# SISTEMA OFFLINE DE RECONOCIMIENTO DE COMANDOS DE VOZ PARA RASPBERRY PI USANDO APRENDIZAJE AUTOMÁTICO

**IBRAHIMME MORELO MEJIA**
**ANDRÉS RAMIRO VILLEGAS OYOLA**


**UNIVERSIDAD EL BOSQUE**
**PROGRAMA DE INGENIERÍA DE SISTEMAS**
**FACULTAD DE INGENIERÍA**
**Bogotá, 2021**

# SISTEMA OFFLINE DE RECONOCIMIENTO DE COMANDOS DE VOZ PARA RASPBERRY PI USANDO APRENDIZAJE AUTOMÁTICO

**IBRAHIMME MORELO MEJIA**
**ANDRÉS RAMIRO VILLEGAS OYOLA**

**Desarrollo Tecnológico presentado como requisito para optar al título de INGENIERO DE SISTEMAS**

**Director**
**FRAN ERNESTO ROMERO ALVAREZ**
Magíster en Modelado y Simulación

**UNIVERSIDAD EL BOSQUE**
**PROGRAMA DE INGENIERÍA DE SISTEMAS**
**FACULTAD DE INGENIERÍA**
**Bogotá, 2021**

Dedicatoria

Ibrahimme Morelo Mejía

Agradezco a mi familia que siempre han estado a mi lado apoyándome, en especial a mis pilares que son mi madre, pareja, hija y hermana. También agradezco a nuestro director y representante del cliente, que siempre estuvo dispuesto para ayudarnos. A los amigos y compañeros que hice durante toda la carrera, hicieron de mi una mejor persona y profesional. La gloria y honra siempre sea para Dios, aquel todo lo puede y que siempre me dio las fuerzas para continuar.

Andrés Ramiro Villegas Oyola

En memoria de mis abuelas quienes siempre quisieron verme graduado como un profesional, a mis padres, por su amor, sacrificio y apoyo incondicional, por enseñarme los valores y principios que me han permitido ser quien soy actualmente, por inculcar en mí la perseverancia y el siempre seguir hacia adelante sin importar la dificultad de los obstáculos, gracias a todo ello hoy puedo cumplir un sueño más. También quiero agradecer a mis profesores por su arduo trabajo, dedicación y guía durante mi carrera académica. Por último, quiero agradecer a Dios por permitirme culminar una nueva etapa en mi vida.

## AGRADECIMIENTOS

Queremos agradecer al grupo de investigación Osiris y Bioaxis de la universidad el bosque por permitirnos hacer parte de este proyecto. De igual forma queremos agradecer al MEng. Fran Ernesto Romero Álvarez por su dedicación, apoyo, enseñanzas, paciencia y orientación durante todo el desarrollo de este proyecto, mediante los cuales fue posible culminar nuestro trabajo de grado, a través del cual obtuvimos un gran aprendizaje, lecciones y conocimientos valiosos para nuestra profesión.

## CONTENIDO

## LISTA DE TABLAS

## LISTA DE FIGURAS

## GLOSARIO

**Aprendizaje automático:** Es un campo de la ciencia de la computación y sub campo de la inteligencia artificial conocido como aprendizaje de máquina, este permite el desarrollo de técnicas para que las máquinas puedan aprender mediante el entrenamiento de datos.

**Computación al borde de la red:** Es una infraestructura tecnológica muy utilizada en proyectos cuyos datos serán procesados en periferia de la red, dicho de otra forma, son procesados lo más cercano a la fuente de origen.

**Espectro:** Es una distribución de onda a través de diferentes frecuencias.

**Espectrograma:** Es un diagrama obtenido de un espectro acústico, permite visualizar de manera gráfica las variaciones de frecuencia e intensidad del sonido por medio de colores que muestran en qué secciones se presenta mayor intensidad de datos.

**Google Coral:** Es un pequeño hardware desarrollado por Google, el cual permite acelerar los tiempos de respuesta en proyectos de inteligencia artificial.

**Inteligencia artificial:** Es un campo de la informática mediante la cual las máquinas pueden aprender capacidades como el aprendizaje y razonamiento humano, esto mediante el entrenamiento de datos.

**Latencia:** Es una suma de retardos que se presentan en una red, estos se producen por la demora existente en la transmisión y recepción de información de la red.

**Reconocimiento de voz:** Es una especialidad de la inteligencia artificial y su objetivo es el poder originar una comunicación hablada entre un humano y una máquina.

**Red neuronal convolucional:** Es una red neuronal que cuenta con aprendizaje supervisado y varias capas intermedias que permiten predicciones mejoradas con un conjunto de datos.

**Redes neuronales:** Es un modelo que permite el procesamiento de información de manera similar al cerebro humano.

## RESUMEN

SISTEMA OFFLINE DE RECONOCIMIENTO DE COMANDOS DE VOZ PARA RASPBERRY PI

Morelo Ibrahimme, Villegas Andrés, Romero Fran

**Resumen**

Con el auge exponencial de la computación en la nube cada vez son más los sistemas que la adoptan sin tener en cuenta la latencia producida en la internet, el reconocimiento de voz es uno de los campos donde comúnmente se hace uso de la nube dado el volumen de datos que usa para tal fin. En el presente documento se propuso el desarrollo de un sistema de reconocimiento de voz offline orientado a comandos de desplazamiento terrestre en idioma español para un robot construido sobre arquitecturas de bajo costo como Raspberry Pi y la aceleradora USB Coral. Se usaron las redes neuronales convolucionales para el entrenamiento del modelo, así como dos conjuntos de datos obtenidos a través de voluntarios y usuarios del sistema en cuestión. En alineación con la modalidad del proyecto (desarrollo tecnológico), se utilizó la metodología ágil SCRUM para sostener una mejor comunicación y alineación con el cliente. El sistema de reconocimiento de voz alcanzó una eficiencia promedio expresada en tiempo de 0.31 segundos necesarios para realizar la acción solicitada, la eficacia del sistema medida en precisión fue del 86% considerada alta para el contexto. Estos resultados fueron obtenidos en entornos sin ruidos externos y con usuarios que hicieron parte del entrenamiento del modelo. El estudio mostró cómo se pueden desarrollar sistemas de reconocimiento de voz desconectados de internet y con buen desempeño.

Palabras clave: Reconocimiento de voz, inteligencia artificial, robot, aprendizaje automático, español, redes neuronales convolucionales.

## Abstract

With cloud computing's exponential rise, more and more systems are using it without considering the latency generated, for speech recognition the cloud is generally used because of the amount of data. In the present paper, a system of speech command recognition in Spanish was proposed to perform land displacement operations in a robot built with low-cost tools like Raspberry Pi and USB Accelerator Google Cora. A Neural network convolutional was used to train two datasets obtained through forms and a module developed for this purpose. According to project mode (technological development), the agile SCRUM methodology was used to have better communication with the project client. The system reached an efficiency (response time) average of 0.31 seconds, on the other hand, the efficacy measured in precision was 86% with it is high for the test context. These results were obtained in environments without external noise and with users who were part of the model's training. The study showed how developing speech commands recognition systems offline can be a good option to reach good performance.

**Keywords:** Speech recognition, artificial intelligence, robot, machine learning, Spanish, convolutional neural network.

# INTRODUCCIÓN

La interacción humana con las máquinas se ha incrementado exponencialmente en las últimas décadas. Existen muchas formas en las que un individuo puede interactuar con una máquina. Una interacción interesante en el ambiente académico se da entre una persona y un robot. Dicha relación, se ve motivada en gran medida por la necesidad de automatizar tareas o procesos que para el ser humano representan beneficios. Para que pueda existir un enlace de comunicación, se necesita establecer un medio. Si bien existen varios, en el presente proyecto nos atiende la comunicación a través de voz o como comúnmente se le conoce "Reconocimiento de Voz (RV)". Este campo del conocimiento es especialmente apetecido en la literatura, es así como muchas investigaciones giran en torno a esta. Los objetivos del RV pueden variar de acuerdo a las necesidades: emociones, reconocer el estado anímico de un individuo; automatización, delegación de tareas; y realizar acciones específicas; entre otras.

El auge de la inteligencia artificial (IA) ha democratizado el uso de esta misma, hoy en día se pueden encontrar diversas herramientas que permiten construir sistemas inteligentes. La IA tiene muchos sub campos de aplicación como lo son percepción y actuación, aprendizaje automático (ML), razonamiento automatizado y resolución de problemas. El ML ha potenciado avances en el campo del RV, los asistentes virtuales como Alexa, Siri, y Google Assistant son una prueba de ellos. Estos sistemas son mundialmente conocidos por facilitar la vida cotidiana de las personas, a través de la realización de tareas.

Las tecnologías de RV en combinación con el ML, por lo general son soluciones basadas en la nube, ubicadas en infraestructuras como Amazon Web Services, Google Cloud y Azure. Una de las razones detrás de tal situación, puede ser el alto poder computacional que poseen estas infraestructuras, que permite el procesamiento de grandes cantidades de datos. Sin embargo, esto genera efectos colaterales, por ejemplo, las altas latencias en la internet, dado que nuestros recursos son limitados. Entre más datos viajan hacia la nube, peores latencias son obtenidas por los usuarios, más aquellos que no cuentan con una buena conexión a internet. Por lo tanto, es necesario buscar alternativas de bajo costo que permitan implementar estas tecnologías sin perder sus bondades. Es aquí donde la computación al borde de la red (CBR) hace su entrada, es una tendencia que busca realizar tareas que usualmente son ejecutadas en la nube, en dispositivos locales.

Al converger el aprendizaje automático, el reconocimiento de voz, la computación al borde de la red, y la robótica se puede obtener un artefacto autónomo con prestaciones vanguardistas. Por lo tanto, se pretende implementar un subsistema de reconocimiento de comandos de voz en español sobre un Robot construido a partir de una Raspberry Pi 4, para realizar acciones de desplazamiento terrestre.

El robot ya cuenta con otros sistemas como visión artificial y desplazamiento terrestre. Esta necesidad nace del seno del grupo de Investigación Osiris & Bioaxis de la Universidad El bosque, cuyo objetivo en los muchos que tienen, es desarrollar soluciones multidisciplinares.

El ML cuenta con una serie de modelos que permiten orientarlos a temáticas específicas como la visión artificial, el RV, las finanzas computacionales, entre otras. En el ámbito del RV, se emplean varios modelos, en este proyecto se tienen en consideración las redes neuronales convolucionales y la red de memoria a corto plazo. Para el entrenamiento del modelo, se usa un conjunto de datos en español que representan los comandos despacio, adelante, atrás, izquierda, derecha, rápido, y lento. Para el procesamiento de los audios se usan varias técnicas, especialmente el aumento de datos, que ayuda considerablemente a mitigar el déficit causado por la falta de datos para el entrenamiento del modelo. Una vez el modelo es entrenado con resultados satisfactorios, se integra con el robot. Al final el resultado es un robot autónomo capaz de entender comandos a partir de la voz, con el objetivo de realizar acciones de desplazamiento.

Este documento está dividido de la siguiente forma: problema, un análisis del contexto de la situación; solución de la ingeniería, presentación de los objetivos; marco referencial, sustentación del proyecto a partir del estado del arte y marco teórico; desarrollo metodológico, procesos y actividades realizadas a lo largo del desarrollo; resultado, una síntesis de lo obtenido; discusión, un análisis de los resultados obtenidos; conclusiones, cierre del proyecto en el marco de objetivos planteados; recomendaciones y trabajos futuros, aspectos a mejorar en la parte técnica; y finalmente lecciones aprendidas, una reflexión sobre los aprendizajes obtenidos a lo largo de todo el trabajo realizado.

# 1. PROBLEMA

El objetivo de este capítulo es explicar la problemática identificada que justifica la realización de este proyecto. Para ello, se presenta el contexto a ser intervenido, los detalles del problema, un árbol de problema (causas y consecuencias), el análisis desde el modelo Biopsicosocial y Cultural (BPSC) visto desde el punto del problema y la solución.

## 1.1 DESCRIPCIÓN DEL CONTEXTO A INTERVENIR

Los robots pueden ser empleados para realizar diferentes tareas automatizadas. El grupo de investigación Osiris & Bioaxis cuenta con un Robot llamado Kenito, que realiza tareas de desplazamiento a través de visión artificial en entornos controlados. Sin embargo, el robot no tiene la capacidad de desplazarse a partir de comandos de voz, lo que implica que los movimientos deben ser emitidos de forma alternativa por un control remoto.

Kenito está construido sobre Raspberry Pi 4, esto permite tener la capacidad mínima para usar las herramientas antes mencionadas, sin embargo, estas tecnologías de RV al estar ubicadas en la nube, dependen de una conexión a internet. En consecuencia, el robot está expuesto a riesgos que pueden representar una mala experiencia para el usuario. La cantidad de paquetes que viajan al intentar hacer RV en la nube, pueden ser muy grandes, ocasionando que los desplazamientos del robot sean demorados o simplemente no se ejecuten [1]. Naturalmente, los resultados son usuarios insatisfechos, sistemas inestables, redes saturadas y la percepción de un sistema poco autónomo. Esto se ve aún más agravado con el incremento de este tipo de sistemas en el mercado tecnológico actual. Los robots, particularmente, necesitan de respuestas inmediatas para la ejecución de acciones.

Por lo tanto, para mitigar este riesgo, se pretende desarrollar un sistema independiente de la red, aplicando el concepto de computación al borde de la red. El sistema permitirá reconocer una cantidad limitada de comandos de voz en español, para el robot en cuestión, así se logrará obtener un entorno con menores tiempos de respuesta. En consecuencia, se podrán enriquecer las experiencias de intercomunicación por voz [2] para este tipo de robot basado en Raspberry Pi. Adicionalmente, será posible automatizar sus procesos de reconocimiento de voz sin la necesidad de depender de la internet [3]. Al tener un sistema con mayor autonomía, se lograría disminuir los datos que son expuestos al viajar a la nube [4], ganando en el camino mayor seguridad ante potenciales ataques de cibernéticos que buscan extraer la información.

El enfoque principal que se pretende abordar con este proyecto, es, por lo tanto, reducir costos computacionales en la implementación del RV, a través del desarrollo de un sistema con mayor independencia de la nube.

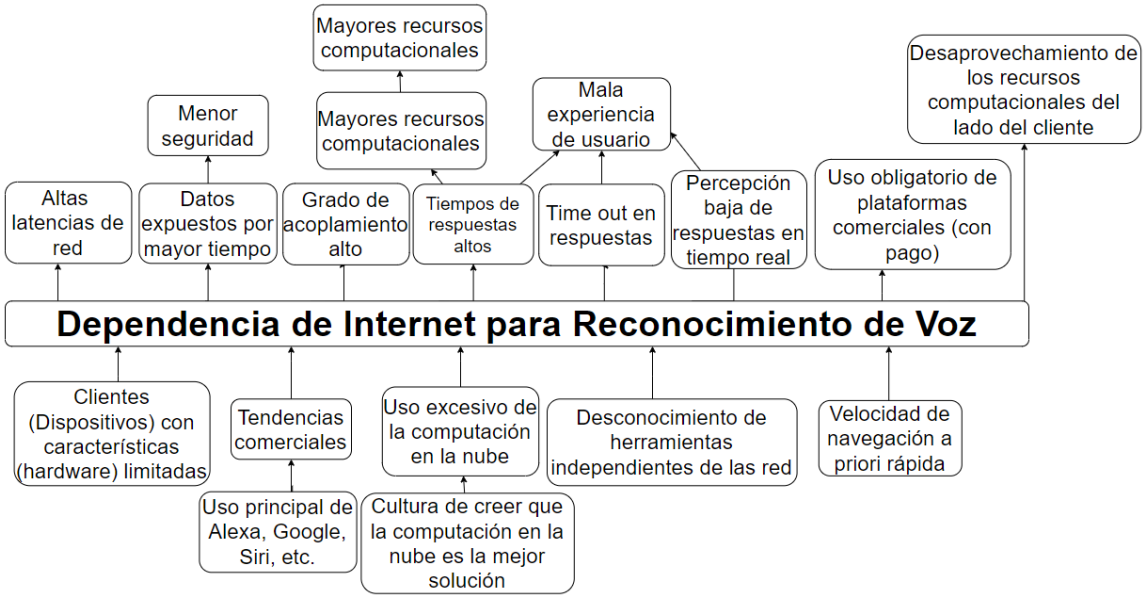
## 1.2 PREGUNTA DEL PROBLEMA

La pregunta de investigación es la siguiente:

¿Cómo se puede implementar el subsistema de reconocimiento de comandos de voz en español para un robot gobernado por una tarjeta Raspberry Pi de tal forma que no exista dependencia alguna de componentes externos al robot mismo?

## 1.3 ÁRBOL DE PROBLEMA

**Figura 1. Problemática identificada**




Fuente: Elaboración propia

El RV en la nube tiene presente diferentes riesgos o aspectos que afectan su funcionamiento, estos pueden ser: latencias, costos de servicios, conexión a internet, entre otras. Dichos factores pueden interferir en la correcta interpretación de la voz de una persona. Para una mejor representación de lo anterior, se elaboró un árbol del problema como se puede ver en la Figura 1. El gráfico está dividido así: parte central, problema del contexto; zona inferior, causas del problema; y zona superior, consecuencias del problema. Este tipo de árboles ayudan a entender el contexto del proyecto desde el punto de vista de la problemática.

## 1.4 ANÁLISIS DEL CONTEXTO DESDE EL MODELO BIOPSICOSOCIAL Y CULTURAL

La evolución de la tecnología ha llevado a que los procesos sean cada vez más inmediatos, ocasionando un cambio en la cultura. Se necesitan obtener respuestas de los sistemas en el menor tiempo posible, demandando un cumplimiento adecuado ante cada petición, motivando el hábito de construir herramientas cada vez más eficientes que puedan automatizar la realización de una tarea. Se ha instaurado la creencia que la computación en la nube es la mejor opción para satisfacer esta necesidad.

La robótica en este aspecto es un gran aliado para facilitar la vida de los humanos, permitiendo delegar tareas con un alto grado de complejidad. Por lo tanto, cada día se busca mejorar la comunicación humano-máquina, procurando interconectar los dispositivos orientados al RV con la nube.

La creación de herramientas que facilitan cada vez más estos procesos, son la respuesta de los desarrollos tecnológicos realizados en diferentes disciplinas, por ejemplo, desde la Ingeniería de Sistemas. La inmediatez y precisión que se requiere en cada proceso informático, ha llevado al hombre a construir herramientas que sustituyan el trabajo manual. De tal forma, se puedan mitigar las fallas humanas presentes en los procesos de negocios, sin embargo, el uso desmesurado de la nube en algunos escenarios representa un alto costo de tráfico en la red. Este medio de comunicación basado en Internet, como lo es la nube, puede afectar la seguridad de los artefactos, entre mayor es la distancia recorrida por los datos, mayores son los riesgos de exposición de datos. Con el desarrollo de un sistema inteligente desconectado de internet se busca mitigar las saturaciones de las redes y evitar la pérdida de seguridad.

Lo anterior, sin la necesidad de hacer consultas a medios externos, disponiendo de la información necesaria para realizar las tareas de desplazamiento terrestre de Kenito a través de RV. Siendo el objetivo del artefacto planteado, eliminar la necesidad de estar conectado a la red y aumentar la eficiencia de las respuestas. Así como también, ser una alternativa de bajo costo, frente a los sistemas existentes en el mercado. Esto se logra al tener todo el sistema desplegado sobre el Robot basado en Raspberry Pi [5], dando como resultado un sistema autónomo por medio del aprendizaje automático.

