



**UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO**  
***FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS***



## **TITULO DE LA TESIS**

Sistema de Navegación Automático controlado por voz, maniobrable y  
seguro adaptable a sillas de ruedas motorizadas

**PARA OBTENER EL TÍTULO PROFESIONAL DE INGENIERO  
ELECTRÓNICO.**

**PRESENTADO POR**

Bach. Veronica Jackelin Guisela Farro Herrera

Bach. Luz Mariela Villegas Malca

LAMBAYEQUE, PERÚ

2017

***FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS***  
***ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA***



Sistema de Navegación Automático controlado por voz, maniobrable y  
seguro adaptable a sillas de ruedas motorizadas

**PARA OBTENER EL TÍTULO PROFESIONAL DE INGENIERO  
ELECTRÓNICO.**

Bach. Veronica Jackelin Guisela Farro Herrera

Bach. Luz Mariela Villegas Malca

Asesor

Ing. Manuel Javier Ramirez Castro


**UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO**  
***FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS***

**TITULO DE LA TESIS**

Sistema de Navegación Automático controlado por voz, maniobrable y  
seguro adaptable a sillas de ruedas motorizadas

Como requisito para obtener el Título Profesional de Ingeniero Electrónico.

**Aceptada por la Escuela Profesional de Ingeniería Electrónica**



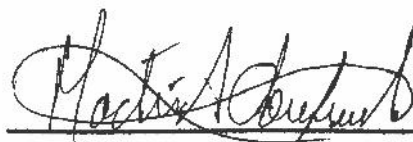
---

Ing. Hugo Javier Chiclayo Padilla  
PRESIDENTE



---

Ing. Segundo Francisco Serrano Altamirano  
SECRETARIO



---

Ing. Martín Augusto Nombora Lossio  
VOCAL

LAMBAYEQUE, PERÚ

2017

**UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO**  
***FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS***

**TITULO DE LA TESIS**

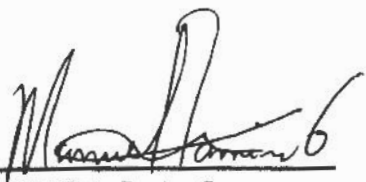
Sistema de Navegación Automático controlado por voz, maniobrable y  
seguro adaptable a sillas de ruedas motorizadas

Como requisito para obtener el Título Profesional de Ingeniero Electrónico.

**Sustentada por:**

  
Bach. Veronica Jackelin Guisela Farro Herrera  
TESISTA  
Bach. Luz Mariela Villegas Malca  
TESISTA

**Asesorado por:**

  
Ing. Manuel Javier Ramirez Castro  
ASESOR

LAMBAYEQUE, PERÚ

2017

# Agradecimientos

---

**E**n primer lugar deseamos expresar nuestro agradecimiento a ti Dios por bendecirnos para cumplir un más de nuestras metas, hacer realidad este sueño anhelado.

Agradecemos a nuestros profesores que durante toda nuestra carrera profesional estuvieron allí brindándonos su aporte, su esfuerzo, dedicación y motivación por ello hemos logrado culminar nuestros estudios con éxito.

Gracias a nuestras familias, a nuestros padres por su apoyo incondicional para con nosotras, por ser nuestro pilar y estar siempre para nosotras sobre todo en los momentos más difíciles.

A mis amigos sin importar donde estén quiero agradecerles por formar parte de mi vida.

A todos, muchas gracias

Los Autores.



# Dedicatoria

---

*Con mucho cariño y amor para las personas que  
hicieron posible llegar hasta aquí gracias a su  
esfuerzo, dedicación y estuvieron a mi lado  
brindándome su apoyo.*

*A mi amada mamita y mi amado Juan Carlos.*

***Verónica Jackelin Farro Herrera***

*Mi agradecimiento es ante todo a Dios por darme la  
oportunidad de vivir y por estar conmigo en cada pa-  
so que doy, por fortalecer mi corazón e iluminar mi  
mente, a mis padres y a mi hermano Jonathan por darme  
siempre su apoyo incondicional y a todos aquellos que  
con sus palabras de aliento me ayudaron a terminar este  
paso de vida.*

***Luz Mariela Villegas Malca***





# Resumen

---

**S**e ha determinado que, en el Perú, el 40.6 % de las personas con discapacidad necesitan del apoyo de terceros para realizar sus actividades y un 24.1 % está entre 65 y 74 años, lo que hace difícil su locomoción, pues no cuentan con suficiente fortaleza para hacerlo por ellos mismos.

Nos propusimos desarrollar un sistema de navegación automática controlada por voz para mejorar la movilidad de estas personas.

Para esto realizamos un estudio de las características mecánicas y motrices de las sillas motorizadas y también de las técnicas de clasificación de comandos de voz, siendo las más utilizadas redes neuronales y aprendizaje profundo, además de que existen en el mercado diferentes soluciones offline y online que facilitan el desarrollo y adaptación sistemas, pues ambos sistemas son fácilmente asimilados por los usuarios de las sillas motorizadas.

Esto permitió desarrollar un prototipo de una interfaz compuesta de hardware y software adaptable a estas sillas motorizadas, para lo cual se construyó un prototipo de silla de ruedas a la cual se le adapta esta interfaz.

Finalmente se consiguió un sistema de navegación automática controlada por voz que permite mejorar la movilidad de las personas con severas discapacidades motrices, facilitando su traslado con un esfuerzo mínimo de aprendizaje de la forma de control lo que aumenta su autonomía.



# Abstract

---

It has been determined that, in Peru, 40.6 % of people with disabilities need the support of third parties to carry out their activities and 24.1 % are between 65 and 74 years of age, which makes it difficult for them to move around, as they do not have the strength to do so themselves.

We set out to develop a voice-controlled automatic navigation system to improve the mobility of these people.

For this purpose, we carry out a study of the mechanical and motor characteristics of motorized chairs and also of the voice command classification techniques, the most commonly used being neural networks and deep learning, in addition to the fact that there are different offline and online solutions on the market that facilitate the development and adaptation of systems, since both systems are easily assimilated by the users of motorized chairs.

This allowed to develop a prototype of an interface composed of hardware and software adaptable to these motorized chairs, for which a prototype of wheelchair was built to which this interface is adapted.

Finally, a voice-controlled automatic navigation system was developed to improve the mobility of people with severe motor disabilities, facilitating their transfer with a minimum effort to learn how to control them, which increases their autonomy.

*Translated with [www.DeepL.com/Translator](https://www.DeepL.com/Translator)*



# Índice

---

<i>Resumen</i>	V
<i>Abstract</i>	VII
<b>1. Introducción</b>	<b>1</b>
1.1. Descripción del Problema	1
1.2. Formulación del Problema	2
1.3. Justificación	2
1.4. Objetivos de la Investigación	3
1.4.1. Objetivo General	3
1.4.2. Objetivos Específicos	3
1.5. Hipotesis	3
<b>2. Marco Teórico</b>	<b>5</b>
2.1. Sillas de Ruedas	5
2.1.1. Bloques de Silla de Ruedas Motorizada	7
2.2. Redes Neuronales	8
2.2.1. Introducción	8
2.2.2. Los Elementos principales de las RNA	9
2.2.3. La Arquitectura de las RNAs	11
Clasificación de RNA según su estructura en capas	11
Clasificación de las RNA según el flujo de datos de la red	12

---

	Clasificación de las RNA según el grado de conexión	12
	Clasificación de las RNA según el tipo de respuesta de red	12
2.3.	Python	13
2.3.1.	Introducción	13
2.3.2.	Instalación de Python	13
2.3.3.	Primeros Pasos con Python	14
2.4.	Arduino	15
2.4.1.	La plataforma Arduino	16
	La Tarjeta Arduino	17
	Arduino Uno y Genuino Uno	17
	Arduino Mega 2560 y Genuino Mega 2560	17
	La IDE Arduino	18
2.5.	Raspberry	19
2.6.	Sistemas de Reconocimiento Automático de voz - ASR	21
2.6.1.	Comunicación Humano-Humano	21
2.6.2.	Comunicación Humano-Computadora	22
2.6.3.	Arquitectura de los Sistemas ASR	22
2.7.	Movi de Audeme	23
<b>3.</b>	<b>Desarrollo de la Ingeniería del Proyecto</b>	<b>25</b>
3.1.	Descripción del Proyecto	25
3.2.	Construcción de Prototipo	27
3.3.	Diseño del algoritmo de Control por Voz	30
3.3.1.	Obtención de las tablas de entrenamiento	32
	Procesamiento de Señales de Entrada - Ordenes grabadas	32
3.3.2.	Definir la Estructura de la Red Neuronal	37
3.3.3.	Inicializar Parámetros de la red neuronal	38
3.3.4.	Lazo de Entrenamiento	40

---

Propagación hacia adelante	40
Función Lineal	40
Función Activación Lineal	40
Modelo de L capas ocultas	41
Cálculo de Errores	42
Implementar el algoritmo de propagación hacia atrás	42
Función lineal hacia atrás.	43
Función Activación Lineal hacia atrás	44
Modelo de L capas ocultas hacia atrás	45
Actualizar parámetros	46
3.3.5. Pruebas de Desempeño	46
3.3.6. Consolidado de Arquitectura de Red Neuronal	47
3.3.7. Conexionado de Tarjetas	49
Solución 01	49
Solución 02	50
3.3.8. Programa de Control	51
<b>4. Conclusiones</b>	<b>57</b>
<b>Apéndice A. Datasheet de Equipos</b>	<b>59</b>
<i>Índice de Figuras</i>	63
<i>Índice de Tablas</i>	65
<i>Bibliografía</i>	67
<i>Glosario</i>	69
<b>Glossary</b>	<b>69</b>





# 1 Introducción

---

*“En algún lugar, algo increíble está esperando ser conocido”..*

CARL SAGAN

**E**n este capítulo fundamentaremos la investigación, identificando la problemática, la que finalmente se sintetizara en la pregunta del problema, que nos ayudara a establecer las variables de la investigación para las cuales estableceremos una hipótesis y objetivos que guiaran el resto de nuestro trabajo.

## 1.1 Descripción del Problema

Los resultados de Primera Encuesta Nacional Especializada sobre Discapacidad[INEI, 2012, pag 9], muestran que en el Perú, existen 1 millón 575 mil 402 personas que padecen algún tipo de discapacidad o limitación física y/o mental. Siendo los más afectados, los mayores de 65 años (50.4 %) y las personas comprendidas entre 15 a 64 años (41.3 %).

En el mismo documento indica que la limitación más frecuente son las discapacidades para moverse o caminar y/o para usar brazos o piernas (59.2 %) y las de tipo visual (50.9 %). Inclusive el 40.6 % de personas con discapacidad necesita del apoyo de terceros para realizar sus actividades diarias, siendo sus propios familiares del hogar los que asisten con mayor frecuencia en sus actividades diarias.

Ademas en el año 2012, un total de 931 mil 993 personas declaran tener al menos una discapacidad de locomoción y/o destreza. Esto representa el 3.1 % de la población total. De este total el 53.1 % no mantiene el equilibrio, se mueve y camina con dificultad dentro de la casa, también del total de personas con discapacidad de locomoción o destreza según grupos de edad, la mayor presencia se aprecia en los grupos de 30 a 64 años (31.5 %), de 65 a 74 años (24.1 %) y de 75 a 84 años (24.1 %).

Es necesario pues brindar una alternativa de movimiento asistido a las personas mayores de 65 años, que usualmente no tienen la suficiente fortaleza para hacerlo por ellos mismo, y este debe ser amigable, con una gran facilidad de programación que permita una libertad de maniobra con seguridad y autonomía dándole la posibilidad de completar todas sus actividades diarias.

## **1.2 Formulación del Problema**

¿De que manera un sistema de navegación automática controlada por voz mejorara la movilidad de las personas con severas discapacidades motrices permitiéndoles un mayor grado de autonomía?

## **1.3 Justificación**

Este trabajo es importante pues estaremos proporcionando una alternativa de control de silla de ruedas motorizada que permitirá mejorar la movilidad de personas con severas discapacidades motrices, dándoles una mayor autonomía y menor dependencia de las personas.

A nosotros nos permitirá experimentar con algoritmos de Machine learning y redes neuronales aplicándolos a un problema real, además de mejorar nuestras habilidades de programación de mini computadoras como la Raspberry; utilizando un lenguaje de programación como Python, y los sistemas embebidos tales como la Arduino.

Una vez concluida la tesis se estará brindado una solución de hardware y software que fácilmente puede adaptarse a las sillas de ruedas motorizadas.

## 1.4 Objetivos de la Investigación

### 1.4.1 Objetivo General

Desarrollar un sistema de navegación automática controlada por voz con alto grado de maniobrabilidad y seguridad, adaptable a sillas de ruedas motorizadas para mejorar las movilidad de las personas con severas discapacidades motrices permitiéndoles un mayor grado de autonomía.

### 1.4.2 Objetivos Específicos

1. Identificar las características de las señales de control necesarias para las sillas motorizadas y desarrollar un prototipo de unidad móvil.
2. Identificar los principales algoritmos de caracterización de la voz y posterior clasificación usando técnicas de machine learning y redes neuronales.
3. Diseñar y construir una interfaz de hardware y software que permita adaptar la micro computadora con el control por voz al prototipo de unidad móvil.
4. Realizar las pruebas que demuestren la maniobrabilidad, seguridad y autonomía del prototipo construido.

## 1.5 Hipotesis

Si desarrollamos un sistema de navegación automática controlada por voz con alto grado de maniobrabilidad y seguridad; adaptable a sillas de ruedas motorizadas se mejorara la movilidad de las personas con severas discapacidades motrices permitiéndoles un mayor grado de autonomía



## 2 Marco Teórico

---

*En inteligencia artificial, los investigadores usan modelos computacionales para obtener una percepción profunda de la psicología humana así como para reflexionar sobre ésta como fuentes de ideas para generar mecanismos que emulen la inteligencia humana....*

SEYMOUR PAPERT

### 2.1 Sillas de Ruedas

Las sillas de ruedas, están diseñadas para permitir el desplazamiento de las personas con problemas de locomoción o movilidad reducida, debido a una lesión o enfermedad física[Wikipedia, 2017b]. Existen básicamente dos tipos de sillas de ruedas:

- **Manuales:** Son impulsadas por el propio ocupante que hace girar las ruedas traseras, con unos aros adaptados a las mismas. Pueden ser plegables o rígidas, siendo construidas con material ultraligeros y resistentes, tales como aluminio de aviones y el titanio al carbono con revestimiento de Kevlar para una larga duración y bajo peso, para que sea manipulada fácilmente por el usuario, consiguiendo así cierto grado de autonomía y autosuficiencia. (ver figura 2.1)

- **Motorizadas:** O eléctricas, que son impulsadas por baterías recargables de 40 o 50 Amperios. El ocupante controla la silla por medio de un mando o un panel de control que tienen muchas opciones de configuración. (Ver figura 2.2)



**Figura 2.1** Sillas de Ruedas Manuales..



(a) Silla Eléctrica Plegable. (b) Silla Eléctrica Plegable. Precio 2788.00 dólares [ama, 2017a].  
Precio 3529.00 dólares [ama, 2017b].

**Figura 2.2** Sillas de Ruedas Motorizadas..

En [Castelao and Faes, 2008], se realiza un análisis de los productos de apoyo con personas con discapacidad, siendo fundamentalmente la información usada del estándar UNE-

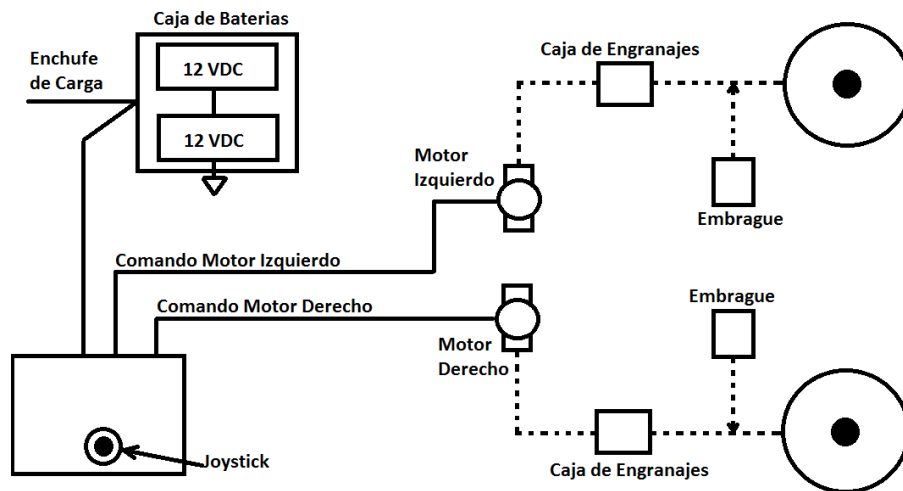
**EN ISO 9999:2007**, que en las paginas 49, 50 y 51(Ver apéndice A) tiene una lista extensa en lo referente a sillas de ruedas manuales (ahí las llaman *sillas de rueda de propulsión manual*) y a las sillas motorizadas (llamadas *sillas de rueda de propulsión motorizada*), ademas en el apartado 12 24 03 menciona los *Sistemas de dirección y de control* como los dispositivos para controlar los movimientos de la silla de ruedas y dirección de recorrido, también menciona los sistemas de frenado o estacionamiento de las sillas de ruedas.

