UNIVERSIDAD DE ESPECIALIDADES ESPÍRITU SANTO

Facultad de Ingeniería

“Comparación y uso de modelos de redes neuronales convolucionales aplicado al desarrollo de tecnología asistiva para identificación de productos”

Trabajo de Titulación que se presenta como requisito para el título de Ingeniero en Sistemas

Autor: Carlos Enrique Ríos González

Tutor: Ing. Washington Antonio Caraguay Ambuludi, M.Sc.

Samborondón, abril de 2020

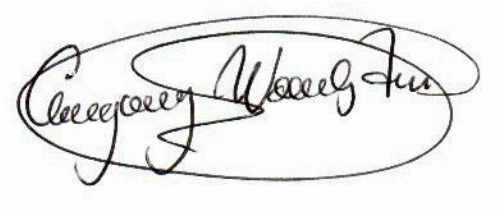
APROBACIÓN DEL TUTOR

En mi calidad de Tutor del estudiante Carlos Enrique Ríos González, que cursa estudios en el programa de TERCER nivel: Ingeniería en Sistemas, dictado en la Facultad de Ingeniería de la UEES, en modalidad presencial.

CERTIFICO:

Que he revisado el Trabajo de Titulación denominado: “Comparación y uso de modelos de redes neuronales convolucionales aplicado al desarrollo de tecnología asistiva para identificación de productos”, presentado por el estudiante Carlos Enrique Ríos González, como requisito previo para optar por el Grado Académico de Ingeniero en Sistemas CERTIFICO que el Trabajo de Titulación ha sido analizado y reúne todos requisitos para ser presentado y sometido a los procesos de revisión estipulados por la Facultad.

Atte.



Ing. Washington Antonio Caraguay Ambuludi, M. Sc

AGRADECIMIENTO

En primer lugar, quiero agradecer a Dios por darme la oportunidad de vida y permitirme culminar mi carrera profesional. Segundo, a mis padres que con tanto esfuerzo y sacrificio siempre estuvieron apoyándome en cada etapa. Tercero, a mi esposa y a mi hija por ser la inspiración para ser mejor persona cada día. Cuarto, a mi tutor Ing. Washington Caraguay por su apoyo incondicional para culminar mi carrera universitaria. Y, por último, agradezco a mis familiares, maestros y amistades que siempre me estuvieron apoyando.

Índice General

Índice de Tablas

Índice de Figuras

RESUMEN

La evolución tecnológica a lo largo del tiempo ha fomentado el desarrollo de diversas herramientas para diferentes áreas del saber humano. Considerando esta evolución las herramientas asistivas se han ido incorporando nuevos elementos más sencillos de emplear para personas con capacidades especiales, logrando ofrecerles hasta cierto punto una mejora en su calidad de vida. Mediante el estudio del aprendizaje profundo y el uso de redes neuronales convolucionales se han logrado construir soluciones que se caracterizan por su alta precisión.

El presente trabajo de titulación describe el desarrollo de una herramienta tecnológica que permita a las personas no videntes facilitar el proceso de identificación de productos de primera necesidad en una tienda o supermercado. Para lograrlo, se utilizó una metodología con enfoque cualitativo de alcance exploratorio, además de un diseño investigativo conformado por dos fases, configuración del modelo y desarrollo de la herramienta tecnológica; cuyas actividades estuvieron marcadas por la implementación de métodos ágiles y de un análisis comparativo de dos modelos de redes neuronales convolucionales, como son *InceptionV3* y *MobileNet* que se caracterizan por tener un alto rendimiento en la clasificación de imágenes.

Los resultados obtenidos de este trabajo de investigación evidencian que el modelo de red neuronal convolucional *MobileNet,* es idóneo para su implementación en la herramienta tecnológica desarrollada para ofrecer mejores resultados en cuanto a la precisión y velocidad para identificar productos. Por medio de las pruebas realizadas con personas en un ambiente que emuló el interior de una tienda, se determinó que la herramienta cumple la finalidad de dotarles de autonomía durante el proceso de compras. Como trabajo futuro, la herramienta puede ser potenciada mediante la inclusión de un conjunto de datos más extenso para otorgar a la persona no vidente de un mayor espectro de productos a elegir y comprar.

CAPÍTULO 1 INTRODUCCIÓN

En la actualidad, se puede evidenciar que la tecnología está presente en nuestras actividades diarias, tanto a nivel personal como profesional. Gracias a este auge y su acceso a todo público, han facilitado la creación de herramientas tecnológicas orientadas a varios grupos que conforman la sociedad; siendo uno de estos el integrado por las personas con discapacidad.

A lo largo del tiempo se han desarrollado varias herramientas que reciben el nombre de tecnologías asistivas, y han contribuido a superar las barreras que sufren estas personas y lograr su participación igualitaria en varios de los aspectos de la sociedad (Lilit , Lumsden, O'Sullivan, & Bartlett, 2013).

Particularmente, el uso de tecnologías asistivas para personas no videntes, buscan mejorar su calidad de vida dotándolos de autonomía y seguridad. También buscan fomentar su inclusión, apartándolos de su entorno normal para que interactúen con otras personas y reducir su miedo al aislamiento social (Hersh & Johnson, 2008).

Si bien existen varios avances tecnológicos, las personas con discapacidad visual se enfrentan a problemas como su dificultad de acceso a estas tecnologías o limitan a las personas a desempeñarse en sus actividades cotidianas; como es el proceso de compra de productos de primera necesidad en tiendas o supermercados. Por tal razón, el presente trabajo de titulación analiza modelos de redes neuronales convolucionales utilizando técnicas de visión artificial empleadas en el desarrollo de una herramienta tecnológica que permita a las personas no videntes facilitar el proceso de compra. Se ha implementado una metodología de enfoque cualitativo ya que se realizará un análisis comparativo de las características de dos modelos de redes neuronales convolucionales, como son *InceptionV3 y MobileNet*. Adicional, tiene un alcance exploratorio en el que se hará una revisión sistemática de los conceptos de las redes neuronales convolucionales, que permitan definir la solución tecnológica a

implementarse para la ubicación e identificación de productos; que justifiquen el marco de trabajo de la herramienta tecnológica. Como metodología de desarrollo se utiliza *Scrum* para generar parcialmente entregables funcionales de la solución planteada.

Este trabajo de titulación está constituido por cinco capítulos en los cuales se van detallando contenido teórico y procedimientos necesarios para la implementación final de la herramienta tecnológica.

El capítulo 1 contiene los antecedentes, la descripción del problema que se afronta, el alcance del tema, sus objetivos y la importancia por las cuales este trabajo debe ser estudiado.

El marco teórico se detalla en el capítulo 2, donde se realiza una revisión bibliográfica de los conceptos y componentes a emplearse en la presente investigación. Adicional se incluye un análisis comparativo de las características de los modelos de redes neuronales convolucionales utilizados para la ubicación e identificación de productos mediante técnicas de visión artificial.

En el capítulo 3 se describe la metodología empleada para alcanzar los objetivos descritos para este trabajo. Se define el enfoque, diseño de la investigación e instrumentos empleados, así como también la arquitectura, etapas y actividades para el desarrollo de la herramienta tecnológica.

Dentro de los contenidos del capítulo 4 se realizan pruebas entre los modelos indicados y se registran los resultados obtenidos de las comparaciones. Se revisa el diseño, codificación e implementación del software. Se registran los resultados de las pruebas del software.

Finalmente, en el capítulo 5 se presentan las conclusiones a partir de los resultados obtenidos y el aporte de la herramienta tecnológica para mitigar el problema planteado. Adicionalmente se indican las recomendaciones para trabajos futuros.

1.1 Antecedentes

De acuerdo con la Organización Mundial de la Salud (2011), aproximadamente el 15% de la población mundial vive con alguna forma de discapacidad siendo una de estas la ceguera. El desarrollo y la accesibilidad de la tecnología han fomentado que gran parte de estas personas actualmente puedan desenvolverse dentro de la sociedad. Este avance se ha logrado gracias a las tecnologías asistivas.

De acuerdo con Hans-Jörg Bullinger (2009), las tecnologías asistivas son cualquier dispositivo, sistema o servicio tecnológico, cuya finalidad sea aumentar, mantener o mejorar las capacidades funcionales de las personas con discapacidad, o reducir sus limitaciones al realizar alguna actividad.

Estas tecnologías asistivas permiten a las personas discapacitadas participar de manera independiente en actividades que cualquier otra persona puede hacer, buscando compensar la discapacidad (Toro-Hernández, y otros, 2019). Una de esas actividades es la de realizar sus propias compras en una tienda o supermercado. En la mayoría de estos establecimientos, la información sobre los productos no se encuentra adaptada para personas con discapacidad visual, por lo que se ven obligados a depender de otra persona que les ayude a identificarlos (Duarte, Cecílio, & Furtado, 2014).

En la actualidad, se han desarrollado muchas soluciones tecnológicas con el objetivo de abordar esta problemática y contribuir en la mejora del estilo de vida de las personas no videntes (Lilit , Lumsden, O'Sullivan, & Bartlett, 2013). Una de estas soluciones fue *RoboCart*, un prototipo que consistía en una adaptación de asistente robótico a un carrito de compras que se integraba a una computadora portátil, un láser para buscar y un lector de *RFID* para identificar los productos (Kulyukin, Gharpure, & Nicholson, 2005).

Como una alternativa mejorada de *RoboCart*, dos de sus creadores desarrollaron *ShopTalk*. Esta herramienta constaba de un lector de código

de barras, un ultra-computador portátil que se guardaba en una mochila y un auricular para la orientación en el supermercado (Nicholson, Kulyukin, & Coster, 2009).

ShopMobile intentó reducir las limitantes de *ShopTalk*. Funcionaba como un asistente de compras integrando a un teléfono inteligente del cual utilizaba su cámara integrada como lector de código de barras para los productos (Kulyukin & Kutiyanawala, 2010). Otro dispositivo que utilizaba esta funcionalidad era *Trinetra*, que buscaba convertirse en una tecnología rentable que aproveche la capacidad colectiva de varios dispositivos integrados en una red para mejorar la actividad de compra (Lanigan, Paulos, Williams, Rossi, & Narasimhan, 2006).

Otra solución tecnológica fue *BlindShopping* que ofrecía la posibilidad a las personas no videntes de navegar a través de los supermercados e identificar los objetos requeridos mediante un teléfono celular para lectura de código QR y un bastón adaptado a un lector *RFID* para identificar señales en el piso que le permitan la navegación (López-de-Ipiña, Lorido, & López, 2011).

EyeBuy se desarrolló como una solución que utiliza tecnología de realidad aumentada en una aplicación para asistir a las personas no videntes mediante la capacidad de identificación y reconocimiento de los productos en percha de una tienda por medio de la cámara de una tablet (Forte, Luciano, Pazini, & Marengoni, 2013).

Una solución más avanzada fue *Third Eye*, combinaba cámaras portátiles en gafas inteligentes y un guante que podía capturar el video que era transmitido inalámbricamente a un servidor para su análisis. Posteriormente el sistema utilizaba estos resultados para poder guiar a la persona discapacitada con comandos de audio y vibraciones en el guante para ubicar el producto en la percha para tomarlo (Zientara, y otros, 2017).

El Ecuador no ha sido ajeno a estas innovaciones, dentro de los proyectos más representativos se encuentran *HandEyes* y *Speakliz Vision*. La primera solución tecnológica es un dispositivo que basa su

funcionamiento en el sistema de ecolocalización, emitiendo ondas ultrasónicas que rebotan en los objetos y regresan al dispositivo para que a su vez alerte a la persona con sonidos y vibraciones de manera que pueda crear un mapa mental de su entorno (MINTEL/JLV, 2016). Por su parte *Speakliz Vision* consiste en una aplicación que emplea la cámara de un celular y mediante inteligencia artificial permite identificar objetos y sus distancias, lugares, colores, y billetes de diferentes divisas (Ecuador TV, 2019).

Por tal razón, se pretende investigar y comparar modelos de redes neuronales convolucionales para orientación y detección de productos por visión artificial que contribuyan a la inclusión de las personas no videntes en actividades laborales y cotidianas como la de realizar compras en supermercados.

1.2 Descripción del Problema

Aproximadamente 1300 millones de personas en el mundo viven con alguna forma de deficiencia visual, de las cuales 36 millones son invidentes (Organización Mundial de la Salud, 2018). Estas personas en la mayoría de los casos requieren un apoyo constante en varios escenarios en los que se desenvuelven, así como también durante sus actividades diarias.

Algunos de los principales desafíos que enfrentan incluyen la dificultad de moverse de un lugar a otro, la dificultad para reconocer a las personas, detectar obstáculos, identificar objetos, entre otros (Monika, y otros, 2019). Es por esta razón que recurren al uso de elementos como bastón blanco o perro guía, sin embargo, esto presenta incomodidad frente al hecho de utilizar un objeto que restringe el uso de sus extremidades o que no le permite generar sensación de autosuficiencia (Hersh & Johnson, 2008).

Las tecnologías asistivas han influido de manera positiva en el estilo de vida de las personas no videntes buscando superar estos desafíos por medio de sistemas de apoyo para acciones rutinarias como son leer,

escribir, caminar, navegar por internet. Sin embargo, en la realidad suelen seguir dependiendo de otra persona para realizar sus compras puesto que las tiendas y supermercados no prestan atención a las necesidades de este tipo de clientes, los cuales requieren de medios que les ayuden a identificar y ubicar los productos que desean adquirir (Jabeen, Muhammad, & Martinez Enriquez, 2015).

Si bien es cierto, a lo largo del tiempo se han desarrollado tecnologías asistivas aplicadas a personas no videntes, sin embargo, el bastón convencional o los perros guía siguen siendo los medios más utilizados por las personas no videntes (Fernandez, Xian, & Tian, 2017). Esta realidad demuestra la necesidad de enfocarse en la construcción de soluciones tecnológicas de bajo costo, que se adapten al contexto local y que contribuyan a reducir la dependencia.

En el contexto local, se puede evidenciar que las tiendas y supermercados no disponen de dispositivos que contribuyan a facilitar la actividad de compra de las personas no videntes, por lo que además de la investigación y la solución tecnológica a desarrollar en el presente trabajo de titulación, es necesario sugerir un estándar sistematizado de ubicación de los productos enfocado exclusivamente a la mejora de la inclusión de este grupo. Consecuentemente, mediante la investigación de los modelos de redes neuronales convolucionales para la ubicación y detección, y su aplicación para personas no videntes, se espera brindar una mejora durante el proceso de compra en un supermercado y garantizar su independencia.

1.3 Alcance

El presente trabajo de investigación se limita a la investigación de modelos de redes neuronales convolucionales y técnicas de visión artificial para su empleo en el diseño de una herramienta tecnológica que permita a las personas no videntes la ubicación e identificación de productos de primera necesidad ubicados en perchas dentro de un supermercado. La

herramienta estará conformada por un soporte para gafas de realidad virtual, en el cual se colocará un smartphone con el software de asistencia virtual. El usuario se colocará este soporte para realizar la interacción con el asistente virtual por medio de comandos de voz; el cual le permitirá preguntar por un producto específico. Una vez, identificado el producto deseado el asistente virtual guiará al usuario a la ubicación del producto y mediante la captura de una imagen se identificará el mencionado producto. Se realizarán pruebas de funcionalidad en un ambiente que emule el interior de una tienda, debido a la cuarenta comunitaria obligatoria especificada en el marco de emergencia por pandemia sanitaria, en el artículo 4 del Decreto ejecutivo 1017 de 16 de marzo del 2020.

1.4 Objetivos

1.4.1 Objetivo General

Implementar una herramienta tecnológica de ubicación e identificación de productos de primera necesidad mediante el uso de modelos de redes neuronales convolucionales y técnicas de visión artificial, contribuyendo a la inclusión de las personas no videntes en el proceso de compra dentro de un supermercado.

1.4.2 Objetivos Específicos

1. Realizar un análisis comparativo de las características de los modelos de redes neuronales convolucionales utilizados para la ubicación e identificación de productos mediante técnicas de visión artificial.
2. Aplicar modelos de redes neuronales convolucionales para la identificación de productos.
3. Implementar una herramienta tecnológica de ubicación e identificación de productos para personas no videntes.

4. Realizar pruebas de funcionalidad del software con personas en un ambiente que emule el interior de una tienda.

1.5 Justificación

De acuerdo con el Ministerio de Salud Pública del Ecuador (2019), existen aproximadamente 473.652 personas con discapacidad registradas, de las cuales el 11,74% cuentan con una discapacidad visual. El Ecuador contempla a estas personas dentro de los grupos vulnerables, estando protegidos bajo el Art. 48 literales 1 y 5 de la Constitución, en los que se expresa la necesidad de fomentar su inclusión y autonomía mediante programas.

En base a la problemática mencionada anteriormente, las personas con discapacidad visual no logran dicha inclusión y autonomía dentro del entorno de supermercados en el Ecuador. Sin embargo, los nuevos avances tecnológicos proporcionan herramientas idóneas para desarrollar soluciones que asistan a este grupo de personas y puedan cubrir sus necesidades y desenvolverse por sus propios medios (Hersh & Johnson, 2008).

Por medio de esta investigación se desea contribuir en la implementación de una herramienta tecnológica, basada en los modelos de redes neuronales convolucionales investigados para proporcionar a las personas no videntes mejoras en el proceso de compra dentro de una tienda o supermercado y garantizar su independencia.

A nivel internacional, este proyecto contribuye al cumplimiento de una de las metas de los objetivos de desarrollo sostenible aprobados por la ONU en 2015. El cual contempla el décimo objetivo que es la reducción de las desigualdades y que dentro de una de sus metas busca en el año 2030, potenciar y promover la inclusión social, económica y política de todas las personas (Organización de las Naciones Unidas, 2015).

La importancia de esta investigación radica, entonces, en el impacto social que pudiera tener en el desenvolvimiento diario de una persona no vidente, dotarlos de una herramienta intuitiva que brinde un aporte para este grupo vulnerable.

CAPÍTULO 2 MARCO TEÓRICO

En este capítulo se presentan los conceptos y teorías necesarias para el análisis comparativo de los modelos de redes neuronales para la navegación asistida e identificación de imágenes. Como punto de partida se revisó el concepto de aprendizaje automático seguido de su relación con el aprendizaje profundo, para posteriormente describir uno de sus componentes, las redes neuronales convolucionales. También se estudian conceptos de los componentes relacionados al desarrollo de la herramienta tecnológica, como son *Tensorflow*, el cual permitirá utilizar de modelos para la identificación de imágenes como *InceptionV3* y *MobileNet* para la tarea específica de navegación y reconocimiento de productos. Otros componentes como *Android, SQLite* y el asistente de voz virtual también son revisados en este capítulo, de la misma manera que *Scrum* como metodología de desarrollo ágil a emplearse.