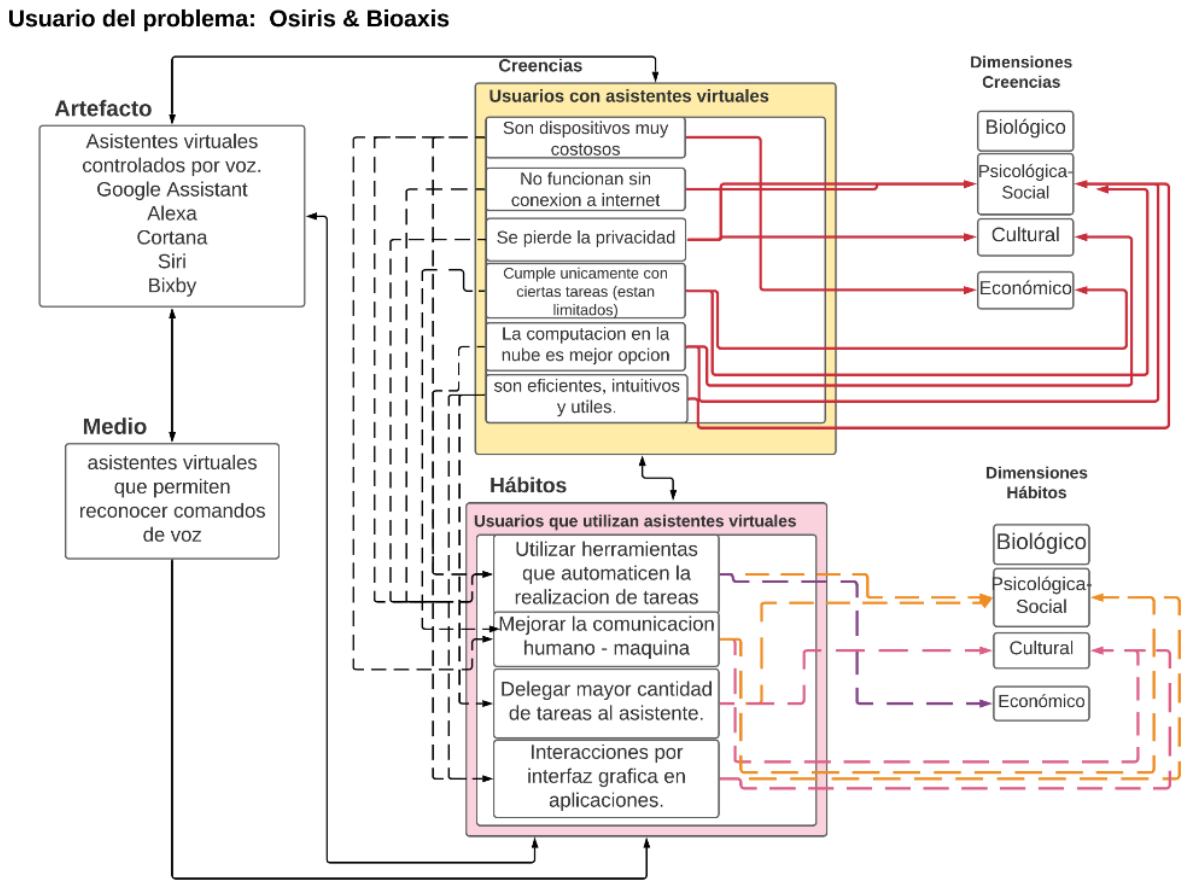
## 1.5 IDENTIFICACIÓN Y DESCRIPCIÓN DE LA PROBLEMÁTICA

Con los constantes avances tecnológicos, el RV ha mejorado y abarcado el mercado significativamente, teniendo cada vez mayor aceptación, tendencia que se ha visto influenciada por la gran variedad de dispositivos existentes. Esto ha permitido a los usuarios cumplir con ciertas tareas o controlar diversos dispositivos, resaltando el

hecho que para controlar múltiples dispositivos le representa al usuario un alto costo de los mismos, sin embargo, hoy en día es más común encontrar personas que interactúan con artefactos tecnológicos por medio de la voz.

Estos dispositivos tecnológicos permiten una comunicación más fluida entre las computadoras y los seres humanos, generando el hábito de utilizarlos para realizar tareas específicas. De igual forma, cada vez se aumenta y delega un mayor número de tareas a estos dispositivos, ocasionando un cambio entre aquellas interacciones a través de una interfaz gráfica al uso de interacciones por voz. Convirtiéndose este en un medio cada vez más relevante para la comunicación entre usuarios y dispositivos, igualmente es importante aclarar que los dispositivos tecnológicos actuales en gran medida dependen de una conexión a la red para poder funcionar, lo que ocasiona en algunos casos malas experiencias de usuario por velocidades de internet inferiores a las recomendadas, presentando altas latencias en los tiempos de respuesta así como errores en la eficiencia de las respuestas, afectando de esa manera el hábito en la mejora de la comunicación entre los seres humanos y las máquinas.

**Figura 2. Problema BPSC**




Fuente: Elaboración propia

Los dispositivos actuales que se encuentran en el mercado son muy eficientes y útiles, pero tienen un alto costo operacional, lo que significa una oportunidad de mejora de cara a nuevas implementaciones. Como el hecho de evitar la pérdida de seguridad y privacidad como lo menciona Anniappa & Kim [6], ya que al viajar los datos a la nube se expone la información del usuario.

Como se mencionó en apartados anteriores, el RV por lo general requiere de una conexión a internet, generando una dependencia, debido a que son sistemas basados en la nube. Asimismo, nace la creencia de contar con un internet de alta velocidad, caso contrario se podría traducir a malas experiencias de usuario. Para una mejor contextualización se presenta la Figura 2, donde se muestran algunas relaciones de artefactos de RV y el cómo afectan las creencias y hábitos de las personas.

# 2. SOLUCIÓN DE INGENIERÍA

En el contexto de este capítulo se abordarán los diferentes objetivos, tanto general como específicos, el análisis de la solución, tabla de entregables, variables a medir, la metodología implementada, los acuerdos alcanzados con el cliente y el componente ético fundamental para la culminación y lineamiento del proyecto.

## 2.1 OBJETIVO GENERAL

Desarrollar un sistema de reconocimiento de comandos de voz en español para Raspberry Pi, el cual funcione de forma desconectada de internet para optimizar el rendimiento y permita a un robot recibir diferentes instrucciones para llevar a cabo operaciones de desplazamiento.

## 2.2 OBJETIVO ESPECÍFICOS

Desarrollar un estado del arte sobre los diferentes modelos existentes para el reconocimiento de voz y su implementación sobre sistemas embebidos, seleccionando el más adecuado para su uso con Raspberry Pi.

Implementar el modelo seleccionado, calibrando sus parámetros para lograr un óptimo desempeño en el reconocimiento de comandos de voz.

Integrar el modelo de reconocimiento de voz con la tarjeta Raspberry Pi y con el robot, a fin de probar el funcionamiento del sistema completo.

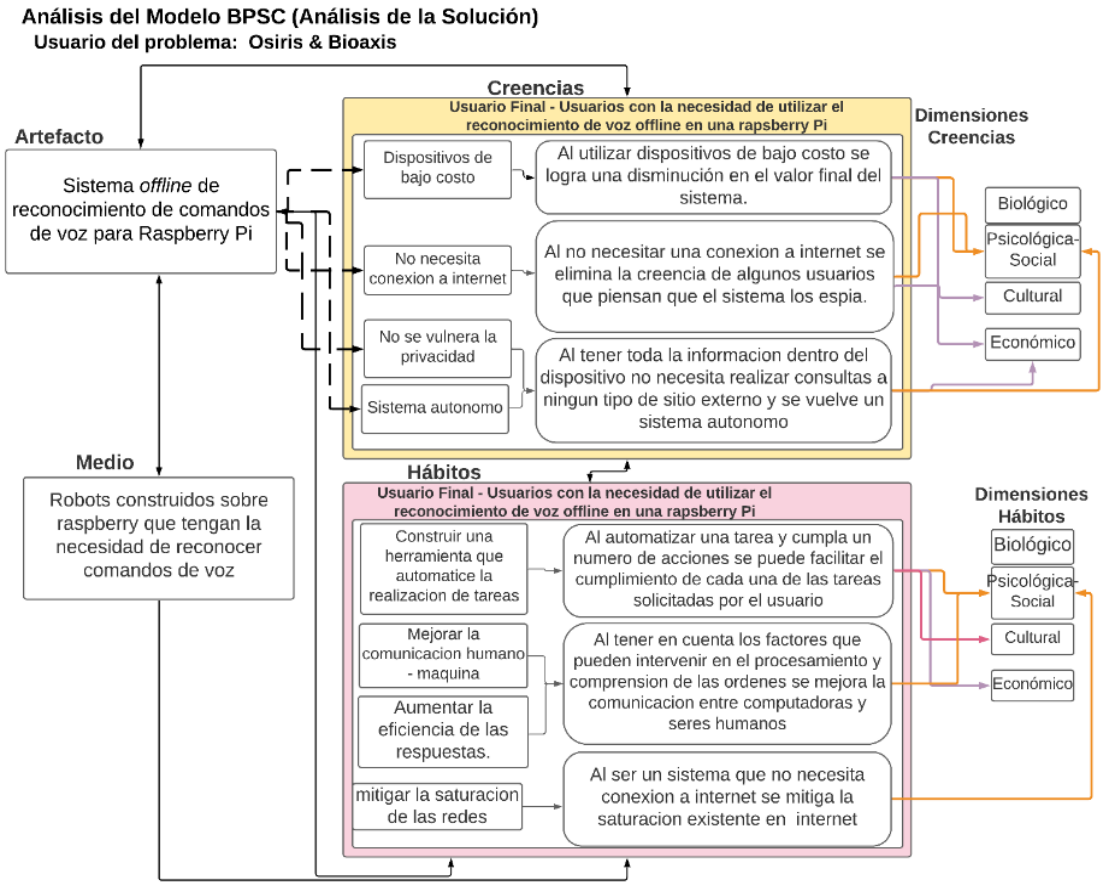
Evaluar la respuesta del robot ante los comandos de voz para determinar su eficacia (que ejecute efectivamente la orden) y eficiencia (tiempos de respuesta) para garantizar el ciclo de transferencia tecnológica.

## 2.3 DESCRIPCIÓN DE LA SOLUCIÓN Y RESULTADOS ESPERADOS

El sistema por desarrollar debe permitir el reconocimiento de comandos de voz en idioma español de manera autónoma e independiente de internet, funcionando en un robot con el hardware basado en la arquitectura de Raspberry Pi. De esta forma, se reducirá el tiempo de ejecución para las tareas solicitadas, al no consultar la información en ningún servidor, teniendo toda la información necesaria en el hardware de la Raspberry Pi. Por lo tanto, se brindaría nuevas posibilidades para entornos o artefactos que no tienen conexión a internet o donde la velocidad de las respuestas es fundamental. La solución será medida con las variables dinámicas: eficacia, que el robot ejecute los comandos correctamente; y eficiencia, el tiempo necesario para realizar cada desplazamiento.

## 2.4 ANÁLISIS DE LA SOLUCIÓN DESDE EL MODELO BPSC

**Figura 3. Solución BPSC**




Fuente: Elaboración propia

Con la problemática identificada, se genera un cambio en la creencia al no vulnerarse la privacidad de cada usuario, al no necesitar hacer consultas a un medio externo o enviar información o datos relevantes a la nube, así como también, se obtiene un cambio en el hábito evitando la saturación de las redes. Lo anterior hace referencia a que se puede desarrollar un sistema de RV utilizando diversas tecnologías de bajo costo. Como también, con la implementación de las variables de medición se busca bajar los tiempos de respuesta sin afectar la eficacia del sistema, considerando de manera crucial las variables eficiencia y eficacia para el proyecto, ante las cuales se busca una alta precisión.

Junto con el cliente se definió un enfoque a abordar, producir cambios favorables tanto en los hábitos como en las creencias al ofrecer un sistema de RV autónomo y seguro para entornos robóticos (Raspberry Pi). Dando como resultado un sistema, a disposición del grupo de investigación Osiris & Bioaxis para ser presentado en contexto de avances tecnológicos a través de la investigación. Logrando

implementarlo en una tarjeta u ordenador de placa reducida como una Raspberry Pi. Para poder entender de una manera gráfica clara y sencilla lo antes mencionado se recomienda ver la Figura 3 donde se evidencia la solución prevista desde el modelo biopsicosocial y cultural.

## 2.5 TABLA DE ENTREGABLES

Los artefactos descritos en la Tabla 1 permiten llevar un control de lo que espera el cliente recibir al final del desarrollo del proyecto.

**Tabla 1. Entregables esperados**

| Nombre | Descripción |
|---|---|
| Sistema de reconocimiento de comandos de voz. | El sistema debe estar implementado en el robot Kenito. De igual forma se entrega el código fuente. |
| Modelo de aprendizaje automático. | El modelo entrenado y evaluado. Esto también incluye los datos utilizados para el entrenamiento. Compilado para ser usado con Tensorflow Lite y Google Acelerador USB Coral. |
| Diagramas UML. | Para una mejor comprensión de la solución, se entregarán los siguientes diagramas: casos de uso, secuencia, componente y despliegue. |

Fuente: Elaboración propia

## 2.6 VARIABLES A MEDIR

Para el desarrollo del sistema no solo es importante trabajar en la parte técnica, también es fundamental garantizar que el cliente (Osiris & Bioaxis) reciba artefactos funcionales, para cumplir con las necesidades solicitadas. Es por ello, que a continuación se definen y presentan las variables en la academia, estáticas y dinámicas.

### 2.6.1 Variable en la académica

Este tipo de evaluación permite asegurar el correcto funcionamiento del desarrollo en una etapa temprana, con el objetivo de corregir o evitar errores. En este proyecto se tienen en cuenta dos variables como se puede observar en la Tabla 2. Reif et al. [7] en referencia al rendimiento mencionan la relevancia de este en el procesamiento de modelos de aprendizaje automático sobre dispositivos de embebidos de bajo costo. Por otro lado, Peng et al. [8] destacan la importancia de que los sistemas desarrollados en ambientes científicos adopten técnicas para asegurar la calidad del desarrollo, tales como: pruebas unitarias y regresiones funcionales.

**Tabla 2. Validación en la academia**

| Variable | Indicadores |
|---|---|
| Rendimiento | Cantidad de recursos necesarios (tiempo) para la ejecución del modelo RV. Reif et al. [7] |
| Calidad | Representa el porcentaje que las pruebas unitarias cubren en el código desarrollado, a fin de asegurar la calidad del programa para evitar errores de funcionamiento. Peng et al. [8] |

Fuente: Elaboración propia

### 2.6.2 Variables estáticas

De acuerdo a la modalidad del proyecto (desarrollo tecnológico) y la metodología escogida (SCRUM), periódicamente se han mantenido reuniones con el cliente beneficiario Osiris & Bioaxis. Uno de los objetivos de estos encuentros es poder socializar los avances con el cliente. Esto se ha realizado a través de los indicadores exactitud y precisión ubicados en la Tabla 3, ampliamente usados en la evaluación de modelos de aprendizaje automático.

**Tabla 3. Validación estática**

| Nombre | Indicadores |
|---|---|
| Exactitud | Representa la calidad del modelo para clasificar los comandos de voz. Brena et al. [9] |
| Precisión | Describe la capacidad que tiene el modelo de reconocer correctamente los comandos de voz. Brena et al. [9] |

Fuente: Elaboración propia

### 2.6.3 Variables dinámicas

Para que el ciclo de transferencia tecnológica sea concretado, se debe demostrar que el sistema cumple con el cuarto objetivo específico del proyecto. Por lo tanto, se definen las variables de medición descritas en la Tabla 4.

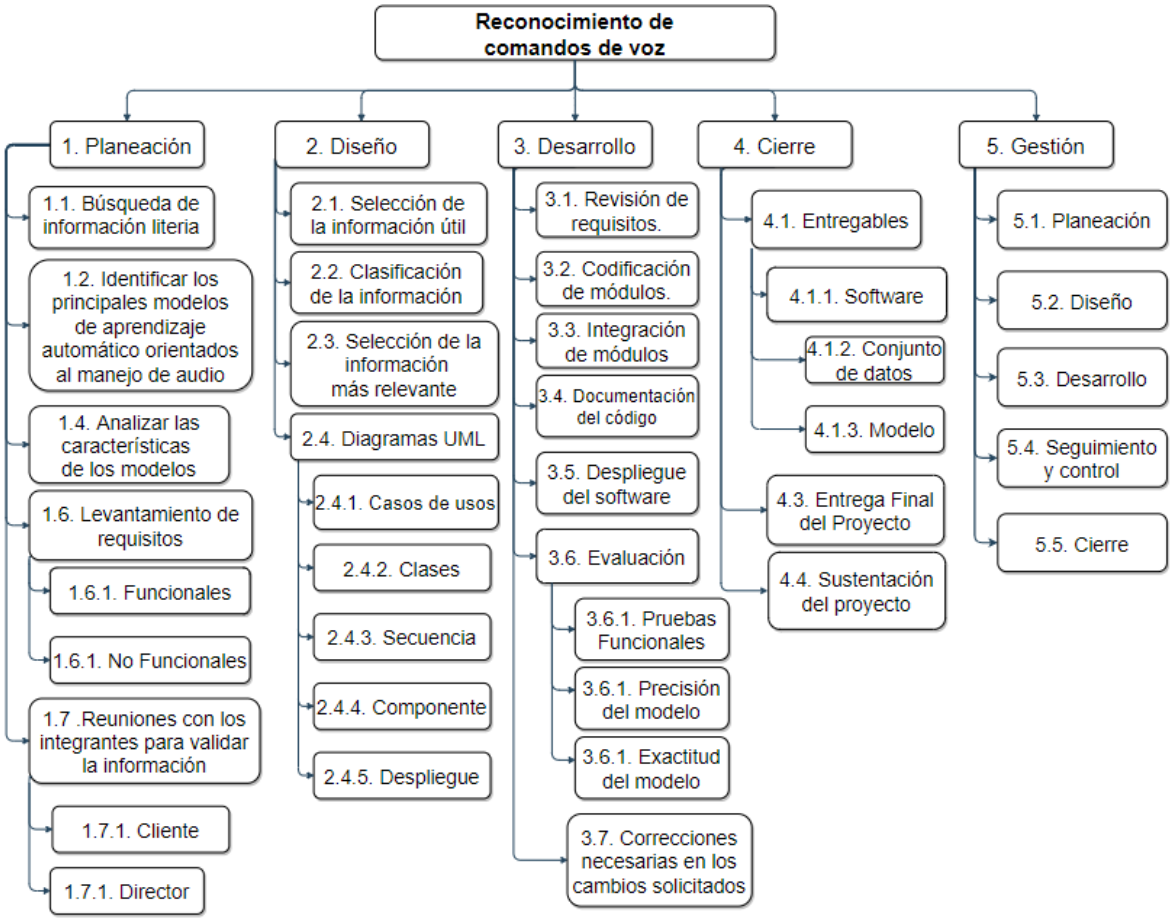
**Tabla 4. Variables para asegurar la transferencia tecnológica**

| Nombre | Indicadores |
|---|---|
| Eficiencia | Precisión del programa integrado al robot, medido a partir de pruebas con usuarios. Sharan et al. [10] |
| Eficacia | Latencia de los procesos ejecutados en el reconocimiento de voz. Oh et al. [11] |

Fuente: Elaboración propia

## 2.7 METODOLOGÍA

**Figura 4. EDT**




Fuente: Elaboración propia

A fin de realizar un proceso de desarrollo de sistema coherente con el proyecto, es necesario la implementación de una o varias metodologías. Estas pueden determinar el éxito o fracaso del proyecto, debido a que muchas veces es más importante la gestión del flujo del trabajo que las habilidades propias de los integrantes. El presente trabajo está basado en gran medida en el aprendizaje automático. La construcción de modelos de ML por lo general tiende a ser cíclica y compleja, es decir, un modelo ya entrenado puede necesitar volver a ser entrenado. Amershi et al. [12] advierten de estas complejidades que integran la construcción de sistemas de ML, estos no son lineales, y se diferencian del desarrollo de los sistemas tradicionales. Los sistemas de ML giran en torno al modelo, manejo de los datos, entrenamiento del modelo, la evaluación del modelo, entre otros. Estos procesos son dinámicos, necesitan repetirse en múltiples ocasiones para obtener un modelo de calidad. Rivero et al. [13] afirman que la evolución del ML obliga a que también los procesos de desarrollo evolucionen. Ellos aplicaron SCRUM en su

proyecto de ML, obteniendo resultados considerablemente exitosos, alcanzaron una precisión del 93.89%; 95,58% de especificidad; 88,84% de sensibilidad; y una precisión de 87,09%.