También [sun, 2016] se tiene un completo manual de sillas de ruedas motorizadas de la marca *Sunrise Medical*, detallando el funcionamiento del control VR2, advertencias sobre seguridad y sugerencias de usuario, bloqueo y desbloqueo de sillas de ruedas, como manejar la silla en aceras, bordillos o escalones, las precauciones en rampas y pendientes, también el cuidado de las baterías y carga de las mismas. Finalmente y no menos importante es la limpieza de la silla de ruedas.

### 2.1.1 Bloques de Silla de Ruedas Motorizada

Una silla motorizada incluye los siguientes componentes principales(Ver figura 2.3):

- Una estructura propia.
- Un sistema de transmisión mecánico.
- Un controlador.
- Un motor izquierdo, un motor derecho.
- Baterías.
- Interfaces de Control y Operación. Usualmente, la interface de operación usa un joystick como dispositivo de entrada para controlar la operación de los motores.



**Figura 2.3** Esquema de Silla Motorizada.

## 2.2 Redes Neuronales

### 2.2.1 Introducción

Las redes neuronales representan modelos simples del sistema nervioso central; son un conjunto de elementos altamente interconectados que tienen la habilidad de responder simultáneamente a distintas entradas y aprender en entornos cambiantes.

Las redes neuronales artificiales han demostrado ser efectivas como procesos computacionales en varias tareas[Raquel Flóres López, 2009], como por ejemplo, en el reconocimiento de patrones.

Estas exponen un gran número de características deseables, algunas de las cuales no se encuentran en los sistemas convencionales. A continuación se enumeran estas propiedades:

- Robustez y tolerancia a fallas.
- Posibilidad de manejar información difusa, con ruido, incompleta o inconsistente.
- Alto grado de paralelismo.
- Capacidad de generalizar.
- Aprendizaje adaptativo.



### 2.2.2 Los Elementos principales de las RNA

Las Redes Neuronales Artificiales o RNAs, en inglés Artificial Neuronal Networks o ANNs, son modelos computacionales que surgieron como intento de conseguir formalizaciones matemáticas acerca de la estructura y el comportamiento del cerebro humano. Se basan en el aprendizaje a través de la experiencia, con la consiguiente extracción del conocimiento a partir de la misma.

El fin perseguido por una RNA es la emulación del sistema central biológico a través de procesadores artificiales, que incluso permitan evitar fallas o errores humanos. Así una RNA puede considerarse como un modelo de las actividades mentales, basado en la explotación del procesamiento local en paralelo y en las propiedades de la representación distribuida.

Los elementos básicos de un sistema neuronal biológico son las neuronas, agrupadas en redes compuestas por millones de ellas y organizadas a través de una estructura de capas. En un sistema neuronal artificial puede establecerse una estructura jerárquica similar, de forma que una RNA puede concebirse como una colección de procesadores elementales (neuronas artificiales), conectados entre sí o bien a entradas externas y con una salida que permite propagar la señal por múltiples caminos. Un conjunto de neuronas artificiales, tales que sus entradas provienen de la misma fuente y sus salidas se dirigen al mismo destino, conforma lo que se denomina capa o nivel.[Raquel Flóres López, 2009]

La agrupación de estos conjuntos constituye el sistema neuronal completo.

Cada procesador pondera las entradas que recibe. La modificación de estas ponderaciones es la clave del aprendizaje de la red. De esta forma, la red neuronal artificial aprende de sus propios errores a través de un procedimiento inductivo basado en la presentación de un conjunto de patrones informativos que permiten al sistema la generalización de conceptos a partir de casos particulares.

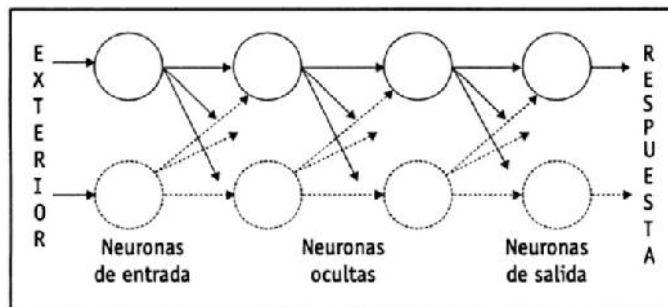
Una red neuronal puede definirse como un grafo dirigido con las siguientes propiedades:

- A cada nodo  $j$  se le asocia una variable de estado  $x_j$ .
- A cada conexión  $(i,j)$ , entre los nodos  $i$  y  $j$ , se le asocia un peso  $w_{i,j} \in R$ .

- En muchos casos a cada nodo se le asocia un umbral de disparo  $\theta_j$ .
- Para todo nodo  $j$  se define una función  $f_j(x_i, w_{i,j}, \theta_j)$  que depende del estado de todos los nodos unidos a él, de los pesos de sus conexiones y del umbral de activación para proporcionar un nuevo estado.

Considerando el lenguaje habitual de los grafos pueden establecerse las siguientes equivalencias:(Ver figura 2.4)

- Un nodo se representa mediante una neurona.
- Una conexión se representa mediante una sinapsis.
- Una neurona de entrada es aquella sin conexiones entrantes.
- Una neurona de salida es aquella sin conexiones salientes.
- Las neuronas que no son de entrada ni de salida se denominan neuronas ocultas.

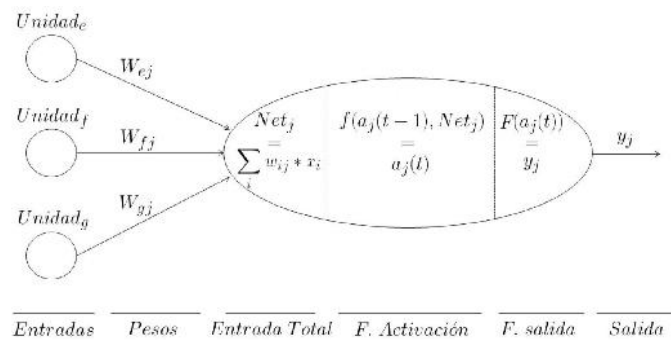


**Figura 2.4** Red Neuronal y sus tipos.

Tratando de mimetizar las características más relevantes de las neuronas biológicas, cada neurona artificial se caracteriza por los siguientes elementos:[Haykin, 2009] (Figura 2.5)

- Un valor o estado de activación inicial  $a_{t-1}$  anterior a la recepción de estímulos.
- Unos estímulos o entradas a la neurona  $(x_j)$ , con unos pesos asociados  $w_{i,j}$ .
- Una función de propagación que determina la entrada total a la neurona  $Net_j$ .
- Una función de activación o de transferencia  $f$  que combina las entradas a la neurona con el estado de activación inicial para producir un nuevo valor de activación.

- Una función de salida ( $F$ ), que transforma el estado final de activación en la señal de salida
- Una señal de salida que se transmite, en su caso, a otras neuronas artificiales  $y_i$ .
- Una regla de aprendizaje, que determina la forma de actualización de los pesos de la red.



**Figura 2.5** Modelo de Neurona Artificial.

### 2.2.3 La Arquitectura de las RNAs

La topología o arquitectura de una RNA hace referencia a la organización y disposición de las neuronas de la red y a las conexiones entre ellas. La arquitectura de una red neuronal depende de cuatro parámetros principales:

1. El número de capas del sistema.
2. El número de neuronas por capa.
3. El grado de conectividad entre las neuronas.
4. El tipo de conexiones neuronales.

Las arquitecturas neuronales pueden clasificarse de acuerdo a distintos criterios que se detallan a continuación.

#### **Clasificación de RNA según su estructura en capas**

- *Redes monocapas*: Compuestas por una única capa de neuronas entre las cuales se establecen conexiones laterales y en ocasiones recurrentes. Este tipo de redes suele utilizarse para la resolución de problemas de auto asociación y clusterización.

- *Redes multicapa* (layered networks): Se corresponde con las RNAs cuyas neuronas se organizan en varias capas: de entrada, oculta(s) y de salida. La capa a la que pertenece una neurona puede distinguirse mediante la observación del origen de las señales que recibe y el destino de la señal que genera.

### **Clasificación de las RNA según el flujo de datos de la red**

- *Redes unidireccionales o de propagación hacia adelante (feedforward)* : En éstas, ninguna salida neuronal es entrada de unidades de la misma capa o de capas precedentes. Por lo tanto, la información circula en un único sentido: desde las neuronas de entrada hacia las neuronas de salida de la red.
- *Redes de propagación hacia atrás (feedback)* : En éstas las salidas de las neuronas pueden servir de entradas a unidades del mismo nivel (conexiones laterales), o de niveles previos. Las redes de propagación hacia atrás que presentan lazos cerrados se denominan sistemas recurrentes.

### **Clasificación de las RNA según el grado de conexión**

- *Redes neuronales totalmente conectadas* : En este caso cada una de las neuronas de una capa se encuentran conectadas con todas las neuronas de la capa siguiente (redes no recurrentes), o con todas las neuronas de la capa anterior (redes recurrentes).
- *Redes neuronales parcialmente conectadas* : En este caso no se da la conexión total entre neuronas de diferentes capas.

### **Clasificación de las RNA según el tipo de respuesta de red**

- *Redes heteroasociativas* : Redes entrenadas para que ante la presentación de un determinado patrón A, el sistema responda con otro diferente B. Estas RNAs precisan al menos dos capas: una para captar y retener la información de entrada y otra para mantener la salida con la información asociada. Las redes heteroasociativas pueden clasificarse a su vez, según el objetivo pretendido con su utilización, distinguiéndose las RNAs destinadas a computar una función matemática a partir de las entradas que reciben, las redes utilizadas para tarea de clasificación y las redes empleadas para la asociación de patrones, entre otras.

- *Redes autoasociativas* : Redes entrenadas para que se asocie un patrón consigo mismo. Su interés reside en que, ante la presentación de un patrón Afectado por ruido, su respuesta sea el patrón original A. Estas redes pueden implementarse con una única capa de neuronas que comenzará reteniendo la información de entrada y terminará representando la información autoasociada. Si se desea mantener la información de entrada y salida, deberán añadirse capas adicionales. Estos modelos suelen emplearse en tareas de filtrado de información, para analizar las relaciones de vecindad entre los datos considerados ( clustering ) y para resolver problemas de optimización.

## 2.3 Python

### 2.3.1 Introducción

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible. Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos.[Duque, 2013]

Un lenguaje interpretado o de script es aquel que se ejecuta utilizando un programa intermedio llamado intérprete, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados).

La ventaja de los lenguajes compilados es que su ejecución es más rápida. Sin embargo los lenguajes interpretados son más flexibles y más portables.

### 2.3.2 Instalación de Python

Existen varias implementaciones distintas de Python: CPython, Jython, IronPython, PyPy, etc.

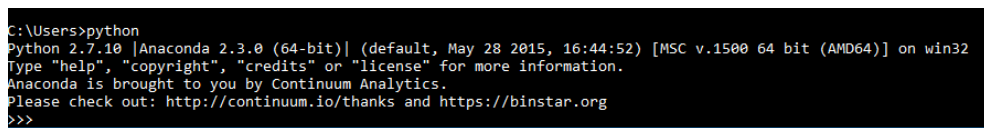
CPython es la más utilizada, la más rápida y la más madura. Cuando la gente habla de Python normalmente se refiere a esta implementación. En este caso tanto el intérprete como los módulos están escritos en C.

Jython es la implementación en Java de Python, mientras que IronPython es su contrapartida en C# (.NET). Su interés estriba en que utilizando estas implementaciones se pueden utilizar todas las librerías disponibles para los programadores de Java y .NET.

PyPy, por último, como habréis adivinado por el nombre, se trata de una implementación en Python de Python.

CPython está instalado por defecto en la mayor parte de las distribuciones Linux y en las últimas versiones de Mac OS. En Windows se puede instalar una Anaconda que permite tener una versión funcional con la mayoría de paquetes de uso común.

Para comprobar si está instalado abre una terminal y escribe python. Si está instalado se iniciará la consola interactiva de Python y obtendremos parecido a lo que muestra la figura 2.6



```
C:\Users>python
Python 2.7.10 [Anaconda 2.3.0 (64-bit)] (default, May 28 2015, 16:44:52) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://binstar.org
>>>
```

**Figura 2.6** Python en Windows usando Anaconda.

La primera línea nos indica la versión de Python que tenemos instalada. Al final podemos ver el prompt (> > >) que nos indica que el intérprete está esperando código del usuario. Podemos salir escribiendo exit(), o pulsando Control + D.

### 2.3.3 Primeros Pasos con Python

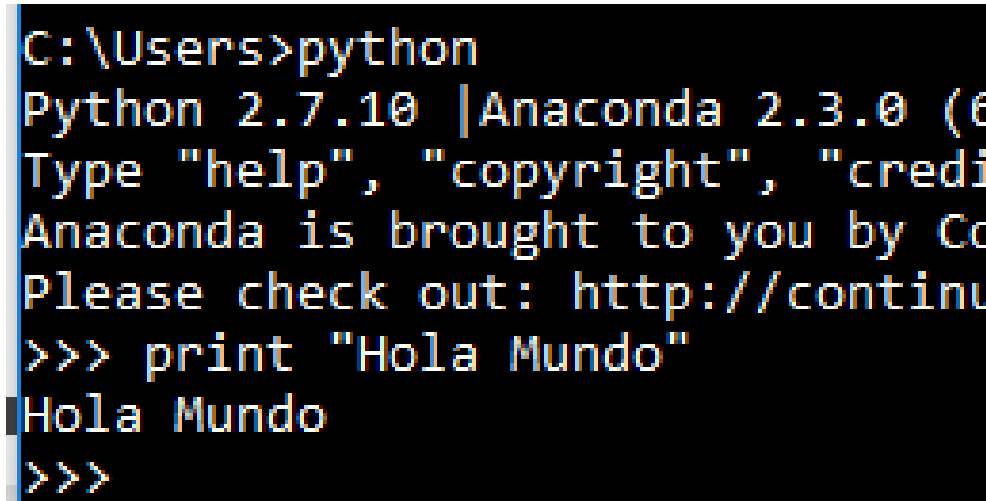
Existen dos formas de ejecutar código Python, bien en una sesión interactiva (línea a línea) con el intérprete, o bien de la forma habitual, escribiendo el código en un archivo de código fuente y ejecutándolo.

El primer programa que vamos a escribir en Python es el clásico Hola Mundo, y en este lenguaje es tan simple como:

```
print "Hola Mundo"
```

Vamos a probarlo primero en el intérprete. Ejecuta python o ipython según tus preferencias, escribe la línea anterior y pulsa Enter. (Figura 2.7)

El intérprete responderá mostrando en la consola el texto Hola Mundo.

A screenshot of a terminal window with a black background and white text. The prompt 'C:\Users>' is followed by the command 'python'. The output shows 'Python 2.7.10 |Anaconda 2.3.0 (64-bit)|' and a welcome message from Anaconda. Below this, the command '>>> print "Hola Mundo"' is entered, and the output 'Hola Mundo' is displayed. The prompt '>>>' appears again at the bottom.

```
C:\Users>python
Python 2.7.10 |Anaconda 2.3.0 (64-bit)|
Type "help", "copyright", "credits()" or "quit()" for more
Anaconda is brought to you by Continuum Analytics, Inc.
Please check out: http://continuum.io
>>> print "Hola Mundo"
Hola Mundo
>>>
```

**Figura 2.7** Python en la consola.

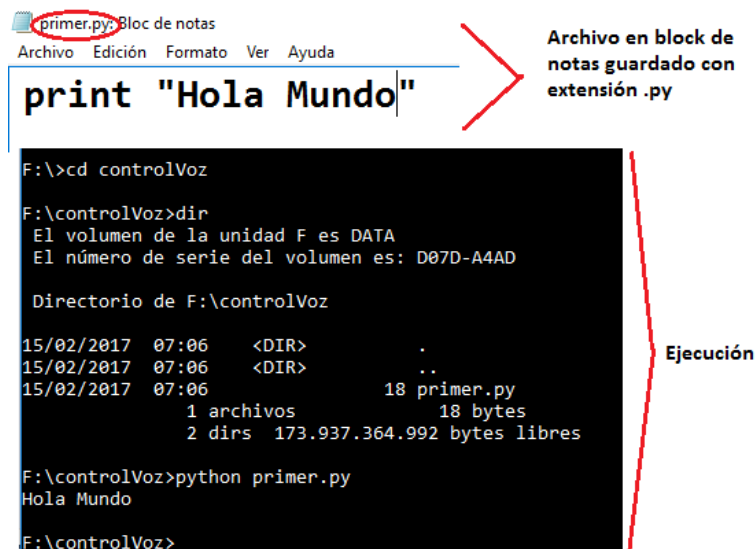
Vamos ahora a crear un archivo de texto con el código anterior, de forma que pudiéramos distribuir nuestro pequeño gran programa entre nuestros amigos.

