DE LA RED PARA EL RECONOCIMIENTO DE VOZ GRABADA POR CODIGO EN MATLAB.

OBJETIVO:

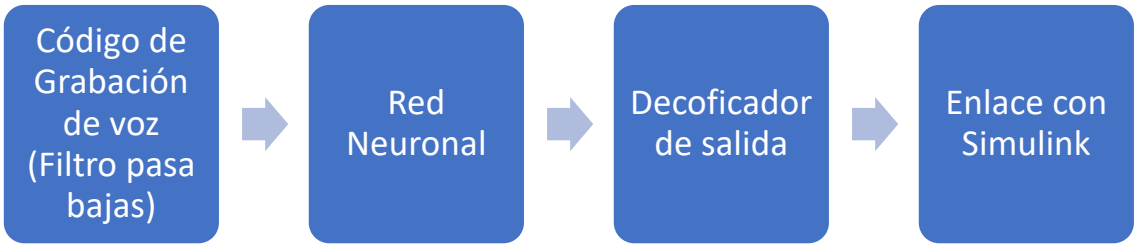
Crear un código que pueda grabar un archivo en formato .*wav, con un filtro pasa bajas, para su decodificación, luego poder obtener la dimensión de las matrices para entrenar la red neuronal en Toolbox, y así poder enlazar con Raspberry pi; para poder obtener la dimensión de la matrices y poder entrenar la red neuronal en el Toolbox , luego enlazar con Simulink.

32. Código de Entrenamiento.

El primer paso para crear nuestro código de entrenamiento, es realizar el

código de grabación de la voz.

A continuación el diagrama de bloques que de lo que vamos hacer.



Código de grabación de la voz.

```
%% Procesamiento de una se?al de audio usando MATLAB
%% Crear se?al de audio
% Frecuencia fundamental
f0=1e3; % 1KHz
% Amplitud
a=4; % V=4
% Frecuencia de muestreo
fs=44.1e3; % Frecuencia de una se?al de audio
% Tiempo de duraci?n en segundos
T=1.5;
L = round(T*fs); % N?mero de muestras
% Frecuencia normalizada
fn=f0/fs;
y = a*sin(2*pi*fn*(0:L-1))+0.5*a*sin(2*pi*2*fn*(0:L-1));
% Graficar la se?al original
subplot(411)
plot((0:L-1)/fs,y)
title('SE?AL ORIGINAL')% T?tulo
xlabel('Tiempo (s)')          % Etiqueta del eje X
ylabel('Amplitud (V)')       % Etiqueta del eje Y
xlim([0 10/1000])             % L?mite de la se?al
%% Grabar y reproducir la se?al de audio
%wavwrite(y,fs,'audio')
% wavplay(y,fs)
%% FFT de la se?al
subplot(412)
% Llamado a la funci?n que calcula la FFT
fft_signal(y,fs);title('ESPECTRO DE LA SE?AL ORIGINAL')
xlim([0 2500])
%% Filtrado de la se?al
% Frecuencia normalizada
fNorm = 1500 / (fs/2);
```

```
% C?lculo de los coeficientes del filtro (filtro pasa bajas)
[b,a] = butter(10, fNorm, 'low');
% Filtrado de la se?al
y_Low = filtfilt(b, a, y);
% Graficaci?n de la se?al en el tiempo
subplot(413)
plot((0:L-1)/fs,y_Low)
title('SE?AL FILTRADA')
xlabel('Tiempo (s)')
ylabel('Amplitud (V)')
xlim([0 10/1000])
% Graficaci?n de la se?al en frecuencia
subplot(414)
% Llamado a la funci?n que calcula la FFT
fft_signal(y_Low,fs);title('ESPECTRO DE LA SE?AL FILTRADA')
xlim([0 2500])
%% Gr?ficas del filtro
% Respuesta en frecuencia del filtro
[H,w]=freqz(b,a,512,1);
figure(2)
%Trazado de la respuesta en Magnitud
subplot(221)
plot(w,20*log10(abs(H)));
grid on;
title ('Filtro pasa-altos, Respuesta en magnitud');
xlabel('frecuencia');
ylabel('H(f) db')
xlim([0 0.4])
% Respuesta en fase
subplot(222)
plot(w,angle(H));
grid on;
title ('Filtro pasa-altos, Respuesta en fase');
xlabel('frecuencia')
ylabel('?ngulo de H rad')
xlim([0 0.4])

%Respuesta al impulso
subplot(223)
[y_eje,t]= impz(b,a,60);
stem(t,y_eje);
title ('Filtro pasa-altos, Respuesta al impulso');

%Ploteo de los polos y ceros
z= roots(b);   % Ceros
p = roots(a);  % Polos
subplot(224)
zplane(z,p)
title('Polos y ceros')
legend('Ceros','Polos')
%% Reproducci?n de audio de entrada y salida
pause(2)
disp('Audio de entrada')
wavplay(0.5*y,fs)
disp('Audio de salida (se?al filtrada)')
wavplay(0.5*y_Low,fs)
```

Luego de ejecutar nuestra programa para grabar la voz, nos muestra esa gráfica

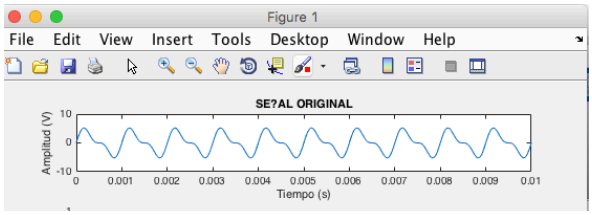


Ilustración 35.

.

El procedimiento se repite el entrenamiento, como en la práctica Nro. 1 y 2.