En consecuencia, para el presente desarrollo tecnológico, se elige la metodología ágil SCRUM, sustentada en los éxitos obtenidos por otros equipos en este campo de ML. Así mismo se tuvo en cuenta que la metodología ASUM-DM está basada en la metodología CRISP-DM y están orientadas a la minería y analítica de datos, como menciona la alianza Caoba. [**14**] estas metodologías son ampliamente conocidas en la realización de proyectos con enfoque en la minería de datos, donde su principal objetivo es poder encontrar información oculta en grandes cantidades de datos, los cuales no son posibles de identificar con formas o métodos estadísticos comunes, siendo el proceso realizado por ambas metodologías como explotación o minería de datos, por tal motivo según la finalidad del proyecto la metodología más acorde es SCRUM, resaltando que los datos no se obtuvieron a través de la minería de datos, sino mediante un formulario y un módulo desarrollado para tal fin, en esa misma línea, esta parte de obtención de datos sólo significó una etapa del proyecto y no a la totalidad del mismo. Finalmente, esta metodología también se adapta y contribuye al avance de los paquetes de trabajos presentados en la EDT de la Figura 4.

### 2.7.1 Estructura de desglose de trabajo

Esta estructura de trabajo permite dividir jerárquicamente los entregables correspondientes a cada etapa del proyecto. En dichas etapas, se tiene claro que se debe entregar, a fin de cumplir con objetivos generales y específicos del proyecto.

### 2.7.2 Aplicación de la metodología

La metodología SCRUM es un marco de trabajo ágil, eso significa que es flexible en su implementación. En el presente proyecto se tiene en cuenta el siguiente proceso: planificación, revisión, retrospectiva, y reuniones diarias. Para una mejor aplicación de la metodología, se eligió la aplicación tecnológica Jira de la compañía Atlassian.

#### 2.7.2.1 Planificación

En esta etapa se realiza el listado de tareas (Pila de Producto - PP) que se trabajarán en el ciclo, este hace referencia al tiempo de ejecución de las actividades. Las tareas reciben el nombre de historias de usuarios (HU). Las HU son asignadas a los integrantes, conjuntamente se determinan los esfuerzos y prioridades necesarias. En la Tabla 6 se presenta el PP definido para este proyecto junto con el número de historias por ciclo. Para la estimación de esfuerzos se puede usar la secuencia de Fibonacci de 1, 2, 3, 5, 8 y 13. Entre mayor sea el número en la secuencia, más esfuerzo requiere la HU. Las historias de usuarios deben ser

explícitas para su correcta realización por parte del responsable, en la Tabla 5 se observa una de estas con su descripción y criterios de aceptación.

**Tabla 5. Ejemplo de historia de usuario**

| **Diagramas UML** |
|---|
| **Descripción** |
| Como cliente quiero poder entender visualmente la estructura del sistema para entender cómo está construido. |
| **Criterios de aceptación** |
| Dado que soy un cliente y quiero entender el flujo de interacción del sistema, entonces debo poder ver el diagrama de secuencia.<br>Dado que soy un cliente, quiero conocer los componentes del sistema a través de un diagrama de componentes.<br>Dado que soy un cliente con la necesidad de entender las acciones realizadas por el sistema, se debe poder visualizar casos de uso.<br>Dado que soy un cliente, debo saber cómo serán desplegados los componentes mediante un diagrama de despliegue. |

Fuente: Elaboración propia

**Tabla 6. Listado de tareas con asignaciones**

| Nombre | Responsable | Ciclo |
|---|---|---|
| Investigación sobre modelos de aprendizaje automático. | Andrés Villegas | 1 |
| Diagramas UML. | Ibrahimme Morelo | |
| Desarrollo del modelo | Ibrahimme Morelo | |
| Crear módulo para aumentar los datos. | Ibrahimme Morelo | 2 |
| Obtener los datos de entrenamiento. | Andrés Villegas | |
| Aumentar los datos de entrenamiento. | Andrés Villegas | |
| Entrenar el modelo RCV. | Ibrahimme Morelo | 3 |
| Crear módulo para capturar y clasificar audios. | Andrés Villegas | |
| Validaciones en la academia. | Ibrahimme Morelo | |
| Validaciones estáticas. | Andrés Villegas | 4 |
| Desplegar el modelo en el robot (Kenito). | Ibrahimme Morelo | |
| Experimentos para validaciones dinámicas. | Ibrahimme Morelo | 5 |
| Documentar el código. | Andrés Villegas | |
| Deuda técnica | Ibrahimme Morelo | |

Fuente: Elaboración propia

#### 2.7.2.2 Revisión

Este punto constituye una fase importante dentro de la metodología, porque revisa las HU completas durante el ciclo. En función de los resultados obtenidos se realizan realimentaciones al proyecto, para no comprometer su éxito. Las revisiones se realizan de acuerdo con la duración del ciclo, en este caso cada 4 semanas. El cliente es un actor fundamental para la ejecución de esta actividad. A raíz de estas

reuniones se obtuvieron recomendaciones y acuerdos con el cliente beneficiario. Se acordó la realización del modelo de ML con TensorFlow y su despliegue en el robot Kenito con formato TensorFlow. También se determinó obtener los datos a través de formularios y aumentar estos mismos con la modificación de las características del audio cómo: velocidad, tono, ruido, duración, entre otras.

#### 2.7.2.3 Retrospectiva

La reflexión de lo bueno y malo hecho durante el ciclo permite mejorar aspectos de cara al futuro inmediato del proyecto. Para este desarrollo tecnológico los involucrados son los autores.

#### 2.7.2.4 Reuniones diarias

A partir de estas reuniones realizadas cada semana, se sincronizan esfuerzos y se eliminan bloqueos. Aspectos en cuanto a la construcción del modelo de ML, diagrama UML, y obtención de datos se han resuelto durante esta ceremonia.

### 2.7.3 Cronograma

En la Figura 5 se contempla la distribución del cronograma para el desarrollo tecnológico propuesto. Contemplando un tiempo de vida de veinticinco (25) semanas. Iniciando en la semana uno (1) con la fase de planificación y finalizando con la entrega de sistema en la semana veinticinco (25). Dando culminación al proyecto del desarrollo tecnológico propuesto.

**Figura 5. Cronograma**

| Actividad | Semana | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Inicio | Fin | Duración | | | | | | | | | | | | | | | | | | | | | | | | | |
| Etapa de planeación | 1 | 3 | 3 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Revisión de requisitos | 4 | 5 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Investigación sobre modelos para reconocimiento de voz | 6 | 7 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Elaboración de diagramas UML | 7 | 9 | 3 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Desarrollo del modelo | 8 | 9 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Obtención/Tratamiento del conjunto de datos | 10 | 13 | 4 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Entrenamiento del modelo | 15 | 16 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Desarrollo de los módulos e integración del modelo en ambiente de pruebas | 15 | 17 | 3 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Evaluación en la academia | 17 | 17 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Evaluación estática | 18 | 18 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Integrar el progama al robot | 19 | 21 | 3 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Evaluación dinámica | 22 | 23 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Documentación del código | 24 | 24 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Sustentación del proyecto | 25 | 25 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | Ciclo 1 | | | | Ciclo 2 | | | | Ciclo 3 | | | | Ciclo 4 | | | | Ciclo 5 | | | |

Fuente: Elaboración propia

## 2.8 ACUERDO CON EL CLIENTE

La distribución de derechos patrimoniales y morales se encuentran en el Anexo 1.

## 2.9 COMPONENTE ÉTICO

Dando cumplimiento al apartado ético, el desarrollo de este trabajo tiene como finalidad el poder ayudar mejorando la comunicación entre las computadoras y los humanos, Orientando el progreso y cumplimiento de cada fase del proyecto bajo los principios éticos del código de ética ACM, generando un enfoque en tres de los ocho principios fundamentales estipulados por Gotterbarn et al. [15], siendo los principios para resaltar; la sociedad, el cliente y el producto, dada la finalidad del proyecto en el entorno de la ingeniería, permitiendo bajo el código de ética una conducta adecuada e íntegra ante el cómo se debe actuar como ingenieros y como profesionales en el desarrollo de software durante el transcurso investigativo, de diseño, elaboración y entrega del sistema de reconocimiento de voz.

Bajo el primer principio basado en la importancia ante la sociedad, se enfatizan los intereses del cliente con el desarrollo del proyecto y el bien social. El propósito del mismo es poder contribuir en la elaboración de un sistema desconectado de bajo costo accesible a un mayor número de personas. Logrando así un sistema que contenga toda su información embebida y totalmente autónoma en una Raspberry Pi sin depender de alguna conexión a internet, respetando y evitando de esta manera la vulneración a la privacidad de cada uno de los usuarios finales y dando cumplimiento al bien social.

Mediante la implementación del segundo y tercer principio del código orientados al cliente y el producto, se dará cumplimiento a los diferentes objetivos del trabajo sin incumplir ningún aspecto ético en cuanto a ser honestos y confiables. Así como el mantener un alto estándar de competencia profesional. Siendo íntegros en la conducta y práctica ética, manteniendo secreta cualquier información confidencial adquirida a lo largo del proyecto. De la misma manera se usará de forma consciente y propiamente permitida el software y hardware del cliente bajo previa autorización cada vez que este sea necesario para la realización de pruebas.

# 3. MARCO REFERENCIAL

Para poder justificar el desarrollo tecnológico del presente trabajo, así como también su alcance, a continuación, se presentarán: los antecedentes, estado del arte, y marco teórico.

## 3.1 ANTECEDENTES Y ESTADO DEL ARTE

Para la realización y conceptualización del presente desarrollo tecnológico se tuvieron en cuenta algunos trabajos realizados con anterioridad, esto en el marco de la temática principal del proyecto.

### 3.1.1 Reconocimiento de voz

La comunicación humano-máquina se ha convertido en un campo de investigación de gran interés en la literatura tecnológica a nivel mundial, más específicamente la comunicación a través de voz. Esto se debe en gran parte a los grandes avances tecnológicos que se han dado en los últimos años, un claro ejemplo de ello, es el uso y mejoría de los modelos de inteligencia artificial. Lo cuales le han concedido una superioridad considerable al reconocimiento de voz (RV) sobre cualquier otro método de comunicación entre humano y máquinas. Es así que Abdulkareem et al. [16] presentan un sistema de RV para interactuar con electrodomésticos y puertas. Para tal desarrollo hacen uso del sistema embebido Raspberry Pi (RPi), las tecnologías de Google para almacenamiento en la nube, modelos Markov y procesamiento de señales digitales. Con 150 muestras de voz, el sistema logró alcanzar una eficiencia del 80%, un porcentaje considerado alto. Por otro lado, Patel et al. [17] muestran la implementación de RV para manipular un robot que cuenta con un brazo, se hace énfasis principal en el reconocimiento de comandos de voz (RCV) en ambientes ruidosos, cómo, por ejemplo: ruido de carretas, aires acondicionados, y música. Para la fase de entrenamiento, se tuvo en consideración tanto hombres como mujeres, para finalmente alcanzar una eficiencia del 89%.

### 3.1.2 Computación al borde la red (*Edge computing*)

Actualmente los asistentes virtuales cómo Siri, Alexa, y Google Assistant viven un gran auge debido a su habilidad de comunicación a través de voz, pero estos artefactos y en general los sistemas RV tienen algo en común, se trata de nada más y nada menos que una conexión a internet. Estas tecnologías se basan en el poder computacional que ofrece la nube, sin embargo, esto ha generado una carga exponencial en los servidores y en algunas ocasiones, malas experiencias de usuario. Afortunadamente el crecimiento de otras tecnologías menos costosas cómo las tarjetas electrónicas, teléfonos inteligentes, computadoras personales, y demás, han otorgado alternativas al uso de la nube. A lo anterior se le conoce cómo la computación en el borde de la red, es decir, el procesamiento de los datos cerca de lugar de origen. En ese sentido, Cheng et al. [18] coinciden con esta visión del

problema, lo hacen esencialmente para aliviar la carga del servidor y mejorar la calidad de las experiencias de los usuarios (EU). Ellos implementan un algoritmo de red neuronal convolucional en equipos locales para la extracción de datos de audios, teniendo en cuenta las restricciones computacionales de cada equipo. Al usar un sistema distribuido local se logra mejorar la carga computacional, por consiguiente, el retraso total de la red disminuye. En esta misma línea, Yang et al. [19] nos expone el desarrollo del sistema llamada EdgeRNN, el cual funciona en el borde de la red, haciendo uso también de un algoritmo de red neuronal convolucional. El funcionamiento del sistema es verificado con un conjunto de comando de voz proporcionado por Google y una tarjeta programable Raspberry Pi 3B +. El conjunto anterior, según los autores, mejoró tanto el reconocimiento de palabras claves cómo el reconocimiento de emociones.

### 3.1.3 Aprendizaje automático en RV

Otro factor determinante en el éxito del RV es la inteligencia artificial (IA), el RV tuvo un impacto positivo por la explosión de la IA, aprendizaje automático (AA), y aprendizaje profundo. Radzikowski et al. [20] destacan cómo la eficiencia del RV puede verse afectada cuando los usuarios no son nativos del idioma en que funciona el RV. Esto a raíz de las diferencias en los acentos y pronunciaciones de la lengua original de los usuarios, y la poca disponibilidad de hablantes no nativos al momento de entrenar un modelo. Es aquí donde entra uno de los algoritmos de aprendizaje profundo, la transferencia de estilo neural (TEN), esta permite modificar el acento de los usuarios para asemejarse a los esperados por el modelo. Cómo consecuencia la eficiencia del sistema mejora notablemente, incrementando la precisión de voz. El lenguaje utilizado cómo nativo fue el inglés, las pruebas del algoritmo de TEN fueron realizadas con hablantes no nativos de Japón. Este desarrollo es bastante importante dado la escasez de datos de formato auditivo que se pueden presentar con bastante frecuencia en el entrenamiento de los modelos.

Las redes neuronales convolucionales (*CNN*) y memorias largas a corto plazo (*LSTM*) son otros de los algoritmos de inteligencia artificial usados para mejorar el RV. Por ello, Sabab et al. [21] usan estos algoritmos para el desarrollo de un modelo de RV en idioma bengalí. Aquí los audios son convertidos a espectrogramas, para posteriormente ser clasificados con RNC y LSTM. Antes de realizar la clasificación de los datos, éstos pasan por un proceso de reducción de dimensiones (RDD), a fin de mejorar la precisión del modelo. Con la RDD la eficiencia alcanza valores de 94.58% y 83.12% para CNN y LSTM respectivamente. Mientras que sin RDD los valores son 97,26% y 93,83 respectivamente. Por lo que concluyen que las CNN y LSTM por sí solas tienen mejores resultados que al usar RDD.

En la literatura se puede encontrar que las CNN son ampliamente usadas para el reconocimiento de comandos de voz o palabras claves. Warden et al. [22] describen su proceso realizado desde la recolección de datos hasta el entrenamiento del modelo con CNN, este último es usado como referencia en la página oficial de

TensorFlow, lo cual demuestra su relevancia. La exactitud del modelo de los autores alcanzó un porcentaje del 88.2%, siendo este un porcentaje significativamente relevante. Por otro lado, los autores usaron grabaciones de ruidos para filtrar el conjunto de datos, a fin de mejorar la calidad de los audios y posterior reconocimiento de voz. Este avance ha servido como fuente de inspiración para posteriores investigaciones y proyectos de innovación. El idioma del conjunto de datos empleado por los autores es el inglés, como la gran mayoría de los existentes. Para necesidades en países de habla hispana, por lo general esto siempre es un inconveniente dada la ausencia de conjunto de datos en español, esto crea la necesidad de hacer recolecciones de datos desde 0.

Van der Westhuizen et al. [23] también usaron CNN para el reconocimiento de comandos de voz, esta decisión la tomaron con base en la eficiencia de este tipo de redes. Dicha característica hace a las CNN apropiadas para aplicaciones en tiempo real, justo como la que desarrollaron los autores. Ellos implementaron su modelo entrenado en una Raspberry Pi para la identificación de palabras de interés en un sistema de radio abierto. Finalmente, el autor concluye que este acercamiento contiene un rendimiento competitivo, que se puede adaptar a muchos escenarios de bajo costo.

De acuerdo a los anteriores antecedentes relacionados al uso de aprendizaje automático para el RV/RCV en conjunto con Raspberry Pi, se decidió avanzar con las CNN para el presente proyecto. Esto a raíz de las necesidades de este mismo, las cuales incluye contar con un modelo de ML que funcione de forma eficiente y eficaz en un robot construido sobre componentes de bajo costo. Con esta elección se da cumplimiento al objetivo específico 1; la selección del modelo de ML soportada en la revisión de antecedentes o estado del arte.

### 3.1.4 ML aplicado en robots

El RV en entornos ruidosos no es tan efectivo, sobre todo cuando hablamos de robots, los ruidos emitidos por las partes de estos son un problema para el RV. Por tal razón, Nagoev et al. [24] propone un modelo basado en una arquitectura cognitiva recursiva multiagentes, dicho modelo le permite al sistema de reconocimiento de voz separar y analizar los sonidos unos de otros, para poder conocer quién es el hablante. Por lo tanto, el nivel de eficacia de los sistemas de RV puede aumentar haciendo uso de esta arquitectura. Sin embargo, para el presente desarrollo tecnológico se parte del hecho de que el sistema será contextualizado en un entorno controlado.

Indudablemente el reconocimiento de voz es un campo que cuenta con un gran interés por parte de la comunidad científica, es así como numerosas investigaciones giran en torno a este tipo de sistemas. Por ello, es importante presentar una alternativa de bajo costo que pueda brindar: autonomía, seguridad, velocidad de respuesta, y finalmente una buena experiencia de usuario.

### 3.1.5 ML con TensorFlow

La construcción de modelos de ML es más sencilla hoy en día gracias a las múltiples herramientas que hay a disposición, una de ellas es TensorFlow, orientada específicamente al aprendizaje automático, esta biblioteca de código es abierta y pertenece al gigante tecnológico Google, es un recurso altamente usado en ambientes académicos e investigativos. Park et al. [25] la usa en la construcción de un modelo médico para la predicción de enfermedades. Dicho modelo hace parte de un modelo más grande, es decir, se combina con otros modelos construidos sobre otras tecnologías, esto demuestra la interoperabilidad de Tensorflow. El modelo es entrenado a partir de atributos extraídos de pruebas de laboratorio. El modelo final obtuvo resultados exitosos, con puntuaciones de F1 del 81%, precisión del 92%, en la clasificación de cinco enfermedades.

Por otro lado, Suganeswaran et al. [26] aplican un modelo ML en conjunto con Raspberry Pi, OpenCV y TensorFlow orientado a la visión artificial. Su contribución se encuentra en el marco de la pandemia causada por COVID-19, tanto que este modelo detecta cuando una persona tiene puesta la mascarilla. La imagen obtenida por una cámara ubicada en la Raspberry Pi es clasificada a través del modelo previamente entrenado con un conjunto de datos de imágenes de personas con y sin mascarillas. Esta solución es propuesta para lugares públicos donde el flujo de personas sea alto.

### 3.1.6 Metodología SCRUM

Como se observó en la sección de metodología previamente expuesta. SCRUM puede ser usado en proyectos de aprendizaje automático, porque contribuye a la flexibilidad que requieren estos mismos [12][13]. SCRUM se adapta mejor al desarrollo de modelos ML en comparación a los métodos tradicionales. La complejidad e incertidumbre de las tareas involucradas en los procesos de desarrollo de aprendizaje automático es mayor que el desarrollo de otros sistemas.

### 3.1.7 Aumento de datos

En la etapa de generación de los conjuntos de datos también encontramos avances investigativos. Estos apuntan a la reproducción de nuevos datos a partir de datos bases, es decir, aumento de datos (*Data Augmentation*). Su relevancia recae en la dificultad de encontrar datos de entrenamientos pertinentes para un modelo, por ejemplo, encontrar datos para algunos lenguajes es una tarea titánica. Kadyan et al. [27] desarrollaron un sistema de reconocimiento de voz para niños en idioma panyabí. Se encontraron con el problema de no tener suficientes datos para entrenar el modelo, intentaron usar audios grabados por adultos, pero el resultado fue un modelo que no reconocía a los niños. Las características acústicas de los adultos son diferentes a la de los niños, sin embargo, a través del aumento de datos,

modificaron las características del audio: tono, duración, y longitud del tracto vocal. Los cambios lograron alcanzar similitudes con el habla de los niños. Con estos nuevos datos, el modelo concretó tasas de error significativamente inferiores.