Abre tu editor de texto preferido o bien el IDE, nosotros los hacemos en el block de notas y nos aseguramos de guardar usando la extensión .py.

Ejecutar este programa es tan sencillo como indicarle el nombre del archivo a ejecutar al intérprete de Python `python primer.py` (Ver el proceso en la figura 2.8)

## 2.4 Arduino

Es una plataforma abierta de computación física basada en una tarjeta de entrada/salida y ambiente de desarrollo que implementa el lenguaje Processing ([www.processing.org](http://www.processing.org)). Arduino puede ser usado para desarrollar proyectos interactivos o pueden ser conectados al software de una PC o Laptop. Tanto la plataforma de Hardware como de Software son abiertos y pueden ser descargados libremente.[Banzi and Shiloh, 2014]. Las características principales que han hecho de Arduino la plataforma preferida de muchos es:



**Figura 2.8** Python con Editor.

- Es un ambiente multiplataforma; es decir puede correr en Windows, Macintosh, y Linux.
- El IDE está basado en Lenguaje Processing, un ambiente de desarrollo de fácil uso de diseñadores y artistas.
- La programación se realiza vía cable USB, no por puerto serial. Esta característica es útil puesto que las computadoras modernas no tienen puerto serial.
- Es una plataforma de Hardware y Software abierta.
- El hardware es barato.
- Hay una comunidad muy activa de usuarios.
- El Proyecto Arduino fue desarrollado en un ambiente educacional y por consiguiente es fácil iniciar su aprendizaje.

La filosofía de Arduino se basa en hacer diseños en lugar de hablar de ellos. Es una búsqueda constante de rapidez y maneras más potentes de construir mejores prototipos[Banzi and Shiloh, 2014].

### 2.4.1 La plataforma Arduino

Esta compuesta de la tarjeta Arduino, que es una pieza de hardware, y el IDE Arduino, la pieza de software que corre en una computadora.



La Tarjeta Arduino

Arduino Uno y Genuino Uno [ard, 2017b]

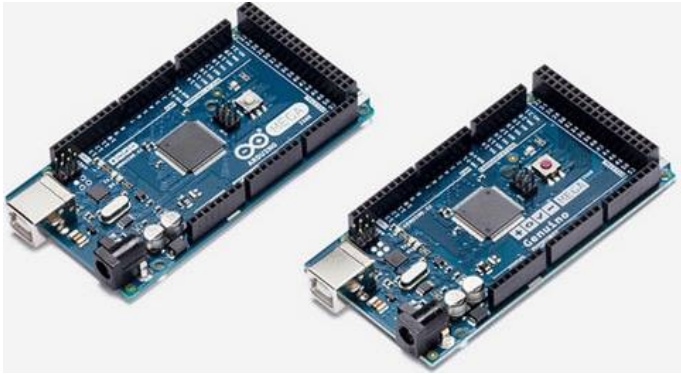


Figura 2.9 Tarjetas Arduino Uno y Genuino Uno.

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Figura 2.10 Especificaciones de Arduino Uno y Genuino Uno.

Arduino Mega 2560 y Genuino Mega 2560 [ard, 2017a]



**Figura 2.11** Tarjetas Arduino Mega 2560 y Genuino Mega 2560.

Microcontroller	Atmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm
Width	53.3 mm
Weight	37 g

**Figura 2.12** Especificaciones de Arduino Mega 2560 y Genuino Mega 2560.

**La IDE Arduino**

La IDE Arduino puede ser descargada libremente de <https://www.arduino.cc/en/Main/Software> donde podemos seleccionar diferentes opciones tales como un editor WEB, o la IDE standalone que actualmente esta en su versión 1.8.1(Ver figura 2.13) y se puede seleccionar para instalar en Windows, Linux o Mac OS X.



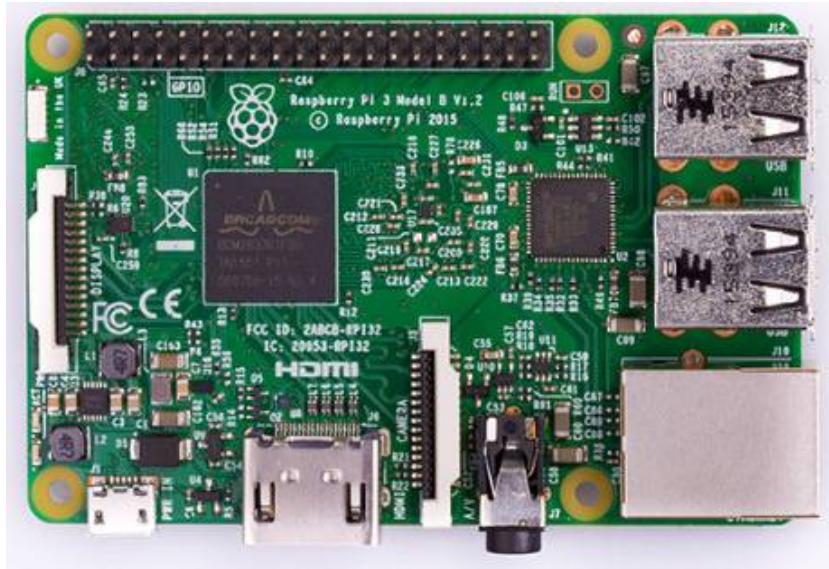
**Figura 2.13** Descarga de Arduino IDE.

## 2.5 Raspberry

Raspberry Pi es un computador de placa reducida, computador de placa única o computador de placa simple (SBC) de bajo coste desarrollado en Reino Unido por la Fundación Raspberry Pi, con el objetivo de estimular la enseñanza de ciencias de la computación en las escuelas[Wikipedia, 2017a].

El software es open source, siendo su sistema operativo oficial una versión adaptada de Debian, denominada Raspbian, aunque permite otros sistemas operativos, incluido una versión de Windows 10. El diseño incluye un System-on-a-chip Broadcom BCM2835, que contiene un procesador central (CPU) ARM1176JZF-S a 700 MHz (el firmware incluye unos modos “Turbo” para que el usuario pueda hacerle overclock de hasta 1 GHz sin perder la garantía), un procesador gráfico (GPU) VideoCore IV, y 512 MB de memoria RAM (aunque originalmente al ser lanzado eran 256 MB). El diseño no incluye un disco duro ni unidad de estado sólido, ya que usa una tarjeta SD para el almacenamiento permanente; tampoco incluye fuente de alimentación ni carcasa. El 29 de febrero de 2012 la fundación empezó a aceptar órdenes de compra del modelo B, y el 4 de febrero de 2013 del modelo A.

Actualmente están en su tercera generación con la Raspberry Pi 3 modelo B (ver figura 2.14).



**Figura 2.14** Raspberry Pi 3 Modelo B.

Sus características son:

- Procesador Quad-core de 64 bit ARMv8 CPU
- Lan Inalámbrica 802.11.
- Bluetooth 4.1.
- Bluetooth Low Energy (BLE).
- 1GB Ram.
- 4 puertos USB.
- 40 puertos GPIO.
- Puerto full HDMI.
- Puerto Ethernet.
- Entrada combinada de audio de 3.5mm.
- Interfaces para cámara.

El Raspberry Pi usa mayoritariamente sistemas operativos GNU/Linux. Raspbian, una distribución derivada de Debian que está optimizada para el hardware de Raspberry Pi,

se lanzó durante julio de 2012 y es la distribución recomendada por la fundación para iniciarse.

## 2.6 Sistemas de Reconocimiento Automático de voz - ASR

Es una importante tecnología para habilitar y mejorar las interacciones entre humanos y humano-computadora[Yu and Deng, 2014].

### 2.6.1 Comunicación Humano-Humano

En este caso sirve para remover barreras en las interacciones entre humanos. Así, en el pasado, las personas que hablan diferentes idiomas necesitan un humano interprete que sea capaz de hablar por los otros. Por ejemplo personas que no conocen China, a menudo es difícil que viajen a China solos. Actualmente existen dispositivos de traslación speech-to-speech, como el dispositivo *Travis the Translator*, presentado en el “Mobile World Congress” un dispositivo, que según sus creadores, traduce hasta 80 idiomas en tiempo real y a viva voz.(Ver figura 2.15)



**Figura 2.15** Traductor Travis.

Otras aplicaciones incluyen sistemas unificados de mensajes, con sub-sistemas de transcripción de voz que pueden ser usados para convertir voz a mensajes y estos fácilmente enviados a los correos electrónicos. En otros ejemplos, ASR puede ser usado para dictar mensajes cortos para reducir el esfuerzo en el envío de mensajes cortos.

### **2.6.2 Comunicación Humano-Computadora**

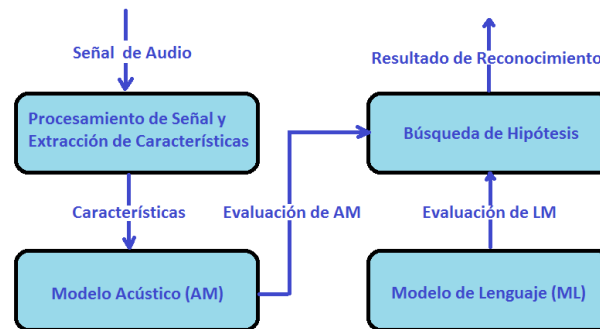
En este ramo las aplicaciones más populares son búsquedas por voz, asistente digital personal, juegos, sistemas de interacción en el hogar, Sistemas de información y entretenimiento en el vehículo[Yu and Deng, 2014].

- Búsquedas por voz: Estas aplicaciones permiten a los usuarios búsqueda de información, tales como restaurantes, direcciones y revisión de productos. Estas aplicaciones son muy populares en dispositivos móviles tales como iPhone, teléfonos con Windows y Android.
- Asistente digital personal (PDA): Conocen la información almacenada en nuestros dispositivos móviles, el historial de interacciones con el sistema y pueden servir mejor a los usuarios.
- Juegos: La experiencia de Juego puede ser mejorada si en los juegos son integrados con tecnologías de voz.
- Sistemas de interacción en el hogar, Sistemas de información y entretenimiento en el vehículo: Son de funcionalidad similar. Estos sistemas permiten al usuario interactuar con los sistemas a través de la voz, pudiendo reproducir música, preguntar por información o controlar el sistema.

### **2.6.3 Arquitectura de los Sistemas ASR**

Está compuesta de 04 componentes principales: Procesamiento de Señal y extracción de características, Modelo Acústico (AM), Modelos del Lenguaje (ML) y búsqueda de hipótesis(Ver figura 2.16).

La componente de procesamiento de señal y extracción de características, tomo como entrada una señal de audio, mejora la voz removiendo el ruido y distorsiones del canal, convierte la señal del dominio del tiempo al dominio de la frecuencia, y extrae vectores de características resaltantes disponibles para el Modelo Acústico. El Modelo Acústico integra conocimiento acerca de acústica y fonética, toma como entrada las características generadas desde el bloque de extracción de características, y genera una Evaluación de



**Figura 2.16** Arquitectura de los Sistemas ASR.

AM. El Modelo del Lenguaje estima la probabilidad de la palabra u oración, esta es la evaluación de LM, por un aprendizaje de la correlación entre palabras desde un conjunto de entrenamiento. La Búsqueda de Hipótesis combina las evaluaciones de AM y LM, es decir los puntajes dadas la secuencia vectorial característica y la secuencia de palabras hipotética y como salida la palabra u oración con la evaluación más alta, como resultado del reconocimiento[Yu and Deng, 2014].

## 2.7 Movi de Audeme

MOVI significa *My Own Voice Interface*, que se puede traducir como *Mi propia Interface de voz*. Es el primer reconocedor y sintetizador autónomo de voz para Arduino, aunque también se interconecta a Raspberry Pi.(Ver figura 2.17)



**Figura 2.17** Movi de Audeme.

Sus características principales son:

- Mucho espacio para frases personalizable en ingles(han sido probados hasta 200, se ha reportado hasta 1000).
- Independiente del hablante.
- Independiente, privado y funciona sin conexión.
- Fácil de programar.
- Diferentes, configurables sintetizadores de voz incluidos.



## 3 Desarrollo de la Ingeniería del Proyecto

---

*Un buen científico es una persona con ideas originales. Un buen Ingeniero es una persona que hace diseños que funcionan con mínimo posible de ideas originales.*

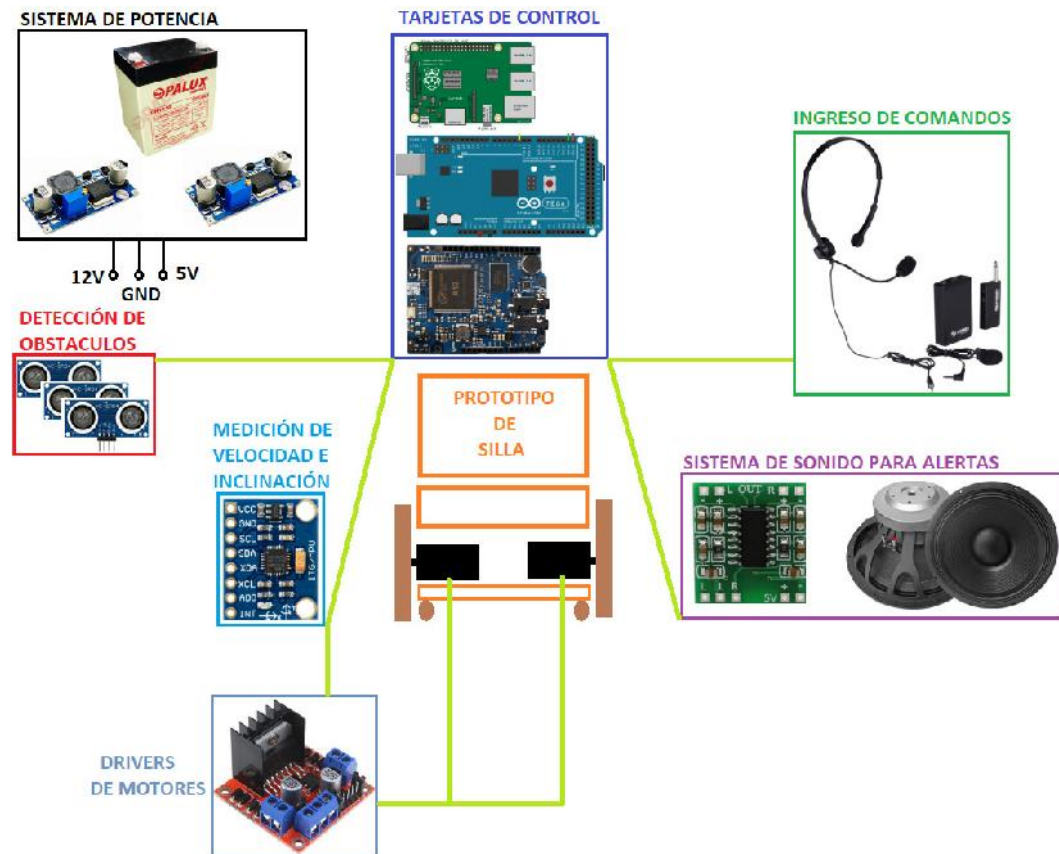
FREEMAN DYSON

### 3.1 Descripción del Proyecto

Después de realizar la revisión de las principales alternativas y opciones que deberíamos incluir en nuestro sistema de navegación automático, se optó por la solución mostrada en la figura 3.1, donde se ve claramente los diferentes bloques que conforman este proyecto, cuya integración será detallada en este capítulo.

Los bloques son los siguientes:

1. Prototipo de Silla de Ruedas: Se construirá un prototipo de silla de ruedas constituido de una estructura de madera y para la locomoción contará con dos motores de DC.