2.1 Aprendizaje automático

Gavilán (2019), expresa que el incremento en la capacidad de aprendizaje de las máquinas se debe a que el aprendizaje automático (*Machine learning* en inglés) y el aprendizaje profundo (*Deep learning* en inglés) son pilares de la inteligencia artificial, tal como se muestra en la Figura 2.1.

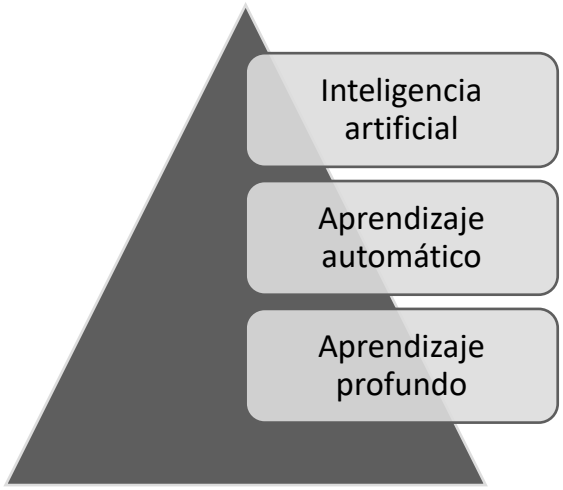


Figura 2.1 Relación entre inteligencia artificial, aprendizaje automático y aprendizaje profundo
Fuente: Modificación propia basado en (Sze, Chen, Yang, & Emer, 2017)

Por medio de la relación mostrada en la Figura 2.1, Mitchell (2019) señala que el aprendizaje automático está presente en varios aspectos de la vida cotidiana por ejemplo en sitios web para recomendar películas, sistemas de reconocimiento de voz o automóviles autónomos; mientras que el aprendizaje profundo se puede aplicar en tecnologías asistivas que permitan el reconocimiento de objetos mediante lentes o teléfonos inteligentes (Mulfari, Palla, & Fanucci, 2017).

Según Paluszek & Thomas (2017), el aprendizaje automático es una rama de la inteligencia artificial donde los datos existentes se utilizan para predecir o responder a datos futuros. Es decir, proporcionan un conjunto de técnicas que facilitan el análisis de estos datos y permiten la construcción de modelos (Smith, et al.,2019). Nayak & Dutta (2017) señalan que las áreas donde se puede aplicar el aprendizaje automático se han extendido gracias a los avances tecnológicos; llegando a convertirse en una técnica clave para la búsqueda de soluciones como se muestra en la Tabla 2.1.

Tabla 2.1
Aplicaciones con aprendizaje automático

APLICACIÓN	
Reconocimiento y detección de rostros	Las cámaras pueden detectar estados de ánimo y gestos, como una sonrisa.
Percepción visual	Reconocimiento de patrones y análisis de escenas.
Clasificación	Segregación de la información recibida en función del contenido que tiene.
Modelado	Para predecir el comportamiento entre objetos reales y objetos mediante sistemas de resolución de problemas.
Reconocimiento de voz	Reconocimiento por idioma hablado y texto, procesamiento semántico.
Detección de anomalías	Aplicaciones para detección de fraudes, envío de correo malicioso.
Biología computacional	Detección de tumores, anomalías cardiacas, entre otros.

Fuente: Modificación propia basado en (Nayak & Dutta, 2017)

El aprendizaje automático emplea algoritmos que permiten que las computadoras aprendan iterativamente de los datos. A continuación, podemos observar en la Figura 2.2 que el proceso de aprendizaje automático empieza con la selección de datos, seguido de la extracción manual de sus características más relevantes que serán utilizadas para la creación de un modelo, por ejemplo, que permita categorizar objetos a partir de una imagen. Finalmente, se entrena el modelo para que brinde una respuesta precisa. Una vez entrenado el modelo se podrá proporcionar la imagen de un objeto para que proceda a clasificarlo (Reddy, 2019).

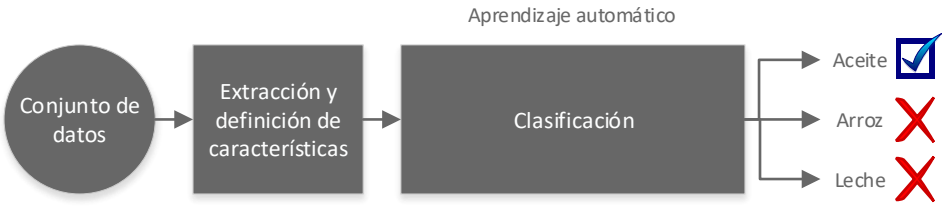


Figura 2.2 Flujo de trabajo del aprendizaje automático
Fuente: Adaptación de (Binhua, Zixiang, Kang, & Asif, 2019)

2.2 Aprendizaje profundo

El aprendizaje profundo (*Deep learning* en inglés) al ser uno de los pilares de la inteligencia artificial, es usado en múltiples aplicaciones como son detección de malware en teléfonos inteligentes (Naway & Li, 2018), detección de enfermedades gastrointestinales a partir de imágenes de biopsias (Srivastava, y otros, 2019) e incluso para la detección y clasificación de vehículos (Orozco & Corazón, 2019). Particularmente, las técnicas de aprendizaje profundo también se emplean para el desarrollo de tecnologías asistivas para personas no videntes, como: sistemas embebidos para la detección de productos (Mulfari, Palla, & Fanucci, 2017) o para la clasificación de imágenes que permitan entender el lenguaje de señas (Daroya, Peralta, & Naval, 2018).

En el campo de las tecnologías asistivas, los teléfonos móviles se han convertido en una plataforma óptima para el desarrollo de aplicaciones que empleen aprendizaje automático buscando mejorar la calidad de vida de las personas con discapacidades durante sus tareas diarias (Bauer, y otros, 2019). Esto se logra gracias a los niveles de precisión que se han conseguido, de tal forma que hasta los últimos avances del aprendizaje han logrado superar a las personas en tareas como la clasificación de objetos e imágenes, prescindir de personas en vehículos que puedan desplazarse a un punto determinado detectando señales de tránsito, peatones y luces de un semáforo (Sewak, 2019).

A diferencia del aprendizaje automático, el proceso del aprendizaje profundo empieza con la extracción de las características más relevantes directamente del conjunto de datos proporcionado, por ejemplo, una imagen. Una vez seleccionada la arquitectura apropiada, se procede a entrenar el modelo y se evalúa su respuesta (Li, Habesab, Wolk, & Fana, 2019); cómo se puede apreciar en la Figura 2.3. Cabe recalcar que a mayor conjunto de datos proporcionados mayor será la precisión con la que estos modelos clasifiquen objetos o productos.

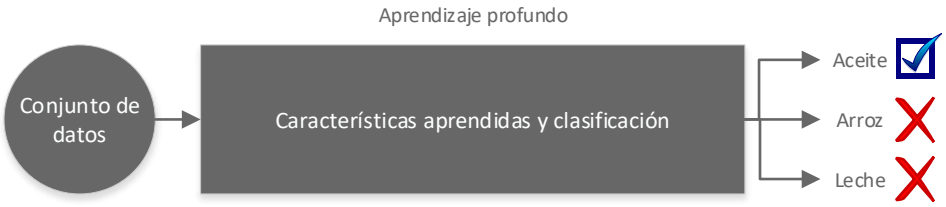


Figura 2.3 Flujo de trabajo del aprendizaje profundo
Fuente: Adaptación de (Binhua, Zixiang, Kang, & Asif, 2019)

Para lograr este nivel de aprendizaje, se utiliza un componente fundamental conocido como redes neuronales profundas. Las redes neuronales son algoritmos inspirados en la estructura del cerebro humano, constituidas por pequeños nodos computacionales como neuronas artificiales con una función en base a las entradas, como se muestra en la Figura 2.4, simulando de esta manera la sinapsis de las neuronas cerebrales (Fawzy Gad, 2019).

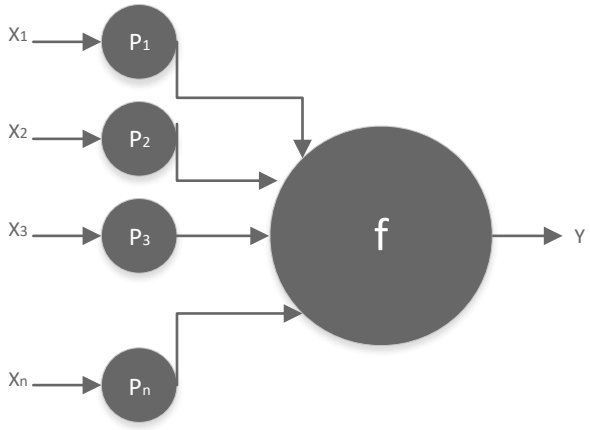


Figura 2.4 Representación gráfica de una neurona artificial
Fuente: Modificación propia basado en (Sze, Chen, Yang, & Emer, 2017)

Las entradas (Xi) son matizadas por un peso (Pi) y obtener un valor de salida se aplica una función (f). El aprendizaje se traduce en la modificación de los pesos que hay entre cada conexión de neuronas. Estas neuronas están conectadas unas a otras, procesando y propagando la

señal de entrada proporcionada entre capas. A esta agrupación se la conoce como una red neuronal, representada gráficamente en la Figura 2.5 y de acuerdo con Fawzy Gad (2019), pueden procesar una gran cantidad de conjuntos de datos como entrada y operar a través de múltiples capas.

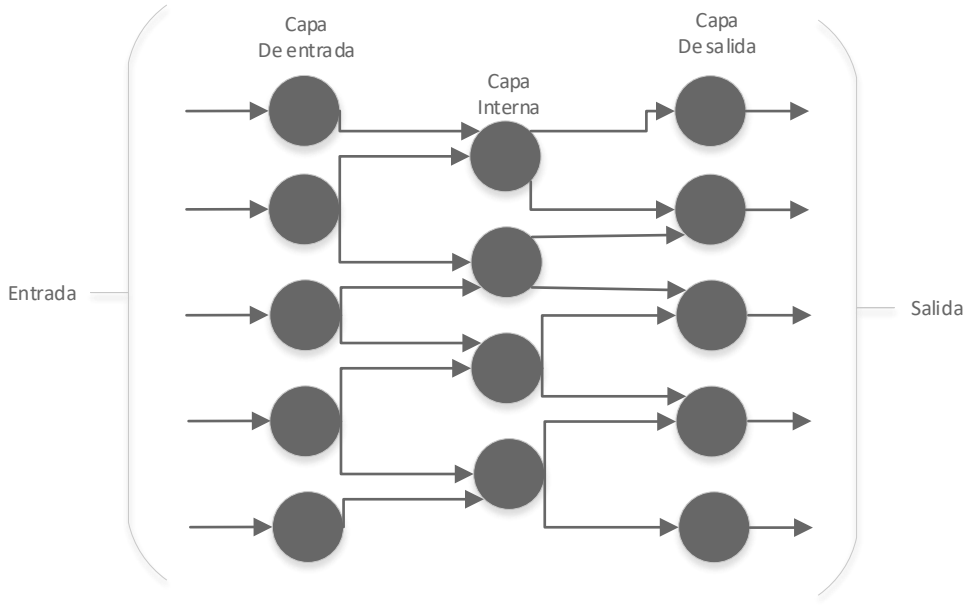


Figura 2.5 Representación gráfica de una red neuronal
Fuente: Modificación propia basado en (Sze, Chen, Yang, & Emer, 2017)

Los algoritmos de redes neuronales cuentan con al menos una capa de entrada y una capa de salida, entre estas dos capas puede haber un número variable de capas internas. Como menciona Fawzy Gad (2019), el número de capas es lo que diferencia a los varios tipos existentes de redes neuronales, a partir de esta definición nace el término red neuronal profunda, refiriéndose al número de capas que posee; cuyo número de redes neuronales tradicionales puede ir entre dos a tres capas y en redes neuronales profundas puede llegar hasta más de 150 capas (Michelucci, 2018). Una de estas, son las redes neuronales convolucionales, las cuales se describen en el siguiente apartado.

2.2.1 Redes neuronales convolucionales

Las redes neuronales convolucionales (*CNN*, por sus siglas en inglés) son una de las redes neuronales profundas más populares y conocidas. Son consideradas un caso particular de las redes neuronales de aprendizaje profundo y surgieron en los años 90 (Peña, Bonet, Manzur, Góngora, & Caraffini, 2019), no obstante, su uso se ha incrementado en las últimas décadas. De acuerdo con Peña, Bonet, Manzur, Góngora, & Caraffini (2019) se emplean principalmente para la identificación y clasificación de características en imágenes, sonidos, videos y texto.

Ahmed, Mahmud, & Yeasin (2019) añaden que el reconocimiento por medio de imágenes fomenta el desarrollo de tecnologías asistivas, por ejemplo, en la detección de objetos para personas no videntes (Mulfari, Palla, & Fanucci, 2017), aplicaciones móviles que brinden apoyo a personas con dislexia o disgrafía (Avishka, Kumarawadu, Kudagama, Weerathunga, & Thelijjagoda, 2019), asistentes de compras para no videntes (Pintado, y otros, 2019), entre otras.

Su uso extendido en las herramientas tecnológicas mencionadas anteriormente se debe a que estas redes se caracterizan por poseer una gran precisión gracias a que los modelos generados se pueden entrenar constantemente (Mocanu, Tapu, & Zaharia, 2019) y a la capacidad de contar con recursos que facilitan el desarrollo de aplicaciones que se basen en estas técnicas. En la actualidad, el uso de las GPUs permite reducir el tiempo de entrenamiento que necesita un modelo, de manera que se pueden aprovechar cientos, miles o millones de imágenes como conjuntos de datos de entrada (Mulfari, Palla, & Fanucci, 2017).

Adicionalmente, como expresan Peña, Bonet, Manzur, Góngora, & Caraffini (2019) las redes neuronales convolucionales tienen características que las convierten en una solución para problemas complejos en los que se necesita reconocer patrones en imágenes, transformar problemas en mapas de características y la integración de funciones para una mejor identificación de las características.

Este tipo de redes pueden tener decenas o cientos de capas mediante las cuales se lleva a cabo el aprendizaje y detección de las características de una imagen que se ha aprendido, de esta manera, por ejemplo, la primera capa podría aprender la detección de formas complejas correspondientes al objeto a reconocer, una segunda capa podría centrarse en la detección de bordes (McClure, 2017).

El flujo de trabajo de una red neuronal convolucional, como lo explican Lopez, Vieira, Garcia-Dias, y Mechelli (2019), empieza con un conjunto de datos, los cuales son procesados durante la fase de convolución en la que se realizan varias operaciones y luego se le atribuye una etiqueta. El flujo de trabajo básico de este tipo de red se puede apreciar en la Figura 2.6.

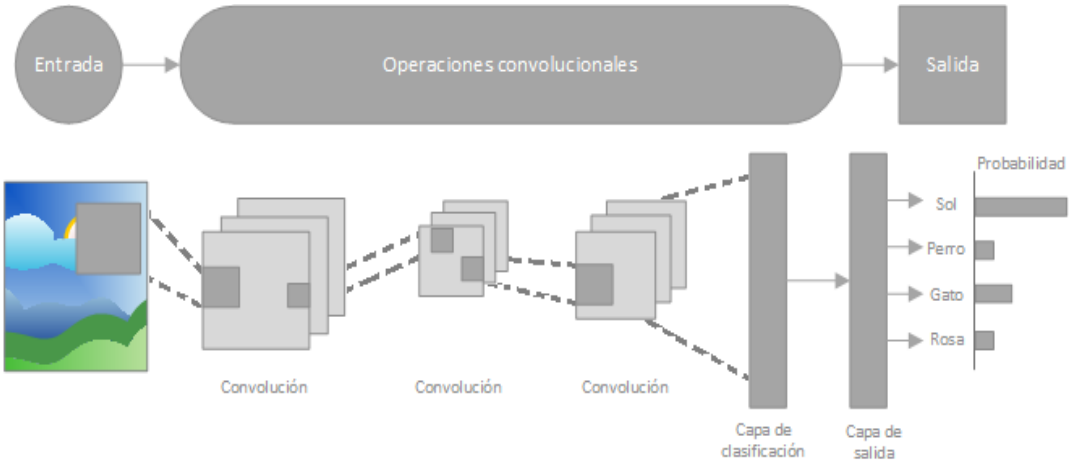


Figura 2.6 Flujo de trabajo básico de una red neuronal convolucional
Fuente: Modificación propia basado en (Ameen & Vadera, 2017)

Las capas realizan operaciones que van alterando el conjunto de datos, con la finalidad de aprender las características específicas que tienen estos datos. Según Fawzy Gad (2019), en una red neuronal convolucional generalmente se tiene como capas básicas:

- La de convolución, en la que el conjunto de datos de la imagen es pasado a través de un conjunto de filtros convolucionales, para poder activar determinadas características que posea la imagen.

- La unidad lineal rectificada (*ReLU*) que permite que el entrenamiento sea rápido y efectivo, solo las características apropiadas siguen su curso a la siguiente capa.
- La de agrupación que reduce la cantidad de parámetros que emplea la red para el aprendizaje. Estas capas básicas son esenciales para la extracción de las características de un conjunto de datos de una imagen.

2.2.2 Marcos de trabajo para aprendizaje profundo

En los últimos años, el aprendizaje profundo ha demostrado tener potencial para el desarrollo de mejores herramientas y productos de software orientados a dar soluciones a problemas reales como la asistencia a personas no videntes (Gramajo, Ballejos, & Ale, 2018). Por ejemplo, cuando el programador desea implementar un modelo para identificación de imágenes o productos como componente de un software recurre a un marco de trabajo para aprendizaje profundo. Paluszek y Thomas (2017) señalan que este tipo de marcos de trabajo consisten en una interfaz, biblioteca o herramienta que permite a los programadores crear e integrar modelos de aprendizaje automático de una forma más sencilla.

De acuerdo con Orozco & Corazón (2019) aplicar estos modelos es menos abrumador y desafiante de lo que solía ser, debido a que en la actualidad estos marcos integran varias características que los hacen viables para la elaboración de proyectos orientados a la identificación de productos. Entre los marcos para aprendizaje profundo más comunes podemos encontrar: *Keras, Infer.net, PyTorch, Theano, Tensorflow, Microsoft Cognitive Toolkit* (*CNTK*), *Caffe*, *MXNet*. A continuación, se describe cada uno de estos marcos de trabajo.

Moolayil (2019) define a *Keras* como un API que implementa redes neuronales de alto nivel, escrita en *Python* lo que brinda un gran soporte por parte de la comunidad de usuarios que emplean este lenguaje. Fue

desarrollada con un enfoque que permite la experimentación rápida y es compatible con otros marcos de trabajo como Tensorflow (Keras, 2018).