### 3.1.8 Evaluación de sistemas

El estudio de Reif et al. [7] estuvo enfocado en predecir el impacto que tiene el uso del acelerador USB Coral de Google, en términos de tiempo y consumo energético. Los autores consideraron clave la medicación de estos factores dado los recursos limitados con los que operan los sistemas embebidos. Dicho enfoque se ve aún más reforzado si tenemos en cuenta la carga de trabajo de los modelos de aprendizaje automático. Para este proceso de estimación, ellos crearon una configuración física, a fin de automatizar la medición. Con los datos obtenidos, entrenaron un modelo de ML, con la meta de predecir el comportamiento de las variables elegidas sobre la ejecución de una red del tipo CNN. El estudio concluyó que es valioso y viable predecir el comportamiento del tiempo de ejecución que tienen las redes neuronales en dispositivos dedicados como Google Coral.

Puchtler y Peinl [28] en su trabajo miden el rendimiento también en términos de tiempo y consumo energético que tiene el uso de modelo de ML en sistemas embebidos. Su enfoque varía en la comparación y combinación de diferentes dispositivos, tales como: Raspberry Pi 4, Google Coral, Nvidia Jetson Nano y Intel Neural Compute Stick 2. Estas tecnologías fueron empleadas para la ejecución de un modelo de ML, en sus resultados, se encontraron que la Google Coral fue la que mejor tiempo obtuvo, por el contrario de la Raspberry Pi 4, que fue la más lenta.

Por otro lado, Peng et al. [8] resaltan la importancia de la calidad del software en ámbitos científicos. Es por ello que, dentro de las pruebas realizadas por los autores, se encuentran las pruebas unitarias y pruebas de regresión. A través de la implementación de estas pruebas alcanzaron un 39,7% de cobertura de código, mientras que con las pruebas de regresión el valor de fue de 44,9% de cobertura. Los resultados anteriores se dieron gracias a un programa orientado a la atomización de pruebas.

Como se puede ver según la literatura, conocer el tiempo de inferencia de un modelo sobre diferentes arquitectas es importante para la toma de decisiones. Contar con dispositivos como Google Coral aparte de mejorar los tiempos también ayuda a reducir la carga de trabajo que tienen las Raspberry Pi en este sentido. Por el lado de la calidad, es importante contar con métricas que la aseguren en una etapa temprana, las pruebas unitarias son una buena forma de asegurar que el código desarrollado como se espera

### 3.1.9 Evaluación de Modelos

Brean et al. [9] en su trabajo orientado a la evaluación automática de fluidez y pronunciación en inglés, hacen uso del aprendizaje automático. A través de un modelo de ML clasifican en 3 clases el nivel de inglés; bajo, medio y alto. El modelo estuvo basado en muestras de audio, que permitían extraer un conjunto de características relevantes para la fase de entrenamiento. Para medir la calidad de los resultados, usaron dos variables de medición sobre el modelo entrenado, por un lado, la precisión alcanzó un 99.9% y la exactitud un 85%. Estos dos componentes son de gran insumo para la medición de modelos de ML, son ampliamente usados en trabajos de ML.

### 3.1.10 Evaluación de Sistemas de Reconocimiento de voz.

Sharan et al. [10] en su trabajo presentan un sistema de reconocimiento de voz para controlar un robot llamado Roswitha a través de comandos. Ellos destacan el interés latente y naciente de los seres humanos por controlar robots a través de la voz, hasta el punto de convertirse en un tema de gran relevancia en el desarrollo de robots asistentes. Para la evaluación realizan experimentos en tiempo real a fin de medir la precisión del desarrollo en diferentes entornos.

Por su parte, Oh et al. [11] plantean un servidor de RV en idioma coreano, con el objetivo de medir y disminuir la latencia de las respuestas. Esto lo logran a través de un detector de voz basado en aprendizaje profundo, y el procesamiento en paralelo de fragmentos de audios. Finalmente, los tiempos de respuesta obtenidos por el trabajo fueron de 3,09 segundos a 5,41 segundos, tiempos notablemente mejores en comparación con los 11,7 segundos inicialmente obtenidos por los autores.

Dado que uno de los enfoques del proyecto fue que trabaje de forma offline, la comparación de tiempos de respuesta con sistemas basados en la nube fue importante, para verificar si estos mejoran. Sin embargo, también se verificó que una reducción de tiempo no afectara la precisión (eficacia) del sistema, es decir, no disminuir tiempos a costa de sacrificar otros aspectos del desarrollo. A través de estas validaciones se completó el ciclo de trasferencia tecnológica, debido a que estas estuvieron orientadas a los usuarios del sistema (validación dinámica).

## 3.2 MARCO TEÓRICO

A continuación, se presentan las consideraciones teóricas necesarias para una mejor comprensión del proceso de desarrollo del proyecto.

El reconocimiento de comandos de voz (RV) se ha convertido en una gran tecnología para comunicarse con los robots. Al mismo tiempo, se pueden encontrar

grandes opciones en el mercado que sirven para tal propósito, cómo lo son Alexa, Siri, Google Assistant, entre otros. Sin embargo, la gran mayoría están basadas en la nube, es decir, dependen de una conexión a internet para su correcto funcionamiento. S. Cheng et al. [18] proponen un sistema de RV para redes móviles, combinando la nube y el borde de la red para minimizar la carga del servidor y mejorar los tiempos de respuesta. No obstante, el uso de la nube para procesos de RV puede deteriorar la experiencia de usuario. Destacando que cada vez es más común que las tecnologías que se usan a diario, estén conectadas a internet. De esta forma el ancho de banda, la latencia, y la rentabilidad, G. Sun et al [29] se han convertido en principales temas de interés a la hora de proponer soluciones que involucren el RV.

La computación al borde de la red (CBR) se está posicionando cómo una aliada relevante para la ejecución de tareas complejas en dispositivos relacionados con el internet de las cosas (IoT). Es tanto así, que es considerada como una de las grandes tendencias tecnológicas para los próximos años. Como lo indica G. Chalapathi et al. [30] uno de sus principales aportes es que intenta evitar que las redes se saturen, puesto que procesa los datos generados por los usuarios tan cerca cómo sea posible. Brindando, en ese sentido, mejores experiencias de usuarios, mayor seguridad de los datos, menos costos de procesamiento y análisis de datos en tiempo real.

Sin embargo, los dispositivos de IoT cuentan con recursos que por lo general son limitados, por ello Yao et al. [31] presentan un sistema basado en CBR que busca poder ejecutar cantidades de datos considerables en estos dispositivos, se propone un mecanismo a través del cual los datos son codificados antes de ser ejecutados en los dispositivos finales, de esta forma los costos en términos de peso se convierten en insignificantes. Así mismo, para más profundidad los datos pueden ser decodificados por servidores ubicados en nube; el proceso de codificación y decodificación demostró ser muy útil con pérdidas de 1% de eficiencia en algoritmos de redes neuronales.

Los algoritmos de redes neuronales (RN) han evidenciado una notable mejoría en términos de eficiencia en los sistemas de RV. Además, dentro del conjunto de algoritmos pertenecientes a las RN, se encuentran las CNN, estas son especialmente útiles para trabajar con imágenes. Permitiendo que las señales de audio pueden ser convertidas en espectrogramas que representen los cambios de frecuencia a través del tiempo en formato de imagen de 2D. Por lo tanto, el RV puede usar las CNN para la clasificación de audios o RCV con porcentajes altos de eficacia. Z. Mushtaq [32] demuestra con porcentajes de 99,04%, 99,49%, y 97,57% la eficiencia, destacando la alta precisión de estos algoritmos orientados a la temática principal de este proyecto, el RCV. Esto lo lograron con diferentes conjuntos de datos, diferentes capas correspondientes a CNN, y en gran parte a la cantidad y calidad de los datos. De hecho, se establece una relación proporcional entre la cantidad de datos y la precisión del modelo.

El uso de tarjetas programables (TP) son una buena herramienta para el desarrollo de robots de bajo costo en comparación con otros dispositivos electrónicos. Sin embargo, aún dentro del mundo de las TP existen alternativas más o menos costosas en función de sus características. En la Tabla 7 se encuentran distintas opciones presentes en el mercado, todas ellas con características relativamente similares. La Jetson Nano [33] y Odroid [34] presentan los procesadores más potentes, lo cual las convierte en las más costosas. Al mismo tiempo banana Pi [35] posee características similares y con un precio inferior. Sin embargo, la Raspberry Pi 4 [36] no queda muy lejos de ellas, con un precio más justo en cuanto al concepto de calidad-precio, por otra parte, la TP más pobre en cuanto características es sin duda alguna la Orange Pi Zero [37], de ahí que tenga el precio más bajo. En los apartados de conectividad, puertos, salida de video, RAM, GPU, y soporte de cámara las diferencias son notablemente cortas. El procesador determina en gran medida el precio de las TP, por ello al analizar estas características en el marco de lo que pueden ofrecer en función de su precio, las Raspberry Pi 4 se aprecian cómo una excelente opción para ser implantada en el presente proyecto.

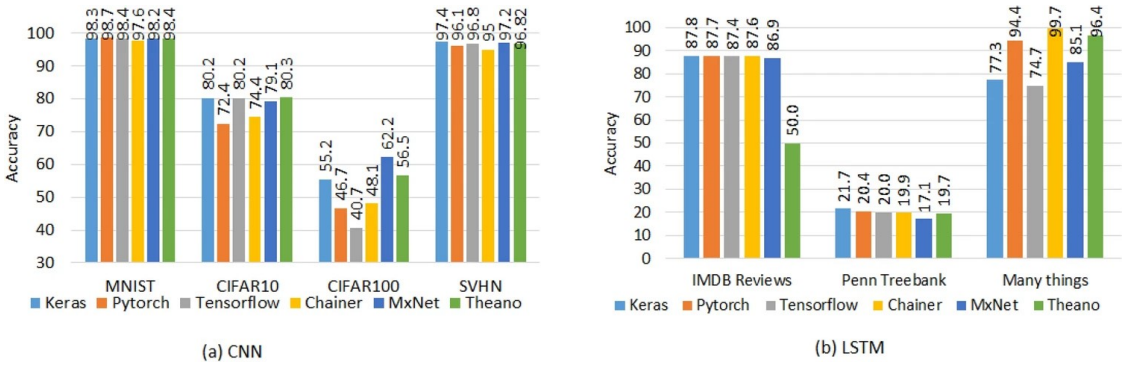
Las CNN son una gran opción para la construcción de modelos orientados al aprendizaje automático. Elshawi et al. [38] realizaron una serie de pruebas entre las CNN y LSTM con las principales librerías de ML: Keras, Pytorch, TensorFlow, Chainer, MxNet, y Theano. De dichos experimentos hay dos variables que soportan la elección de las CNN sobre las LSTM, estas son la precisión y tiempo de entrenamiento del modelo. En la Figura 6 se observa cómo las CNN en general alcanzan mejores porcentajes de precisión frente a las LSTM con conjuntos de datos similares. Para la precisión también se observa que, en las CNN suelen ser más cercanas entre las librerías, con LSTM se aprecian más cambios de precisiones entre las diferentes plataformas. Por otro lado, para los tiempos de entrenamiento de la Figura 7, las CNN necesitan considerablemente mucho menos tiempo que las LSTM. Esto se debe a cómo las CNN normalizan los datos, manteniendo las características más representativas de estos mismo.

## Tabla 7. Principales TP con sus características

| Tarjeta | Raspberry Pi 4 | Jetson Nano | Odroid N2 | Banana Pi M5 | Orange Pi Zero |
|---|---|---|---|---|---|
| **RAM** | 2GB – 4GB – 8GB | 4 GB 64-bit LPDDR4 25.6 GB/s | 2GB – 4GB | 4 GB LPDDR4 | 256MB/512MB DDR3 |
| **CPU** | Broadcom BCM2711, Quad-Core Cortex-A72 64 bits a 1.5GHz | Quad-core ARM A57 @ 1.43 GHz | Amlogic S922X hexa-core | Amlogic S905X3 Quad-Core Cortex-A55 | H2 Quad-core Cortex-A7 H.265/HEVC 1080P. |
| **GPU** | VideoCore VI 3D | 128-core Maxwell | GPU Mali-G31 | Mali-G31 MP2 | Mali400MP2 GPU @600MHz |
| **Conectividad** | Wi-Fi b/g/n/ac (2.4GHz y 5GHz), Bluetooth 5.0 con BLE | Gigabit Ethernet, M.2 Key E | Adaptador USB Wifi | 10/100/1000 Mbit/s Ethernet Optional WiFi USB dongle | XR819, IEEE 802.11 b/g/n y Ethernet RJ45 |
| **Puertos** | 2 puertos USB 3.0 y 2 puertos USB 2.0 | 4x USB 3.0, USB 2.0 Micro-B | 4x USB 3.0 y 1x USB 2.0 OTG | 4 USB | USB 2.0 y USB 2.0 OTG |
| **Salida de video** | 2 puertos micro HDMI (hasta 4K a 60 FPS) | HDMI y DP | HDMI 2.0 y conector RCA | HDMI 2.0 | HD video |
| **Soporte para cámara** | Si | Si | Si | Si | Si |
| **Precios** | ~ 35 USD | 99 USD | 140 USD | 63 USD | 18 USD |

Fuente: Elaboración propia

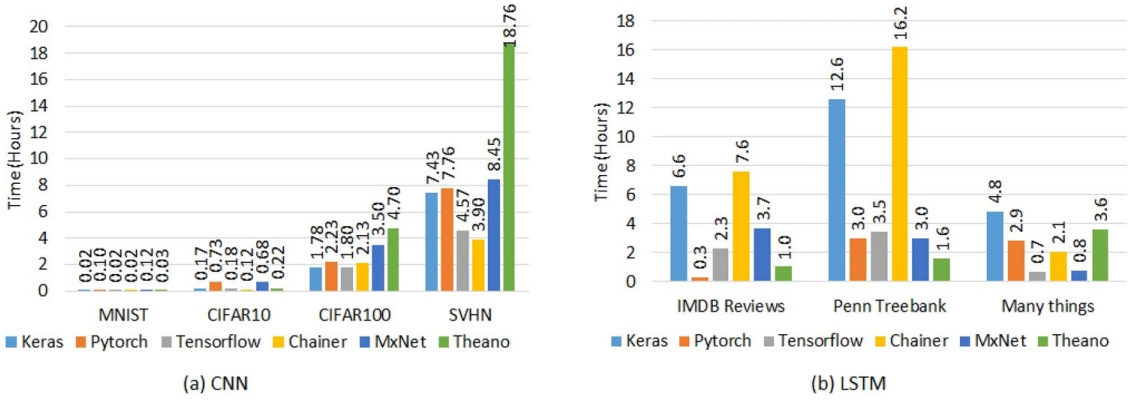
**Figura 6. Precisión CNN vs LSTM**




Fuente: Elshawi et al. [38]

TensorFlow es una herramienta relevante para el desarrollo de sistemas basados en aprendizaje automático, puntualmente, para modelos de reconocimiento de voz. En la Figura 6 se ve cómo TensorFlow logra en la gran mayoría de escenarios resultados sobresalientes en cuanto a precisión se refiere. En algunos escenarios se ve superado, por ejemplo, para el conjunto de datos CIFAR100. Sin embargo, sobresale visiblemente con los tiempos de entrenamiento observados en la Figura 7, tanto para las CNN cómo para las LSTM.

**Figura 7. Tiempo de entrenamiento CNN vs LSTM**




Fuente: Elshawi et al. [38]

# 4. DESARROLLO METODOLÓGICO

En este apartado se explica cómo fue desarrollado el presente proyecto, partiendo del uso de la metodología ágil SCRUM. En coherencia con la metodología, los ciclos se dividieron en cinco, estos son presentados a detalle más adelante. Se muestran los requerimientos funcionales y no funcionales creados a partir de las necesidades del cliente en la etapa de planeación. También se presenta la documentación realizada desde el punto de vista de ingeniería de sistemas: casos de usos, diagrama de secuencia, diagrama de componentes y diagrama de despliegue. No se tienen en cuenta algunos otros diagramas usados en ingeniería de sistema dado el contexto del proyecto, por ejemplo, diagramas de clases y de objetos. El sistema gira en gran medida alrededor del modelo de ML, este tipo de modelo no se destaca por estar construidos en clases y estar orientada a objetos. Posteriormente, los componentes físicos (hardware) así como también las herramientas tecnológicas (sistema) son presentadas y justificadas dado el contexto del proyecto. El desarrollo del artefacto es mostrado y explicado a fin de que el lector pueda comprender cómo se realizó cada detalle de este mismo. Finalmente, se presentan las pruebas realizadas durante y después del desarrollo con el objetivo de asegurar un buen ciclo de transferencia tecnológica.

## 4.1 CICLOS

A continuación, se presenta la explicación de los ciclos empleados para el proceso de desarrollo, esta corresponde a lo visto en la Tabla 6.

### 4.1.1 Primer ciclo

En este ciclo, el foco principal fue orientar parte del estado del arte a la investigación de modelos empleados en el RV, al fin de seleccionar uno para su implementación en el proyecto. Por otro lado, se representó desde el punto de vista de la ingeniería de sistema, el funcionamiento del artefacto objetivo. Por último, se desarrolló el modelo de ML para su posterior entrenamiento. Con este ciclo, se pudo dar cumplimento al primer objetivo específico.

### 4.1.2 Segundo ciclo

Fue un ciclo bastante importante, debido a que giró en torno a los datos de entrenamiento, desde su obtención hasta su aumentación. Lo anterior incluyó el desarrollo de un módulo para recolectar audios, otro para el aumento de datos, y el uso de un formulario para la recolección de datos.

### 4.1.3 Tercer ciclo

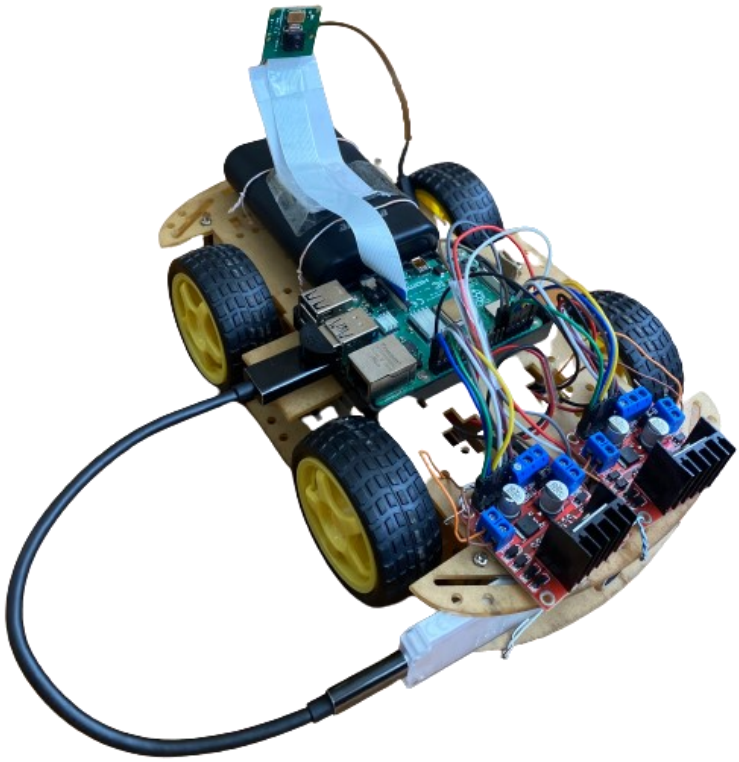
Una vez los datos fueron obtenidos, el paso más lógico fue entrenar el modelo calibrando sus parámetros de entradas. Por otro lado, fueron necesarios dos

módulos: clasificación, para usar el modelo ya entrado; audio, capturar las señales de audio emitidas en tiempo real. Este ciclo permitió el cumplimiento del segundo objetivo específico del proyecto, en referencia a la implementación del modelo. También se realizaron las validaciones en la academia sobre los módulos desarrollados.