**Figura 3.1** Componentes del Sistema de Navegación controlado por Voz.

2. Tarjetas de Control: Constituidos por la micro computadora de propósito general Raspberry Pi 3, una tarjeta para adquisición de datos que en este caso se usará la Arduino Mega 2560 y una tarjeta para el procesamiento y síntesis de los controles de voz, constituido por la tarjeta Audeme MOVI.
3. Ingreso de Comandos: Se instalará un micrófono inalámbrico, tipo solapero y/o vincha que facilita la adquisición de los comandos de voz.
4. Sistema de Potencia: Representado por una batería de 12 / 4AH, con tarjetas convertidores DC-DC Step Down del que se puede obtener cualquier voltaje menor a 10V y mayor a 1.2V, con una corriente máxima de 3A y eficiencia del 92%.
5. Detección de obstáculos: Un conjunto de sensores de ultrasonidos permitirá detectar obstáculos que posiblemente no puedan ser observados por el usuario.
6. Medición de Velocidad e Inclinación: Se usará un acelerómetro y giroscopio que

permitirá regular la velocidad y el giro de la silla dentro de límites considerados seguros para el usuario.

7. Sistema de Sonido para alertas: Se incluye un amplificador de audio de 3W con un sistema de parlantes para que se realice una interacción amigable con el sistema de navegación.
8. Drivers para los motores: Este bloque facilita la interconexión de los motores a las salidas digitales de la tarjeta de adquisición de datos.

## 3.2 Construcción de Prototipo

Debido al costo de las sillas de ruedas motorizadas se ha optado por realizar el control por voz sobre una plataforma que tiene la misma arquitectura de una silla de ruedas, es decir; un motor izquierdo y un motor derecho y un par de ruedas de giro libre (ver figura 3.2)

En la figura 3.3, se observa la plataforma lista y con elementos mecánicos que permitan colocar las tarjetas electrónicas, sensores y baterías que permitirán el diseño del control por voz.

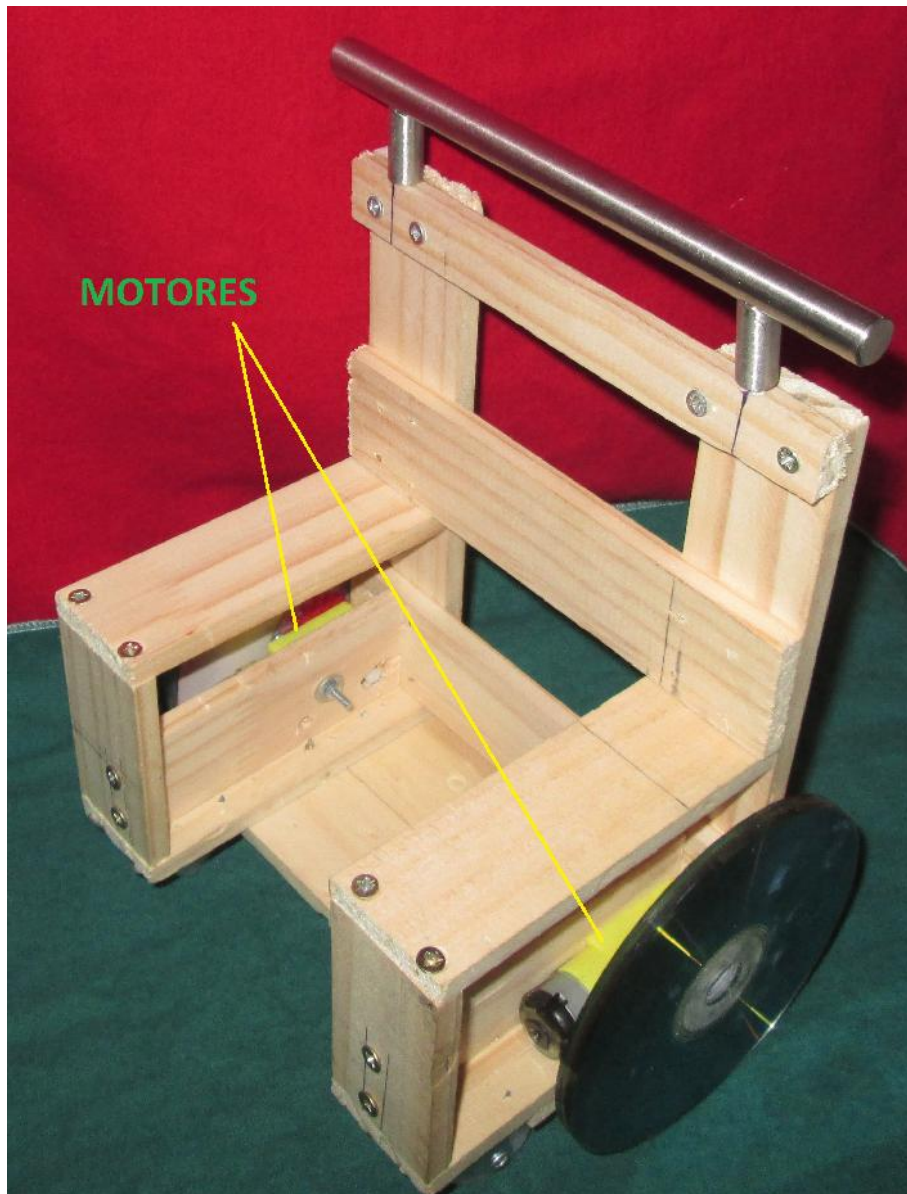
Con este primer prototipo desarrollado se tuvieron dificultades para un adecuado control, por lo que se optó por el desarrollo de otro prototipo de silla de ruedas usando motores de 12V DC 15RPM, con el que se tiene un control más preciso y seguro de las funciones de la silla de ruedas. En la figura 3.4 y 3.5 se ve el prototipo con los nuevos motores.

El diámetro de las ruedas, en este caso CDs es de 12.5cm o 0.125m, lo que nos permite calcular la velocidad desarrollada por este prototipo

$$Diametro = 2\pi 0.125m = 0.785m$$

$$V(m/s) = 0.2m/s$$

En el caso del modelo real las ruedas tienen 0.6m, la velocidad sería



**Figura 3.2** Prototipo de Silla de Ruedas - Motor Izquierdo y Derecho.

$$Diametro = 2\pi 0.125m = 0.785m$$

$$V(m/s) = 1m/s$$

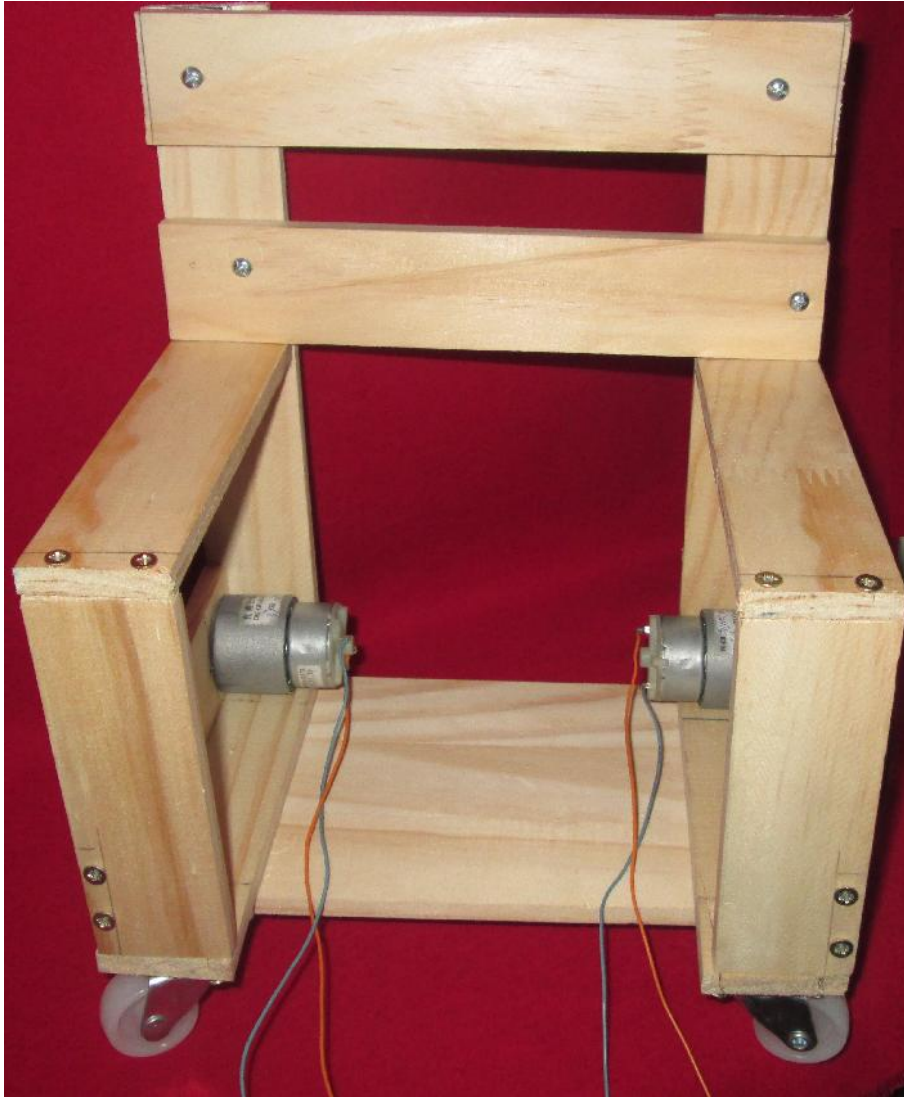
Esta velocidad es relativamente alta, pero recordemos que este es caso de máxima velocidad, las velocidades obtenidas serán menores a este valor pues se realizara un control



**Figura 3.3** Prototipo de Silla de Ruedas.

de potencia de los motores usando PWM y en consecuencia se regulara la velocidad del prototipo de acuerdo a las condiciones de manejo del mismo. Con esto se ha querido mostrar que el prototipo tiene las condiciones de seguridad para un manejo adecuado ya que mecánicamente está limitado a una velocidad segura.





**Figura 3.4** Prototipo de Silla de Ruedas - Motor Izquierdo y Derecho.

### 3.3 Diseño del algoritmo de Control por Voz

El diseño del algoritmo de control por voz, se realiza usando aprendizaje profundo con redes neuronales (Deep Learning), y aprendizaje supervisado para lo cual seguiremos la siguiente metodología:[Géron, 2017]

1. Obtención de las tablas de entrenamiento.
2. Definir la estructura de la red neuronal.
3. Inicializar los parámetros de la red neuronal.



**Figura 3.5** Prototipo de Silla de Ruedas.

**4.** Lazo de entrenamiento:

- a) Implementar propagación hacia adelante.
- b) Calcular los errores.
- c) Implementar el algoritmo de propagación hacia atrás.
- d) Actualizar parámetros

Ahora pasaremos a detallar cada uno de los pasos mencionados.

### 3.3.1 Obtención de las tablas de entrenamiento

Al tratarse de un aprendizaje profundo con redes neuronales, y aprendizaje supervisado, es necesario contar con las tablas de entradas y salidas deseadas.

En este caso de control por voz, la tabla consiste, en principio; de una orden grabada y la consecuente activación de la salida correspondiente, como se muestra en la figura 3.6, es decir, las ordenes de control rojo que corresponden a lo mismo solo activan la salida en la primera columna, de la misma forma las ordenes de color azul activaran la salida en la segunda columna y así sucesivamente. Un dato importante es tener las dimensiones de esta tabla, estas dimensiones involucran lo siguiente:

- Número de ejemplos totales. En este caso sería número de filas.
- Las dimensiones de cada orden de entrada, que dependen de la naturaleza de la entrada y del pre-procesamiento realizado. Explicaremos detalladamente este proceso.
- Número de salidas, es decir el numero de columnas de SALIDAS, en la tabla de la figura 3.6

Es importante indicar que usualmente las tablas obtenidas se dividen en dos partes, como en este caso se tiene un valor pequeño de ejemplos se divide en :

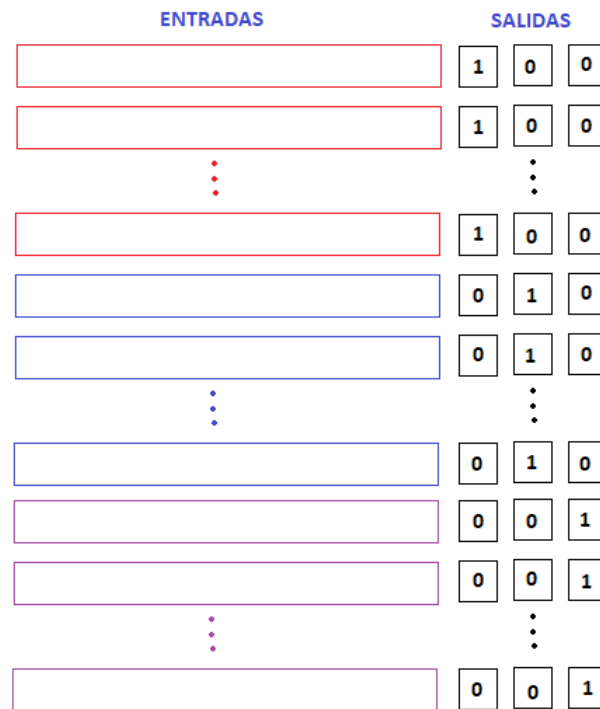
- 70 % para entrenamiento.
- 30 % para validación.

La recomendación consiste en que una vez obtenidas las tablas se desordenan para que básicamente las ordenes se distribuyan uniformemente en toda las filas de la tabla y finalmente se separan en los porcentajes indicados de entrenamiento y validación.

#### **Procesamiento de Señales de Entrada - Ordenes grabadas**

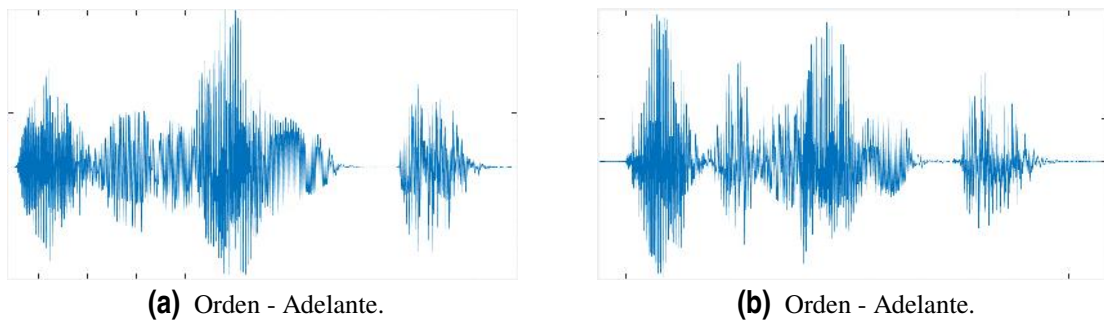
Las ordenes grabadas consisten de señales de voz del comando que se pretenda la red neuronal aprenda, en las figuras 3.7, 3.8, 3.9, 3.10 se observa dos versiones de algunas ordenes que fueron grabadas usando una velocidad de muestreo de 44200 muestras por segundo, grabada durante un segundo. En todas ellas se puede notar ciertas similitudes,





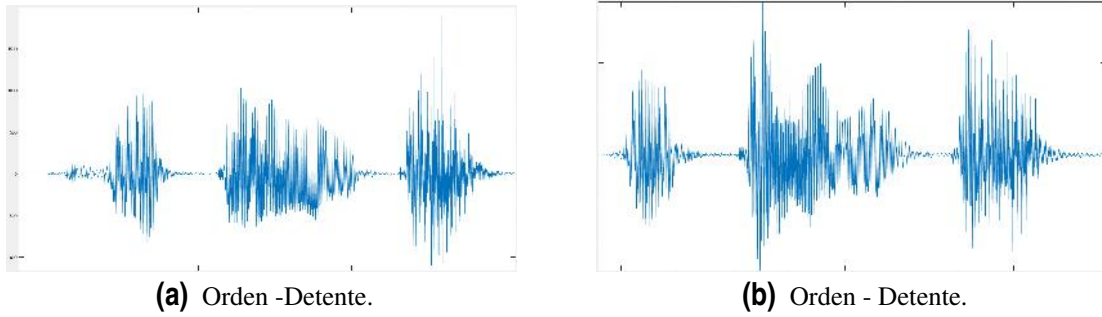
**Figura 3.6** Intuición de Tabla de entrada salidas para aprendizaje supervisado.

pero aún así es difícil poder clasificarlas correctamente, debido a posibles diferencias temporales y a las grandes variaciones de amplitud de estas señales.