Infer.net es un marco de trabajo de código abierto desarrollado por Tom Minka y John Winn con el fin de ejecutar inferencia bayesiana en modelos gráficos. Está diseñado para ofrecer algoritmos para el modelado probabilístico y herramientas analíticas como el análisis bayesiano, la cadena oculta de Markov, la agrupación (NET Foundation and Contributors, 2019).

PyTorch por su parte ofrece un amplio soporte de algoritmos de aprendizaje automático. Fue desarrollado por el grupo de inteligencia artificial de *Facebook* y se suele emplearse en proyectos que requieren un corto tiempo de desarrollo y sin grandes flujos de trabajo complejos (Pytorch, 2018).

Theano fue desarrollado por la Universidad de Montreal en 2007, se caracteriza por ofrecer rápida capacidad de cómputo y puede ser ejecutado tanto en el CPU como GPU; se utiliza particularmente para el desarrollo y entrenamiento de modelos basados en redes neuronales profundas (Theano, 2018).

De acuerdo a Pattanayak (2017), *Tensorflow* es una buena opción para emplearse en proyectos que empleen redes neuronales multicapa. Creado por Google y escrito en C++ y Python, Tensorflow es un marco de trabajo de código abierto especializado en aprendizaje profundo que cuenta con su propia versión para ejecutar procesos en dispositivos móviles y permite la implementación conjunta de otros marcos como Keras (Google, 2018).

Caffe es un marco de trabajo implementado en C++, soporta el desarrollo de proyectos basados en redes neuronales convolucionales. Además, tiene un buen rendimiento para el entrenamiento de modelos sin necesidad de codificación adicional y procesamiento de imágenes. Fue desarrollado por Yangqing Jia (Caffe, 2015).

MXNet al disponer de una alta escalabilidad, es ideal para proyectos de aprendizaje profundo a nivel industrial. Se emplea para el reconocimiento de voz y texto, redes neuronales y predicciones. Fue creado por Apache y soporta varios lenguajes. Es utilizado por compañías como Amazon e Intel (Apache Software Foundation, 2017).

En la Tabla 2.2 se muestra una comparación de los marcos de trabajo descritos y sus características más relevantes.

Tabla 2.2
Comparación de marcos de trabajo para aprendizaje automático.

Marco	Desarrollo con Android	Escrito en	Disponibilidad	Entrenamiento de modelos	CNN
Keras	No	*Python*	Código abierto	Sí	Sí
Infer.net	No	*.NET*	Código abierto	No	No
PyTorch	No	*Python, CUDA, C++*	Código abierto	Sí	Sí
Theano	No	*Python*	Código abierto	Sí	Sí
Tensorflow	Sí	*Python, C++*	Código abierto	Sí	Sí
Caffe	No	*C++*	Código abierto	Sí	Sí
MXNet	No	*C++*	Código abierto	Sí	Sí

Fuente: Elaboración Propia

Los marcos de trabajo descritos permiten el desarrollo de proyectos que implementan componentes del aprendizaje profundo, sin embargo, para el desarrollo de la herramienta tecnológica propuesta se ha optado por Tensorflow. Mulfari, Palla, & Fanucci (2017) indican que este marco se ajusta al hardware y al proceso de clasificación de imágenes aprovechando las características y propiedades de los potentes modelos de aprendizaje automático basados en redes neuronales convolucionales. Entre una de

sus ventajas al ser uno de los marcos de uso más extendido, permite el desarrollo de proyectos móviles en Android y cuenta con gran soporte documental. En la siguiente sección se describen los principales detalles de este marco.

Tensorflow

Para Fatih & Galip (2017) *Tensorflow* es un gran marco de trabajo no solo por ser de código abierto, sino porque incorpora diferentes APIs para implementar arquitecturas de aprendizaje profundas como las redes neuronales convolucionales o redes neuronales recurrentes. Xiaoling, Cui, & Bing (2017) añaden que cuenta con alta disponibilidad, flexibilidad y apoyo por parte de los propios investigadores, lo que permite que se actualice constantemente y añadan nuevas características, por ejemplo, modelos pre-entrenados para el desarrollo de aplicaciones y tecnologías asistivas.

En las aplicaciones orientadas a tecnologías asistivas, permite su integración para trabajar en conjunto con otros componentes y tecnologías, logrando ofrecer, como por ejemplo, a una persona no vidente de dispositivos para la detección de objetos basados en el internet de las cosas (Mulfari, 2018) o emplear los modelos entrenados en aplicaciones para dispositivos móviles inteligentes que permitan escuchar indicaciones de su entorno para poder desplazarse (Shehieb, Osama Nasri, Mohammed, Debsi, & Arshad, 2018).

Para el área de interés de este proyecto, correspondiente a la detección de objetos en una herramienta tecnológica, *Tensorflow* cuentan con soporte para la implementación de modelos para identificación de imágenes como son *InceptionV3* y *MobileNet* los cuales se describen en los siguientes apartados.

InceptionV3

Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna (2015) señalan que es un modelo creado por Google e introducido en el 2015. Actualmente ha sido perfeccionado para brindar resultados óptimos en la labor de reconocimiento y clasificación de imágenes (McClure, 2017). Este modelo tiene una precisión del 94.39% (He, et al., 2018). En el campo del desarrollo de software puede ser implementado fácilmente empleando los marcos de trabajo Tensorflow y Keras. Bankar & Gavai (2018) indican que en conjunto con *Tensorflow*, posee la documentación necesaria para poder reentrenar este modelo añadiendo nuevas categorías.

InceptionV3 tiene un uso particular en el campo de la medicina en aplicaciones que sirven como soporte para la detección de enfermedades a partir de la clasificación de imágenes, logrando encontrar estudios en los que, por ejemplo, se emplea para la detección de cáncer de mama (Chougrad, Zouaki, & Alheyane, 2014), para diferenciar la adenopatía cervical mediante imágenes (Guan, y otros, 2019) o clasificar las imágenes pulmonares para detectar anomalías evitando diagnósticos erróneos (Wang, y otros, 2019).

Este modelo, se caracteriza por incorporar una estructura denominada inception, como se puede apreciar en la Figura 2.7, con la finalidad de actuar como un extractor de características de varios niveles mediante el cálculo de convoluciones de 1x1, 3x3 y 5x5; los resultados de estas convoluciones se apilan antes de continuar la siguiente capa de la arquitectura del modelo.

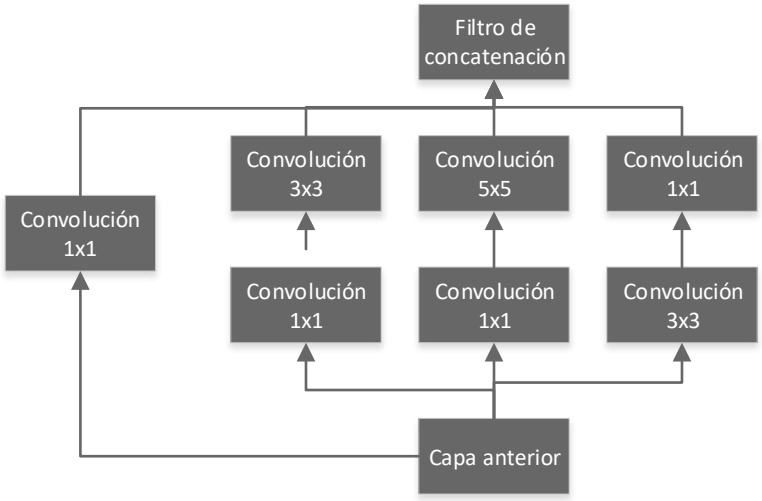


Figura 2.7 Representación gráfica de una estructura inception
Fuente: Modificación propia basado en (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2015)

MobileNet

Es uno de los modelos más populares y ampliamente usados para la implementación de redes neuronales convolucionales (Howard, y otros, 2017), enfocado específicamente en la labor de reconocimiento y detección imágenes como se aprecia en la figura 2.8.

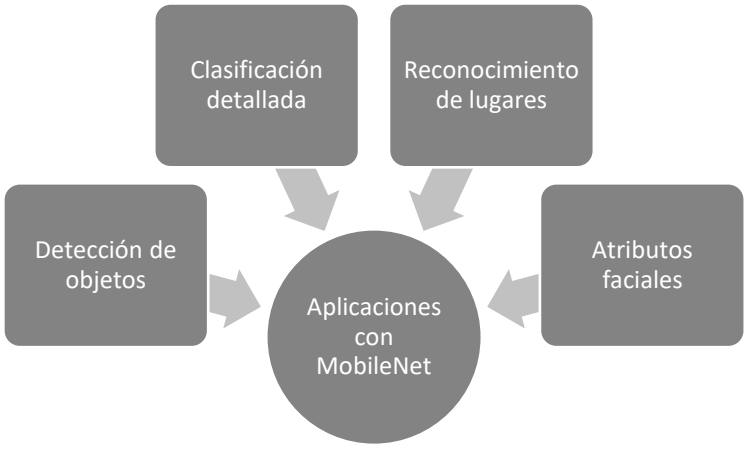


Figura 2.8 Aplicaciones del modelo MobileNet en dispositivos móviles
Fuente: Modificación propia basado en (Howard, y otros, 2017)

El modelo *MobileNet* tiene una precisión del 91.30% (He, et al., 2018) y puede ser empleado con *Tensorflow*, ofreciéndolo pre-entrenado con un conjunto de datos de ImageNet (Nitin, Yashashree, Seema, & Rashmi, 2017). De acuerdo con Wannipa, Wiphada, & Pattara (2019), *MobileNet* fue diseñado para poder ejecutarse en dispositivos móviles que cuentan con recursos limitados como teléfonos, gafas y relojes inteligentes. Considerando esta propiedad, se puede emplear para desarrollar herramientas para personas con discapacidades como por ejemplo, en una solución asistida de gafas inteligentes basada en realidad aumentada para personas con trastorno del espectro autista (Machado, Carrillo, Saldana, Chen, & Chen, 2019), diseñando una red neuronal eficiente y liviana para el reconocimiento visual del habla en dispositivos móviles (Shrivastava, y otros, 2019), aplicaciones ejecutadas en Android para el reconocimiento de objetos utilizando redes neuronales para personas con discapacidad visual (Dosi, Sambare, Singh, Lokhande, & Garware, 2018).

En lo referente a su arquitectura está compuesto por 30 capas y se caracteriza por usar convoluciones separables en profundidad en lugar de convoluciones estándar, lo cual se traduce en una reducción significativa del número de parámetros comparado con una convolución normal, de esta manera se puede emplear una red neuronal liviana. Solo se usa una convolución estándar en la primera capa y el resto de las capas son las que realizan la convolución separable en profundidad (Hong-Yen & Chung-Yen, 2018). Se estandariza y normaliza los *inputs* de cada capa de la red por medio de un *Batchnorm y ReLU;* excluyendo la capa *FC* (*Fully Connected* por siglas en inglés). Luego, se realiza un *Pooling* promedio para disminuir la resolución espacial de la imagen a 1 y enviar los datos a la capa *FC,* la cual también recibe información de la capa de clasificación *Softmax* (Muñoz Moreno, 2019). La arquitectura descrita se visualiza a continuación en la Tabla 2.3.

Tabla 2.3
Arquitectura de MobileNet

Tipo / Paso	Forma del filtro	Tamaño de la entrada
Conv / s2	$3\ x\ 3\ x\ 3\ x\ 32$	$224\ x\ 224\ x\ 3$
Conv dw / s1	$3\ x\ 3\ x\ 32\ dw$	$112\ x\ 112\ x\ 32$
Conv / s1	$1\ x\ 1\ x\ 32\ x\ 64$	$112\ x\ 112\ x\ 32$
Conv dw / s2	$3\ x\ 3\ x\ 64\ dw$	$112\ x\ 112\ x\ 64$
Conv / s1	$1\ x\ 1\ x\ 64\ x\ 128$	$56\ x\ 56\ x\ 64$
Conv dw / s1	$3\ x\ 3\ x\ 128\ dw$	$56\ x\ 56\ x\ 128$
Conv / s1	$1\ x\ 1\ x\ 128\ x\ 128$	$56\ x\ 56\ x\ 128$
Conv dw / s2	$3\ x\ 3\ x\ 128\ dw$	$56\ x\ 56\ x\ 128$
Conv / s1	$1\ x\ 1\ x\ 128\ x\ 256$	$28\ x\ 28\ x\ 128$
Conv dw / s1	$3\ x\ 3\ x\ 256\ dw$	$28x\ 28\ x\ 256$
Conv / s1	$1\ x\ 1\ x\ 256\ x\ 256$	$28x\ 28\ x\ 256$
Conv dw / s2	$3\ x\ 3\ x\ 256\ dw$	$28x\ 28\ x\ 256$
Conv / s1	$1\ x\ 1\ x\ 256\ x\ 512$	$14\ x\ 14\ x\ 256$
$5\ x$ Conv dw / s1 $5\ x$ Conv / s1	$3\ x\ 3\ x\ 512\ dw$ $1\ x\ 1\ x\ 512\ x\ 512$	$14\ x\ 14\ x\ 512$ $14\ x\ 14\ x\ 512$
Conv dw / s2	$3\ x\ 3\ x\ 512\ dw$	$14\ x\ 14\ x\ 512$
Conv / s1	$1\ x\ 1\ x\ 512\ x\ 1024$	$7\ x\ 7\ x\ 512$
Conv dw / s2	$3\ x\ 3\ x\ 1024\ dw$	$7\ x\ 7\ x\ 1024$
Conv / s1	$1\ x\ 1\ x\ 1024\ x\ 1024$	$7\ x\ 7\ x\ 1024$
Avg Pool / s1	Pool $7\ x\ 7$	$7\ x\ 7\ x\ 1024$
FC / s1	$1024\ x\ 1000$	$1\ x\ 1\ x\ 1024$
Softmax / s1	Classifier	$1\ x\ 1\ x\ 1000$

Fuente: Elaboración propia basado en (Muñoz Moreno, 2019)

A partir de la revisión de los modelos para identificación de imágenes *InceptionV3* y *MobileNet* en los aparatados anteriores se han comparado sus principales características en la Tabla 2.4.

Tabla 2.4
Comparación de modelos de identificación de imágenes.

Modelo	Tipo de convolución	Precisión	Rendimiento
InceptionV3	Convolución estándar	94.39 %	Alto
MobileNet	Convolución separable en profundidad	91.30 %	Alto optimizado para dispositivos móviles

Fuente: Elaboración propia basado en (Dosi, Sambare, Singh, Lokhande, & Garware, 2018)

En base a la revisión y comparación realizada de los modelos para identificación de imágenes, se considera que *MobileNet* es un modelo apropiado para efectos del desarrollo de la herramienta tecnológica que servirá para asistir a las personas no videntes, al generar modelos de reducido tamaño, contar con baja latencia y baja potencia parametrizados para cumplir con las limitaciones de recursos en dispositivos móviles (Howard, y otros, 2017).

2.3 Android

Android es el resultado de la propuesta de *Google* y *Open Handset Alliance* para el desarrollo de estándares abiertos para teléfonos inteligentes con la finalidad de lograr su evolución masiva en el mundo de los dispositivos móviles. De acuerdo a *International Data Corporation*, actualmente Android ronda el 87% de la cuota de mercado de teléfonos inteligentes (Melissa & Reith, 2019) convirtiéndose en un sistema operativo móvil dominante en los últimos años.

Según Allen & Murphy (2011) este sistema operativo no solo se limita a estar presente en teléfonos móviles inteligentes, sino que también se lo emplea en *tablets*, relojes inteligentes, automóviles y electrodomésticos.

2.4 SQLite

Jesse Feiler (2015) define SQLite como una librería de software escrita en lenguaje C. Fue desarrollada por D. Richard Hipp en el 2000 con la finalidad de proporcionar una manera conveniente para que las aplicaciones puedan administrar datos sin la necesidad de contar con un sistema dedicado de administración de base de datos relacionales (Allen & Owens, 2010).

Se caracteriza por ser altamente portátil, fácil de usar, compacta, eficiente y confiable por lo que suele emplearse ampliamente en la gestión

de datos de entornos integrados, como teléfonos inteligentes (Qinlong, Xingmei, Weiwei, & Minghai, 2010).

2.5 Asistentes de voz virtuales

La empresa norteamericana Apple introdujo en el 2011 a Siri, un asistente de voz incorporado a un dispositivo móvil. Actualmente en el mercado estos asistentes no solo están incorporados en teléfonos inteligentes, sino que también se han desarrollado altavoces inteligentes y otros dispositivos que los incorporan, como por ejemplo Google y su asistente Google Home, Microsoft con Cortana o Amazon con su asistente Alexa (Guzman, 2019).

Los asistentes de voz virtuales emplean tecnología para el reconocimiento de voz, cuentan con una gran aceptación al poseer un gran potencial para apoyar a sus usuarios en sus acciones como llamar a personas, pedir direcciones o buscar información en internet, activar una alarma, entre otras acciones (Burbach, y otros, 2019).

Para implementar un asistente de voz virtual en una aplicación se requiere emplear APIs para convertir audio en texto y realizar el reconocimiento de voz. Entre las más empleadas se encuentran: *Cloud Speech, Mycroft, Api.ai*, entre otros.

2.5.1 Cloud Speech

Cloud Speech es una *API* desarrollada por *Google* que permite a los programadores emplear el aprendizaje automático para convertir voz en texto aplicando modelos de redes neuronales. Se caracteriza por reconocer 110 idiomas (Google, 2018), también es posible realizar transcripciones usando el micrófono de como interfaz de entrada, habilitar el comando y control de voz.

2.6 Scrum

Scrum fue creado por Jeff Sutherland y Ken Schwaber entre 1993 y 1995. En la actualidad es una de las metodologías más adaptadas por los equipos de trabajo para proyectos de desarrollo de software (Lei, Ganjeizadeh, Jayachandran, & Ozcan, 2017). Se basa en flexibilidad, adaptabilidad y productividad, permitiendo que los desarrolladores elijan las técnicas específicas de desarrollo de software, métodos y prácticas para el proceso de implementación.

Enric Senabre (2019) menciona que *Scrum* facilita la actividad coordinada de los programadores que dividen su trabajo en pequeñas tareas que son completadas en ciclos de duración fija o también conocidos como *sprints*, en los que se va registrando el progreso y planificando reuniones periódicas para desarrollar el software de forma incremental. El objetivo de cada *sprint* es crear un producto que sea utilizable y potencialmente puede liberarse. Al final de cada *sprint* se realiza una inspección para identificar mejoras en el próximo (Liu, Ho, Chang, & Chia-An Tsai, 2019).