### 4.1.4 Cuarto ciclo

El resultado de cada iteración fue presentado al cliente, sin embargo, fue en este ciclo que el cliente pudo comenzar a ver avances significativos. Se presentaron los resultados de las validaciones estáticas del proyecto, a través de las variables precisión y exactitud. Así mismo, el programa fue desplegado en el Robot Kenito en conjunto con el cliente, dicho Robot puede ser observado en la Figura 8. En este punto, fue posible el cumplimento del tercer objetivo específico del proyecto, enfocado en la integración del desarrollo con el artefacto del cliente.

**Figura 8. Robot del cliente llamado "KENITO"**



Fuente: Elaboración propia

**Tabla 8. Reuniones destacadas con cliente.**

| Fecha | Participantes | Observaciones |
|---|---|---|
| 8/03/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Investigación y realización de un algoritmo que permita el reconocimiento de voz por medio de un micrófono |
| 5/04/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Investigar cómo se manejan los streams de audio en Python, frecuencias y aumento de datos. |
| 19/04/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Investigar y aumentar el conjunto de datos obtenidos en los formularios de Jtform. |
| 3/05/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Programar los tiempos de escucha del modelo y en el documento estandarizar las tablas e imágenes |
| 24/05/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Realizar la medición de variables midiendo exactitud, distancia y volumen. |
| 04/09/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Sincronización en cómo debería ser la integración del desarrollo con el robot Kenito. Se dan pauta y consideraciones para realizar este proceso. |
| 28/09/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Acuerdo para realizar experimentos con estudiantes del semillero de investigación de la facultad a fin de realizar las validaciones dinámicas. |
| 12/10/2021 | Director.<br>Estudiante 1.<br>Estudiante 2. | Se realizan las primeras pruebas del modelo desarrollado integrándose con el robot Kenito, realizando pruebas de funcionamiento y evidenciando puntos a mejorar. |
| 16/10/2021 | Director.<br>Estudiante 1.<br>Estudiante 2.<br>Estudiantes Semillero de investigación. | Se realizan nuevas pruebas del modelo desarrollado e implementado en el robot Kenito con el director y miembros del semillero de investigación. |

Fuente: Elaboración propia

### 4.1.5 Quinto ciclo

Finalmente, para poder garantizar el ciclo de transferencia tecnológica, así como también dar cumplimiento y respuesta al cuarto objetivo específico y pregunta de investigación respectivamente, se realizaron las validaciones dinámicas. También se realizó la documentación del código, y se cubrió deuda técnica dejada a través de todos los ciclos.

## 4.2 CLIENTE BENEFICIARIO

Todas las decisiones a nivel tecnológico han sido consensuadas en común acuerdo con el cliente del proyecto, Osiris & Bioaxis, un grupo de investigación de la Universidad el Bosque orientada al desarrollo tecnológico multidisciplinar. Los aportes del cliente han sido significativos, a través de las ceremonias de revisión, el proyecto nunca ha perdido de vista el objetivo general. En estas reuniones se han realizado ajustes y acuerdos de cara a las tareas por hacer. Las reuniones tienen una duración de media hora, se celebran de forma virtual en Google Meet. En la Tabla 8 se puede observar el agendamiento de las reuniones más destacadas que tuvieron lugar a lo largo del proyecto.

## 4.3 INGENIERÍA DE REQUERIMIENTOS

Para esta sección se presenta un ejemplo del formato utilizado para la redacción de los requerimientos funcionales y no funcionales obtenidos de las reuniones mantenidas con el cliente del proyecto. Se puede observar en la Tabla 9 el enfoque por priorización y tipo de requerimiento. El resto de estos, el lector los puede ubicar en el Anexo 4 del documento.

**Tabla 9. RF01**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RF01 | | | | | |
| **Nombre** | Desarrollo de un modelo de reconocimiento de voz | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El sistema a desarrollar será un modelo de inteligencia artificial para el reconocimiento de comandos de voz en español que funcione de manera offline para Raspberry Pi. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

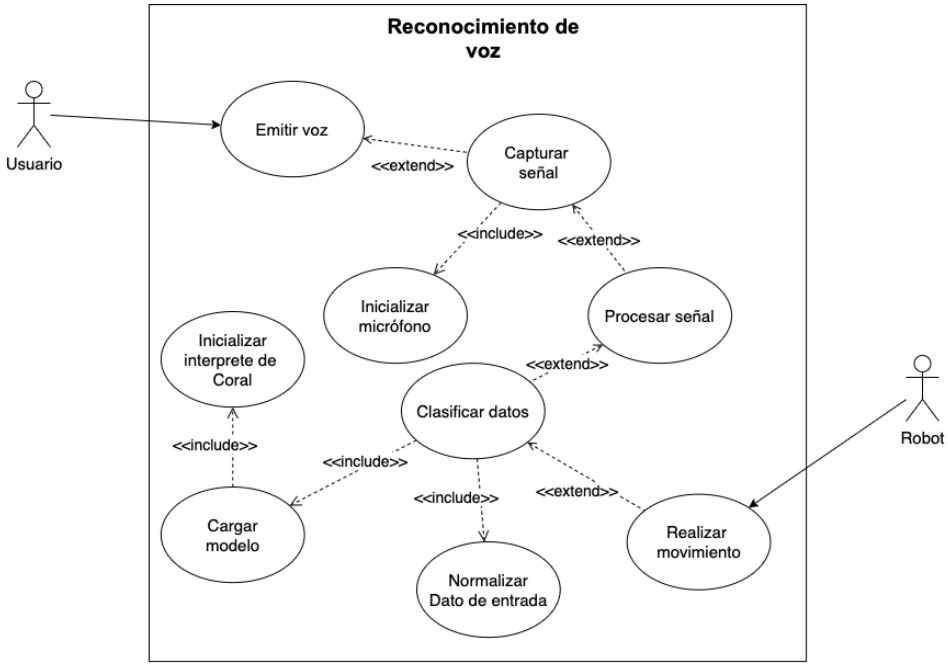
Fuente: Elaboración propia

## 4.4 DIAGRAMAS UML

A continuación, se presentan los diagramas propuestos, que ayudan a documentar la construcción del sistema.

### 4.4.1 Casos de uso

**Figura 9. Casos de usos del proyecto**



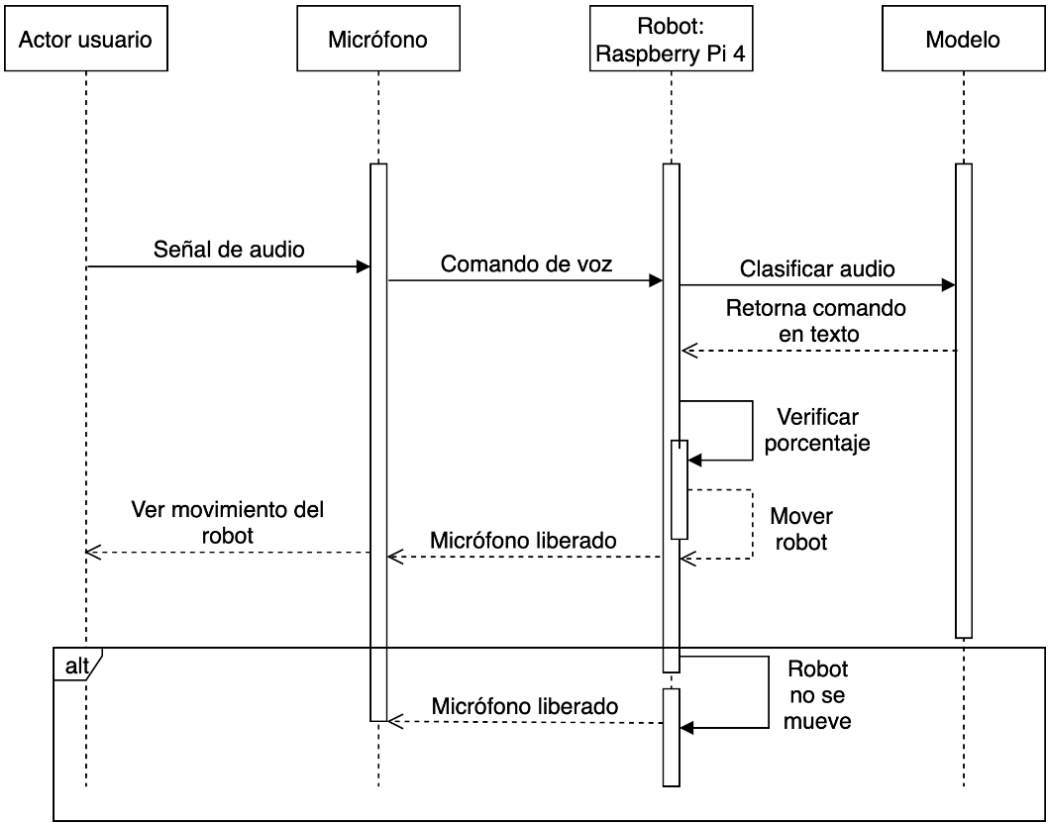

Fuente: Elaboración propia

En la Figura 9 se observa la participación de dos actores (Usuario y Robot) dentro del marco del RV que se llevará a cabo. El usuario, de aquí en más conocido cómo hablante, es quien activará el caso de uso, al emitir las instrucciones (comandos de voz). Posteriormente la señal es capturada a través de un micrófono y convertida a un formato válido para el modelo de ML, que previamente debe estar cargado en el interpretador de la Google Coral. Finalmente, si la instrucción es clasificada correctamente el robot realiza la acción solicitada.

### 4.4.2 Secuencia

Conocer el orden secuencial en que se ejecutan las tareas en el artefacto desarrollado es fundamental, por tal razón se propone el diagrama de la Figura 10. Esta muestra el flujo que comienza con la emisión de sonido por parte del usuario, y finaliza con la ejecución o no del movimiento por parte del robot.

**Figura 10. Diagrama de secuencia para el RV**

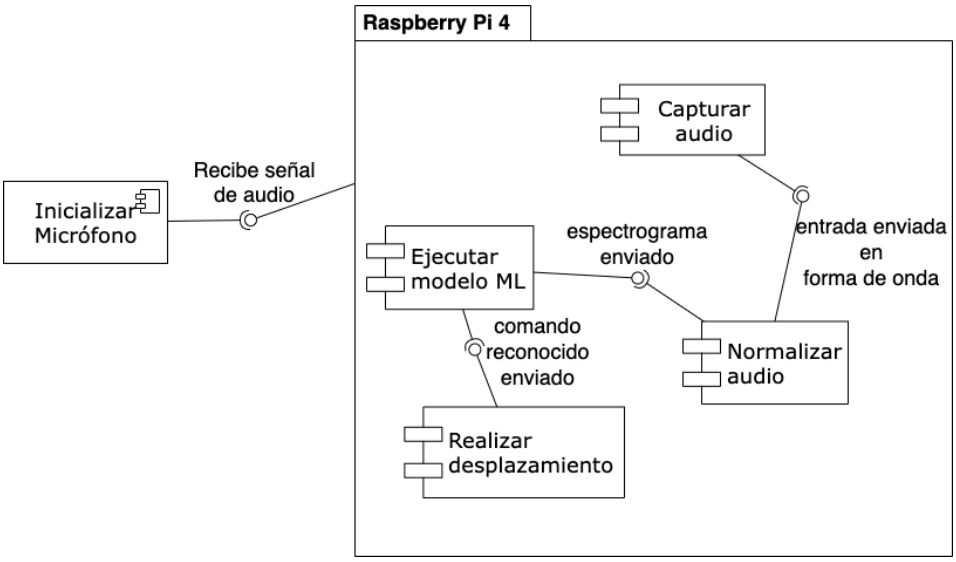



Fuente: Elaboración propia

### 4.4.3 Componentes

Otro de los aspectos importantes a nivel de ingeniería, son los componentes, estos permiten conocer cuáles son los recursos con los que se cuentan. Para este proyecto el micrófono permite capturar la señal de audio, la señal es transformada a un formato válido, el modelo clasifica esa entrada, y el robot posteriormente realiza la ejecución de un comando. La descripción visual de lo anterior se observa en la Figura 11.

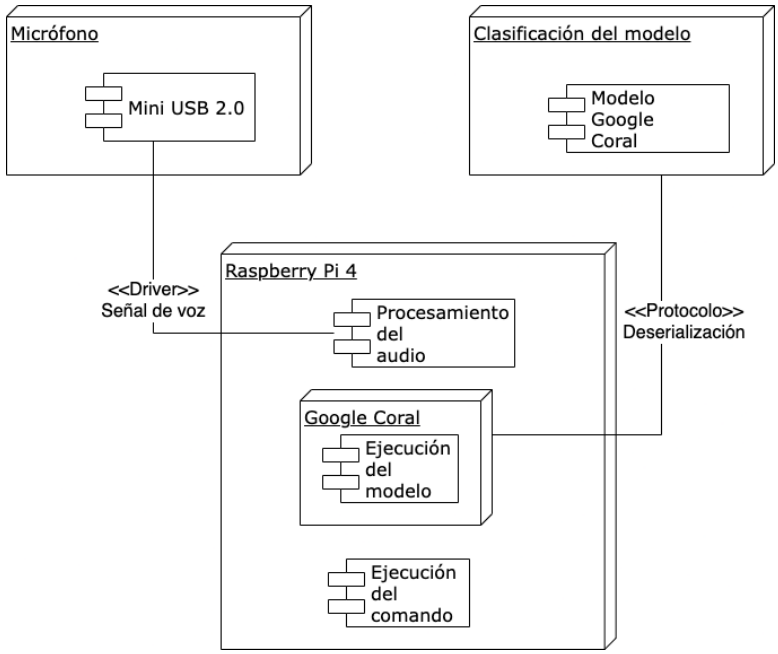
**Figura 11. Componentes tecnológicos involucrados**




Fuente: Elaboración propia

### 4.4.4 Despliegue

**Figura 12. Diagrama de despliegue**




Fuente: Elaboración propia

El despliegue es punto crítico en cualquier desarrollo tecnológico, porque es el paso final para llevar todo lo trabajado al usuario final. En ese sentido, el despliegue para

este proyecto se realizará sobre una Raspberry Pi 4, en ella se instalará a priori el modelo, y los siguientes módulos: captura y procesamiento de audio, ejecución del modelo y comandos. Lo anterior se puede ver representado en la Figura 12.

## 4.5 AMBIENTE DE PRUEBAS

Como en todo desarrollo de sistema, siempre es importante contar con un ambiente que se acerque al escenario real. Por lo tanto, se realizó la compra de una Raspberry Pi 4 y un micrófono para simular la arquitectura básica del robot Kenito observado en la Figura 8. Este también cuenta con una USB aceleradora Coral, orientada a la inferencia rápida de modelos de aprendizaje automático, dicha herramienta fue proveída por el cliente a fin de medir el rendimiento del modelo con y sin este componente.

### 4.5.1 Raspberry Pi 4

La Raspberry Pi es un ordenador de placa reducida del tamaño de una tarjeta de crédito como se puede observar en la Figura 13. Fue desarrollado por la fundación Raspberry Pi en Reino Unido, con la finalidad de poder llegar a más personas, estimular el desarrollo y la enseñanza de la informática.

**Figura 13. Tarjeta adquirida para pruebas**


Fuente: Elaboración propia

Sus características técnicas son:

- Procesador: Chip de arquitectura 32 bits y 1.5GHz.
- Almacenamiento: Tarjeta micro SD de 32Gb.
- RAM: Memoria LPDDR4 de 8Gb.

- Conectividad: WIFI de doble banda 2.4Ghz y 5GHz, así como conexión Bluetooth 5.0 y puerto de ethernet RJ 45.
- Conexiones: Dos puertos USB 3.0, dos puertos USB 2.0, una conexión estándar de 3.5mm y dos puertos de conexión Micro HDMI.

### 4.5.2 Mini Micrófono

Este dispositivo fue adquirido en conjunto con el cliente, es decir, el robot incorpora el mismo micrófono que se puede ver en la Figura 14. La frecuencia de muestreo del micrófono es de 48000 kHz. Este dispositivo es una opción de bajo costo, por lo tanto, sus prestaciones no son las mejores del mercado, sin embargo, por cuestiones de presupuesto se decidió comprar.

**Figura 14. Micrófono utilizado en las pruebas**

Fuente: Elaboración propia

### 4.5.3 Acelerador USB

Como se mencionó anteriormente, es un dispositivo USB como se ve en la Figura 15, diseñado para el proceso de inferencia rápida de modelos de aprendizaje automático.

**Figura 15. Google USB Coral**


Fuente: Elaboración propia

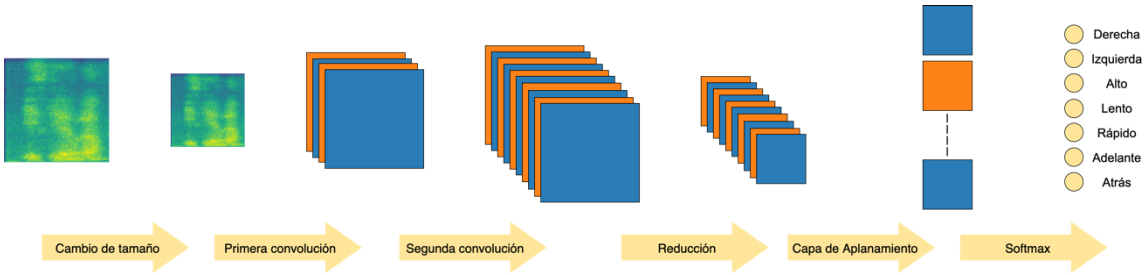
## 4.6 MODELO ML

Para la construcción del modelo de ML se investigó sobre los principales modelos de reconocimiento de voz. Dichos resultados fueron mostrados en el Marco Referencial, en ese sentido, se tomó la decisión de construir el modelo con las siguientes librerías:

- Tensorflow: Librería para construir modelos de ML
- Python: Principal lenguaje de programación en ciencias de datos
- Tensorflow Lite: Librería que ayuda a ejecutar modelos de ML en dispositivos embebidos.
- Numpy: Librería de matrices y vectores multidimensionales.
- OS: Librería para acceder a características de los sistemas operativos.
- Keras: Es una librería que optimiza la creación de modelos de ML.
- Sklearn: Se usó para la generación de estadísticas del modelo.

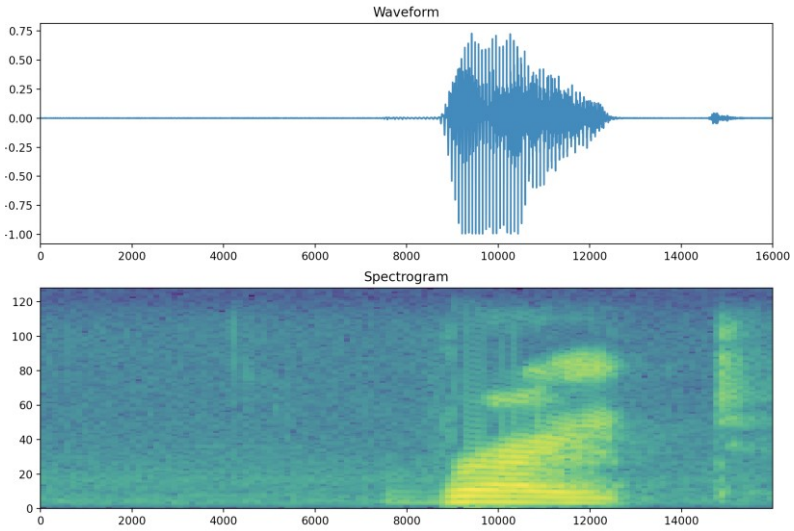
El modelo seleccionado es una CNN, algoritmo utilizado en aprendizaje automático para dar la capacidad de ver a las máquinas. Desde que los audios son formas de ondas, es decir, cambios de amplitud a través del tiempo, estos pueden ser plasmados cómo los vemos en la Figura 17a. Sin embargo, se pueden obtener imágenes con más información para un mejor modelo. Es aquí, donde entran los espectrogramas, imágenes que representan los cambios de frecuencia en el tiempo, añadiendo además la amplitud, representada cómo cambios de color, es decir, mapas de calor. Este formato contiene más información a ser extraída con objeto de entrenar un mejor modelo, en la Figura 17b se puede observar un ejemplo de espectrograma. Las CNN son modelos supervisados, aplican filtros en cada interacción (convolución) para obtener las características que mejor representan los datos de entrenamiento. Por otro lado, este tipo de redes cuentan con etapas de reducción que permiten disminuir el tiempo de procesamiento de los datos, sin afectar la calidad de los mismos. En la Figura 16 vemos la arquitectura de la CNN empleada para el desarrollo del proyecto, es una red neuronal básica recomendada por TensorFlow para el reconocimiento de palabras claves [39]. Primero el tamaño de las imágenes es reducido para mejores tiempos de entrenamiento, luego se usan dos convoluciones para la extracción de características, una capa de reducción de datos para disminuir tiempos de procesamiento, y finalmente una capa de aplanamiento para obtener el modelo final. Desde que el modelo entrega los resultados en forma de probabilidades, una función llamada softmax es usada para obtener el comando con mayor porcentaje de probabilidad.