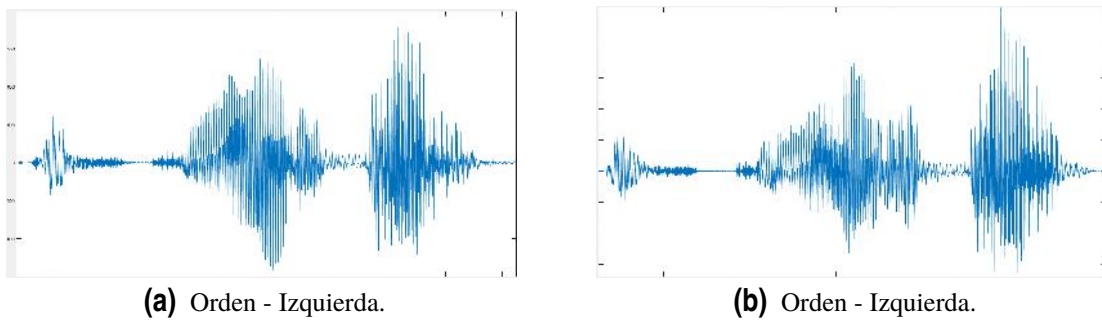


**Figura 3.7** Un segundo de una orden grabada..

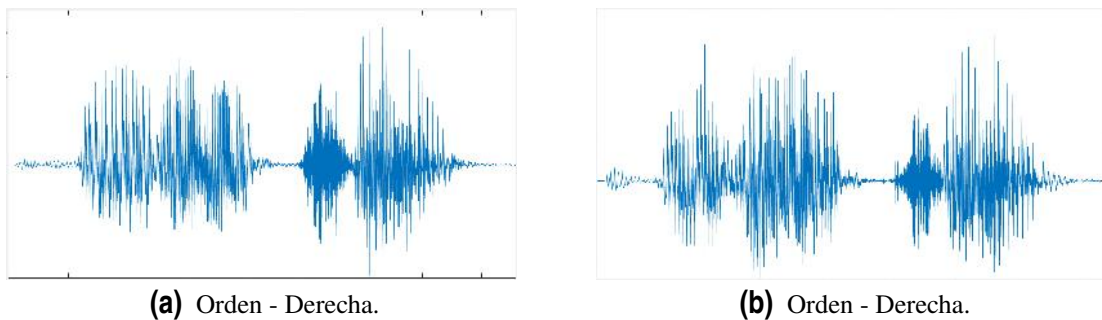
Una forma sencilla de caracterizar estas señales es la obtención de su representación el dominio de la frecuencia, esto se puede realizar aplicando la transformada rápida de fourier (**FFT- Fast Fourier Transform**), donde no es relevante la información temporal, si no mas bien el contenido de frecuencias de la señal. La herramienta usada en realidad es la densidad espectral de potencia (**PDF - Power Density Frequency**).



**Figura 3.8** Un segundo de una orden grabada..



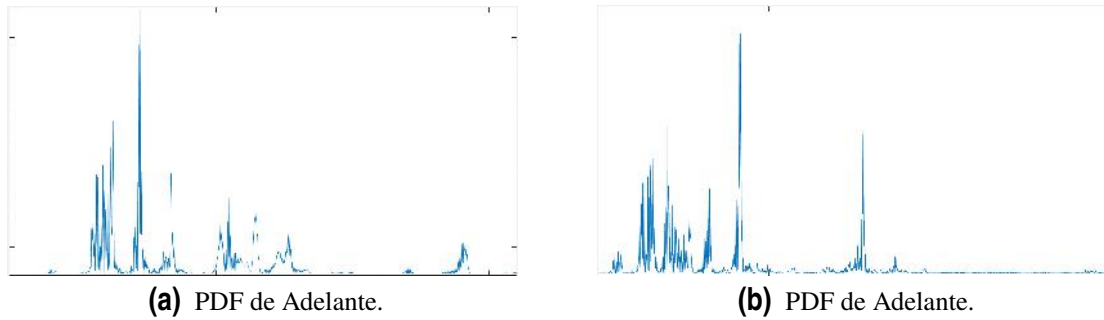
**Figura 3.9** Un segundo de una orden grabada..



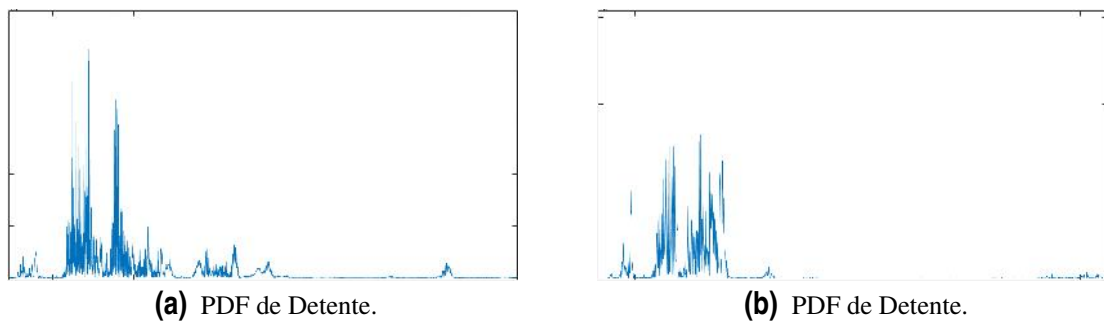
**Figura 3.10** Un segundo de una orden grabada..

En las figuras 3.11, 3.12, 3.13, y 3.14 se puede ver el resultado de obtener los PDF de ocho señales de voz.

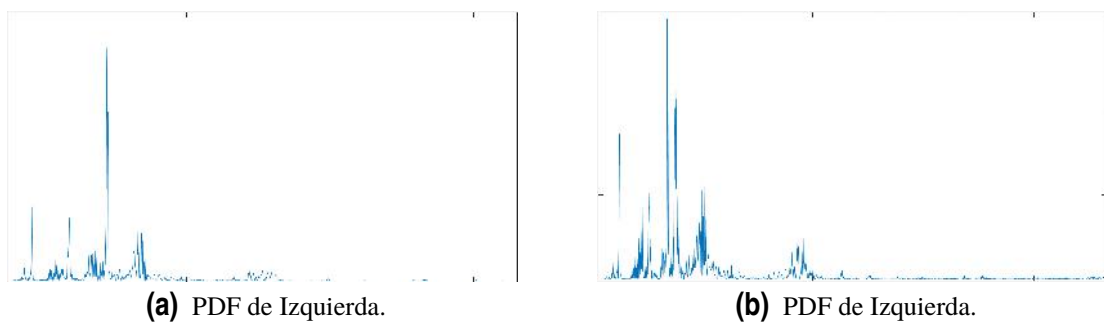
El resultado es de nuevo vectores que tendrán las mismas dimensiones de la señal grabada en el tiempo. Esto es un inconveniente puesto que diferentes valores de la señal de entrada tendrían diferentes dimensiones lo cual no es apropiado como entrada a la red neuronal. Para eso se usa un estimador de densidad espectral de potencia que consiste en dividir el intervalo de evaluación (acá no importa su valor), en 129 rangos (también denominadas



**Figura 3.11** Dos versiones de PDF de señales de entrada..



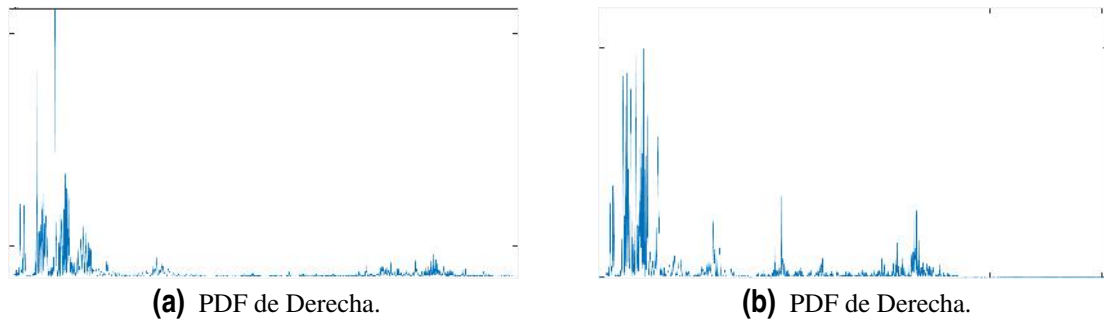
**Figura 3.12** Dos versiones de PDF de señales de entrada..



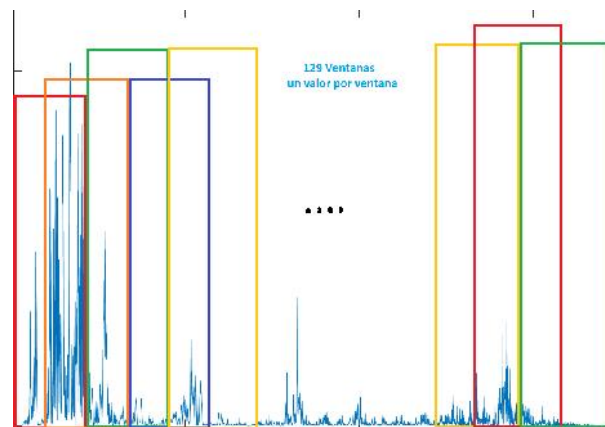
**Figura 3.13** Dos versiones de PDF de señales de entrada..

ventanas) y obtener un valor promedio por cada ventana, de esta manera terminaríamos con solo entradas de 129 valores, sin importar su duración. En la figura 3.15 se muestran intuitivamente como estas ventanas se traslapan y de esta manera se puede obtener un promedio del PDF de la señal.

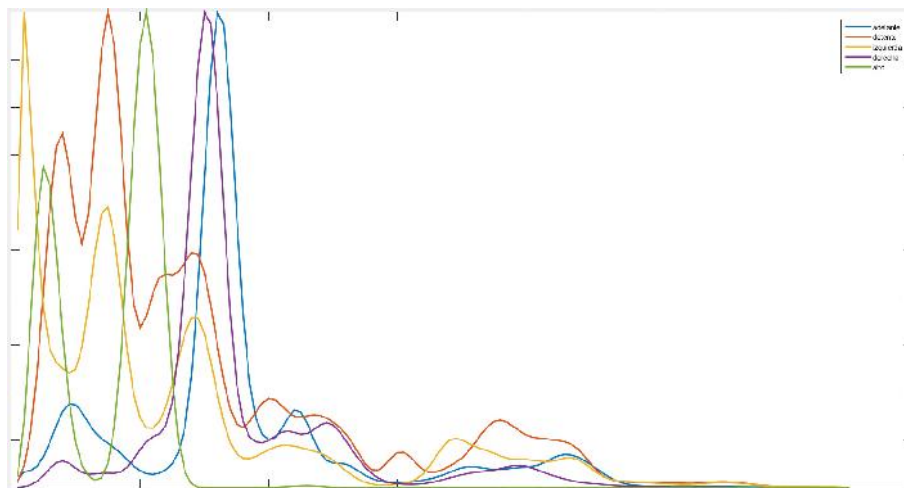
Luego de aplicar el estimador de PDF a las señales de entrada, se observa que entre las diferentes ordenes consideradas existen una diferencias que posibilita el entrenamiento de una red neuronal que permita clasificar correctamente las ordenes (Ver figura 3.16 )



**Figura 3.14** Dos versiones de PDF de señales de entrada..



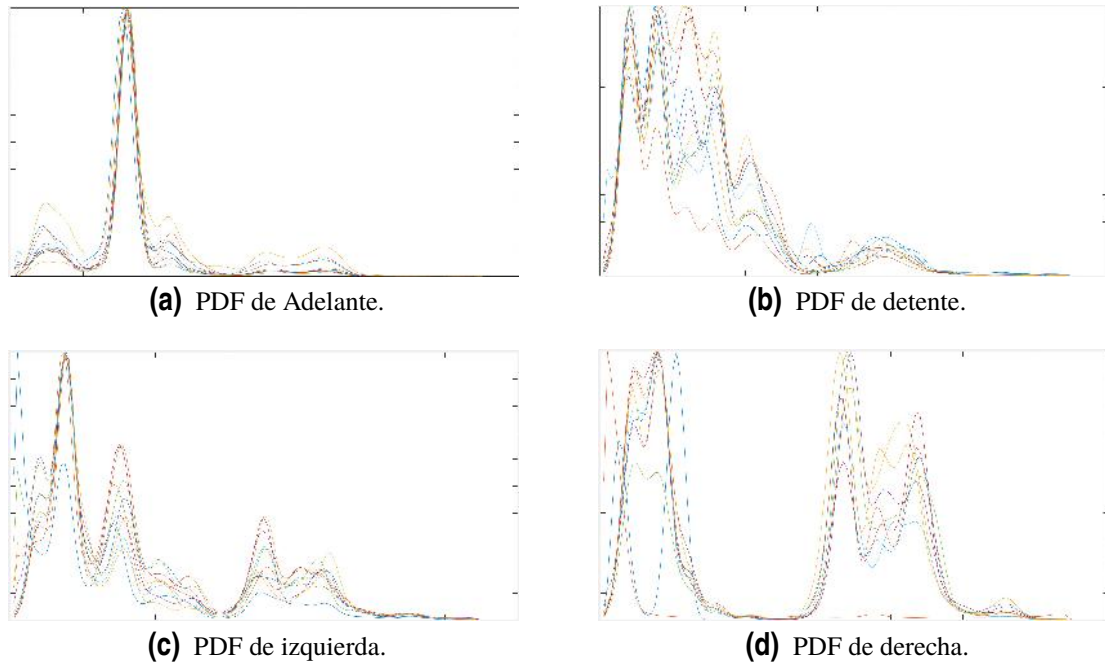
**Figura 3.15** Intuición de estimador de PDF.



**Figura 3.16** Estimación de PDF para cinco ordenes diferentes.

Podemos verificar también que para una misma orden la estimación del PDF son bastante similares (Ver figura 3.17)

Entonces, aplicando el estimador de PDF, descrito se obtiene finalmente las entradas



**Figura 3.17** Similitud de la estimación de PDF de señales de entrada para la misma señal.

para la red neuronal que consiste en vectores de 129 valores y la cantidad de ejemplos deber ser la máxima posible que se puede obtener. En nuestro caso nos fijamos la meta de tener 50 ordenes grabadas por comando de al menos diez de nuestros compañeros o familiares, lo que nos daría 500 ordenes por comando esto multiplicado por el numero de comandos proyectados.

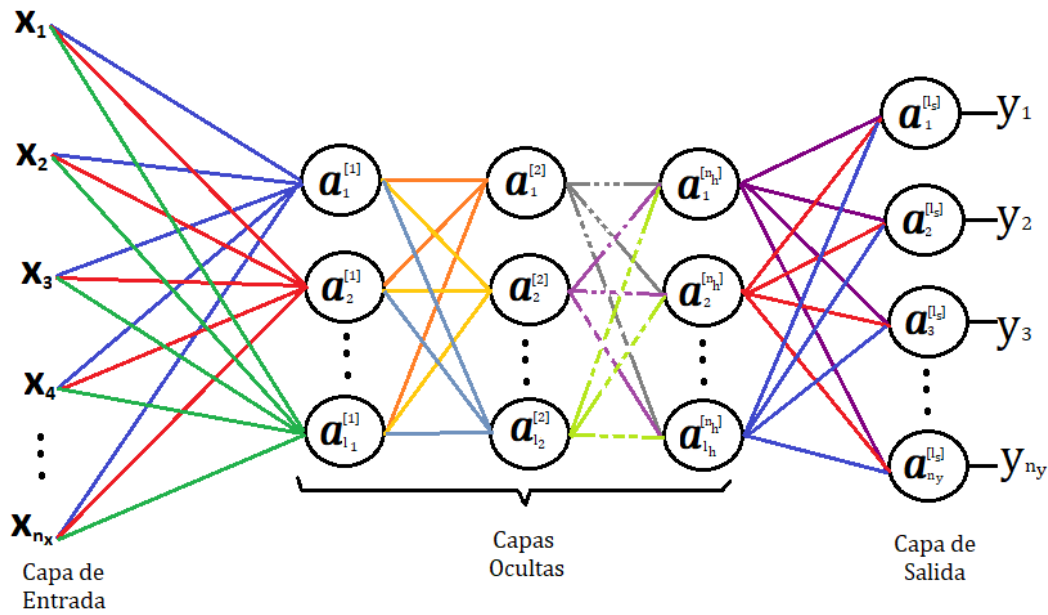
### 3.3.2 Definir la Estructura de la Red Neuronal

En la figura 3.18 se observa una red neuronal, donde se identifican claramente los parámetros de la red, tales como:

- El tamaño del vector de entrada, es decir la cantidad de elementos de cada muestra de entrada. En nuestro caso es determinado por la salida de estimador de Densidad Espectral de Potencia. Esto es  $n_x = 129$
- La cantidad de neuronas en cada capa oculta y la cantidad de capas ocultas, en este caso se entrega en la forma de un vector  $[l_1, l_2, l_h]$  donde  $h$  es el número de capas ocultas y  $l_1$  es la cantidad de neuronas en la primera capa oculta,  $l_2$  es la cantidad de neurona en la segunda capa oculta y  $l_h$  es la cantidad de neuronas en la capa  $h$ . Las

dimensiones de este arreglo serán establecidas de los experimentos y del desempeño obtenido.