Como se ha podido apreciar en este capítulo, el aprendizaje profundo se ha extendido en múltiples aplicaciones que permiten la clasificación de objetos, reconocimiento facial, desarrollo de automóviles autónomos, entre otros. En el área de las tecnologías asistivas, las aplicaciones basadas en el aprendizaje profundo que emplean redes neuronales convolucionales han permitido conjugar las capacidades de clasificación y reconocimiento en imágenes que son tomadas a pacientes para determinar la posible existencia de enfermedades como el cáncer, logrando convertirse en herramientas útiles para su diagnóstico gracias a su precisión. De igual manera esta tecnología se ha empleado en conjunto con dispositivos móviles, siendo la herramienta preferida para el desarrollo de aplicaciones enfocadas a personas discapacitadas. Para las personas no videntes, este tipo de aplicaciones ayudan a comprender su entorno,

detectar obstáculos en su casa o espacios abiertos, o como el caso particular de este trabajo de titulación, permitirle identificar productos dentro un supermercado.

CAPÍTULO 3 METODOLOGÍA

En este capítulo se describe a detalle el trayecto que seguirá para cumplir con el propósito de la investigación acorde al alcance previamente establecido. En los siguientes párrafos, se detallan: el enfoque, alcance y diseño de este trabajo.

La presente investigación tiene un enfoque exploratorio ya que se realizará una revisión sistemática de los conceptos con relación al aprendizaje automático, el aprendizaje profundo, las redes neuronales convolucionales y los marcos de trabajo para aprendizaje profundo entre los cuales se destaca *Tensorflow* para la implementación de modelos de identificación de imágenes como son *InceptionV3 y MobileNet*. Luego de esta revisión, se realizará un análisis cualitativo de las características de los modelos mencionados anteriormente y en base al modelo seleccionado, se desarrollará e implementará una herramienta tecnológica que permita la ubicación e identificación de productos dentro de una tienda o supermercado con la finalidad de contribuir a mejorar la autonomía e inclusión de las personas con discapacidad visual.

El diseño de la investigación consiste en dos etapas; configuración del modelo y desarrollo de la herramienta tecnológica. En la etapa de configuración del modelo se levantará un ambiente con los recursos necesarios para el entrenamiento de los modelos y su posterior comparación en base a tareas de reconocimiento de productos; además, se recopilará la información necesaria para el desarrollo de la herramienta tecnológica. En la etapa de desarrollo de la herramienta se empleará un marco de trabajo ágil que permita resolver problemas complejos, facilitar la entrega del software, gestionar los cambios y aumentar la productividad (Vijayasarathy & Butler, 2016). La metodología ágil a utilizar es *Scrum*.

3.1 Etapa 1: Configuración del modelo

Esta etapa comprende las siguientes actividades:

1. Implementación de los modelos para la identificación de imágenes.
    a. Instalación del marco de trabajo.
    b. Selección del conjunto de datos.
    c. Proceso de entrenamiento.
    d. Prueba de los modelos.
2. Análisis comparativo de los modelos para la ubicación e identificación de imágenes.
3. Selección y justificación del modelo a implementar.

En la Tabla 3.1 se presenta un resumen de las actividades realizadas en esta etapa.

Tabla 3.1
Actividades de configuración del modelo

Tarea	Entrada	Resultado
Análisis comparativo de los modelos para la ubicación e identificación de imágenes.	Ejecución de pruebas de los modelos entrenados con el conjunto de datos previamente establecido.	Registro de resultados de las pruebas realizadas con los dos modelos entrenados.
Selección y justificación del modelo a implementar.	Tabla comparativa en base a los resultados de las pruebas realizadas a los dos modelos entrenados.	Modelo entrenado a implementarse en la herramienta tecnológica.

Fuente: Elaboración Propia

3.2 Etapa 2: Desarrollo de la herramienta tecnológica

La herramienta por desarrollar está conformada por tres componentes: (a) un smartphone, (b) unas gafas soporte de cabeza para smartphone y (c) auriculares inalámbricos. En la Figura 3.1 se muestran estos componentes en un diagrama que corresponde a la arquitectura de la herramienta.

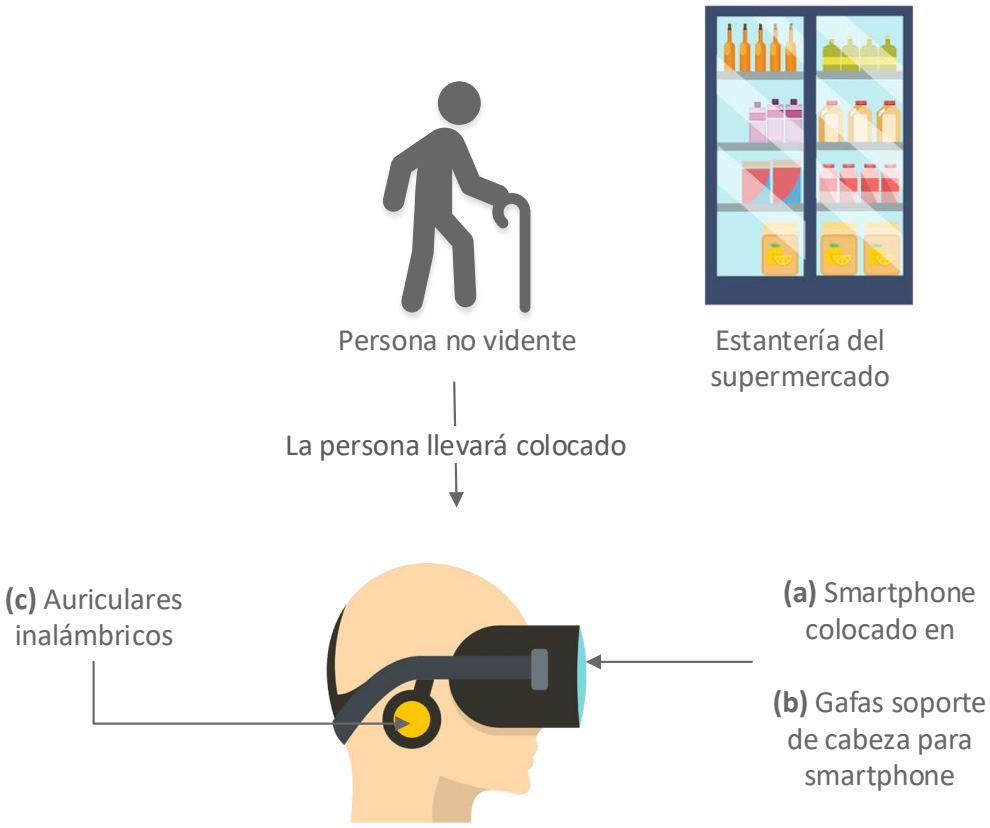


Figura 3.1 Diagrama de funcionamiento de la herramienta tecnológica
Fuente: Elaboración propia

La interacción de estos componentes permite cumplir con el objetivo de ubicar e identificar el producto que desea la persona no vidente de la siguiente manera:

1. La persona no vidente se coloca las gafas y auriculares inalámbricos.
2. Mediante un comando de voz activa el asistente virtual.
3. La persona no vidente le indica al asistente que producto desea buscar.

4. El asistente procesa la solicitud, valida si hay stock del producto e indica su ubicación.
5. La herramienta guía a la persona no vidente hacia la ubicación del producto mediante la identificación de señales (flechas) ubicadas en los pasillos del supermercado.
6. La persona no vidente toma el producto y solicita al asistente que valide si es el producto deseado.

Para el desarrollo del asistente que se instalará en el smartphone y que permitirá emplear el modelo de clasificación de imágenes previamente entrenado, se aplica la metodología ágil *Scrum* que según Lei, Ganjeizadeh, Jayachandran, & Ozcan (2017), es una de las metodologías más adoptadas por los equipos de trabajo para proyectos de desarrollo de software en la actualidad. Esta metodología posee artefactos y herramientas, cada uno con su propósito específico y que son de vital importancia para la consecución del desarrollo de la herramienta tecnológica.

Para desarrollar la herramienta se emplearán *Sprints*, los cuales se describen en la Tabla 3.2.

Tabla 3.2
Sprints planificados para el desarrollo de la herramienta

# Sprint	**Duración**	**Resultado**
1	2 semanas	Desarrollo del módulo de administración de comandos con conexión a base de datos.
2	1 semana	Desarrollo del asistente por voz.
3	5 semanas	Desarrollo y configuración del módulo de identificación de productos y navegación asistiva

Fuente: Elaboración Propia

Scrum también posee etapas en las cuales se planifican y revisan las actividades de los *sprints* mediante: reuniones de planificación, reuniones diarias, revisiones del sprint y retrospectiva del *sprint*.

Las reuniones de planificación de *sprints* consisten en una reunión en la que se definen las tareas que se llevarán a cabo en cada *sprint*. Para la planificación de cada tarea se realiza una estimación del esfuerzo en horas empleadas en la misma. El resultado de esta reunión se presenta en una tabla por planificación, en la que detallan las tareas con la estimación y el número total de horas que se han planificado por *sprint*.

En las reuniones diarias se revisan las actividades del día y las posibles dificultades que se pueden suscitar durante su ejecución y que pudiesen afectar su cumplimiento. A partir de estas reuniones se puede evitar estas dificultades o brindar asistencia especializada para mitigarlas. Para el presente trabajo de investigación, una vez concluida cada reunión se procederá a realizar cada tarea asignada en el día, conforme a la lista de pendientes de cada *sprint*.

En la etapa de revisión del sprint se revisa lo que se ha realizado durante un *sprint*, se detallan las tareas que se han concluido y las tareas pendientes. El resultado de esta revisión es una lista de tareas concluidas y pendientes.

En la etapa de retrospectiva del sprint se valora el *sprint* para determinar su éxito o fracaso y una estrategia de mejora. También se identifican los elementos eficientes y se realiza una planificación para implementar mejoras. Terminada esta etapa se libera un incremento del producto y se empieza un nuevo sprint. Al concluir con los *sprints* planificados, se evalúa el producto para su posterior aprobación y finalización del desarrollo.

3.3 Resumen de la metodología

En la metodología presentada se va a emplear la metodología ágil Scrum, para el desarrollo de una herramienta asistiva que permita la ubicación e identificación de productos con la finalidad de contribuir a mejorar la

autonomía e inclusión de las personas con discapacidad visual. A continuación, en la Figura 3.2 se muestra el flujo del proceso.

Figura 3.2 Resumen de la metodología
Fuente: Elaboración propia

CAPÍTULO 4 DESARROLLO, IMPLEMENTACIÓN Y RESULTADOS

En este capítulo se describe el desarrollo, implementación y resultados de las pruebas realizadas a los modelos de clasificación de imágenes y la herramienta asistiva. De acuerdo con lo establecido en el tercer capítulo de este trabajo de titulación, las definiciones incluidas en el marco teórico conforman la revisión sistemática para la configuración del modelo a implementar.

La herramienta tecnológica por desarrollar está conformada por módulos, como se puede apreciar en la Figura 4.1. El módulo del asistente virtual (a) permite la interacción por voz entre la persona no vidente y la herramienta. El módulo de navegación asistiva (b) brinda a la persona no vidente autonomía para que pueda orientarse y desplazarse hasta la ubicación del producto requerido. El módulo de identificación de productos (c) reconoce el producto escogido por la persona vidente. Dichos módulos interactúan entre sí y se ejecutarán en el smartphone.

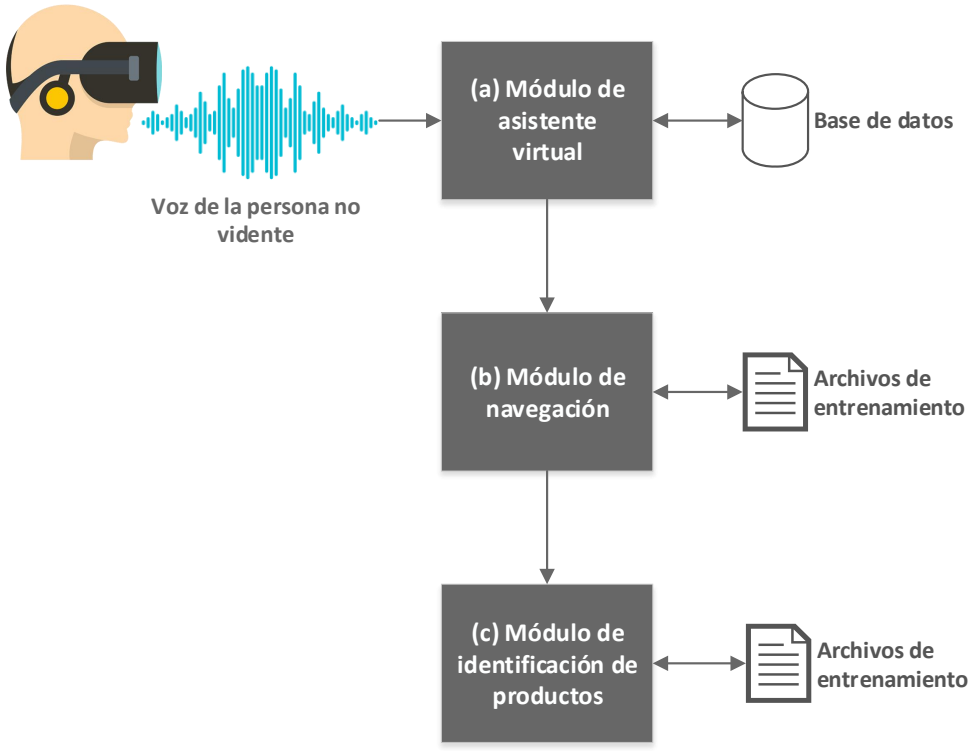


Figura 4.1 Diagrama de módulos de la herramienta tecnológica.
Fuente: Elaboración propia

El módulo del asistente virtual se activará por medio del botón del auricular. Posteriormente, el asistente consultará a la persona no vidente que producto desea buscar. La librería *SpeechRecognizer* procesará el comando de voz indicado y se validará si está registrado en la base de datos. A continuación, la librería *TextToSpecch* convierte el resultado en voz; para que finalmente el asistente le indique a la persona no vidente el texto por audio. Esta secuencia se muestra en la Figura 4.2.

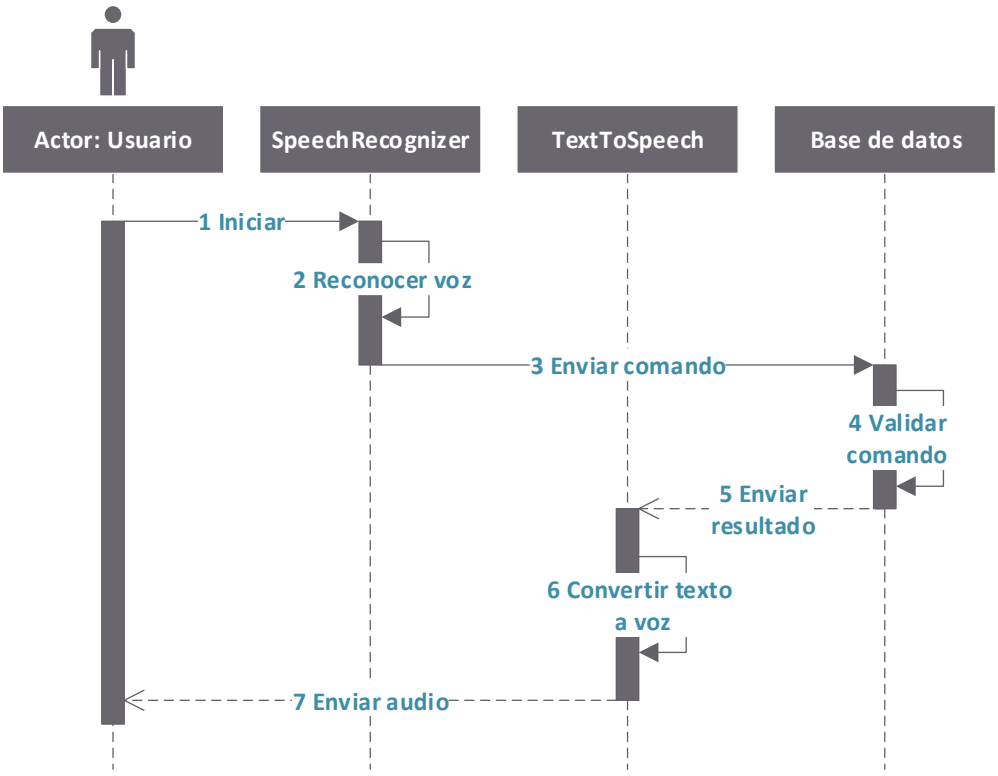


Figura 4.2 Diagrama de secuencia para el módulo del Asistente virtual
Fuente: Elaboración propia

El módulo de navegación asistiva (b) y el módulo de identificación de productos (c) emplean archivos generados por el proceso de entramiento de modelos para clasificación de imágenes. Sin embargo, usan conjunto de datos diferentes para dicho entrenamiento, es decir, para el primer módulo mencionado se utilizan imágenes de señales (fechas) y para el segundo módulo se utilizan fotos de productos.

El módulo de navegación asistiva, integra el funcionamiento del asistente virtual para la interacción por voz con la persona no vidente. Dicho módulo se activa por medio del comando "*Encontrar*". Luego, la persona no vidente indica el producto deseado y se valida si existe en tienda. A continuación, se ejecutará la librería *ImageClassifier* para activar la cámara del smartphone y el proceso de detección de señales, el cual permite la orientación y navegación de la persona no vidente. Una vez que se detecta la señal (flecha) correspondiente al producto, se indica su ubicación. Esta secuencia se muestra en la Figura 4.3.