**Figura 16. Red neuronal CNN usada**




Fuente: Elaboración propia

**Figura 17. Audio en forma de ondas y espectrograma**




Fuente: Elaboración propia

## 4.7 CONJUNTO DE DATOS

Los datos suponen uno de los aspectos más importantes en el entrenamiento de modelo de aprendizaje automático. Los comandos de voz acordados con el cliente fueron: alto, adelante, atrás, izquierda, derecha, rápido, y lento. A continuación, veremos cómo se trabaja esta parte.

### 4.7.1 Solicitud de datos

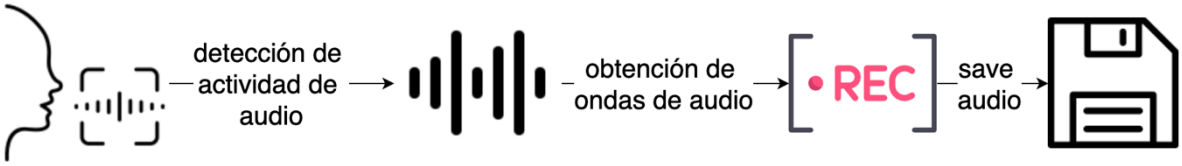
Se parte del hecho de que no contamos con ese conjunto de datos, por lo tanto, es necesario obtenerlos. Se eligieron dos formas para la recolección de datos: el formulario observado en la Figura 18, para voluntarios externos al proyecto; y un módulo escrito en Python, orientado a los usuarios finales del desarrollo tecnológico. El formulario fue creado en la plataforma jotform, que cuenta con varios formatos para solicitar información, los datos fueron obtenidos gracias a compañeros de

clase, familiares y amigos. El funcionamiento del módulo para la recolección de audios se presenta en la Figura 19, se desarrolló para que reconociera la actividad de audio emitida por el humano y posteriormente el silencio, a fin de capturar la muestra de audio.

**Figura 18. Audio en forma de ondas y espectrograma**


Fuente: Elaboración propia

**Figura 19. Recolección de audios**




Fuente: Elaboración propia

### 4.7.2 Procesamiento de datos

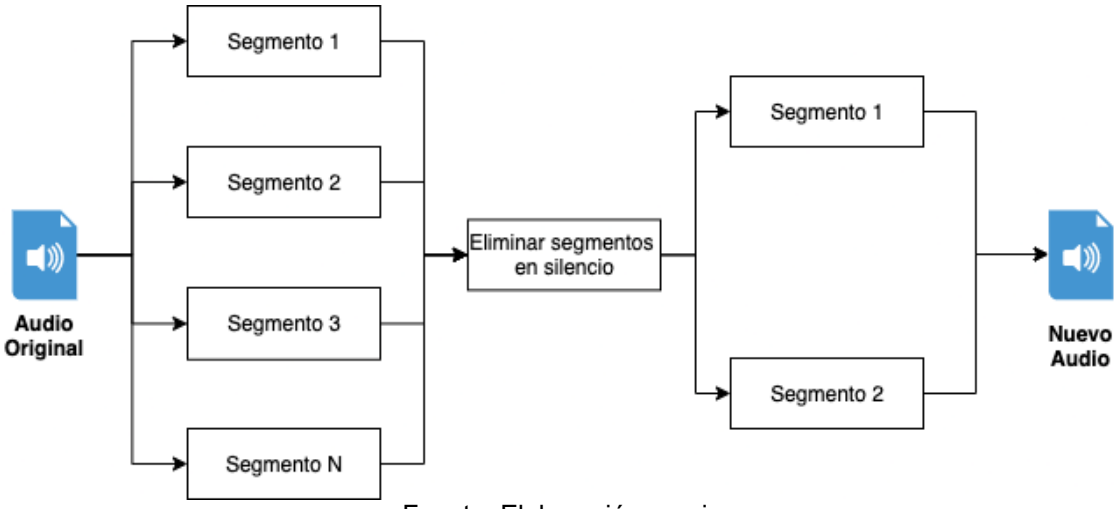
El tiempo de duración de los audios requerido para entrenar el modelo es de 1 segundo, durante la recolección a través del formulario, se observó que el tiempo era insuficiente para los voluntarios, por ello se configuró el formulario para capturar las muestras en 2 segundos. Por tal razón, el primer paso para el procesamiento de datos fue cortar los audios preservando sólo las partes donde no había silencio. Para este fin, se desarrolló un módulo con las siguientes herramientas:

- Python: Lenguaje de programación.
- Scipy IO: Permite leer y escribir tanto los datos cómo la frecuencia de muestreo de los archivos WAV.

- Numpy: Procesa, modifica, y elimina datos de las muestras de audio.
- Path Lib: Ubica los archivos dentro del sistema operativo.

En la Figura 20 vemos la implementación de las herramientas anteriores. Básicamente las muestras de audios se dividen en segmento de energía, cada parte representa un cambio de amplitud en los audios. Luego se eligen los segmentos más representativos para crear la nueva muestra de audio.

**Figura 20. Diagrama para recortar audios**

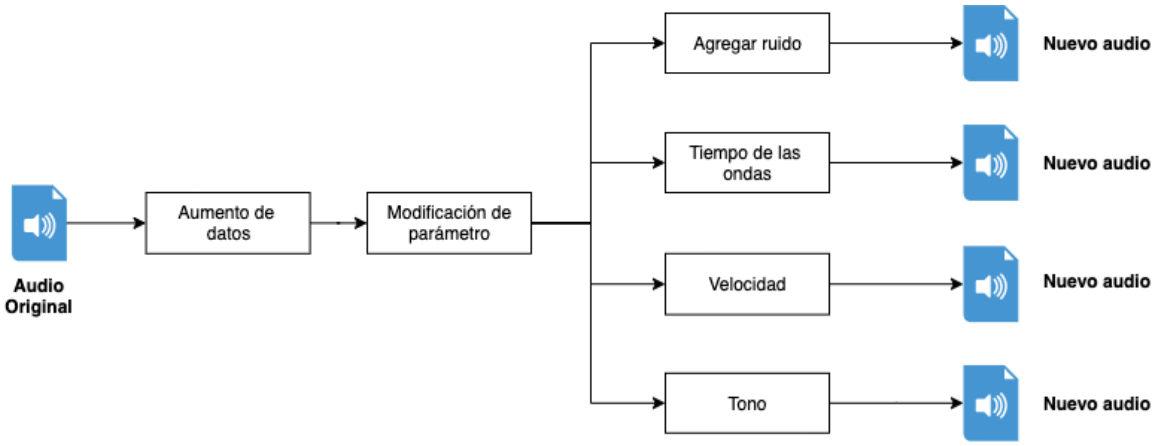



Fuente: Elaboración propia

### 4.7.3 Aumento de datos

Los datos obtenidos fueron insuficientes para el correcto entrenamiento del modelo, para mitigar este problema se aplican técnicas de aumento de datos. Consisten básicamente en añadir ruido, cambiar el tiempo de las ondas, cambiar la velocidad, y variar el tono de las muestras de audio. Con esto, el conjunto de datos incrementa cuatro veces su tamaño. En la Figura 21 se encuentra la implementación de esta técnica.

**Figura 21. Diagrama para aumentar datos**




Fuente: Elaboración propia

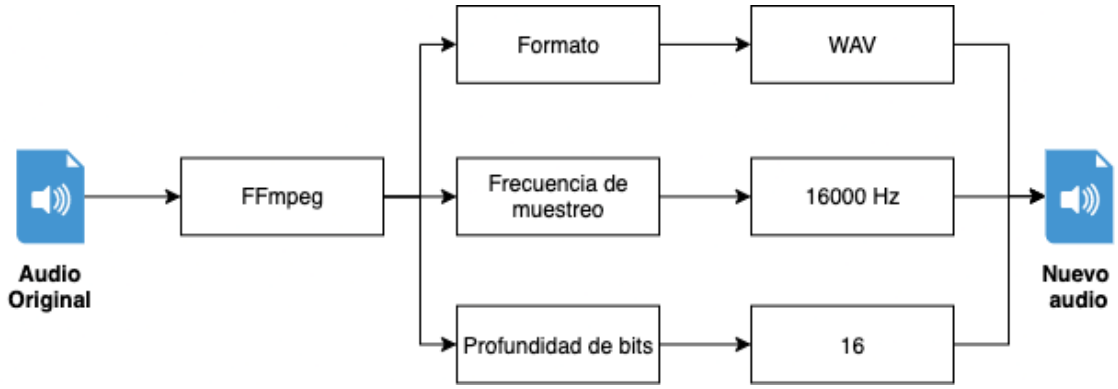
### 4.7.4 Conversión de audios

Tensorflow para el entrenamiento del modelo espera características específicas en las muestras de audios:

- Formato: Se usa WAV porque es un formato que no agrega datos adicionales, es decir, es un formato puro. Es comúnmente usado para edición de audios.
- Frecuencia de muestreo: Se requiere que sea 16000 Hz, esto representa la cantidad de ondas generadas por segundo.
- Profundidad de bits: Trata sobre la resolución de captura de una señal de audio con respecto a la amplitud de la onda. Para el presente modelo se usan 16 bits, esta cantidad garantiza la calidad del audio.

Para convertir los audios a estas características, se usó una herramienta libre llamada FFmpeg. Es una herramienta muy poderosa para grabación, conversión y reproducción tanto de videos como audios. La herramienta se implementó a través de un script escrito en Bash cómo se puede observar en la Figura 22, para automatizar el proceso de conversión.

**Figura 22. Diagrama para convertir audios**




Fuente: Elaboración propia

## 4.8 ENTRENAMIENTO

Esta fase se dividió en 2 momentos: primero, entrenamiento con los datos obtenidos de los voluntarios; y segundo, entrenamiento con los audios de los usuarios finales. Los datos en ambos casos fueron divididos así: 80% entrenamiento, 10% validación, y 10% pruebas, esto se puede observar en la Tabla 11. La razón de tener dos conjuntos de datos, fue poder observar el comportamiento del sistema de RV con y sin las voces de los usuarios finales en la etapa de entrenamiento. Como se observa en la Tabla 10, la cantidad de muestras para los dos conjuntos de datos es similar, lo cual es fundamental para la comparación del comportamiento del modelo.

**Tabla 10. Muestras totales**

| | Voluntarios | Usuarios |
|---|---|---|
| **Muestras totales** | 5990 | 5530 |
| **Muestras por comandos** | 865 | 785 |

Fuente: Elaboración propia

**Tabla 11. Distribución de muestras**

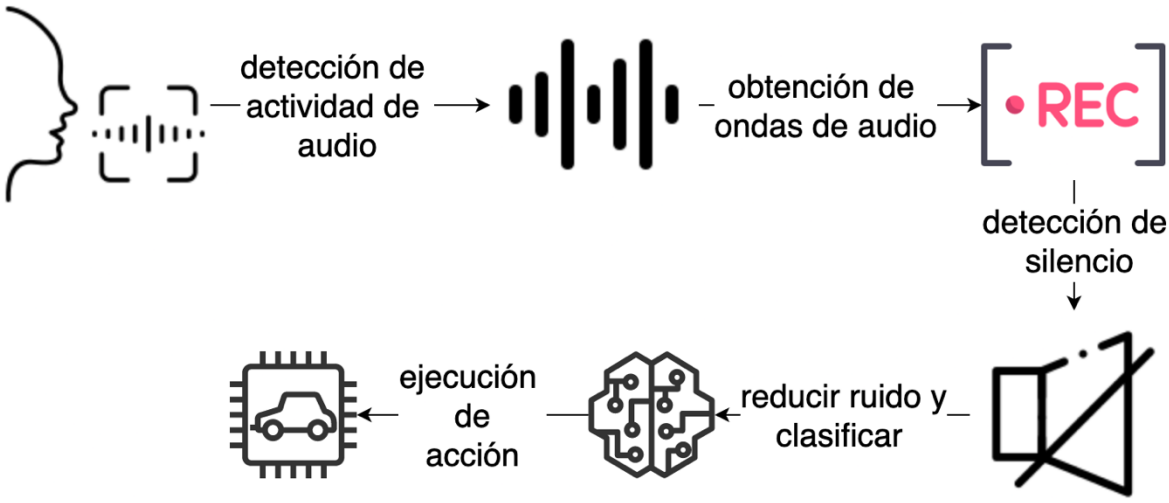
| | Voluntarios | Usuarios |
|---|---|---|
| **Tamaño datos de entrenamiento** | 4792 | 4424 |
| **Tamaño datos de validación** | 599 | 553 |
| **Tamaño datos de pruebas** | 599 | 553 |

Fuente: Elaboración propia

## 4.9 MÓDULO DE CAPTURA DE AUDIO

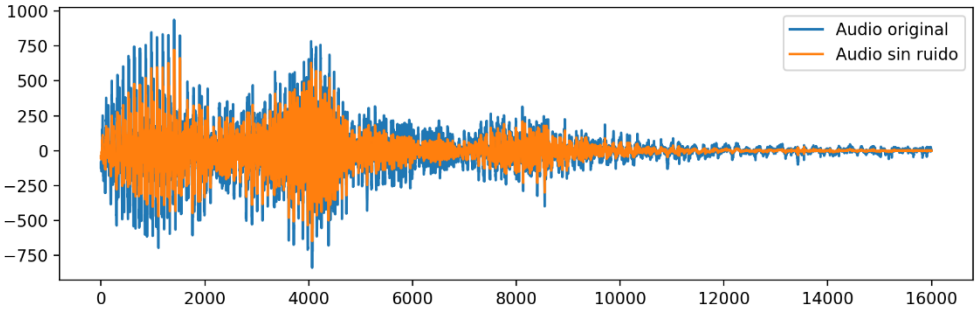
Esta etapa fue sin duda uno de los grandes retos del proyecto, el dominio de conocimiento de señales de audio es un tema que excede lo visto a través de toda la carrera de Ingeniería de Sistemas. Sin embargo, se logró entender lo básico para crear este módulo en integración con el modelo de ML. La recepción de señales digitales en tiempo real funciona a través de frecuencias de muestreo por segundo (FMS), es decir, la cantidad de muestras de audio por segundo. Entre mayor sea el número de FMS más aproximada estará la señal de audio a la onda original emitida. El anterior funcionamiento es el equivalente a los píxeles en las imágenes, a mayor píxeles mayor calidad tiene la imagen. Una FMS comúnmente usada para capturar audio es 48000 kHz, justamente la que posee el micrófono del robot. Estas FMS se dividen en pedazos con el propósito de mejorar la latencia de procesamiento de las señales. Dichos pedazos permiten convertir las ondas a arreglos, a través de estos es posible detectar el ruido definiendo un umbral de ruido mínimo, de igual manera es posible identificar el silencio en las señales. Fue así, que se pudo desarrollar un sistema de detección de sonido, que permite obtener la acción requerida del usuario para ser procesada. Dado que el robot realiza acciones de desplazamiento, estos movimientos emiten ruido a través de los motores, por tal razón fue necesario reducir dicho ruido [40][41] en la recepción de audios como se observa en Figura 24. Finalmente, el audio procesado es enviado al modelo para obtener la respuesta y en conjunto con un módulo desarrollado por el cliente, el robot Kenito ejecuta los comandos de desplazamiento. Todo lo anteriormente descrito está plasmado en la Figura 23.

**Figura 23. Funcionamiento de recepción de audio en tiempo real**




Fuente: Elaboración propia

**Figura 24. Reducción de sonido**




Fuente: Elaboración propia

## 4.10 INTEGRACIÓN

Como bien se ha mencionado, el robot Kenito cuenta con otros módulos tales como: visión artificial, control por comando físico y control por teclado. Por tal razón, se tuvieron sesiones con el cliente para realizar la integración del sub-módulo desarrollo de RV. Este proceso se hizo con el foco de integrar las dos partes de forma escalable, para que en el futuro los cambios o mejoras fueran sencillos de implementar.

## 4.11 POBLACIÓN OBJETIVO

Nuestro cliente Osiris & Bioaxis nos expresó que los usuarios del sistema desarrollado serían: un profesor, cuyo enfoque son los sistemas inteligentes; y estudiantes del semillero de investigación, dirigido por este profesor hacia temáticas de robótica y sistemas inteligentes. Fue con este grupo de personas que se realizaron las validaciones dinámicas del proyecto, con el objetivo de realizar la transferencia del ciclo tecnológico. El beneficio recibido por esta población de usuarios es la nueva capacidad de controlar los movimientos del robot a través de la voz en conjunto con las ya existentes: teclado y control de mando.

## 4.12 EXPERIMENTOS

Se tuvieron varias sesiones con el cliente para realizar las mediciones de las variables dinámicas del proyecto. Sin embargo, en estas sesiones no solo se hicieron tales mediciones, también se detectaron oportunidades de mejoras en el modelo de ML y módulo de captura de señales de audios. En primer lugar, el enfoque fue capturar audios cada 2 segundos, esto fue incorrecto debido a que las palabras podían quedar cortadas entre intervalos. Por lo tanto, el siguiente paso fue detectar la actividad de sonido, para garantizar que las palabras no quedarán cortadas, en ese sentido la recepción de audio mejoró. Hasta este punto las pruebas fueron con el modelo de ML entrenado a partir de los audios de los voluntarios. Con

el problema de recepción de audios solucionado, un nuevo inconveniente fue detectado, la precisión del sistema de RV de cara a los usuarios no fue buena, para algunos comandos esta fue de 0%. Lo anterior abrió una nueva oportunidad de mejora, se decidió junto con el cliente crear un nuevo conjunto de datos basado únicamente en los usuarios finales. También, el comando “lento” fue reemplazado por “despacio” dada su similaridad con otros comandos de voz. A través de esta decisión la precisión del sistema de RV aumentó, lo que permitió avanzar con las validaciones dinámicas. Realmente fueron sesiones enriquecedoras en términos de aprendizaje, que permitieron no encontrar errores sino oportunidades de mejoras.

## 4.13 MEDICIÓN DE VARIABLES

En la sección 2.6 se realizó la presentación de las variables en la academia, estáticas y dinámicas. En ese mismo sentido, en el estado del arte se mostraron estudios relacionados que tuvieron en consideración estos mismos factores de medición. Ahora es el momento de introducir los instrumentos empleados para este propósito, así como también los métodos de generación de datos, y cómo estos fueron cuantificados.