- Un punto importante también es establecer las funciones de transferencia de las capas ocultas y de la capa de salida. Usualmente en las capas ocultas se usa  $\tanh(z)$  o  $\text{ReLU}(z)$  y en la capa salida se usa  $\text{sigmoide}(z)$
- La cantidad de neuronas en la capa de salida  $n_y$ , establecida por la cantidad de comandos que queremos reconocer.



**Figura 3.18** Estructura de la Red Neuronal.

Nuestro algoritmo debería recibir un arreglo, de la forma  $[n_x, l_1, l_2, l_h, n_y]$ , donde claramente se observa que se está indicando el tamaño de cada vector de entrada, con  $n_x$ , la cantidad de neuronas en cada capas oculta y además la cantidad de capas ocultas, con  $[l_1, l_2, l_h]$ , y la cantidad de capas en la salida, con  $n_y$ .

### 3.3.3 Inicializar Parámetros de la red neuronal

Para comprender los parámetros que tenemos que inicializar, mostraremos el proceso con un ejemplo. Se el vector recibido  $[4, 3, 4, 5]$ , del cual deducimos lo siguiente:

- Tamaño del vector de entrada = 4.

- Número de capas ocultas = 2.
- Número de neuronas en las capas ocultas.  $l_1 = 3$ ,  $l_2 = 4$ .
- Número de neuronas en la capa de salida = 5.

En la tabla 3.1 se muestra cuales son las dimensiones de todos los parámetros establecidos a partir del ejemplo y en el listado 3.1 se muestra la función en python creada para tal efecto.

**Tabla 3.1** Dimensiones de los parámetros para una red neuronal dada por [4, 3, 4, 5].

	Forma de W	Forma de b	Activación	Forma de Activación
Capa 1	$[l_1, n_x]$	$[l_1, 1]$	$Z^{[1]} = W^{[1]}X + b^{[1]}$	$[l_1, m]$
Capa 1	$[3, 4]$	$[3, 1]$	$Z^{[1]} = W^{[1]}X + b^{[1]}$	$[3, m]$
Capa 2	$[l_2, l_1]$	$[l_2, 1]$	$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]}$	$[l_2, m]$
Capa 2	$[4, 3]$	$[4, 1]$	$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]}$	$[4, m]$
Capa 3	$[l_3, l_2]$	$[l_3, 1]$	$Z^{[3]} = W^{[3]}A^{[2]} + b^{[3]}$	$[l_3, m]$
Capa 3	$[5, 4]$	$[5, 1]$	$Z^{[3]} = W^{[3]}A^{[2]} + b^{[3]}$	$[5, m]$

**Código 3.1** Función para inicializar parámetros.

```
def inicializar_parametros_red(vector_dim):
    parametros={}
    L=len(vector_dim)

    for l in range(1,L):
        parametros['W'+str(l)] = np.random.randn(vector_dim[l],
            vector_dim[l-1])*0.01
        parametros['b'+str(l)] = np.zeros((vector_dim[l],1))

    return parametros
```

### 3.3.4 Lazo de Entrenamiento

#### Propagación hacia adelante

Después de inicializar parámetros, ahora realizaremos la función que realiza la propagación hacia adelante, es decir propaga las entradas a través de las diferentes capas ocultas hasta la capa de salida para obtener la salida a la entrada aplicada, para esto se completara las siguientes funciones:

- Función lineal.
- Función de activación ReLU o sigmoide.
- Función de L capas.

**Función Lineal** La expresión matemática de esta unidad es  $Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]}$ , donde  $A^{[0]} = X$ , su desarrollo se muestra en el listado 3.2

---

#### Código 3.2 Función lineal.

---

```
def lineal(A, w, b):  
    Z=np.dot(W, A) + b  
    cache=(A, W, b)  
    return Z, cache
```

---

**Función Activación Lineal** Es necesario tener las opciones de activación sigmoide y ReLU que se describe a continuación:

- **Sigmoide:**  $\sigma(Z) = \sigma(WA + b) = \frac{1}{1+e^{-(WA+b)}}$
- **ReLU:**  $A = ReLU(Z) = \max(0,Z)$

La expresión matemática de esta unidad es  $A^{[l]} = g(Z^{[l]}) = g(W^{[l]}A^{[l-1]} + b^{[l]})$ , donde "g" puede ser la función sigmoide o ReLU. (Ver el listado 3.3)



---

**Código 3.3** Función Activación lineal.

---

```
def funcion_lineal_activacion(A_prev, W, b, activacion):  
    if activacion=="sigmoide"  
        Z, lineal_cache = lineal(A_prev, W, b)  
        A, activacion_cache = sigmoide(Z)  
  
    if activacion="ReLU"  
        Z, lineal_cache = lineal(A_prev, W, b)  
        A, activacion_cache = relu(Z)  
  
    return A, cache
```

---

**Modelo de L capas ocultas** De una forma mas conveniente, cuando trabajamos con una red neuronal de L-capas, es necesario replicar las L-1 capas con función de activación ReLU y la ultima capa con función de activación sigmoide. Aplicando esto resulta el código mostrado en el listado 3.4

---

**Código 3.4** Función Activación para L capas.

---

```
def funcion_L_activacion(X, parametros):  
    caches = []  
    A = X  
    L=len(parametros) // 2  
  
    for l in range(1, L) :  
        A_prev = A  
        A, cache = funcion_lineal_activacion(A_prev, parametros['W'  
            + str(l)], parametros['b' + str(l)], activacion='relu')  
        caches.append(cache)
```

```
AL, cache = funcion_lineal_activacion(A_prev, parametros['W' +
    str(L)], parametros['b' + str(L)], activacion='sigmoid')
caches.append(cache)

return AL, caches
```

---

### Cálculo de Errores

Es necesario hallar los errores para implementar el algoritmo de propagación hacia atrás, esta función de costo nos permitirá verificar si nuestra red neuronal esta realmente aprendiendo.

Para calcular el error es necesario realizar la siguiente formula:  $-\frac{1}{m} \sum_{i=1}^m (y^{(i)} \log(a^{[L](i)}) + (1 - y^{(i)}) \log(1 - a^{[L](i)}))$  (Ver el listado 3.5)

---

#### Código 3.5 Función para el cálculo del error.

---

```
def calculo_error(AL, Y):
    m = Y.shape[1]
    error = (-1 / m) * np.sum(np.multiply(Y, np.log(AL)) + np.
        multiply(1 - Y, np.log(1 - AL)))
    return error
```

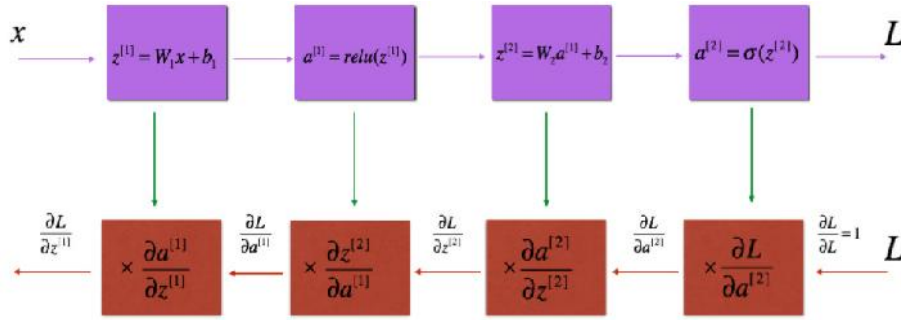
---

### Implementar el algoritmo de propagación hacia atrás

Al igual que en la propagación hacia adelante, ahora se implementara la propagación hacia atrás en pasos (funciones). Es importante recordar que la propagación hacia atrás es usada para calcular el gradiente de la función de error en función de los parámetros de la red. En la figura 3.19 se establece las relaciones entre la propagación hacia adelante y la propagación hacia atrás.

De manera similar a propagación hacia adelante son necesario las siguientes funciones:

- Función lineal hacia atrás.
- Función de activación ReLU o sigmoide hacia atrás.



**Figura 3.19** Propagación hacia adelante y propagación hacia atrás.

- Función de L capas hacia atrás.

**Función lineal hacia atrás.** Para la capa  $l$  la parte lineal es  $Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]}$ , donde  $A^{[0]} = X$ , son necesarios obtener  $dW^{[l]}$ ,  $db^{[l]}$ ,  $dA^{[l-1]}$ , que son calculadas usando como entrada  $dZ^{[l]}$ . Aquí las formulas necesarias:

$$dW^{[l]} = \frac{\partial L}{\partial W^{[l]}} = \frac{1}{m} dZ^{[l]} A^{[l-1]T}$$

$$db^{[l]} = \frac{\partial L}{\partial b^{[l]}} = \frac{1}{m} \sum_{i=1}^m dZ^{[l](i)}$$

$$dA^{[l-1]} = \frac{\partial L}{\partial A^{[l-1]}} = W^{[l]T} dZ^{[l]}$$

---

**Código 3.6** Función lineal hacia atras.

---

```
def lineal_atras(dZ, cache):
    A_prev, W, b = cache
    m = A_prev.shape[1]
    dW = np.dot(dZ, cache[0].T) / m
    db = np.squeeze(np.sum(dZ, axis=1, keepdims=True)) / m
    dA_prev = np.dot(cache[1].T, dZ)

    return dA_prev, dW, db
```

---

**Función Activación Lineal hacia atrás** Es necesario tener las opciones de activación sigmoide y ReLU que se describe a continuación:

---

**Código 3.7** Función Activación lineal hacia atrás.

---

```
def funcion_lineal_activacion_atras(dA, cache, activacion):
    lineal_cache, activacion_cache = cache

    if activacion == "relu":
        dZ = relu_hacia_atras(dA, activacion_cache)

    elif activacion == "sigmoid":
        dZ = sigmoide_hacia_atras(dA, activacion_cache)

    dA_prev, dW, db = lineal_atras(dZ, lineal_cache)

    return dA_prev, dW, db
```

---

El listado 3.8 hace uso de las funciones `relu_hacia_atras(dA, activacion_cache)` y `sigmoide_hacia_atras(dA, activacion_cache)`, las mismas que se muestran en el listado ??

---

**Código 3.8** Función Activación lineal hacia atrás.

---

```
def relu_hacia_atras(dA, cache):
    Z = cache
    dZ = np.array(dA, copy=True)
    dZ[Z <= 0] = 0
    return dZ

def sigmoide_hacia_atras(dA, cache):
    Z = cache
    s = 1/(1+np.exp(-Z))
```

---

```
dZ = dA * s * (1-s)
return dZ
```

---

**Modelo de L capas ocultas hacia atrás** De nuevo en una red neuronal de L-capas, es necesario replicar las L-1 capas con función de activación ReLU y la ultima capa con función de activación sigmoide es necesario desarrollar una función de propagación hacia atrás que use las funciones de transferencia correctas. Los resultados se observa en el listado 3.10

---

**Código 3.9** Función propagación hacia atras para L capas.

---

```
def funcion_L_activacion_atras(AL, Y, caches):
    grados = {}
    L = len(caches)
    m = AL.shape[1]
    Y = Y.reshape(AL.shape)

    dAL = dAL = - (np.divide(Y, AL) - np.divide(1 - Y, 1 - AL))

    current_cache = caches[-1]
    grados["dA" + str(L)], grados["dW" + str(L)], grados["db" + str(L)] = lineal_atras(sigmoide_hacia_atras(dAL, current_cache[1]), current_cache[0])

    for l in reversed(range(L-1)):
        current_cache = caches[l]
        dA_prev_temp, dW_temp, db_temp = lineal_atras(sigmoide_hacia_atras(dAL, caches[l]), caches[l])
        grados["dA" + str(l + 1)] = dA_prev_temp
        grados["dW" + str(l + 1)] = dW_temp
```

```
    grados["db" + str(l + 1)] = db_temp  
    return grados
```

---

### Actualizar parámetros

Ahora actualizamos los parámetros de la red neuronal usando el gradiente descendente:

$$b^{[l]} = b^{[l]} - \alpha db^{[l]}$$

donde  $\alpha$  es la velocidad de aprendizaje. Después de actualizar los parámetros se almacenan los mismos en un diccionario.

---

#### **Código 3.10** Función de actualización de parámetros.

---

```
def actualiza_parametros(parametros, grados, veloc_aprendizaje):  
    L = len(parameters) // 2  
    for l in range(L):  
        parametros["W" + str(l + 1)] = parametros["W" + str(l + 1)]  
            - veloc_aprendizaje * grados["dW" + str(l + 1)]  
        parametros["b" + str(l + 1)] = parametros["b" + str(l + 1)]  
            - veloc_aprendizaje * grados["db" + str(l + 1)]  
  
    return parametros
```

---

### 3.3.5 Pruebas de Desempeño

El modelo de red neuronal se ha entrenado para 04 comandos, el error ha ido disminuyendo y esto ha tomado como 15 minutos para 2500 iteraciones. Esto se muestra en la figura 3.20 y en las figura 3.21 y 3.22 la tendencia a la disminución de error con el aumento de iteraciones para dos comandos diferentes.

```

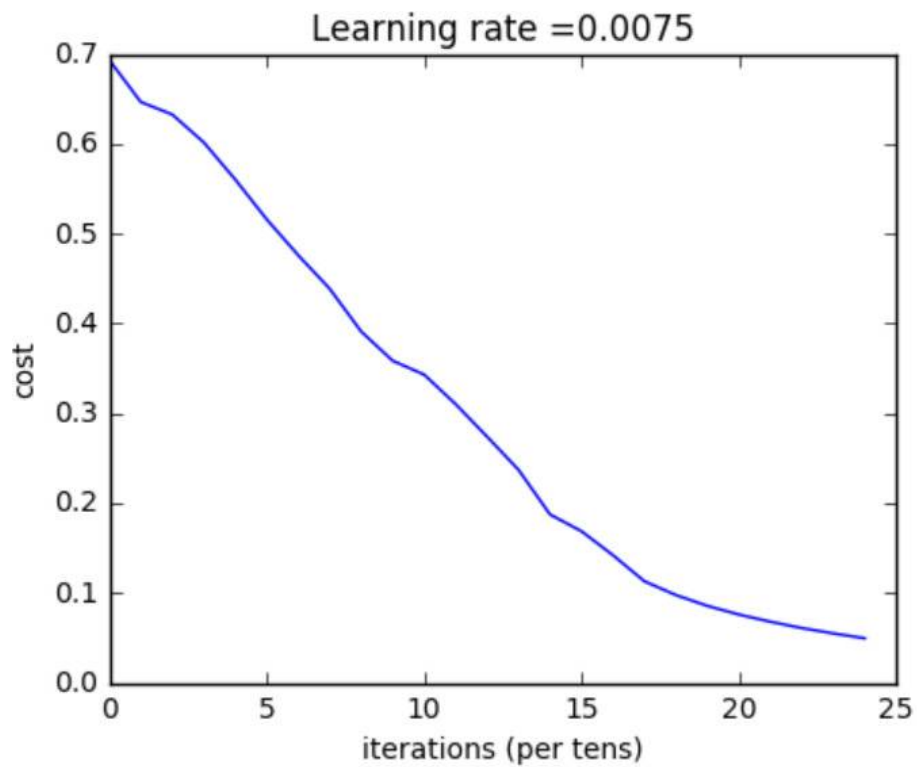
Cost after iteration 0: 0.693049735659989
Cost after iteration 100: 0.6464283150388817
Cost after iteration 200: 0.6325086257948581
Cost after iteration 300: 0.6014932097069347
Cost after iteration 400: 0.5601652326734954
Cost after iteration 500: 0.5157934975899229
Cost after iteration 600: 0.47590238832247345
Cost after iteration 700: 0.43858640997112314
Cost after iteration 800: 0.3910072887566946
Cost after iteration 900: 0.3584220494573506
Cost after iteration 1000: 0.34299735904004847
Cost after iteration 1100: 0.3101904235095467
Cost after iteration 1200: 0.273966132872609
Cost after iteration 1300: 0.2370347086950205
Cost after iteration 1400: 0.18734248651597438
Cost after iteration 1500: 0.16864009816415473
Cost after iteration 1600: 0.14197411005689567
Cost after iteration 1700: 0.11292363594102996
Cost after iteration 1800: 0.09758444844752405
Cost after iteration 1900: 0.08554699338649566
Cost after iteration 2000: 0.07593496617351737
Cost after iteration 2100: 0.06794136245969253
Cost after iteration 2200: 0.06085759338231386
Cost after iteration 2300: 0.054887735653158576
Cost after iteration 2400: 0.04950829635846386

```

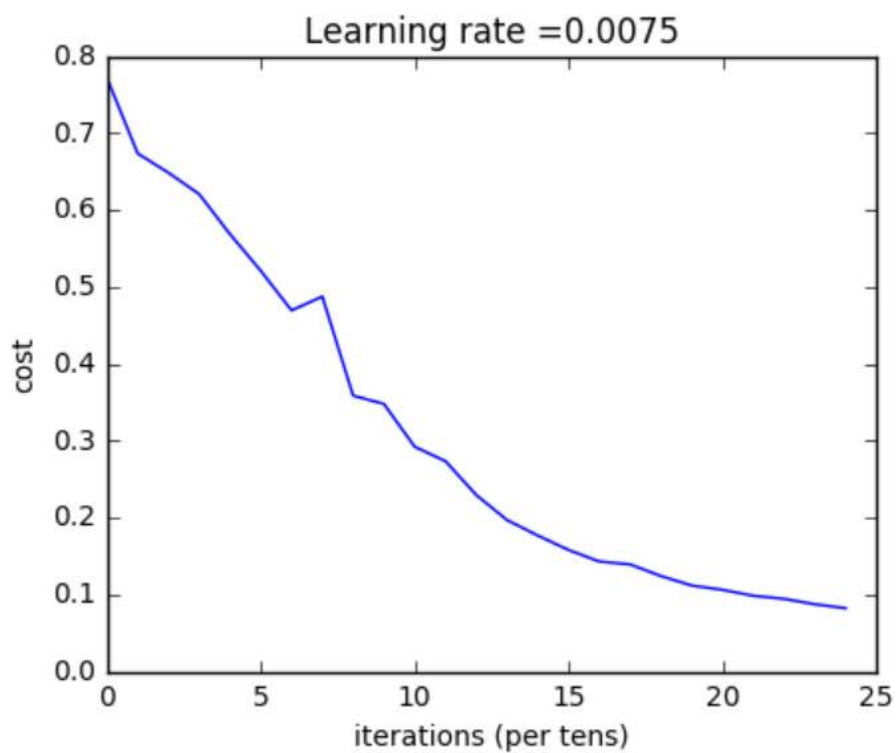
**Figura 3.20** Error versus número de iteraciones.