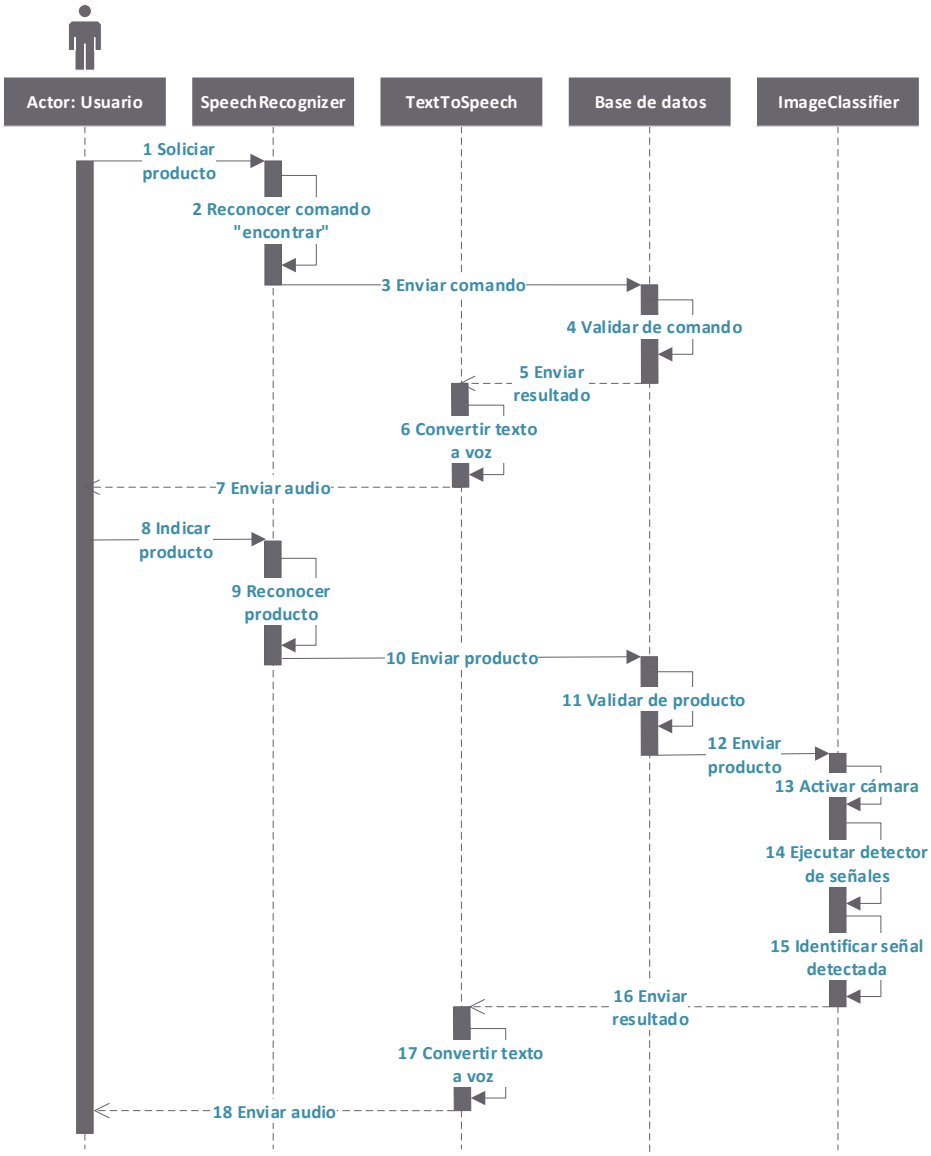


Figura 4.3 Diagrama de secuencia para el módulo de Navegación asistiva.
Fuente: Elaboración propia

El módulo de Identificación de productos, integra el funcionamiento del asistente virtual para la interacción por voz con la persona no vidente. Dicho módulo se activa por medio del comando *"Validar"*. Luego, la persona no vidente indica el producto deseado y se valida si existe en tienda. A continuación, se ejecutará la librería *ImageClassifier* para activar la cámara del smartphone y el proceso de identificación de productos, el cual permite la detección del producto escogido por la persona no vidente. Una vez detectado el producto, el asistente indicará por voz que es el producto deseado. Esta secuencia se muestra en la Figura 4.4.

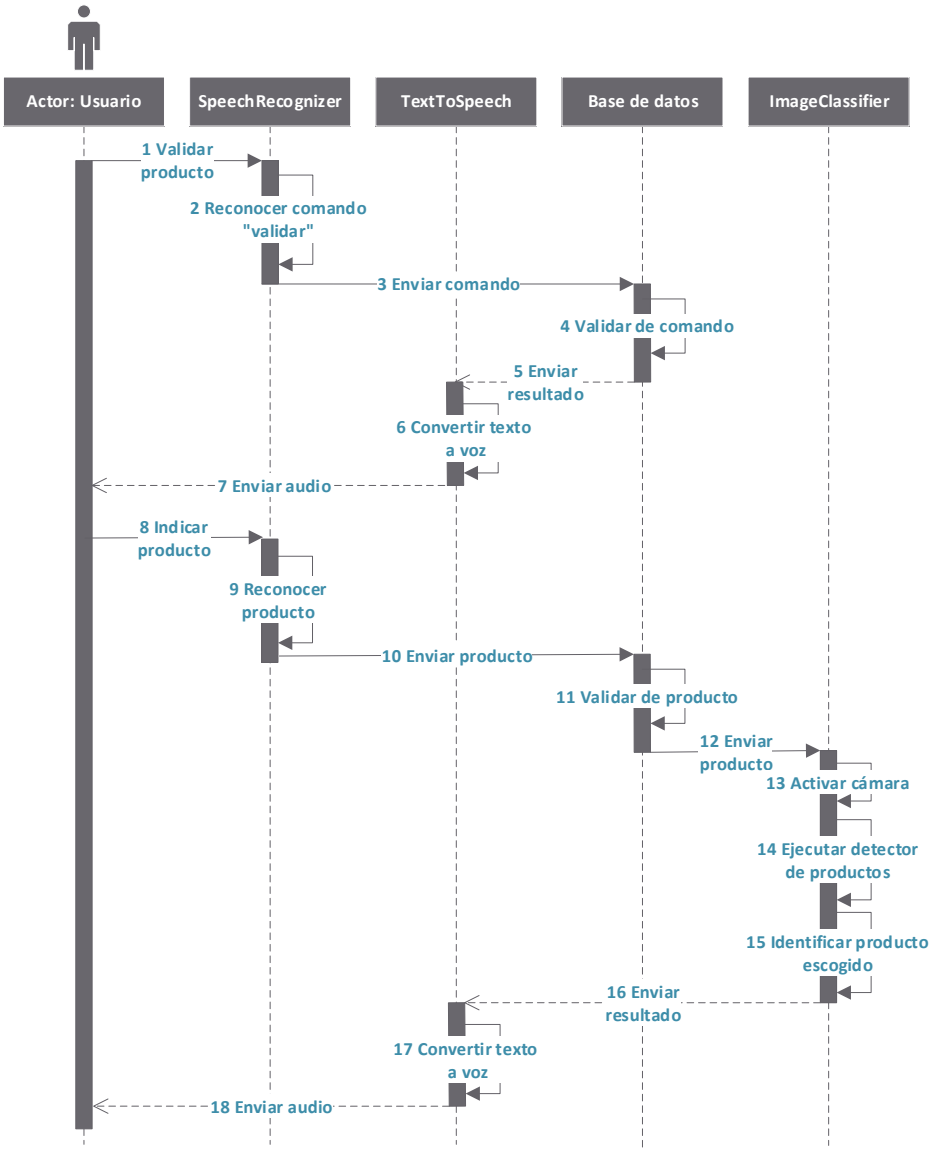


Figura 4.4 Diagrama de secuencia para el módulo de Identificación de productos. Fuente: Elaboración propia

A continuación, se detallan las actividades para el desarrollo de la etapa de configuración del modelo, donde se levantará un ambiente con los recursos necesarios para el entrenamiento, configuración y comparación de los modelos *InceptionV3* y *MobileNet*.

4.1 Configuración del modelo

Esta etapa comprende las siguientes actividades:

1. Implementación de los modelos para la identificación de imágenes.
    a. Instalación del marco de trabajo.
    b. Selección del conjunto de datos.
    c. Proceso de entrenamiento.
    d. Prueba de los modelos.
2. Análisis comparativo entre los modelos para la ubicación e identificación de imágenes.
3. Selección y justificación del modelo a implementar.

4.1.1 Implementación de los modelos para la identificación de imágenes

Para realizar las pruebas de identificación de imágenes para la detección y ubicación de productos se emplearon los modelos *InceptionV3* y *MobileNet* previamente descritos en el Capítulo 2 del presente trabajo de investigación. Ambos modelos consideraron un flujo de trabajo como el que se muestra en la Figura 4.5, en el que se pueden apreciar cuatro etapas: instalación del marco de trabajo, selección del conjunto de datos, proceso de entrenamiento y prueba de los modelos.

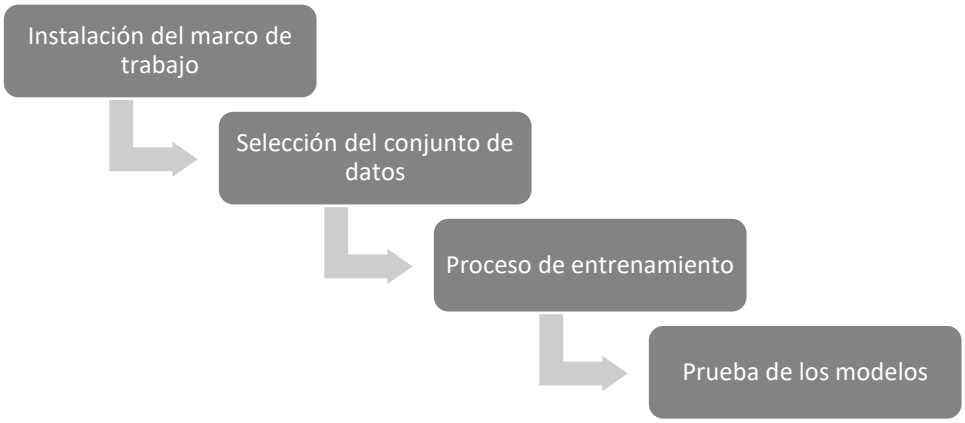


Figura 4.5 Flujo de trabajo para los modelos InceptionV3 y MobileNet
Fuente: Elaboración propia

Instalación del marco de trabajo

Para la instalación del marco de trabajo *Tensorflow* y el desarrollo del software se empleará un equipo de cómputo con las características mostradas en la Tabla 4.1.

Tabla 4.1
Características del equipo empleado

Características	**Equipo**
Procesador	*Intel i7* 4 núcleos 2.8GHz
Memoria	8GB
Disco Duro	1TB
GPU	*GeForce GTX* 1050

Fuente: Elaboración Propia

En la implementación de *Tensorflow* es necesario contar con *Python* instalado en el equipo, para lo cual se descargó el instalador desde su página oficial. Para el presente trabajo se utilizó la versión 3.7.5. Posterior a su instalación se puede ejecutar *Python* desde la ventana de comandos de Windows y acceder a su interfaz para ejecutar scripts e instrucciones en este lenguaje como se aprecia en la Figura 4.6.

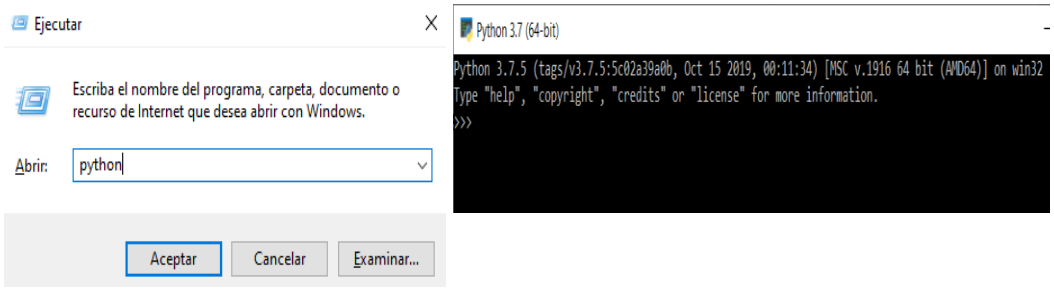


Figura 4.6 Interfaz de línea de comandos con Python
Fuente: Elaboración propia

Luego, se realizó la instalación de *Tensorflow* desde la interfaz de línea de comandos de *Python y* se implementó la versión 1.15.0.

Selección del conjunto de datos

Un aspecto clave para el desarrollo de estas pruebas es el conjunto de datos con el que se propone entrenar los modelos para identificación de imágenes, para su posterior comparación e implementación en la herramienta tecnológica. El conjunto de datos empleado para el módulo de identificación de productos está compuesto de 739 fotos de productos de primera necesidad que se pueden encontrar comúnmente en un supermercado. Se clasificaron estos productos por categorías como: víveres, condimentos, bebidas, lácteos, enlatados y productos de aseo personal. El detalle de esta clasificación se muestra en la Tabla 4.2. Adicional, para cada tipo de producto se escogieron entre dos y tres marcas.

Tabla 4.2
Clasificación de productos de primera necesidad

Clase	Producto	Marcas
Víveres	Azúcar	San Carlos, La Troncal
Condimentos	Aceite	Oro, Girasol, La Favorita
	Salsa de tomate	Los Andes, Facundo
Bebidas	Gaseosa	Coca Cola, Manzana, Pepsi

Lácteos	Leche	Nutri leche, La lechera
Enlatados	Atún	Campos, Real, Van Camp's
Aseo Personal	Jabón	Protex, Jolly, Lux
	Pasta dental	Colgate, Fortident

Fuente: Elaboración Propia

Para cada producto se tomaron entre 30 a 40 fotos utilizando un smartphone. Se consideró el ángulo frontal de cada producto acorde a su ubicación en la percha, como se puede apreciar en la Figura 4.7.

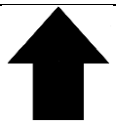
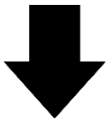
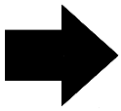
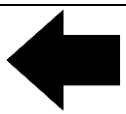
Producto en percha	**Producto en mano**	**Producto en mano (con acercamiento)**
		
		

Figura 4.7 Ejemplo de fotos de productos empleadas como conjunto de datos
Fuente: Elaboración propia.

Para el módulo de navegación asistiva el conjunto de datos está compuesto de 120 fotos de señales, las cuales indican la ruta a seguir y la ubicación en percha de los productos. El detalle de esta clasificación se muestra en la Tabla 4.3.

Tabla 4.3
Clasificación de las señales

Productos	Señales	Indicaciones
Azúcar		**Gris:** Azúcar ubicado en la percha de la izquierda.
Leche		**Rosado:** Leche ubicada en la percha de la derecha.
Aceite		**Amarillo:** Aceite ubicado en la percha de la izquierda.
Sala de tomate		**Naranja:** Salsa de tomate ubicada en la percha de la derecha.
Gaseosa		**Azul:** Gaseosa ubicada en la percha de la izquierda.
Atún		**Café:** Atún ubicado en la percha de la derecha.
Jabón		**Morado:** Jabón ubicado en la percha de la izquierda.
Pasta dental		**Verde:** Pasta dental ubicada en la percha de la derecha.

Avanzar		Indica que la persona no vidente avance.
Retroceder		Indica que la persona no vidente retroceda.
Derecha		Indica que la persona no vidente gire a la derecha.
Izquierda		Indica que la persona no vidente gire a la izquierda.

Fuente: Elaboración Propia

Para cada señal se tomaron entre 15 a 20 fotos utilizando un smartphone, como se puede apreciar en la Figura 4.8.



Figura 4.8 Ejemplo de fotos de señales empleadas como conjunto de datos
Fuente: Elaboración propia.

Las fotos tomadas para ambos conjuntos de datos cuentan con las características que se listan en la Tabla 4.4.

Tabla 4.4
Características de las imágenes tomadas

Características	**Imagen**
Resolución	3120 x 4160
Extensión	JPG
Color	Todas a color
Tamaño	3.5 MB ~ 4 MB

Fuente: Elaboración Propia

Para el entrenamiento de clasificación de imágenes se emplea el aprendizaje supervisado, el cual recibe los valores de salida deseados. Es por esta razón que se organiza el conjunto de datos en subcarpetas, como se muestra de referencia en la Figura 4.9. La misma organización se utiliza para el entrenamiento de identificación de productos y de señales.

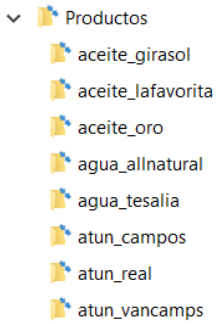


Figura 4.9 Estructura de las imágenes para el entrenamiento de modelos.
Fuente: Elaboración propia.

Proceso de entrenamiento

Para el proceso de entrenamiento se utilizó la interfaz de línea de comandos de Windows para realizar el entrenamiento de modelos del marco de trabajo de *Tensorflow,* donde se indicaron los siguientes parámetros:

- *Image_dir*: Ruta de las fotos de los productos o señales a clasificar.
- *How_many_training_steps*: Cantidad de pasos de entrenamiento que ejecutará el modelo.
- *Model_dir*: Ruta del script del modelo a utilizar, por ejemplo, *InceptionV3*.
- *Output_graph*: Ruta para generar el archivo pb. El archivo pb contiene la definición del gráfico y los pesos del modelo.
- *Output_labels*: Ruta para generar el archivo de categorías de productos o señales.

Para ejecutar esta configuración se ejecuta el script *retrain.py* el cual se obtiene del repositorio *Tensorflow Hub* que permite resolver nuevas tareas con menos tiempo y datos de entrenamiento (Google LLC, 2020). Adicional, se especifica el modelo que se utilizará para el entrenamiento, como se puede apreciar en la Figura 4.10. Durante este proceso de configuración solo se empleó el CPU del equipo de cómputo.

```
python retrain.py
--image_dir=D:\TESIS\Image_classification\tensorflow\tensorflow\examples\image_retraining\Productos
--how_many_training_steps=500
--architecture=inception_v3
--model_dir=D:\TESIS\Image_classification\tensorflow\tensorflow\examples\image_retraining\inception
--output_graph=D:\TESIS\Image_classification\tensorflow\tensorflow\examples\image_retraining\inceptionv3.pb
--output_labels=D:\TESIS\Image_classification\tensorflow\tensorflow\examples\image_retraining\inceptionv3_label.txt
--summaries_dir=D:\TESIS\Image_classification\tensorflow\tensorflow\examples\image_retraining\logs_InceptionV3
```

Figura 4.10 Línea de comando para entrenar modelo de clasificación de imágenes. Fuente: Elaboración propia.

Para ejecutar el entrenamiento del modelo *InceptionV3* se utilizó el conjunto de datos definidos en la sección anterior y en conjunto con la herramienta *TensorBoard* se pudo representar el grafo del modelo *InceptionV3* empleado para estas pruebas como se muestra en la Figura 4.11.