## 4.14 VALIDACIÓN EN LA ACADEMIA

### 4.14.1 Rendimiento

Esta primera variable estuvo orientada a la comparación del tiempo de inferencia del modelo de ML funcionando directamente en la Raspberry Pi versus su ejecución en la USB aceleradora Coral. Los datos fueron obtenidos a partir de registros (logs) que median estos tiempos de inferencia en el sistema de RV, para luego ser promediados ($\bar{r}$).

$$\bar{r} = \frac{\sum_{i=1}^{n} r_i}{n} \qquad (1)$$

### 4.14.2 Calidad

Con el objetivo de garantizar la calidad e integración del sistema de RV se implementaron pruebas unitarias, dichos resultados son cuantificados como el porcentaje de cobertura (c) de líneas probadas (lc) sobre la cantidad total de código (n).

$$c = \frac{lc}{n}\ x\ 100 \qquad (2)$$

## 4.15 VALIDACIÓN ESTÁTICA

### 4.15.1 Precisión y exactitud

Estas dos variables son comúnmente usadas para las métricas de evaluación de modelo de aprendizaje automático, porque permiten entender el rendimiento de estos mismos. La exactitud (e) de un modelo se puede entender como el número de predicciones correctas sobre el total de estas mismas. Por otro lado, la precisión (p) es la dispersión de los resultados obtenidos en mediciones repetidas, a menor dispersión mayor precisión, en otras palabras, es el porcentaje de casos positivos detectados. Los resultados de estas variables fueron obtenidos a través del 10% del conjunto de datos utilizado para el entrenamiento del modelo. Estas variables son calculadas a partir de lo que se denomina matriz de confusión, dicha matriz consiste en comparar los valores de las predicciones o clasificaciones contra los valores observados.

$$e = \frac{VP + VN}{VP + FP + FN + VN} \quad (3)$$

$$p = \frac{VP}{VP + FP} \quad (4)$$

VP = Verdaderos Positivos
VN = Verdaderos Negativos
FP = Falsos Positivos
FN = Falsos Negativos

## 4.16 VALIDACIÓN DINÁMICA

### 4.16.1 Eficacia

A pesar de que el sistema está orientado a entornos controlados, es decir, libres de ruidos, el motor presente en el robot Kenito produce ruido. Por otro lado, el rendimiento del micrófono no es constante, tiene picos de rendimiento bajo y altos, lo cual afecta el funcionamiento del sistema de RV. Por lo tanto, fue importante medir la eficacia (e) de todo el sistema a través de la precisión, con las herramientas de registros en formatos de: texto, resultado del modelo; y audios, acción requerida por el usuario.

$$e = \frac{VP}{VP + FP} \quad (5)$$

### 4.16.2 Eficiencia

Dado que uno de los principales conceptos del presente proyecto es la computación al borde de la red, es importante medir la latencia de todo el sistema de RV. Expresada en términos de tiempo, es el promedio ($\bar{e}$) de la duración que le toma a todo el sistema procesar los comandos, desde la recepción de señales de audios hasta la ejecución de los desplazamientos de navegación. Estos registros son guardados automáticamente por el sistema de RV.

$$\bar{e} = \frac{\sum_{i=1}^{n} e_i}{n} \qquad (6)$$

# 5. RESULTADOS

En esta sección se presentan los resultados de las mediciones en la academia, estáticas y dinámicas.

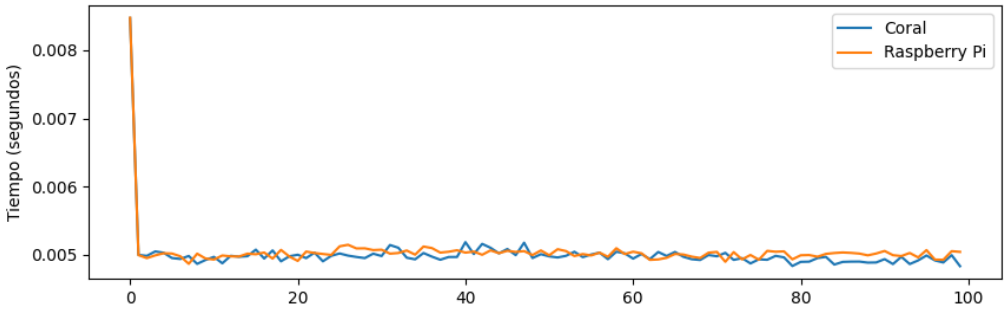
## 5.1 VALIDACIÓN EN LA ACADEMIA

A continuación, se muestran los resultados de las mediciones realizadas en la academia definidas en la sección 2.6.1:

### 5.1.1 Rendimiento

En la Figura 25 se presenta el rendimiento expresado en tiempo de las mediciones realizadas en la Google USB aceleradora Coral y la Raspberry Pi. Se realizaron 100 repeticiones sobre cada arquitectura, obteniendo a través de la Fórmula (***1*** un tiempo promedio de 0.005 (5 ms) segundos igual para ambos escenarios.

**Figura 25. Tiempo de inferencia del modelo**

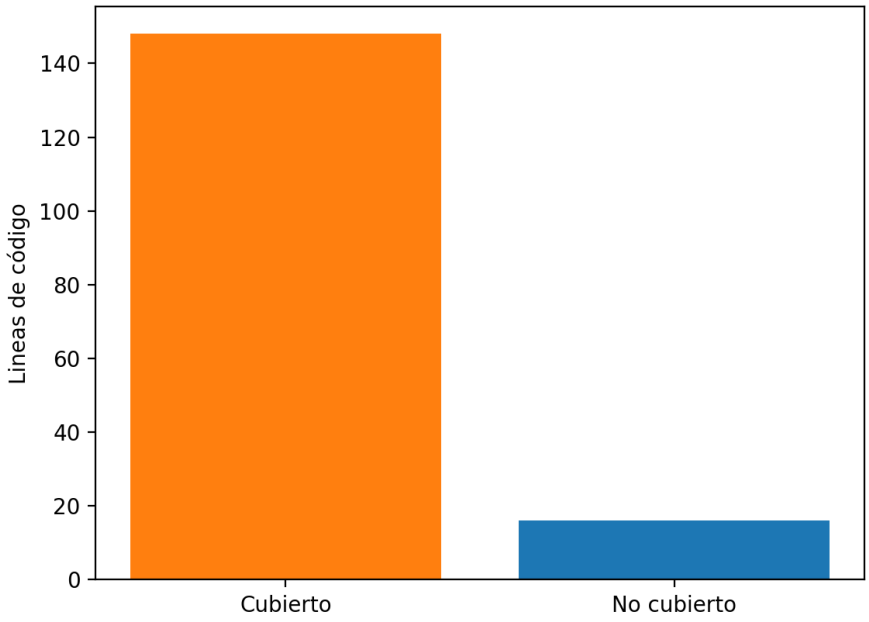



Fuente: Elaboración propia

### 5.1.2 Calidad

La medición de cobertura total de pruebas unitarias se realizó con la utilidad de Python llamada: coverage, el cálculo lo realiza a través de la Fórmula (***2*** y el valor obtenido fue del 92%. Este es el resultado de cubrir 148 líneas de código con pruebas unitarias de un total 164 como se puede ver en la Figura 26.

**Figura 26. Cobertura de código**




Fuente: Elaboración propia

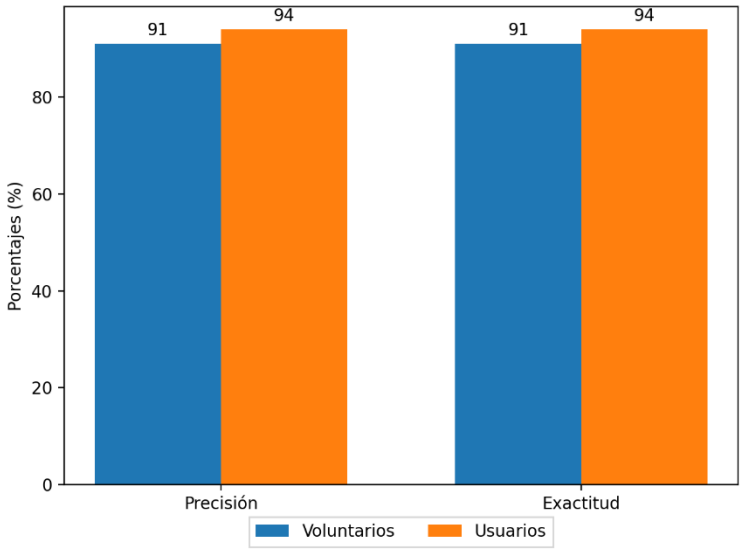
## 5.2 VALIDACIÓN ESTÁTICA

En cuanto a este tipo de validación y sus mediciones previamente definidas en la sección 2.6.2, a continuación, se presentan los resultados obtenidos:

### 5.2.1 Precisión y exactitud

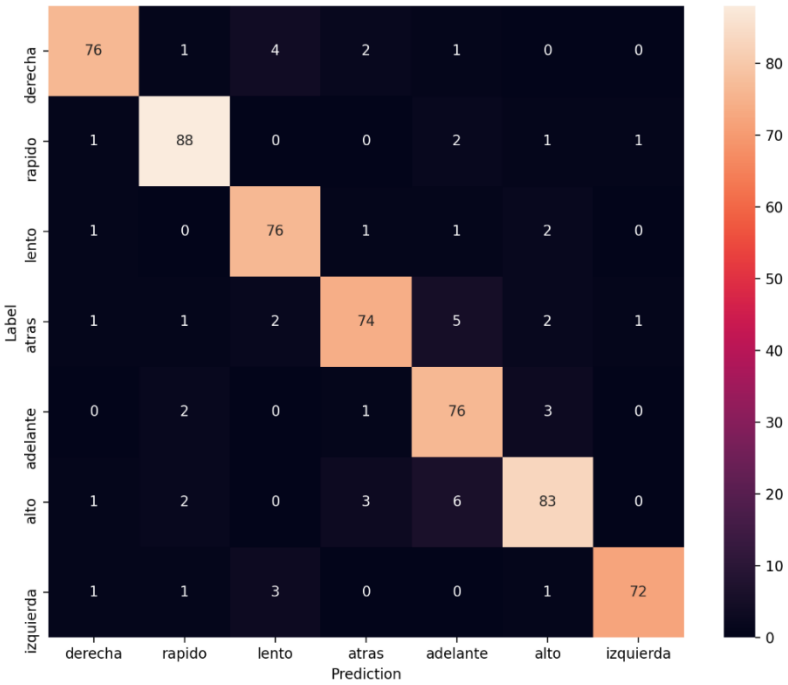
A partir de los modelos basados en los datos de voluntarios y usuarios, se muestran los resultados de la precisión y exactitud en la Figura 26. Los resultados fueron obtenidos por medio de las Fórmulas (***3*** y (***4*** presentadas en la sección 4.15.1, estos sirvieron para justificar el uso del modelo creado a partir de los usuarios. Lo anterior dada la necesidad de mejorar el modelo de cara a los usuarios finales del sistema de RV. Así mismo, las matrices de confusión correspondiente a cada modelo en el proceso de entrenamiento son mostradas en las Figura 28 y Figura 29. Estas representaciones ayudan a entender que tanto el modelo tiende a confundir un comando con otro, dicho problema fue la principal causa para cambiar la fuente de entrenamiento del modelo, desde los datos de voluntarios hacia los datos de usuarios.

## Figura 27. Porcentajes de precisión y exactitud



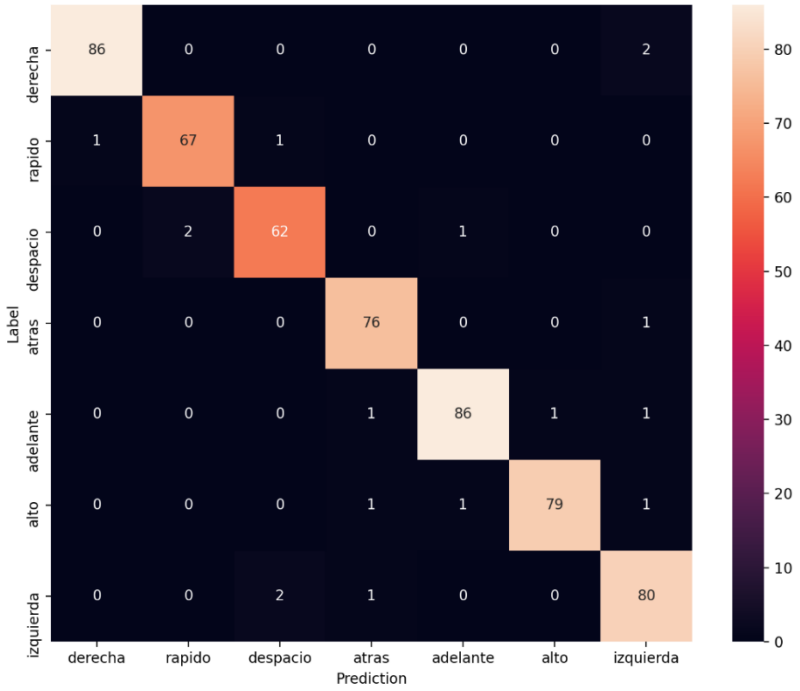

Fuente: Elaboración propia

## Figura 28. Matriz de confusión con datos de voluntarios




Fuente: Elaboración propia

**Figura 29. Matriz de confusión con datos de usuarios**




Fuente: Elaboración propia

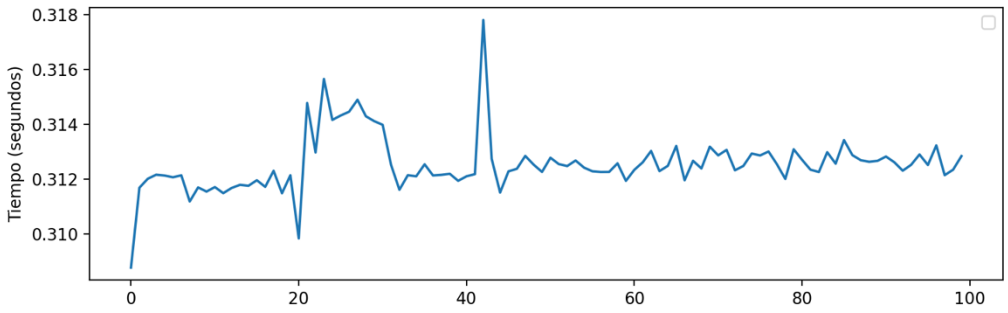
## 5.3 VALIDACIÓN DINÁMICA

El ciclo de transferencia tecnológica muy importante generar cambios en el modelo BPSC se buscó garantizar a través de la mediciones descritas en la sección 2.6.3, estas fueron realizadas en conjunto con la población definida en la sección 4.11.

### 5.3.1 Eficiencia

De cara a la eficiencia del sistema de RV, de 100 comandos de voz solicitados por los usuarios, se obtuvo que el tiempo promedio (ver Fórmula (***5***) en realizar la acción de desplazamiento fue de 0.31 segundos. El comportamiento de tales solicitudes pueden ser observadas en la Figura 30, los tiempos de respuesta son estable, sin embargo, en algunas ocasiones pueden ser superiores debido a latencia propia del micrófono usado para este proyecto. Esta medición ayuda entender la latencia del sistema, a fin de realizar una comparación con sistemas basados en la nube.

**Figura 30. Tiempo de ejecución de comandos de voz**




Fuente: Elaboración propia

### 5.3.2 Eficacia

En la misma línea de pruebas con los usuarios, se obtuvo a través de la matriz de confusión de todo el sistema mostrada en la Tabla 12, que la eficacia (ver Fórmula (***6***) expresada en indicadores de precisión fue del 86%. Este es el resultado de 87 comandos correctamente ejecutados por el sistema de RV y 14 comandos erróneamente realizados. La relevancia de esta medición radica en comprobar que la disminución de la latencia no afecte considerablemente la eficacia del sistema.

**Tabla 12. Matriz de confusión del todo el sistema**

| | | Predicción | |
|---|---|---|---|
| | | Positivos | Negativos |
| Observación | Positivos | 87 | 0 |
| | Negativos | 14 | 0 |

Fuente: Elaboración propia

# 6. ANÁLISIS DE RESULTADOS

Luego de realizar la comparación de tiempos de inferencia sobre los dispositivos USB acelerador Coral y Raspberry Pi, podemos observar que el tiempo promedio para ambos es de 0.005 segundos (Ver sección 5.1.1). Esto puede dar a entender que es innecesario la compra de un dispositivo orientado a procesamiento de modelo de ML. Sin embargo, dado que la Raspberry Pi realiza muchas otras tareas, es relevante delegar tantos procesos que sean posible, para no afectar el resultado del resto de tareas. Por ejemplo, el robot a través de la Raspberry Pi emite los movimientos al robot, captura imágenes y audios para visión artificial y el RV respectivamente, y hace el procesamiento de estos formatos de datos. Es importante resaltar que los resultados obtenidos están limitados al modelo y tamaño del conjunto de datos de los usuarios, cantidades de datos superiores o inferiores pueden modificar considerablemente los tiempos de inferencia en cada dispositivo.

En cuanto a la comparación de modelos se observó que el modelo entrenado con datos de los usuarios finales es más preciso y exacto (Ver sección 5.2.1). Este obtuvo valores del 94% para las dos mediciones, mientras que con los datos de los voluntarios los resultados fueron del 91%. Esto quiere decir que el modelo de los usuarios es capaz de clasificar correctamente más comandos de voz, adicionalmente el modelo de voluntarios tuvo un mal comportamiento con los usuarios del sistema, 1 de cada 10 comandos fueron clasificados correctamente. Como se pudo observar los resultados de precisión y exactitud fueron iguales, esto quiere decir que es un modelo balanceado, con respuestas poco dispersas (preciso) más cerca del valor real (exactitud). De igual forma, se presenta la matriz de confusión para los dos conjuntos de datos, allí se observa que los comandos de voz son menos propensos a ser confundidos con los datos de los usuarios. La razón de los malos resultados del modelo basado en datos de voluntarios es la necesidad de entrenar modelos con los datos de los usuarios cuando el conjunto de datos es pequeño, siendo esto una limitando del presente proyecto. El cambio de conjunto de datos permitió calibrar los parámetros de entrada del modelo para obtener mejores resultados, con esto se dio cumplimento al segundo objetivo específico del proyecto. A pesar de que el modelo basado en los usuarios obtuvo mejor desempeño, es importante resaltar que esto es para el contexto de los usuarios finales del sistema de RV. Fuera de tal contexto los resultados pueden ser diferentes, y quizás el modelo hecho con los voluntarios sería más apropiado para ser usado.

El tercer objetivo específico del proyecto estuvo orientado a la integración del sistema, por tanto, fue importante garantizar la calidad del sistema técnicamente hablando. Se obtuvo un porcentaje de cobertura de pruebas unitarias del 92% (Ver sección 5.1.2) considerado alto. Según Mockus et al. [42] el aumento de cobertura con pruebas unitarias está directamente relacionado con una disminución de problemas en etapas operativas. Dicha afirmación fue confirmada por el presente proyecto, porque a la hora de realizar la integración con el Kenito no se encontraron

errores funcionales. La limitación de la variable medida es el tamaño del código probado, con proyectos que involucren más líneas de código la relación proporcional entre cobertura y error puede ser muy diferente, incluso puede cambiar a una relación inversamente proporcional.

En cuanto a los resultados obtenidos de las pruebas realizadas con los usuarios del sistema, se obtuvo que el sistema tiene alta eficiencia con tiempos de respuesta de 0,31 segundos (Ver sección 5.3.1). Lo anterior es confirmado si lo comparamos con tiempos de respuestas de sistema basados en la nube, por ejemplo, Oh et al. [11] en su trabajo obtuvieron de 3,09 a 5,41 segundos, incluso con las mejoras en procesamientos de datos planteados por los autores. Estos resultados concuerdan con lo que se esperaba en la solución planteada desde el modelo BPSC, aumentar la eficiencia de las respuestas a través de sistema autónomos desconectados de internet.

Por otro lado, la eficacia del sistema alcanzó una precisión del 86% (Ver sección 5.3.2), resultados que se vieron mermados por la calidad del micrófono usado en Kenito. Productos como Amazon Echo o Google Assistant llegan a implementar hasta 7 micrófonos en artefactos, lo cual destaca la relevancia de estos. Por lo tanto, usar micrófonos en mayor calidad y cantidad mejora la eficacia del sistema de RV. Hay alternativas en el mercado para este fin, sin embargo la gran mayoría dejan de ser dispositivos de bajo costo, lo cual era una limitación en el presente proyecto. Sin embargo, a pesar que los resultados de esta medición pueden ser mejores, lo obtenido se considera como un buen porcentaje de eficacia de acuerdo a Sharan et al. [10] Los sistemas de RV basados en la nube evidentemente alcanzan métricas de precisión más elevadas, esto se debe a la gran cantidad de datos que poseen y la facilidad con la que día a día recolectan más datos. La eficacia y también la eficiencia del sistema fueron probadas en entornos controlados libres de ruidos externos, un sistema de RV en ambientes ruidosos requiere otras consideraciones más complejas a las vistas en el proyecto.