### 3.3.6 Consolidado de Arquitectura de Red Neuronal

Finalmente presentamos la arquitectura de la red neuronal que se usa para el reconocimiento de voz. Está compuesta de una entrada que consta de 129 características obtenidas del estimador de densidad espectral de potencia para cada comando de voz, la que esta completamente conectada a una primera capa oculta de 40 neuronas por lo que las dimensiones de los pesos de esta primera capa es  $W^{[1]} = [129, 40]$ . Esta capa se conecta a una segunda capa oculta compuesta de 06 neuronas, así las dimensiones de los pesos son  $W^{[2]} = [40, 6]$ , seguido de una tercera capa oculta compuesta también de 06 neuronas, dando una matriz de pesos de  $W^{[3]} = [6, 6]$ . Finalmente tenemos la capa de salida compuesta de 05 neuronas,



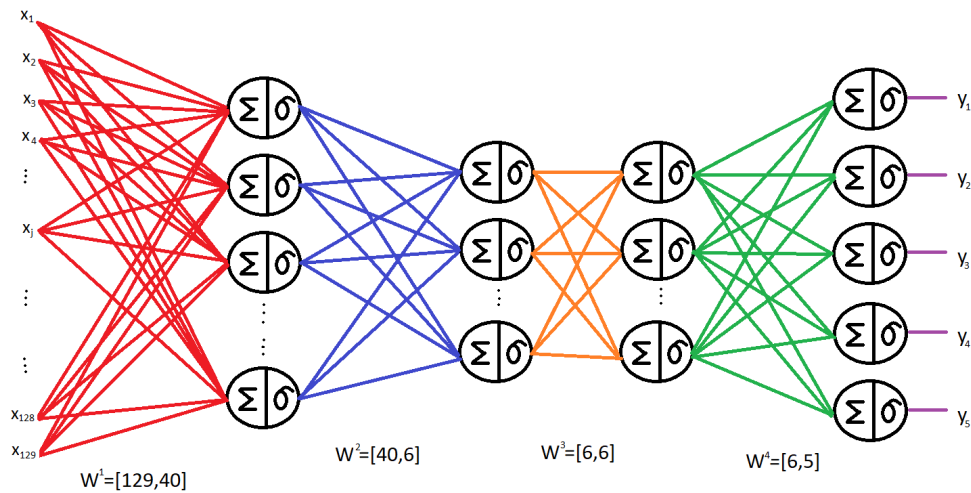
**Figura 3.21** Error versus número de iteraciones.



**Figura 3.22** Error versus número de iteraciones.



en este caso cada neurona de salida se usa para cada comando. es decir se activara solo una neurona en cada comando de voz. Las dimensiones de los pesos de está capa de salida es  $W^{[4]} = [6,5]$



**Figura 3.23** Arquitectura de Red Neuronal.

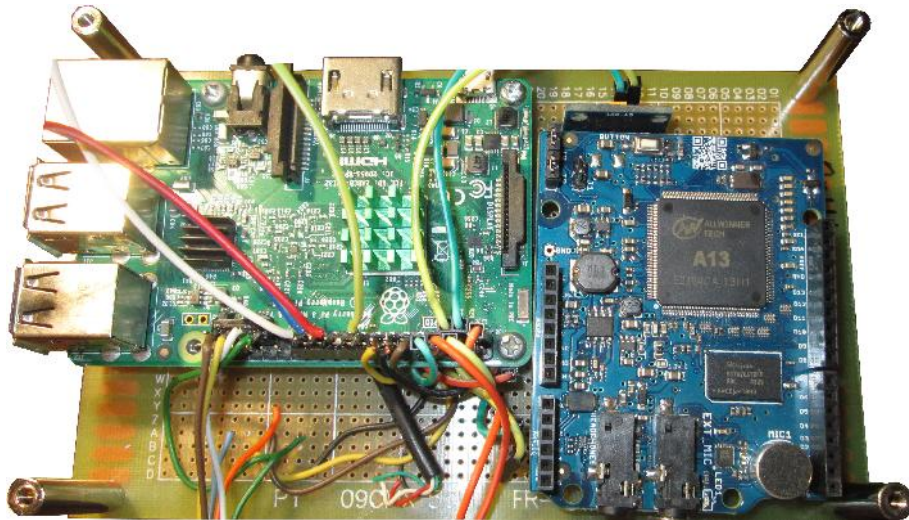
### 3.3.7 Conexionado de Tarjetas

#### Solución 01

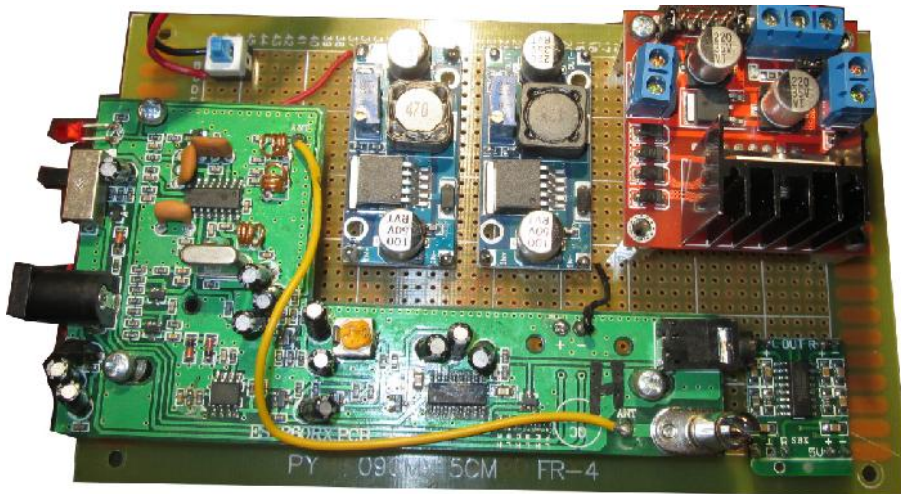
Se realizo las conexiones de la tarjeta de control, donde se usa un Raspberry Pi 3, que se comunica de forma serial a una tarjeta de adquisición de voz y síntesis de respuestas para hacer el sistema más amigable. El resultado se puede ver en la figura 3.24

Ademas se ha construido la Tarjeta de potencia (Ver figura 3.25) donde se puede identificar, la tarjeta de control de motores, que puede manejar cargas de hasta 5A, las tarjetas convertidores de DC-DC con eficiencias cercanas al 90 %, que permiten a partir de los 12V de la batería se pueda obtener 9V DC y 5V DC, voltajes necesarios para alimentar las diferentes tarjetas. En esta tarjeta también se observa el receptor de un sistema de micrófono inalámbrico y amplificador de audio para la salida de las respuestas sintetizadas.

En la figura 3.26 se puede ver el prototipo armado y conexionado de acuerdo a esta alternativa. Se puede ver claramente la tarjeta de control de motores, la tarjetas convertidores de DC-DC, los sensores de ultrasonido y el parlante que s usara para escuchar las respuestas del sistema.



**Figura 3.24** Tarjeta de Control - Alternativa 01.



**Figura 3.25** Tarjeta de Potencia - Alternativa 01.

Desafortunadamente esta alternativa de solución no tubo un desempeño adecuado, pues fallaba constantemente la comunicación serial entre la tarjeta Raspberry PI y la tarjeta de adquisición de voz y síntesis de respuestas que llevaban a un comportamiento errático al sistema. Por estas razones esta solución tubo que ser desechada y se busco otra alternativa

### **Solución 02**

Para esta alternativa se considero una tarjeta Arduino Mega 2560 como tarjeta de control, junto con la tarjeta de adquisición de voz y síntesis de respuestas para hacer el sistema mas amigable. El resultado se puede ver en la figura 3.27

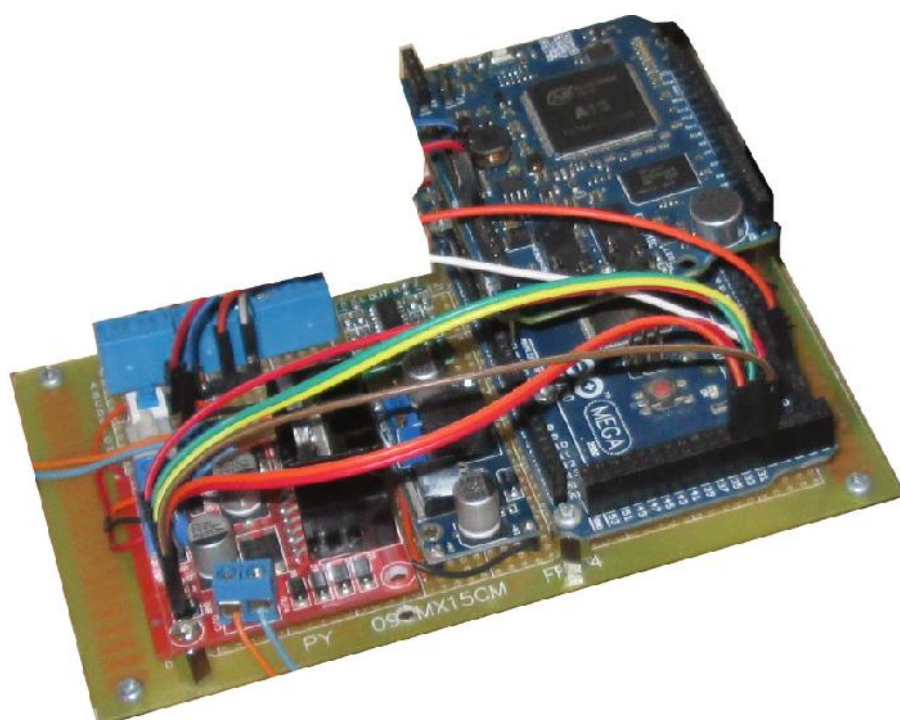


**Figura 3.26** Prototipo - Alternativa 01.

De nuevo se puede observar que la tarjeta de control de motores, la tarjeta de convertidor de DC-DC, el amplificador de audio. Finalmente el prototipo armado se ve en la figura 3.28, donde se puede ver claramente, el parlante y el sensor de ultrasonido en la parte posterior.

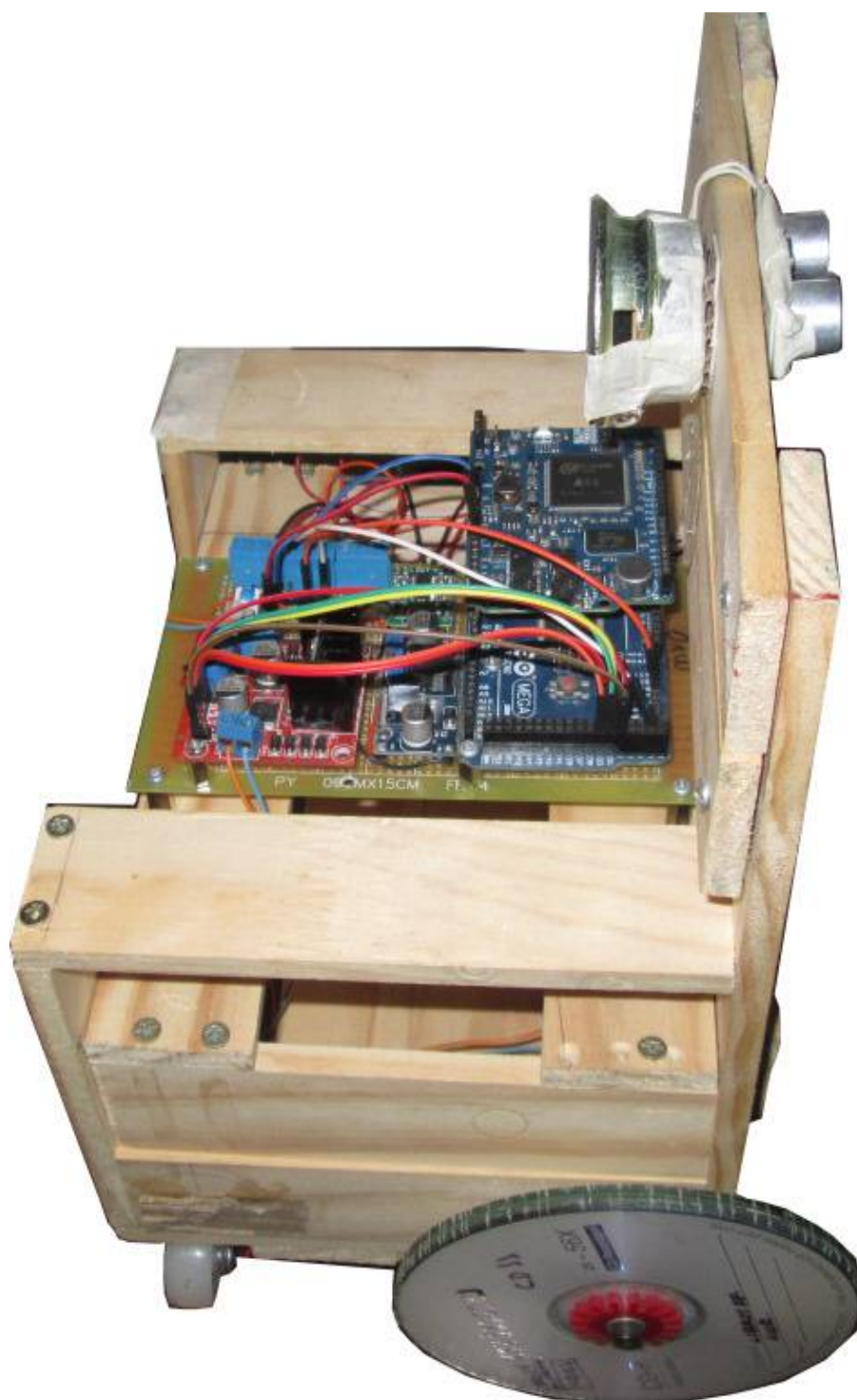
Debido a un mal funcionamiento del parlante y/o amplificador de audio colocado se opto por colocar unos parlantes de computadora, que solo necesitan una alimentación de 5V DC y su conexión a salida correspondiente en la tarjeta de adquisición de voz y síntesis de respuestas. El resultado final se puede observar en la figura 3.29.