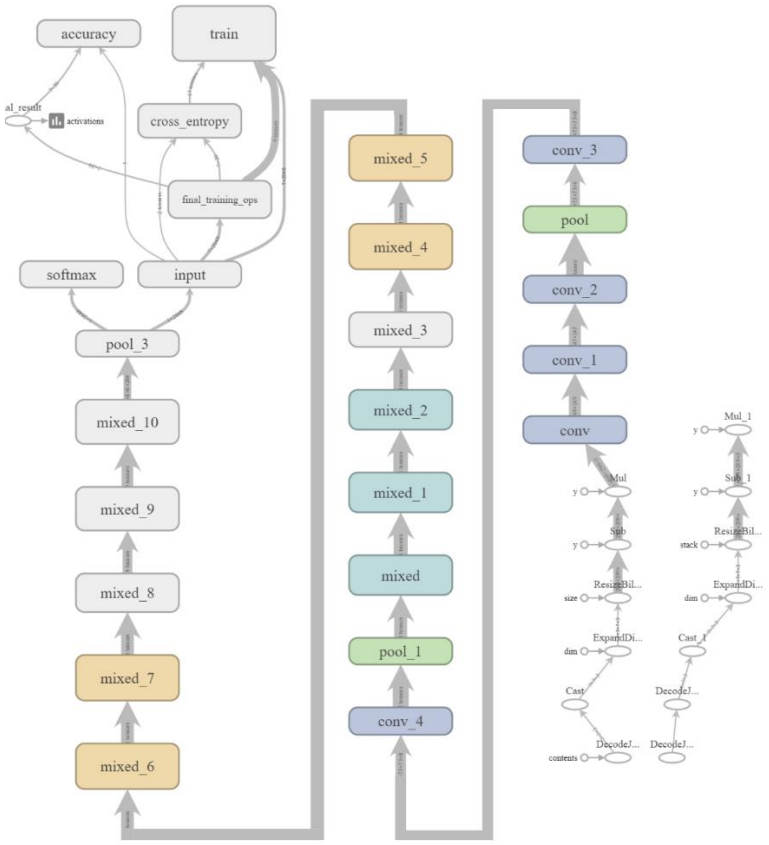


Figura 4.11 Gráfico del modelo InceptionV3.
Fuente: Elaboración propia empleando la herramienta TensorBoard.

En la Figura 4.12 se puede apreciar el resultado de la ejecución del script *retrain.py* con el modelo *InceptionV3* configurado para el entrenamiento de imágenes.

```
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202404.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202405.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202406.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202407.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202414.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202416.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202418.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202420.jpg_inception_v3.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\gaseosa_pepsi\IMG_20200105_202424.jpg_inception_v3.txt
```

```
INFO:tensorflow:2020-03-15 16:32:37.376199: Step 490: Train accuracy = 100.0%
INFO:tensorflow:2020-03-15 16:32:37.376199: Step 490: Cross entropy = 0.159468
INFO:tensorflow:2020-03-15 16:32:37.481191: Step 490: Validation accuracy = 100.0% (N=100)
INFO:tensorflow:2020-03-15 16:32:38.407197: Step 499: Train accuracy = 100.0%
INFO:tensorflow:2020-03-15 16:32:38.408193: Step 499: Cross entropy = 0.153726
INFO:tensorflow:2020-03-15 16:32:38.507226: Step 499: Validation accuracy = 100.0% (N=100)
INFO:tensorflow:Final test accuracy = 100.0% (N=122)
INFO:tensorflow:Froze 2 variables.
Converted 2 variables to const ops.
```

Figura 4.12 Ejecución del proceso de entrenamiento para el modelo InceptionV3
Fuente: Elaboración propia.

Para ejecutar el entrenamiento del modelo MobileNet se utilizó el mismo conjunto de datos para el modelo *InceptionV3*. En conjunto con la herramienta *TensorBoard* se pudo representar el grafo del modelo *MobileNet* empleado para estas pruebas como se muestra en la Figura 4.13.

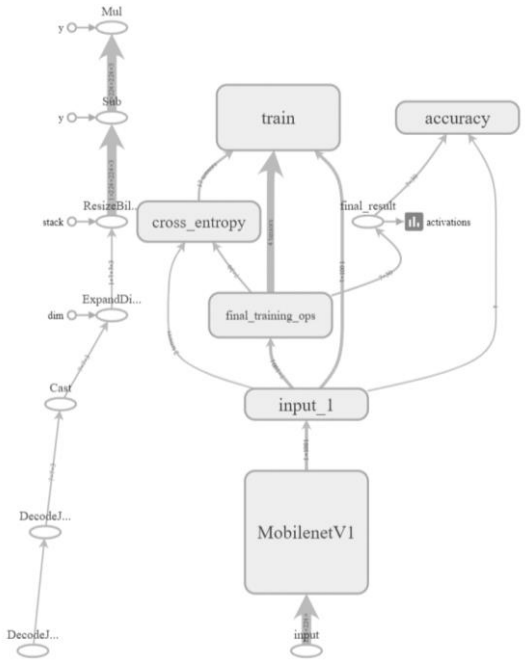


Figura 4.13 Gráfico del modelo MobileNet.
Fuente: Elaboración propia empleando la herramienta TensorBoard.

En la Figura 4.14 se puede apreciar el resultado de la ejecución del script *retrain.py* con el modelo *MobileNet* configurado para el entrenamiento de imágenes.

```
INFO:tensorflow:100 bottleneck files created.
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_lafavorita\IMG_20200105_201706.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_lafavorita\IMG_20200105_201705.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_lafavorita\IMG_20200105_201734.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_lafavorita\IMG_20200105_201743.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_lafavorita\IMG_20200105_201753.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_lafavorita\IMG_20200105_201755.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_lafavorita\IMG_20200105_201800.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_oro\IMG_20200105_201618.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_oro\IMG_20200105_201619.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_oro\IMG_20200105_201628.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_oro\IMG_20200105_201631.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_oro\IMG_20200105_201634.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_oro\IMG_20200105_201635.jpg_mobilenet_1.0_224.txt
INFO:tensorflow:Creating bottleneck at /tmp/bottleneck\aceite_oro\IMG_20200105_201637.jpg_mobilenet_1.0_224.txt
```

```
INFO:tensorflow:2020-03-15 17:34:24.964349: Step 470: Validation accuracy = 100.0% (N=100)
INFO:tensorflow:2020-03-15 17:34:25.505376: Step 480: Train accuracy = 100.0%
INFO:tensorflow:2020-03-15 17:34:25.506350: Step 480: Cross entropy = 0.003534
INFO:tensorflow:2020-03-15 17:34:25.565376: Step 480: Validation accuracy = 100.0% (N=100)
INFO:tensorflow:2020-03-15 17:34:26.115349: Step 490: Train accuracy = 100.0%
INFO:tensorflow:2020-03-15 17:34:26.115349: Step 490: Cross entropy = 0.003679
INFO:tensorflow:2020-03-15 17:34:26.176383: Step 490: Validation accuracy = 100.0% (N=100)
INFO:tensorflow:2020-03-15 17:34:26.662376: Step 499: Train accuracy = 100.0%
INFO:tensorflow:2020-03-15 17:34:26.663349: Step 499: Cross entropy = 0.003079
INFO:tensorflow:2020-03-15 17:34:26.722349: Step 499: Validation accuracy = 100.0% (N=100)
INFO:tensorflow:Final test accuracy = 100.0% (N=122)
INFO:tensorflow:Froze 2 variables.
Converted 2 variables to const ops.
```

Figura 4.14 Ejecución del proceso de entrenamiento para el modelo MobileNet
Fuente: Elaboración propia.

Cada modelo configurado con *Tensorflow* genera dos archivos con las características detalladas en la Tabla 4.5 y que se deben copiar dentro del proyecto para ejecutar la identificación de productos mediante la herramienta tecnológica.

Tabla 4.5
Características de los archivos generados a partir del entrañamiento de los modelos de clasificación de imágenes

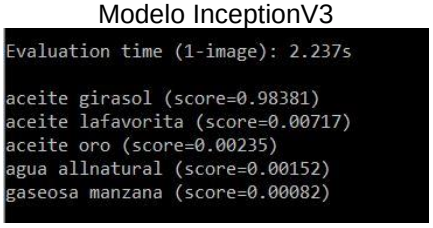
Archivo generado por modelo	Extensión	Tamaño
InceptionV3	Archivo. pb	85MB
	Archivo .txt	1KB
MobileNet	Archivo. pb	17MB
	Archivo .txt	1KB

Fuente: Elaboración Propia

Pruebas del modelo

Para las pruebas de clasificación de imágenes se empleó *Python* con las librerías de *Tensorflow* para ambos modelos. Adicional, se utilizó una imagen del producto aceite girasol para comprobar el tiempo de detección y precisión, como se muestra en la Figura 4.15.

Modelo InceptionV3



Modelo MobileNet

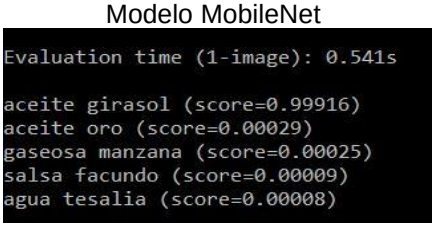


Figura 4.15 Resultado de las pruebas de los modelos para la detección del producto aceite girasol.
Fuente: Elaboración propia.

Como se observa en la Figura 4.15, de las pruebas realizadas se obtuvo que el modelo *MobileNet* tomó 1.70 segundos menos en la detección del producto aceite girasol que el modelo InceptionV3. Adicional, el modelo que tuvo mayor porcentaje de precisión fue el modelo *MobileNet*.

4.1.2 Análisis comparativo de los modelos para la ubicación e identificación de imágenes

Por medio de los archivos generados por cada uno de los modelos en la etapa de configuración y ejecutando una versión beta del módulo de identificación de productos de la herramienta tecnológica se realizaron las respectivas pruebas para comparar los modelos. Dicho análisis evaluó los siguientes parámetros:

Velocidad de entrenamiento

Se consideró 500 pasos para medir el tiempo de entrenamiento de los modelos en el ambiente de trabajo, como se muestra en la Figura 4.16.

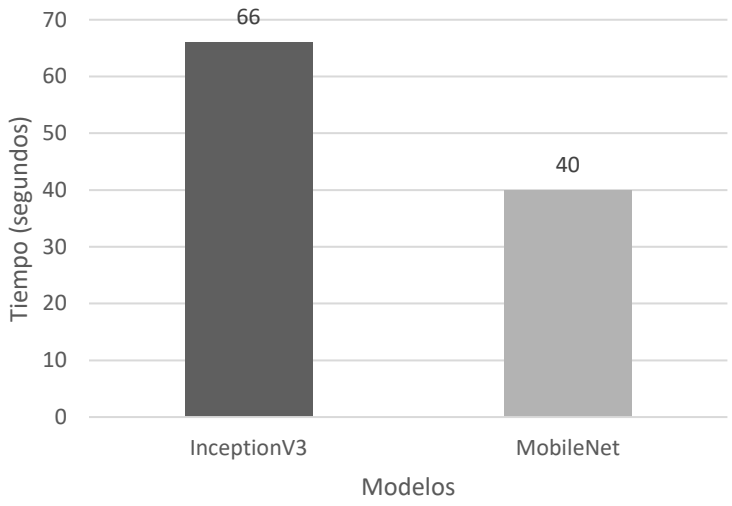


Figura 4.16 Tiempo de entrenamiento por modelo para la clasificación de imágenes. Fuente: Elaboración propia.

En la Figura 4.16 se puede apreciar que el modelo *InceptionV3* demoró 26 segundos más en entrenarse que el modelo *MobileNet*. Por tanto, se corroboró que MobileNet es más rápido.

Precisión

Para determinar el porcentaje de precisión de cada modelo se capturaron imágenes de artículos de primera necesidad y se utilizó el módulo de identificación de productos desarrollado. Este módulo toma una foto del producto y valida el mismo, como se aprecia en la figura 4.17; donde se muestra como ejemplo la ejecución del módulo con el modelo *InceptionV3*.

Figura 4.17 Módulo de identificación de productos de la herramienta tecnológica.
Fuente: Elaboración propia.

Para las pruebas de precisión se utilizó el producto aceite girasol de la categoría condimentos y se realizaron cinco pruebas de identificación del producto por cada modelo, como se muestra en la tabla 4.6.

Tabla 4.6
Pruebas de precisión de identificación de productos por cada modelo

	Porcentaje de precisión				
Modelo	**Prueba 1**	**Prueba 2**	**Prueba 3**	**Prueba 4**	**Prueba 5**
InceptionV3	76.86 %	84.28 %	85.44 %	88.24 %	77.28 %
MobileNet	85.72 %	89.61 %	91.54 %	88.35 %	92.03 %

Fuente: Elaboración Propia

Posterior a las pruebas, se promedió el valor de precisión registrado, cuyos resultados se presentan en la Figura 4.18.

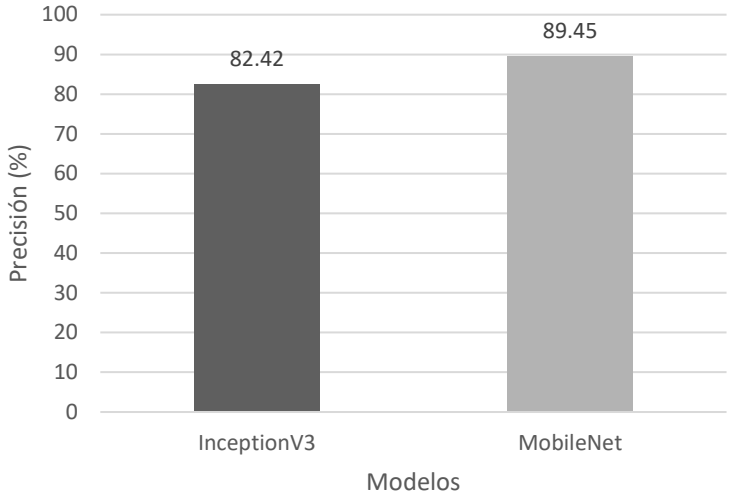


Figura 4.18 Medición de precisión de los modelos
Fuente: Elaboración propia.

Al capturar la imagen del producto aceite girasol se registró que el porcentaje promedio de precisión del modelo *MobileNet* fue mayor al modelo *InceptionV3*, como se puede apreciar en la Figura 4.18.

Velocidad de identificación de productos

Para las pruebas de velocidad se utilizó el producto aceite girasol y se realizaron cinco pruebas de identificación del producto por cada modelo, como se muestra en la tabla 4.7.

Tabla 4.7
Pruebas de velocidad de identificación de productos por cada modelo

	Velocidad de identificación de productos (Segundos)				
Modelo	**Prueba 1**	**Prueba 2**	**Prueba 3**	**Prueba 4**	**Prueba 5**
InceptionV3	7.97	11.50	8.22	7.16	7.20
MobileNet	7.25	8.41	7.08	7.45	7.22

Fuente: Elaboración Propia

Posterior a las pruebas, se promedió el valor de velocidad registrado y se evidenció que el modelo *MobileNet* es en promedio el más rápido para identificar productos. Los resultados se presentan en la Figura 4.19.

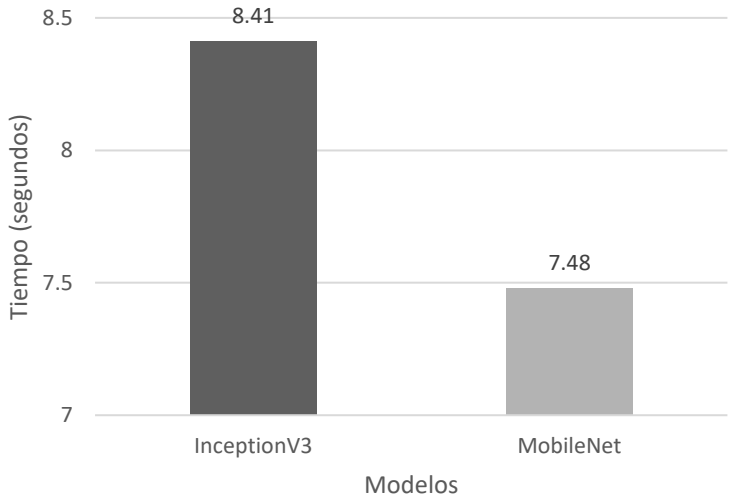


Figura 4.19 Tiempo registrado para identificar un producto
Fuente: Elaboración propia.

Memoria RAM utilizada por los modelos

El promedio de memoria RAM utilizado por cada modelo al ejecutar el módulo de identificación de productos se muestra en la tabla 4.8.

Tabla 4.8
Uso de memoria RAM por cada modelo en el módulo de identificación de productos

Modelo	Uso promedio de RAM
InceptionV3	131 MB
MobileNet	115MB

Fuente: Elaboración Propia

En la tabla 4.8 se puede apreciar que el modelo *MobileNet* consume en promedio menos memoria RAM del smartphone.

Espacio de almacenamiento por modelo

Al ejecutar el entrenamiento de modelos generan archivos que se utilizan como fuente de datos para el módulo de identificación de productos o señales. Dichos archivos generados se almacenan en el proyecto de software y requieren un espacio específico, como se muestra en la figura 4.20.

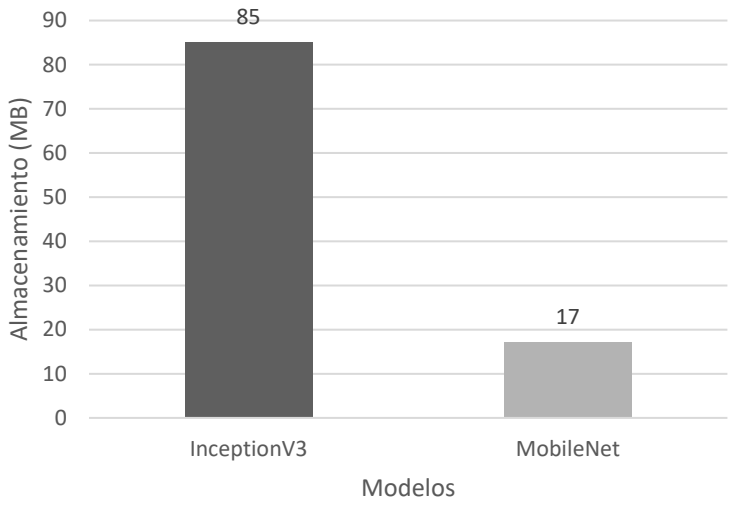


Figura 4.20 Espacio de almacenamiento requerido por los archivos generados por cada modelo
Fuente: Elaboración propia.

En la Figura 4.20 se evidencia que *MobileNet* es el modelo que requiere menos espacio de almacenamiento para los archivos generados posterior al entrenamiento de identificación de imágenes.

4.1.3 Selección y justificación de modelo para identificación de imágenes.

Considerando los resultados registrados en el apartado anterior se ha realizado una tabla comparativa (Tabla 4.9) en la que se detallan los resultados de las pruebas realizadas. Cabe recalcar que se consideró un escenario para la ejecución de una aplicación que permitiese probar los modelos de clasificación de imágenes de interés para el presente trabajo de investigación.

Tabla 4.9
Comparación de modelos de identificación de imágenes.