Las dos anteriores variables permiten realizar el ciclo de transferencia tecnológica con el cliente, dando cumplimento al cuarto objetivo específico del proyecto. En cuanto BPSC, se reemplazan los artefactos basados en la nube para RV por un sistema de offline, así como también la introducción de un robot que integre esta funcionalidad. En consecuencia, las creencias de ser sistemas muy costosos cambian a qué la implementación con dispositivos de bajo costo es posible sin sacrificar privacidad ni seguridad en los datos procesados. En cuanto a los hábitos, se contribuye a mejorar la comunicación entre humano y máquina a través de la voz, con propósitos de automatización en la ejecución de comandos de desplazamiento terrestre. Se obtuvieron tiempos de respuesta menores tal como se planteó en un principio, generando un nuevo hábito de no saturar las redes de internet. Es por esto y basados en el análisis de resultado planteado, se da cumplimiento al objetivo general del proyecto. El cambio de medio desde servicios en la nube ofrecidos por grandes empresas como Google o Amazon hacia robots

con sistemas de RV autónomos, permite entender que es posible el desarrollo de sistemas vanguardistas sin dependencias externas, que finalmente ponen en riesgo la estabilidad de la internet y seguridad de los datos. La introducción de artefactos de RV offline abre la posibilidad de solucionar problemas específicos con soluciones especificas y optimizadas para el contexto, en lugar de emplear soluciones generales que pueden hace uso de recursos computacionales innecesarios. Lo anterior de acuerdo a la necesitad que se planteó en un principio, el reconocimiento de solo siete palabras, el ajuste del sistema a este contexto delimitó la solución haciendo uso de lo justo necesario computacionalmente hablando.

En cuanto a la pregunta de investigación planteada en la sección 1.2, se puede implementar un subsistema o sistema de RV con tecnologías de aprendizaje automático como las redes neuronales convolucionales, entrenadas a partir de muestras de audios de los usuarios. También es importante añadir al cómo de esa pregunta, la investigación previa de cómo funcionan las señales de audios digital para poder hacer un buen procesamiento de estas mismas. A través de este proceso y por lo visto en los resultados se puede obtener un sistema que sea más rápido sin llegar a sacrificar mucho su eficacia.

# 7. CONCLUSIONES

Una primera conclusión es que el sistema de reconocimiento de voz funciona mejor si las muestras de audios de los usuarios del sistema son incluidas en la fase de entrenamiento, eso se ve reflejado en la precisión (94%) y exactitud (94%) del modelo. Así como también, la calidad y cantidad de micrófonos afectan directamente la eficacia del sistema entero.

La inclusión del concepto de computación al borde de la red ayuda a tener mejores tiempos de respuesta, ganando colateralmente más seguridad y privacidad en los datos. Por otro lado, el estado del arte permitió encontrar el modelo de ML más adecuado para ser implementado e integrado en el robot Kenito.

La medición de la variable calidad, permitió una integración sin errores funcionales en el robot en cuestión, y a través de las pruebas unitarias también se asegura que futuras modificaciones por parte del cliente no afecten el funcionamiento del sistema.

En cuanto a la pregunta de investigación, el uso de tecnologías de aprendizaje automático con otras disciplinas referentes al sonido (señales de audios digitales) marcan un camino de como sistemas de RV pueden ser implementados con bajos, sin dependencias de internet y buenos rendimientos. El investigar más allá del campo de dominio permite también encontrar soluciones interdisciplinares que generen cambios en el contexto de acción. Por ejemplo, en los usuarios se mejoró la forma de interacción con Kenito, con tiempos rápidos de respuesta (0,31 segundos) y una buena precisión (86%) del sistema. Siendo Kenito un robot autónomo con características de visión artificial, con la incorporación del reconocimiento de comandos de voz se convierte en aún más autónomo. En otras palabras, el uso del artefacto offline de RV a través del medio Kenito mejoró la comunicación entre este mismo y sus usuarios, ofreciendo una nueva alternativa eficaz y eficiente de comunicación.

Este estudio estuvo limitado a un ambiente controlado a fin de obtener mejores resultados que garantizarán el cumplimiento del objetivo general (Ver sección 2.1) y ciclo de transferencia tecnológica. Por otro lado, la población de usuarios fue muy pequeña, dicha limitación fue marcada por el contexto del cliente. Otra limitación del proyecto fue la cantidad de comandos de voz (7), en futuros trabajos esta la oportunidad de soportan una mayor cantidad de palabras o incluso frases con instrucciones más complejas.

El primer objetivo específico se cumplió gracias a la revisión del estado del arte, lo cual permitió ahorrar esfuerzos en la selección del modelo de aprendizaje automático. En cuanto al segundo objetivo la técnica de aumento de datos permitió efectivamente modificar los datos de entrada del modelo en la fase de

entrenamiento para obtener mejores métricas de evaluación de este mismo. Las pruebas unitarias ayudaron a una correcta y fluida integración del sistema, cumpliendo así el tercer objetivo. Por otro lado, la medición de las variables de interés (dinámicas) junto con sus resultados y análisis ya presentados, permitió cumplir con el cuarto objetivo de forma satisfactoria, obteniendo los resultados que se esperaban desde el modelo BPSC. Finalmente, en coherencia con el cumplimiento de todos los objetivos específicos, el objetivo general fue cumplido de forma satisfactoria con el sistema desarrollado que fue debidamente probado con la población objetivo.

Finalmente, el trabajo también abrió nuevas oportunidades de mejoras que permiten seguir avanzando hacia mejores sistemas de bajo costo offline. Este tipo de proyecto es importante para mostrar que es posible desarrollar tecnológicas vanguardistas desde Latinoamérica.

# 8. RECOMENDACIONES Y TRABAJOS FUTUROS

Futuros estudios pueden incluir la incorporación de un mejor micrófono para aumentar la eficacia del sistema. También poder obtener muestras de audios de un número mayor de usuarios para realizar mediciones más complejas. El estudio estuvo limitado a 7 comandos de voz, un mayor número de comandos puede significar un sistema más completo, incluso soportar oraciones puede derivar en acciones más complejas por parte del robot. El RV no solo se limita a robots, sino a múltiples ambientes, experimentar con casas inteligentes y asistentes de voz de forma desconectada se presentan como grandes enfoques en pro de la privacidad de los usuarios.

# 9. LECCIONES APRENDIDAS

Son varios los aprendizajes obtenidos, no solo a nivel técnico sino en cuanto al manejo de proyecto. El proyecto arrancó con muchas incertidumbres técnicas, en cuanto al cómo desarrollar el sistema. En ese obstáculo, el director del presente proyecto jugó un papel fundamental, entregó orientaciones puntuales y precisas que conllevaron a encontrar claridad en la etapa de planificación. Una vez los requisitos fueron obtenidos, llegó otro inconveniente, la metodología, se partió con Kanban, que no es una metodología, razón por la cual se cambió a SCRUM. Este cambio no fue sencillo, porque implicó empezar a trabajar con más rigurosidad a fin de cumplir con la aplicación de la metodología. Por último, es importante destacar el desafío del procesamiento y entendimiento de las señales de audios digitales. Nuevamente, el director dictó las pautas para realizar este trabajo, con mucho esfuerzo y dedicación se encontró el flujo correcto para procesar las señales de audio. Jugaron un papel importante los parámetros que conforman los audios: amplitud, frecuencia, y tiempo.

## 10. REFERENCIAS BIBLIOGRÁFICAS

[1] S. Ali, "Speech Recognition Based Home Automation System using Raspberry Pi and Zigbee", Universidad de Leeds, Reino Unido, articulo, 10.24081/niger.2017.2.0008, 2018.

[2] S. Upadhyay, "Intelligent system based on speech recognition with capability of self-learning", Gujarat Technological University, Ahmebadad, India, 2014.

[3] M. Walid, "Real-Time Implementation of Isolated-Word Speech Recognition System on Raspberry Pi 3 Using WAT-MFCC", Universidad de Túnez El Manar, Túnez, 2019.

[4] H. Gulustan, "Experimental speech recognition system based on Raspberry Pi 3", Plovdiv university, Bulgaria, 2017.

[5] J. Arthur, “Raspberry Pi: The complete guide to Raspberry Pi for beginners, including projects, tips, tricks, and programming”, CreateSpace Independent Publishing Platform, 2017.

[6] D. Anniappa and Y. Kim, "Security and Privacy Issues with Virtual Private Voice Assistants," 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), 2021, pp. 0702-0708, doi: 10.1109/CCWC51732.2021.9375964.

[7] S. Reif, B. Herzog, J. Hemp, W. Schröder-Preikschat, and T. Hönig, “Ai waste prevention: Time and power estimation for edge tensor processing units: Poster,” in Proceedings of the Twelfth ACM International Conference on Future Energy Systems, ser. e-Energy ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 300–301.

[8] Z. Peng, X. Lin, M. Simon, and N. Niu, “Unit and regression tests of scientific software: A study on swmm,” Journal of Computational Science, vol. 53, p. 101347, 2021.

[9] R. F. Brena, E. Zuvirie, A. Preciado, A. Valdiviezo, M. Gonzalez-Mendoza, and C. Zozaya-Gorostiza, “Automated evaluation of foreign language speaking performance with machine learning,” International Journal on Interactive Design and Manufacturing (IJIDeM), pp. 1–15,2021.

[10] S. Sharan, T. Q. Nguyen, P. Nauth and R. Araujo, "Implementation and Testing of Voice Control in a Mobile Robot for Navigation," 2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), 2019, pp. 145-150, doi: 10.1109/AIM.2019.8868892.

[11] R. Oh, K. Park, and J. G. Park, “Online Speech Recognition Using Multichannel Parallel Acoustic Score Computation and Deep Neural Network (DNN)- Based Voice-Activity Detector,” APPLIED SCIENCES-BASEL, vol. 10, no. 12, JUN 2020.

[12] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagap-pan, B. Nushi, and T. Zimmermann, “Software engineering for machine learning: A case study,” 2019, pp. 291–300.

[13] Rivero, J. Diniz, G. Silva, G. Borralho, G. Braz Junior, A. Paiva, E. Alves, and M. Oliveira, “Deployment of a machine learning system for predicting lawsuits against power companies: Lessons learned froman agile testing experience for improving software quality,” 2020.

[14] Alianza Caoba, Perfil Alianza Caoba, reporte técnico. Universidad de los Andres, Bogotá, 2017.

[15] D. Gotterbarn, K. Miller, and S. Rogerson, “Computer society and acm approve software engineering code of ethics,”Computer, vol. 32, no. 10,pp. 84–88, 1999.

[16] Abdulkareem, T. E. Somefun, O. K. Chinedum, and F. Agbetuyi, “Design and implementation of speech recognition system integrated with internet of things,” International Journal of Electrical and Computer Engineering, vol. 11, no. 2, p. 1796, 2021.

[17] P. Patel, A. S. A. Doss, L. PavanKalyan, and P. J. Tarwadi, “Speech Recognition Using Neural Network for Mobile Robot Navigation,” Trends in Mechanical and Biomedical Design, pp. 665–676, 2021.

[18] S. Cheng, Z. Xu, X. Li, X. Wu, Q. Fan, X. Wang, and V. C. Leung, “Task Offloading for Automatic Speech Recognition in Edge-Cloud Computing Based Mobile Networks,” in 2020 IEEE Symposium on Computers and Communications (ISCC). IEEE, 2020, pp. 1–6.

[19] S. Yang, Z. Gong, K. Ye, Y. Wei, Z. Huang, and Z. Huang, “EdgeRNN: A Compact Speech Recognition Network with Spatio-Temporal Features for Edge Computing,” IEEE Access, vol. 8, pp. 81 468–81 478, 2020.

[20] K. Radzikowski, L. Wang, O. Yoshie, and R. Nowak, “Accent modification for speech recognition of non-native speakers using neural style transfer,” EURASIP Journal on Audio, Speech, and Music Processing, vol. 2021, no. 1, pp. 1–10, 2021.

[21] M. N. Sabab, M. A. R. Chowdhury, S. M. I. Nirjhor, and J. Uddin, “Bangla Speech Recognition Using 1D-CNN and LSTM with Different Dimension Reduction Techniques,” in International Conference for Emerging Technologies in Computing. Springer, 2020, pp. 158–169.

[22] P. Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” arXiv preprint arXiv:1804.03209, 2018.

[23] E. van der Westhuizen, H. Kamper, R. Menon, J. Quinn, and T. Niesler, “Feature learning for efficient asr-free keyword spotting in low-resource languages,” Computer Speech & Language, vol. 71, p. 101275, 2022.

[24] Z. Nagoev, L. Lyutikova, and I. Gurtueva, “Model for automatic speech recognition using multi-agent recursive cognitive architecture,” Procedia computer science, vol. 145, pp. 386–392, 2018.

[25] D. Park, M. Park, H. Lee, Y.-J. Kim, Y. Kim, and Y. Park, “Development of machine learning model for diagnostic disease prediction based on laboratory tests,” Scientific Reports, vol. 11, no. 1, 2021.

[26] Suganeswaran, K. Dhileephan, N. Nithyavathy, P. Ganeshan, S. Arunkumar, and A. Antony, “Machine learning based mechanism for crowd mobilization and control,” 2021, pp. 1334–1339.

[27] V. Kadyan, S. Shanawazuddin, and A. Singh, “Developing children’s speech recognition system for low resource Punjabi language,” Applied Acoustics, vol. 178, p. 108002, 2021.

[28] P. Puchtler and R. Peinl, “Evaluation of deep learning accelerators for object detection at the edge,” in KI 2020: Advances in Artificial Intelligence, U. Schmid, F. Klügl, and D. Wolter, Eds. Cham: Springer International Publishing, 2020, pp. 320–326.

[29] G. Sun, Y. Li, Y. Li, D. Liao, and V. Chang, “Low-latency orchestration for workflow-oriented service function chain in edge computing,” Future Generation Computer Systems, vol. 85, pp.116–128, 2018.

[30] G. Chalapathi, V. Chamola, A. Vaish, R. Buyya, *Industrial Internet of Things (IIoT) Applications of Edge and Fog Computing: A Review and Future Directions,* vol 83, pp 293-325, Springer, Cham, 2021.

[31] S. Yao, J. Li, D. Liu, T. Wang, S. Liu, H. Shao, and T. Abdelzaher, “Deep compressive offloading: speeding up neural network inference by trading edge computation for network latency,” in Proceedings of the 18th Conference on Embedded Networked Sensor Systems, 2020, pp. 476–488.

[32] Z. Mushtaq, S.-F. Su, and Q.-V. Tran, “Spectral images based environmental sound classification using CNN with meaningful data augmentation,” Applied Acoustics, vol. 172, p. 107581, 2021.

[33] NVIDIA Developer, Jetson Nano Developer Kit, Disponible en: https://developer.nvidia.com/embedded/jetson-nano-developer-kit [Accedido oct. 30, 2021].

[34] Odroid, ODROID-N2+ with 4GByte RAM, Disponible en: https://www.hardkernel.com/shop/odroid-n2-with-4gbyte-ram-2/ [Accedido oct.30, 2021].

[35] Banana Pi, Banana Pi M5, Disponible en: https://www.banana-pi.org/m5.html [Accedido oct.30, 2021].

[36] Raspberry Pi, Raspberry Pi 4 Model B specifications, Disponible en: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/ [Accedido oct.30, 2021].

[37] Orange Pi, orange pi zero – Orangepi, Disponible en: http://www.orangepi.org/orangepizero/ [Accedido oct.30, 2021].

[38] R. Elshawi, A. Wahab, A. Barnawi, and S. Sakr, “Dlbench: a comprehensive experimental evaluation of deep learning frameworks,” Cluster Computing, 2021.

[39] TensorFlow Developers, “TensorFlow”. Zenodo, oct. 07, 2021. doi: 10.5281/zenodo.5555139.

[40] T. Sainburg, “timsainb/noisereduce: v1.0,” Jun. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3243139.

[41] T. Sainburg, M. Thielk, and T. Q. Gentner, “Finding, visualizing, and quantifying latent structure across diverse animal vocal repertoires,” PLoS computational biology, vol. 16, no. 10, p. e1008228, 2020.

[42] A. Mockus, N. Nagappan and T. T. Dinh-Trong, "Test coverage and post-verification defects: A multiple case study," 2009 3rd International Symposium on Empirical Software Engineering and Measurement, 2009, pp. 291-301, doi: 10.1109/ESEM.2009.5315981.

# 11. ANEXOS

## Anexo 1. Acuerdo con cliente

Esta sección no puede ser mostrada por acuerdos de confidencialidad.

## Anexo 2. Conjunto de datos

Esta sección no puede ser mostrada por acuerdos de confidencialidad.

## Anexo 3. Repositorio

Esta sección no puede ser mostrada por acuerdos de confidencialidad.

## Anexo 4. Requerimientos funcionales y no funcionales

### Tabla 13. RF02

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RF02 | | | | | |
| **Nombre** | Reconocimiento de los comandos de voz | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El modelo debe contar con la habilidad de reconocer una serie de comandos de voz en idioma español para poder movilizarse, como los son; alto, adelante, atrás, izquierda, derecha, rápido, lento. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

### Tabla 14. RF03

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RF03 | | | | | |
| **Nombre** | Identificación y clasificación de comandos de voz | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El sistema a desarrollar hará el reconocimiento de voz a través de un micrófono que funciona por puerto USB y una vez identificado el comando lo clasifica dentro de los comandos soportados. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

**Tabla 15. RF04**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RF04 | | | | | |
| **Nombre** | Implementación del modelo de reconocimiento de voz | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El modelo debe estar desplegado en una tarjeta programable Raspberry pi 4. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

**Tabla 16. RF05**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RF05 | | | | | |
| **Nombre** | Lenguaje de programación utilizado para el desarrollo del modelo | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El modelo debe estar desarrollado en el lenguaje Python. | | | | | |
| **Prioridad** | | Alta | x | Media | | Baja |

Fuente: Elaboración propia

**Tabla 17. RF06**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RF06 | | | | | |
| **Nombre** | Uso de librerías | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El Modelo debe implementar el uso de la librería Tensor Flow. | | | | | |
| **Prioridad** | | Alta | x | Media | | Baja |

Fuente: Elaboración propia

**Tabla 18. RNF01**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RNF01 | | | | | |
| **Nombre** | Tiempo de respuesta del modelo | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El modelo debe ofrecer tiempos de respuesta rápidos e inmediatos. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

**Tabla 19. RNF02**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RNF02 | | | | | |
| **Nombre** | Dependencia de Internet | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El modelo a desarrollar no debe depender de internet para su utilización. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

**Tabla 20. RNF03**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RNF03 | | | | | |
| **Nombre** | Eficiencia y capacidad de reconocer los comandos de voz | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El modelo debe asegurar el reconocimiento de voz en entornos controlados. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

**Tabla 21. RNF04**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RNF04 | | | | | |
| **Nombre** | Escalabilidad del Modelo RCV | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | Se debe asegurar la escalabilidad del modelo en cuanto a los comandos de voz que puede identificar. | | | | | |
| **Prioridad** | | Alta | x | Media | | Baja |

Fuente: Elaboración propia

**Tabla 22. RNF05**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RNF05 | | | | | |
| **Nombre** | Seguridad de los datos de prueba | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | Los datos de entrenamiento deben permanecer seguros. | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

**Tabla 23. RNF06**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RNF06 | | | | | |
| **Nombre** | Entrenamiento del modelo | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El modelo desplegado debe haber sido previamente entrenado | | | | | |
| **Prioridad** | x | Alta | | Media | | Baja |

Fuente: Elaboración propia

**Tabla 24. RNF07**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Número** | RNF07 | | | | | |
| **Nombre** | Sistema operativo que utiliza la Raspberry Pi. | | | | | |
| **Tipo** | x | Requisito | | | Restricción | |
| **Descripción** | El sistema operativo que debe implementar la Raspberry Pi es NOOBS, una distribución de Linux recomendada por el fabricante de Raspberry. | | | | | |
| **Prioridad** | | Alta | x | Media | | Baja |

Fuente: Elaboración propia