### 3.3.8 Programa de Control



**Figura 3.27** Tarjeta de Control - Alternativa 02.

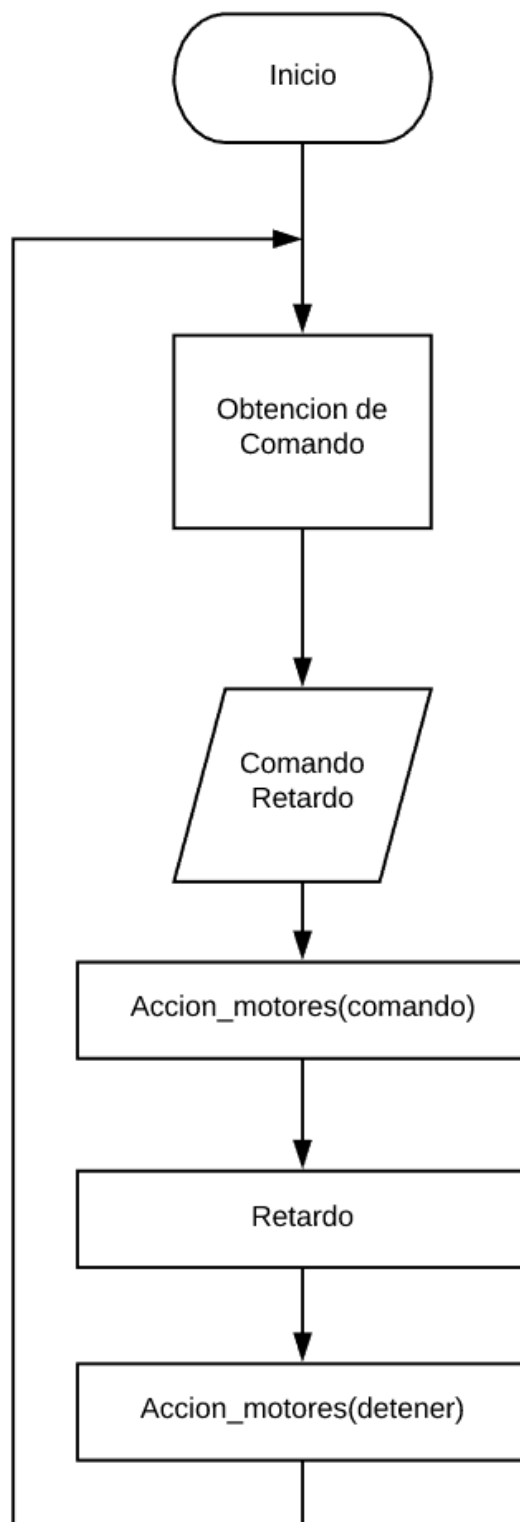




**Figura 3.28** Prototipo - Alternativa 02.



**Figura 3.29** Prototipo Final.



**Figura 3.30** Programa de Control , versión 0.1.





## 4 Conclusiones

---

**S**e desarrollo un sistema de navegación automática controlada por voz que permite mejorar la movilidad de las personas con severas discapacidades motrices, facilitando su traslado con un esfuerzo mínimo de aprendizaje de la forma de control lo que aumenta su autonomía.

Los once comandos disponibles permiten un alto grado de maniobrabilidad y los sensores instalados garantizan la seguridad de los movimientos de la silla de ruedas.

El prototipo desarrollado fácilmente puede adaptarse a las sillas de ruedas motorizadas.

Se utilizan algoritmos de caracterización de voz complejos lo que permite una adecuada clasificación casi independiente del hablante.



## **Apéndice A**

# **Datasheet de Equipos**

---

- 12 16 06 *Motocicletas y ciclomotores de tres ruedas*
- 12 16 09 *Motocicletas y ciclomotores de cuatro ruedas*
- 12 18 **Ciclos**  
Ciclos con propulsión motorizada incluidos  
Productos de apoyo para transporte para usar con bicicletas/sillas de ruedas véase 24 36 15
- 12 18 03 *Bicicletas*
- 12 18 06 *Triciclos con pedales*  
Ciclos de tres ruedas propulsados por los pies
- 12 18 09 *Ciclos propulsados con las manos*  
Ciclos de dos o más ruedas propulsados con las manos
- 12 18 12 *Patinetes no motorizados propulsados por un pie*  
Vehículos propulsados por el pie que consisten en un patín con ruedas y un manillar de dirección
- 12 18 15 *Tándems y ciclos de cuatro ruedas*  
Ciclos con dos o más asientos y pedales que permiten montar a mas de una persona y son propulsados por los pies
- 12 18 21 *Adaptaciones para ciclos*  
Añadidos o modificaciones hechos en los ciclos para facilitar su manejo  
Incluidos por ejemplo, un motor adicional o ruedas de aprendizaje
- 12 22 **Sillas de ruedas de propulsión manual**  
Dispositivos que proporcionan movilidad sobre ruedas y soporte corporal a personas con capacidad limitada para caminar y son manejadas por el usuario o un asistente  
Sillas de bipedestación (sillas capaces de elevar y mantener una a una persona en posición de pie) incluidas  
Sillas para baño/ducha con ruedas, véase 09 33 03  
Sillas con orinal (con o sin ruedas), véase 09 12 03  
Sillas de traslado, véase 12 27 04
- 12 22 03 *Sillas de ruedas bimanuales*  
Sillas de ruedas diseñadas para ser propulsadas por el usuario, empujando con ambas manos sobre las ruedas traseras o los aros de las ruedas traseras  
Sillas de ruedas propulsadas por las ruedas delanteras y por las ruedas traseras incluidas
- 12 22 06 *Sillas de ruedas bimanuales manejadas por medio de palancas*  
Sillas de ruedas diseñadas para ser propulsadas por el usuario, con las dos manos, usando dos palancas
- 12 22 09 *Sillas de ruedas manuales, de conducción mono-lateral*  
Sillas de ruedas diseñadas para ser propulsadas por el usuario usando solamente una mano
- 12 22 12 *Sillas de ruedas manuales de propulsión asistida*  
Sillas de ruedas diseñadas para ser propulsadas por el usuario, empujando con la mano(s) sobre el aro(s) o la rueda(s), con un mecanismo eléctrico para ayudar al giro de la rueda  
Unidades de propulsión, véase 12 24 09
- 12 22 15 *Sillas de ruedas manejadas por el pie*  
Sillas de ruedas diseñadas para ser propulsadas por el usuario usando solamente el/los pie/pies  
Andadores para caminar sentado, véase 12 06 09  
Mobiliario para sentarse, véase 18 09

- 12 22 18 *Sillas de ruedas manuales manejadas por asistente*  
Sillas de ruedas diseñadas para ser propulsadas y conducidas por un asistente empujando con ambas manos sobre las empuñaduras de la silla de ruedas  
Sillas de ruedas para empujar incluidas  
Sillas de traslado, véase 12 27 04
- 12 22 21 *Sillas de ruedas manuales de propulsión asistida manejadas por asistente*  
Sillas de ruedas diseñadas para ser propulsadas por un asistente, empujando con ambas manos sobre las empuñaduras de la silla de ruedas, con un mecanismo eléctrico para ayudar al giro de la rueda.
- 12 23 Sillas de ruedas de propulsión motorizada**  
Dispositivos con propulsión por motor, que proporcionan movilidad sobre ruedas y soporte corporal a personas con capacidad limitada para caminar  
Sillas motorizadas de bipedestación (sillas capaces de elevar y mantener una a una persona en posición de pie) incluidas
- 12 23 03 *Sillas de ruedas con motor eléctrico y dirección manual*  
Sillas de ruedas de propulsión eléctrica con control de la dirección por un sistema mecánico que permite la orientación de las ruedas de dirección sin fuente de energía  
*Scooters* incluidos
- 12 23 06 *Sillas de ruedas con motor eléctrico y dirección eléctrica*  
Sillas de ruedas de propulsión eléctrica con control eléctrico de la dirección
- 12 23 09 *Sillas de ruedas con motor de combustión*  
Sillas de ruedas propulsadas por un motor de combustión
- 12 23 12 *Sillas de ruedas motorizadas manejadas por asistente*  
Sillas de ruedas con propulsión eléctrica diseñadas para ser conducidas por un asistente  
Sillas de ruedas manuales manejadas por asistente, véase 12 22 18
- 12 24 Accesorios para sillas de ruedas**  
Dispositivos asociados al uso de la silla de ruedas  
Incluidos por ejemplo, aquellos accesorios que no forman parte de la gama estándar de accesorios diseñados para la utilización con una silla de ruedas particular (estos últimos se incluyen en el nivel particular de silla de ruedas que corresponda)  
Productos de apoyo para prevención de úlceras por presión (productos antiescaras), véase 04 33  
Asientos, sistemas de sedestación y bloques de abducción, véase 18 09 31  
Respaldos, véase 18 09 34  
Cojines de asiento y soportes, véase 18 09 42  
Cojines y almohadillas para la espalda, véase 18 09 45  
Orugas para escaleras, véase 18 30 12
- 12 24 03 *Sistemas de dirección y de control*  
Dispositivos para controlar los movimientos de la silla de ruedas y la dirección del recorrido
- 12 24 09 *Unidades de propulsión*  
Dispositivos que se añaden a una silla de ruedas manual para proporcionar energía y mecanismos para conducirla  
Sistemas de dirección, de control o de frenado incluidos  
Sillas de ruedas manuales de propulsión asistida, véase 12 22 12

- 12 24 12 *Luces*  
Dispositivos para iluminar los alrededores o señalar la posición de la silla de ruedas
- 12 24 15 *Mesas o bandejas portátiles*  
Dispositivos en los se pueden llevar a cabo actividades o colocar objetos mientras se está sentado en la silla de ruedas  
Recipientes de poca profundidad y tablas incluidos
- 12 24 18 *Frenos*  
Dispositivos para reducir la velocidad, parar una silla de ruedas o mantenerla en una posición fija
- 12 24 21 *Neumáticos y ruedas*
- 12 24 24 *Baterías y cargadores de baterías*  
Dispositivos para proporcionar energía eléctrica
- 12 24 27 *Dispositivos para limpiar las ruedas*  
Cepillos incluidos
- 12 24 30 *Sistemas de seguridad para ocupante de silla de ruedas*  
Dispositivos usados en silla de ruedas para prevenir al ocupante deslizarse o caerse de la silla  
Cinturones, arneses y chalecos incluidos  
Equipo para sujetar una silla de ruedas en un coche, véase 12 12 24
- 12 24 33 *Paraguas y sujeciones para paraguas para silla de ruedas*
- 12 24 36 *Conexiones para bicicletas*  
Dispositivos para acoplar una silla de ruedas a una bicicleta
- 12 27 Vehículos**  
Coche, véase 12 10  
Motocicletas y ciclomotores, véase 12 16  
Ciclos, véase 12 18
- 12 27 04 *Sillas de traslado*  
Dispositivos para trasladar, durante una distancia corta, a una persona en posición sentada controlados por un asistente  
Sillas de ruedas manuales controladas por asistente, véase 12 22 18
- 12 27 07 *Sillas de paseo para niños (cochecitos para niños)*  
Dispositivos con ruedas para llevar a una o más personas en posición tumbada o sentada, diseñados para ser propulsados y conducidos por un asistente
- 12 27 09 *Trineos*  
Vehículos montados sobre patines usados para transportar personas sobre hielo y nieve
- 12 27 12 *Trineos para empujar con el pie*  
Vehículos que consisten en una silla fijada sobre dos patines, con o sin ruedas, propulsadas con el pie por una persona de pie detrás de la silla
- 12 27 15 *Gateadores y tablas para desplazarse*  
Dispositivos con ruedas en los que una persona se sienta o tumba, permitiéndole desplazarse empujando sobre el suelo con sus brazos o piernas

# Índice de Figuras

---

2.1.	Sillas de Ruedas Manuales.	6
2.2.	Sillas de Ruedas Motorizadas.	6
2.3.	Esquema de Silla Motorizada	8
2.4.	Red Neuronal y sus tipos	10
2.5.	Modelo de Neurona Artificial	11
2.6.	Python en Windows usando Anaconda	14
2.7.	Python en la consola	15
2.8.	Python con Editor	16
2.9.	Tarjetas Arduino Uno y Genuino Uno	17
2.10.	Especificaciones de Arduino Uno y Genuino Uno	17
2.11.	Tarjetas Arduino Mega 2560 y Genuino Mega 2560	18
2.12.	Especificaciones de Arduino Mega 2560 y Genuino Mega 2560	18
2.13.	Descarga de Arduino IDE	19
2.14.	Raspberry Pi 3 Modelo B	20
2.15.	Traductor Travis	21
2.16.	Arquitectura de los Sistemas ASR	23
2.17.	Movi de Audeme	23
3.1.	Componentes del Sistema de Navegación controlado por Voz	26
3.2.	Prototipo de Silla de Ruedas - Motor Izquierdo y Derecho	28

---

3.3.	Prototipo de Silla de Ruedas	29
3.4.	Prototipo de Silla de Ruedas - Motor Izquierdo y Derecho	30
3.5.	Prototipo de Silla de Ruedas	31
3.6.	Intuición de Tabla de entrada salidas para aprendizaje supervisado	33
3.7.	Un segundo de una orden grabada.	33
3.8.	Un segundo de una orden grabada.	34
3.9.	Un segundo de una orden grabada.	34
3.10.	Un segundo de una orden grabada.	34
3.11.	Dos versiones de PDF de señales de entrada.	35
3.12.	Dos versiones de PDF de señales de entrada.	35
3.13.	Dos versiones de PDF de señales de entrada.	35
3.14.	Dos versiones de PDF de señales de entrada.	36
3.15.	Intuición de estimador de PDF	36
3.16.	Estimación de PDF para cinco ordenes diferentes	36
3.17.	Similitud de la estimación de PDF de señales de entrada para la misma señal	37
3.18.	Estructura de la Red Neuronal	38
3.19.	Propagación hacia adelante y propagación hacia atrás	43
3.20.	Error versus número de iteraciones	47
3.21.	Error versus número de iteraciones	48
3.22.	Error versus número de iteraciones	48
3.23.	Arquitectura de Red Neuronal	49
3.24.	Tarjeta de Control - Alternativa 01	50
3.25.	Tarjeta de Potencia - Alternativa 01	50
3.26.	Prototipo - Alternativa 01	51
3.27.	Tarjeta de Control - Alternativa 02	52
3.28.	Prototipo - Alternativa 02	53
3.29.	Prototipo Final	54
3.30.	Programa de Control , versión 0.1	55



# Índice de Tablas

---

3.1.	Dimensiones de los parámetros para una red neuronal dada por $[4, 3, 4, 5]$	39
------	---	----



# Bibliografía

---

- [sun, 2016] (2016). Manual de sillas de ruedas motorizadas sunrise medical y sus cuidados.
- [ama, 2017a] (2017a). Amazon.com: Latest 2017 Foldawheel PW-1000xl Power Chair - 57 lbs only with battery (Supports 330 lbs). Foldable in just 2 seconds. Comes with a thick & tuff travel bag. Electric motorized wheelchair.: Health & Personal Care.
- [ama, 2017b] (2017b). Amazon.com: Pride jazzy select elite motorized wheelchair w/captain seat and batteries: Kindle store.
- [ard, 2017a] (2017a). Arduino - arduinoboardmega2560.
- [ard, 2017b] (2017b). Arduino - arduinoboarduno.
- [Banzi and Shiloh, 2014] Banzi, M. and Shiloh, M. (2014). *Getting Started with Arduino: The Open Source Electronics Prototyping Platform*. Maker Media, Inc.
- [Castelao and Faes, 2008] Castelao, G. S. and Faes, R. F. (2008). Productos de apoyo para personas con discapacidad. clasificación y terminología. *Revista asturiana de Terapia Ocupacional*, (6):11–13.
- [Duque, 2013] Duque, R. G. (2013.). *Python para todos*. mundogeek.net, second edition.
- [Géron, 2017] Géron, A. (2017). Hands-on machine learning with scikit-learn and tensorflow: concepts, tools, and techniques to build intelligent systems.

- [Haykin, 2009] Haykin, S. (2009). *Neural Networks and Learning Machines*. Prentice Hall, third edition.
- [INEI, 2012] INEI (2012). *Primera Encuesta Nacional Especializada sobre Discapacidad 2012*. INEI, Lima.
- [Raquel Flóres López, 2009] Raquel Flóres López, J. M. F. F. (2009). *Las Redes Neuronales Artificiales. Fundamentos teóricos y aplicaciones prácticas*. Netbiblo, S. L.
- [Wikipedia, 2017a] Wikipedia (2017a). Raspberry pi — wikipedia, la enciclopedia libre. [Internet; descargado 4-marzo-2017].
- [Wikipedia, 2017b] Wikipedia (2017b). Silla de ruedas — wikipedia, la enciclopedia libre. [Internet; descargado 3-febrero-2017].
- [Yu and Deng, 2014] Yu, D. and Deng, L. (2014). *Automatic speech recognition: A deep learning approach*. Springer.

# Glossary

---

## R

**RNA** Red Neuronal Artificial, p. IX.