Modelo	Velocidad de entrenamiento	Precisión	Velocidad de identificación de productos	Memoria RAM utilizada	Espacio de almacenamiento por modelo
InceptionV3	66 seg	82.42%	8.41 seg	131MB	85MB
MobileNet	40 seg	89.45 %	7.48 seg	115MB	17MB

Fuente: Elaboración Propia

Como se puede apreciar, el modelo *MobileNet* ofrece mayor precisión y rapidez al identificar productos. Además, considerando la revisión de estos modelos en el apartado 2.2.2, el modelo *MobileNet* es idóneo para implementarse en desarrollos móviles, dicho punto se corroboró con los resultados de medir el consumo de memoria *RAM* del smartphone y el espacio de almacenamiento requerido por dicho modelo, los cuales son inferiores al otro modelo mencionado.

4.2 Desarrollo de la herramienta

En esta etapa se definió un diagrama correspondiente a la arquitectura de la herramienta tecnológica, tal como se muestra en la Figura 4.21. En esta imagen se ilustra que la comunicación entre el usuario no vidente y la herramienta tecnológica es por comandos de voz a través del asistente virtual. A su vez, la herramienta está conformada por tres capas. La primera capa la compone el controlador el cual procesa los comandos enviados por el asistente virtual. Luego, la capa de base de datos realiza las consultas y extrae la información que será enviada nuevamente al controlador para que al mismo tiempo se envíe la respuesta a la capa de vista y posteriormente el asistente virtual convierta la respuesta de datos a repuesta por voz y así el usuario no vidente pueda escuchar su requerimiento.

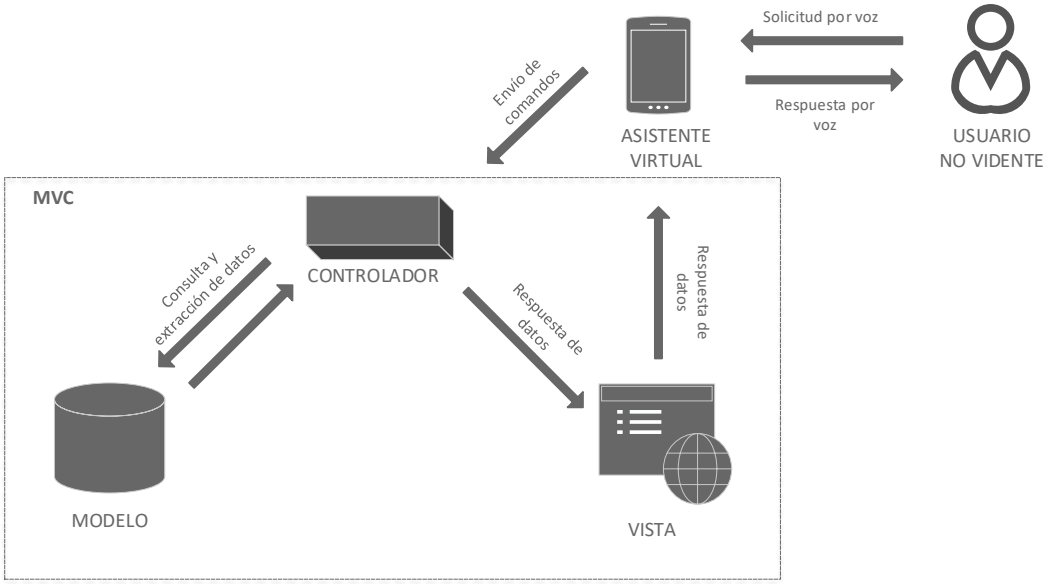


Figura 4.21 Diagrama de arquitectura de la herramienta tecnológica
Fuente: Elaboración propia

4.2.1 Sprint 1

4.2.1.1 Planificación del Sprint 1

Como primer paso se definieron las historias de usuario para posteriormente especificar las tareas que conformarán el *sprint*. Dichas tareas son evaluadas de acuerdo con el esfuerzo en horas estimado a requerirse para su culminación. En la tabla 4.10 se muestra el detalle de estas tareas.

Tabla 4.10
Lista de tareas del sprint 1

# Tarea	Lista de tareas sprint 1	Esfuerzo
Tarea 1	Diseñar arquitectura de base de datos	3h
Tarea 2	Crear tablas para registrar los comandos de voz	2h
Tarea 3	Desarrollar interfaz gráfica para administrar comandos de voz	4h
Tarea 4	Ingresar comando de voz	3h
Tarea 5	Actualizar comando de voz	2h
Tarea 6	Eliminar comando de voz	2h
	Total	16h

Fuente: Elaboración Propia

Con base a estas tareas se realizó una planificación del trabajo en horas necesarias para cumplirlas, como se muestra en la tabla 4.11. Esta planificación del *sprint* 1 busca entregar una versión del producto que contenga la interfaz gráfica para administrar los comandos de voz que utilizará el asistente virtual para interactuar con la persona no vidente.

Tabla 4.11
Planificación de tareas para el desarrollo del Sprint 1

	Tarea 1	**Tarea 2**	**Tarea 3**	**Tarea 4**	**Tarea 5**	**Tarea 6**	**Total**
Día 1	1h	-	-	-	-	-	1h
Día 2	2h	-	-	-	-	-	2h
Día 3	-	2h	-	-	-	-	2h
Día 4	-	-	2h	-	-	-	2h
Día 5	-	-	2h	-	-	-	2h
Día 6	-	-	-	2h	-	-	2h
Día 7	-	-	-	1h	-	-	1h
Día 8	-	-	-	-	2h	-	2h
Día 9	-	-	-	-	-	1h	1h
Día 10	-	-	-	-	-	1h	1h
Total	3h	2h	4h	3h	2h	2h	16h

Fuente: Elaboración Propia

4.2.1.2 Desarrollo del Sprint 1

Para el desarrollo del Sprint 1, se completarán las actividades indicadas en la tabla 4.10. A continuación, se detalla el diagrama de casos de uso que realizará el usuario administrador de la herramienta tecnológica, como se muestra en la Figura 4.22

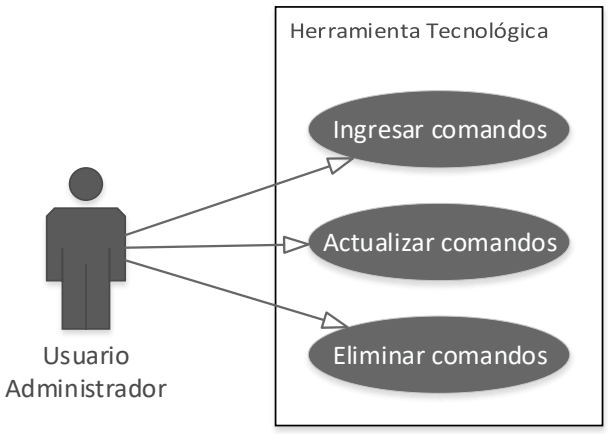


Figura 4.22 Diagrama de casos de uso por el usuario administrador
Fuente: Elaboración propia

1. Diseñar arquitectura de base de datos

El diseño de la arquitectura de base de datos consistió en definir las tablas y sus relaciones entre sí. Luego, de definir las tablas necesarias para el funcionamiento de la herramienta se procede a crearlas para el registro de las instrucciones para los comandos de voz. Este diseño se muestra en la Figura 4.23.

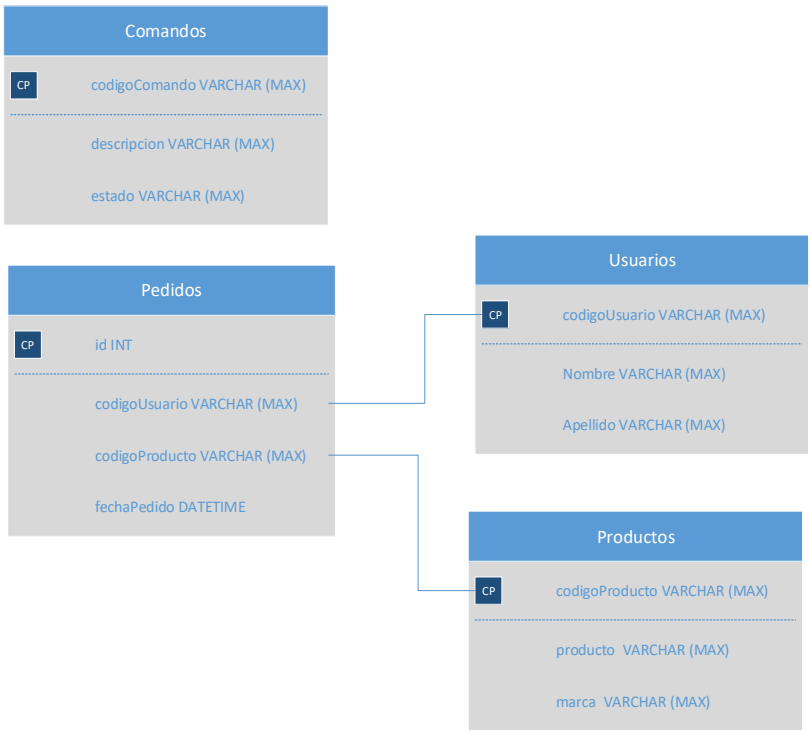


Figura 4.23 Diseño de la arquitectura de base datos
Fuente: Elaboración propia

2. Crear tablas para registrar los comandos de voz

Posterior al diseño de la arquitectura de base de datos, se crean las tablas en la base de datos SQLite. Las tablas se describen a continuación:

- Comandos: La tabla contiene los comandos de voz que sirven para la interacción entre la persona no vidente y el asistente virtual.
- Pedidos: La tabla contiene el detalle de los productos solicitados por la persona no vidente.
- Usuarios: La tabla contiene la lista de personas no videntes que harán uso de la herramienta tecnológica.
- Productos: La tabla contiene la lista de productos disponibles en percha.

3. Desarrollar interfaz gráfica para administrar comandos de voz

La interfaz se desarrolló utilizando la IDE de *Android Studio* con lenguaje *JAVA*. La misma que contiene tres opciones como se muestra en la Figura 4.24.

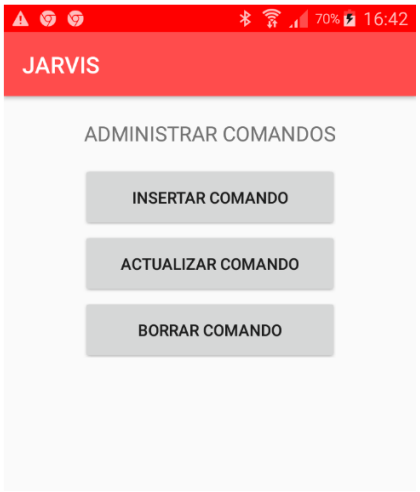


Figura 4.24 Interfaz gráfica para administrar los comandos de voz
Fuente: Elaboración propia

4. Ingresar comando de voz

La opción de ingresar permite al administrador de la herramienta tecnológica configurar múltiples comandos de voz que utilizará el asisten virtual para interactuar con la persona no vidente, como se muestra en la Figura 4.25.

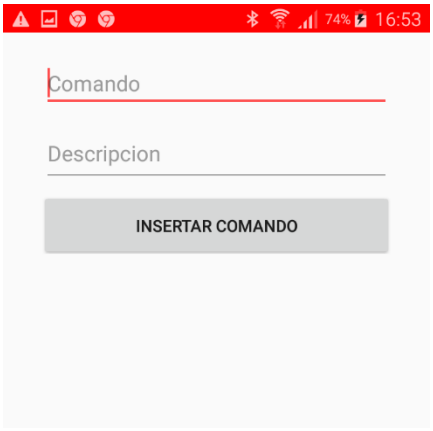


Figura 4.25 Interfaz gráfica para ingresar comandos de voz
Fuente: Elaboración propia

5. Actualizar comando de voz

La opción de actualizar permite al administrador de la herramienta tecnológica modificar los comandos de voz registrados en la base de datos, como se muestra en la Figura 4.26.

Figura 4.26 Interfaz gráfica para actualizar comandos de voz
Fuente: Elaboración propia.

6. Eliminar comando de voz

La opción de eliminar permite al administrador de la herramienta tecnológica borrar algún comando en específico de la base de datos, como se muestra en la Figura 4.27.

Figura 4.27 Interfaz gráfica para eliminar comandos de voz
Fuente: Elaboración propia.

4.2.1.3 Revisión del Sprint 1

En la revisión del Sprint 1, se corroboró que se hayan completado todas las actividades planificadas respecto a la interfaz gráfica para administrar los comandos de voz, como se aprecia en la Tabla 4.12.

Tabla 4.12
Lista de tareas completadas en el sprint 1

# Tarea	Lista de tareas sprint 1	Completado
Tarea 1	Diseñar arquitectura de base de datos	✓
Tarea 2	Crear tablas para registrar los comandos de voz	✓
Tarea 3	Desarrollar interfaz gráfica para administrar comandos de voz	✓
Tarea 4	Ingresar comando de voz	✓
Tarea 5	Actualizar comando de voz	✓
Tarea 6	Eliminar comando de voz	✓

Fuente: Elaboración Propia

A continuación, la Tabla 4.13 muestra en resumen las tareas completas y su tiempo de desarrollo.

Tabla 4.13
Tiempo de desarrollo para las tareas completadas en el Sprint 1

	Tareas completadas	Total
Día 1	Tarea 1	4h
Día 2		
Día 3	Tarea 2	2h
Día 4	Tarea 3	5h
Día 5		
Día 6	Tarea 4	4h
Día 7		
Día 8	Tarea 5	2h
Día 9	Tarea 6	3h
Día 10		
Total		**20h**

Fuente: Elaboración Propia

En la Tabla 4.13 se puede apreciar que el tiempo de desarrollo de las tareas del Sprint 1 tomó cuatro horas adicionales de lo planificado. A continuación, en la Figura 4.28 se muestra la diferencia entre las horas planificadas y las horas desarrolladas del Sprint 1.

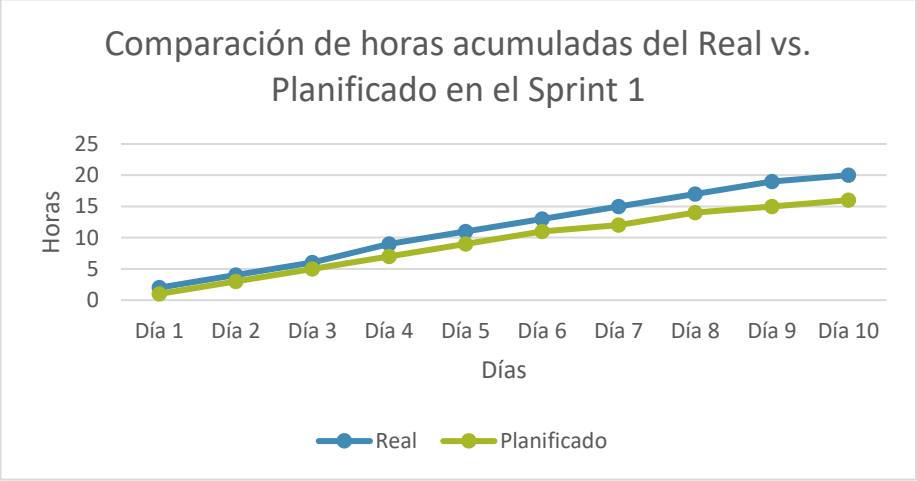


Figura 4.28 Comparación de horas acumuladas en el Sprint 1
Fuente: Elaboración propia

4.2.1.4 Retrospectiva del Sprint 1

Se concluye el desarrollo del Sprint 1 con éxito, puesto que completaron todas las tareas planificadas en los días establecidos. Adicional, se debe considerar los aspectos mencionados a continuación para el desarrollo de los siguientes *sprints*:

- Mantener un desarrollo funcional de las interfaces gráficas
- Conservar el esquema de desarrollado de la herramienta tecnológica para que esté orientado a facilitar la inclusión de las personas no videntes.

4.2.2 Sprint 2

4.2.2.1 Planificación del Sprint 2

En el Sprint 2 se busca desarrollar el medio de comunicación entre la persona no vidente y la herramienta tecnológica, es decir, el asistente por voz. A continuación, en la Tabla 4.14 se definen las tareas necesarias para cumplir el desarrollo del *Sprint* 2.

Tabla 4.14
Lista de tareas del sprint 2

Tarea #	**Lista de tareas sprint 2**	**Esfuerzo**
Tarea 1	Ingresar comandos de voz estándar para el asistente virtual	3h
Tarea 2	Desarrollar interfaz gráfica para administrar el asistente virtual	8h
	Total	11h

Fuente: Elaboración Propia

Luego de definir las tareas, se realizó una planificación del trabajo en horas necesarias para cumplirlas, como se muestra en la Tabla 4.15.

Tabla 4.15
Planificación de tareas para el desarrollo del Sprint 2

	Tarea 1	Tarea 2	Total
Día 1	2h	-	2h
Día 2	1h	-	1h
Día 3	-	3h	3h
Día 4	-	3h	3h
Día 5	-	2h	2h
Total	3h	8h	11h

Fuente: Elaboración Propia

En Tabla 4.15 el mayor esfuerzo invertido en horas se encuentra en la tarea 2, ya que se desarrolla el asistente virtual, es decir, el módulo de comunicación entre la persona no vidente y la herramienta tecnológica.

4.2.2.2 Desarrollo del Sprint 2

Posterior a la planificación del Sprint 2, se detallan las tareas a desarrollarse y el diagrama de casos de uso entre la persona no vidente y la herramienta tecnológica, como se muestra en la Figura 4.29

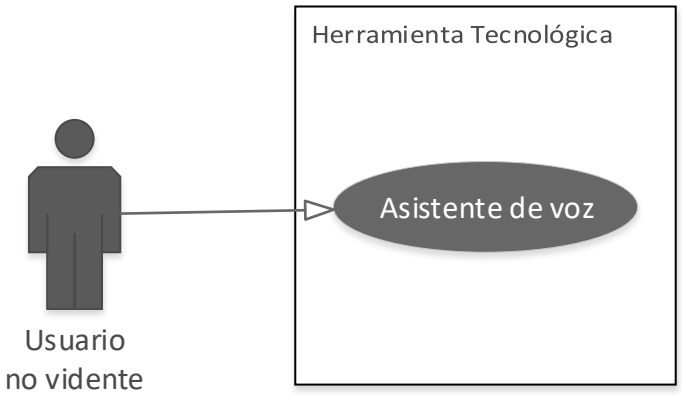


Figura 4.29 Diagrama de casos de uso por el usuario no vidente
Fuente: Elaboración propia

1. Ingresar comandos de voz estándar

   Para la comunicación entre la persona no vidente y la herramienta tecnológica existe el asistente virtual, capaz de ejecutar acciones en base a comandos de voz solicitados por la persona. Los principales comandos de voz son:

   - Buscar: Permite a la persona no vidente verificar a través de la herramienta tecnológica si está disponible un producto en específico dentro del supermercado.
   - Encontrar: La persona no vidente puede solicitar a la herramienta tecnológica que la guíe hasta donde se encuentra el producto.
   - Validar: La persona no vidente puede solicitar a la herramienta tecnológica que valide si el producto tomado es el solicitado.

Los comandos de voz se registrar desde la opción ingresar de la interfaz gráfica para administrador de comandos, como se muestra en la Figura 4.30.

Figura 4.30 Interfaz gráfica para ingresar comandos de voz
Fuente: Elaboración propia.

2. Desarrollar interfaz gráfica para administrar el asistente virtual

Se desarrolló una interfaz gráfica para validar la integridad de los comandos de voz solicitados por la persona no vidente y así realizar las

respectivas pruebas y mejoras del asistente por voz en caso de que amerite. De igual forma, la interfaz gráfica permitirá validar que la respuesta del asistente por voz esté correcta. A continuación, en la Figura 4.31 se muestra la interfaz gráfica del asistente virtual.

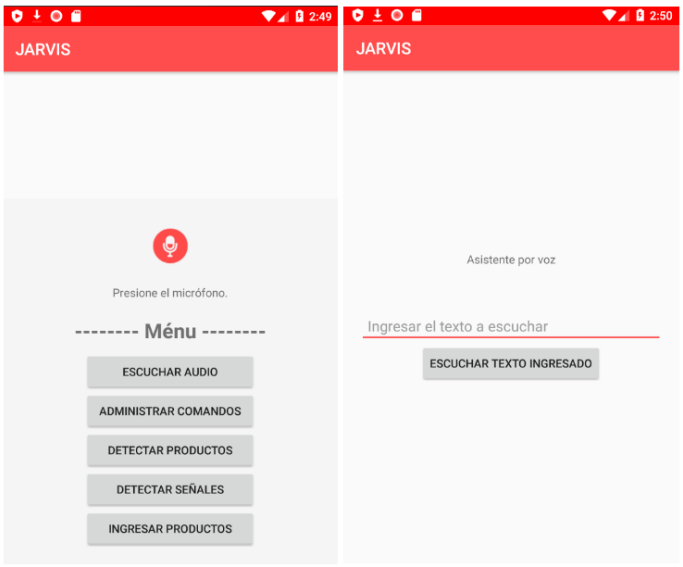


Figura 4.31 Interfaz gráfica del asistente virtual por voz
Fuente: Elaboración propia.

4.2.2.3 Revisión del Sprint 2

En el presente apartado, se detalla el estado de las actividades planificadas durante el *Sprint* 2, como se muestra en la Tabla 4.16. Dichas, actividades permitieron desarrollar el asistente virtual; el mismo que permite la comunicación entre la persona no vidente y la herramienta tecnológica.

Tabla 4.16
Lista de tareas completadas en el sprint 2

Tarea #	Lista de tareas sprint 2	Completado
Tarea 1	Ingresar comandos de voz estándar para el asistente virtual	✓
Tarea 2	Desarrollar interfaz gráfica para administrar el asistente virtual	✓

Fuente: Elaboración Propia

A continuación, la Tabla 4.17 muestra en resumen las tareas completadas y su tiempo de desarrollo.

Tabla 4.17
Tiempo de desarrollo para las tareas completadas en el Sprint 2

	Tareas completadas	Total
Día 1 **Día 2**	Tarea 1	3h
Día 3 **Día 4** **Día 5**	Tarea 2	3h 3h 3h
Total		**12h**

Fuente: Elaboración Propia

En la Tabla 4.17 se puede apreciar que el tiempo de desarrollo de las tareas del Sprint 2 tomó una hora adicional de lo planificado. A continuación, en la Figura 4.32 se muestra la diferencia entre las horas planificadas y las horas desarrolladas del Sprint 2.

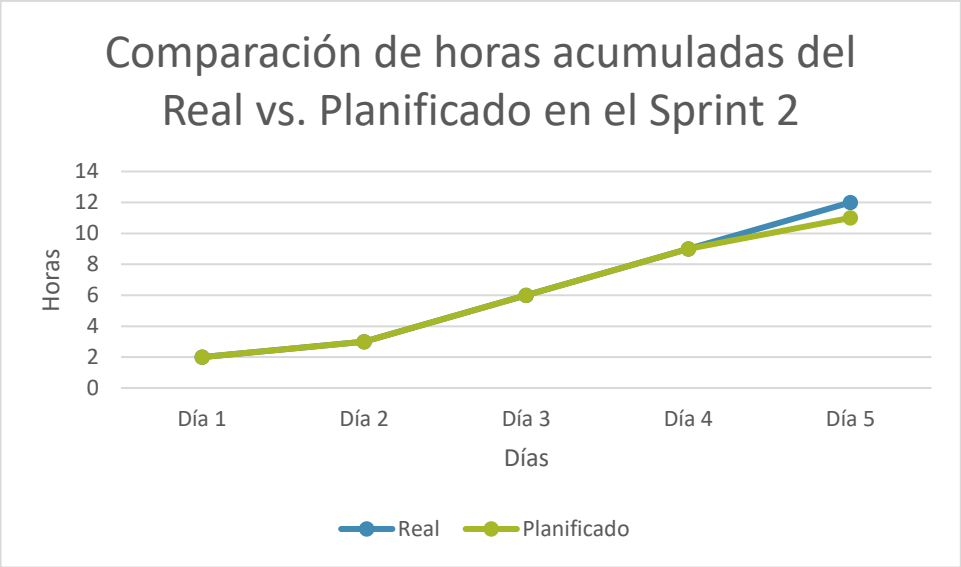


Figura 4.32 Comparación de horas acumuladas en el Sprint 2
Fuente: Elaboración propia

4.2.2.4 Retrospectiva del Sprint 2

Se completó cada tarea planificada para el *Sprint* 2, donde el tiempo en horas de desarrollo sólo se excedió con una hora adicional de lo planificado. Por lo tanto, para el *Sprint* 3 se sugiere tener en cuenta el punto a continuación:

- Mantener un desarrollo funcional de las interfaces gráficas

4.2.3 Sprint 3

4.2.3.1 Planificación del Sprint 3

El presente Sprint está enfocado en desarrollar el módulo principal de la herramienta tecnológica, el mismo que permite al usuario no vidente identificar los productos y encontrarlos mediante una navegación asistiva por voz a través del asistente virtual desarrollado en el *Sprint* 2. Con la finalidad de completar el Sprint 3 se definieron las tareas que se aprecian en la Tabla 4.18

Tabla 4.18
Lista de tareas del sprint 3

Tarea #	Lista de tareas sprint 3	Esfuerzo
Tarea 1	Configurar modelo para identificación de imágenes	3h
Tarea 2	Diseñar interfaz gráfica para el módulo de identificación de imágenes	8h
Tarea 3	Desarrollar proceso de identificación de productos	25h
Tarea 4	Desarrollar proceso de navegación asistiva	20h
	Total	56h

Fuente: Elaboración propia

Luego de definir las tareas, se realizó una planificación del trabajo en horas necesarias para cumplirlas, como se muestra en la Tabla 4.19.

Tabla 4.19
Planificación de tareas para el desarrollo del Sprint 3

	Tarea 1	Tarea 2	Tarea 3	Tarea 4	Total
Día 1	2h	-	-	-	2h
Día 2	1h	-	-	-	1h
Día 3	-	2h	-	-	2h
Día 4	-	2h	-	-	2h
Día 5	-	2h	-	-	2h
Día 6	-	2h	-	-	2h
Día 7	-	-	3h	-	3h
Día 8	-	-	3h	-	3h
Día 9	-	-	3h	-	3h
Día 10	-	-	3h	-	3h
Día 11	-	-	3h	-	3h
Día 12	-	-	3h	-	3h
Día 13	-	-	2h	-	2h
Día 14	-	-	2h	-	2h
Día 15	-	-	2h	-	2h
Día 16	-	-	1h	-	1h
Día 17	-	-	-	3h	3h
Día 18	-	-	-	3h	3h
Día 19	-	-	-	2h	2h
Día 20	-	-	-	2h	2h
Día 21	-	-	-	2h	2h
Día 22	-	-	-	2h	2h
Día 23	-	-	-	2h	2h
Día 24	-	-	-	2h	2h
Día 25	-	-	-	2h	2h
Total	3h	8h	25h	20h	56h

Fuente: Elaboración Propia

En Tabla 4.19 el mayor esfuerzo en horas se centra en el desarrollo de los procesos para identificar los productos y navegación asistiva. Al cumplir, con lo planificado en el Sprint 3 la persona no vidente podrá hacer uso de la herramienta tecnológica.

4.2.3.2 Desarrollo del Sprint 3

Posterior a la planificación. A continuación, se detallan las tareas a desarrollarse y el diagrama de casos de uso entre la persona no vidente y la herramienta tecnológica, como se muestra en la Figura 4.33

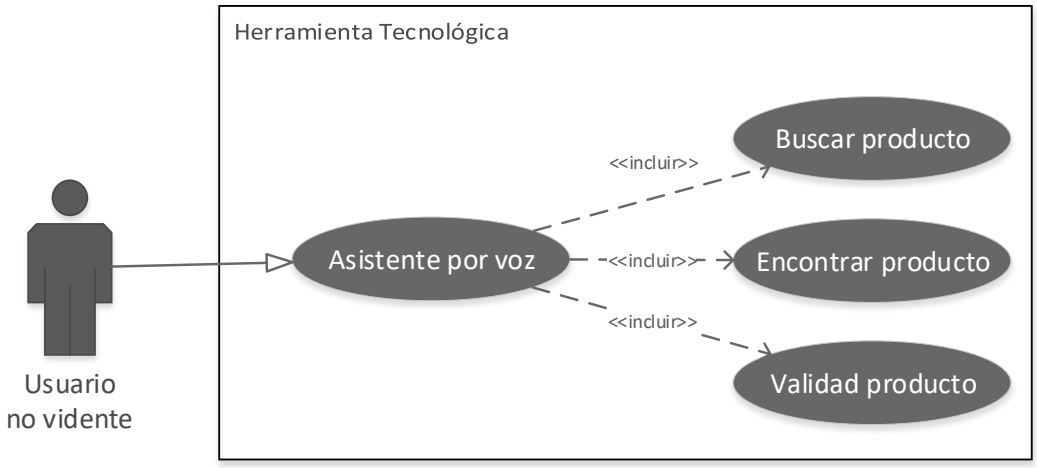


Figura 4.33 Diagrama de casos de uso por el usuario no vidente
Fuente: Elaboración propia

1. <u>Configurar modelo para la identificación de imágenes</u>

Luego de seleccionar el modelo de identificación de imágenes a implementarse, se almacenan los archivos que se obtuvieron como resultado del entrenamiento del modelo en el directorio *assets* del proyecto de software, para el posterior funcionamiento de los módulos de navegación e identificación de productos. A continuación, en la Figura 4.34 se puede visualizar la ruta de almacenamiento del proyecto y los archivos del modelo.

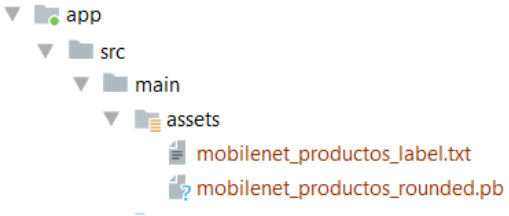


Figura 4.34 Ruta de almacenamiento y archivos generados por el modelo a implementar
Fuente: Elaboración propia

2. Diseñar interfaz gráfica para el módulo de identificación de imágenes

Para el diseño de la interfaz gráfica del modelo de identificación de imágenes se mantiene un esquema funcional como se sugirió en la retrospectiva del Sprint 2, con la finalidad de facilitar las pruebas y desarrollar futuras mejorar al modelo por parte del administrador de la herramienta tecnológica. A continuación, en la Figura 4.35 se puede apreciar un ejemplo de detección de producto.

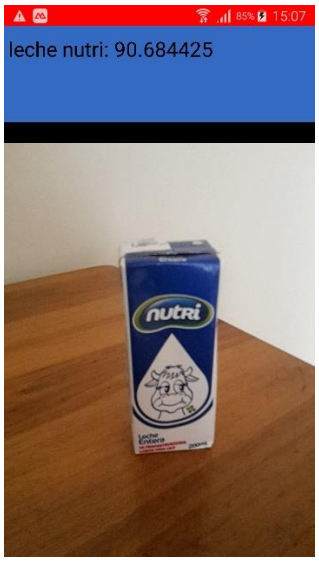


Figura 4.35 Interfaz gráfica del módulo de identificación de imágenes
Fuente: Elaboración propia.

En la parte superior de la interfaz, se puede apreciar información como: el nombre del producto detectado y su porcentaje de precisión por parte de la herramienta tecnológica; información que es de utilidad que el administrador de la herramienta pueda realizar futuros ajustes, verificaciones o pruebas.

3. Desarrollar proceso de identificación de productos

El proceso de identificación de productos permite a la persona no vidente agarrar un producto y validar si es el solicitado. Dicho proceso se activa mediante el comando de voz previamente configurado “Validar”. Una

vez que la herramienta tecnológica ha validado el producto, el asistente virtual comunica a la persona no vidente por medio de comandos de voz el resultado de la validación. Generalmente, se establecen rangos de probabilidad de que sea o no el producto escogido por la persona. En la Figura 4.36 se puede apreciar la validación del producto aceite girasol y el porcentaje de probabilidad del producto escogido por la persona no vidente es el correcto.

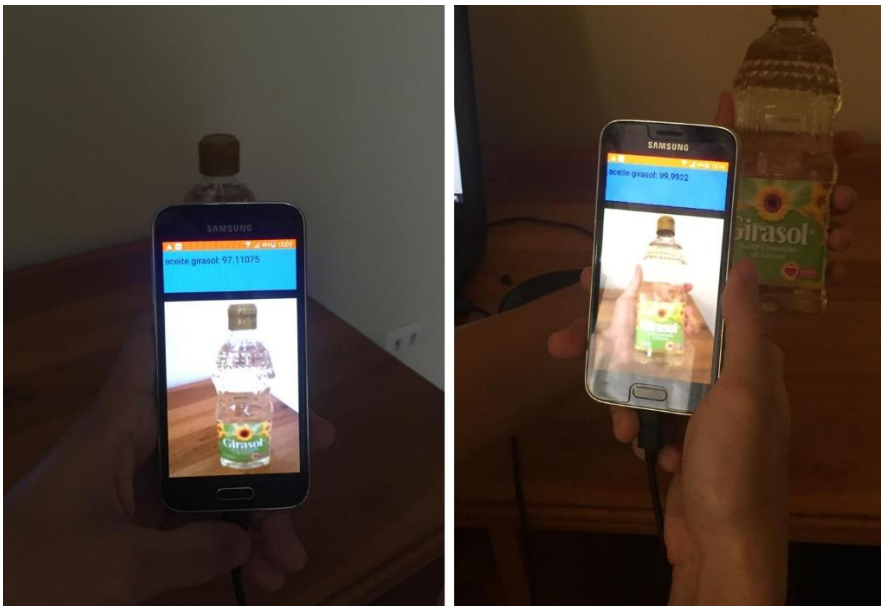


Figura 4.36 Prueba de identificación del producto aceite girasol
Fuente: Elaboración propia.

4. Desarrollar proceso de navegación asistiva

El proceso de navegación asistiva parte del mismo principio de identificación de imágenes, el cual se activa cuando la persona no vidente solicita el producto por medio del comando de voz “Buscar”. La herramienta tecnológica verifica si el producto tiene stock disponible, luego, detecta las señales en el pasillo mediante un código asociado al tipo de producto deseado. El código de la flecha está compuesto por las tres primeras letras de la palabra “código”, seguido de las iniciales de los dos tipos de producto acorde a su ubicación en el pasillo; si está a la “izquierda” del pasillo o a la “derecha” del pasillo, como se muestra en la Figura 4.37.

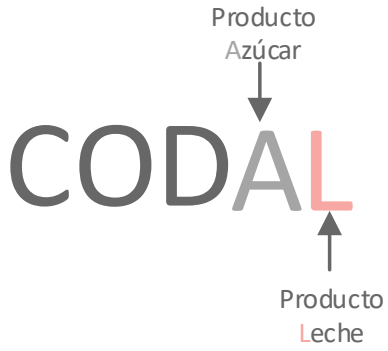


Figura 4.37 Código de señales para navegación asistiva.
Fuente: Elaboración propia.

En la Tabla 4.20 se puede apreciar la codificación establecida para las señales del módulo de navegación asistiva.

Tabla 4.20
Cuadro de codificación de las señales.

Señales	Codificación
	CODAL
	CODAS
	CODGA
	CODJP
	CODAV
	CODRE
	CODDE
	CODIZ

Fuente: Elaboración Propia

4.2.3.3 Revisión del Sprint 3

En la revisión del Sprint 3, se detalla el estado de las actividades planificadas, como se muestra en la Tabla 4.21. Dichas, actividades fueron completadas con éxito y permitieron el desarrollo del módulo principal de la herramienta tecnológica. El mismo que permite la identificación de productos y la navegación asistiva.

Tabla 4.21
Lista de tareas completadas en el sprint 3

Tarea #	Lista de tareas sprint 3	Completado
Tarea 1	Configurar modelo para identificación de imágenes	✓
Tarea 2	Diseñar interfaz gráfica para el módulo de identificación de imágenes	✓
Tarea 3	Desarrollar proceso de identificación de productos	✓
Tarea 4	Desarrollar proceso de navegación asistiva	✓

Fuente: Elaboración Propia

A continuación, la Tabla 4.22 muestra en resumen las tareas completas y su tiempo de desarrollo.

Tabla 4.22
Tiempo de desarrollo para las tareas completadas en el Sprint 3

	Tareas completadas	Total
Día 1	Tarea 1	2h
Día 2	Tarea 1	2h
Día 3	Tarea 2	3h
Día 4	Tarea 2	3h
Día 5	Tarea 2	2h
Día 6	Tarea 2	2h
Día 7	Tarea 3	3h
Día 8	Tarea 3	3h
Día 9	Tarea 3	3h
Día 10	Tarea 3	3h
Día 11	Tarea 3	3h

Día 12	Tarea 3	3h
Día 13	Tarea 3	3h
Día 14	Tarea 3	3h
Día 15	Tarea 3	2h
Día 16	Tarea 3	2h
Día 17	Tarea 4	4h
Día 18	Tarea 4	3h
Día 19	Tarea 4	3h
Día 20	Tarea 4	3h
Día 21	Tarea 4	2h
Día 22	Tarea 4	2h
Día 23	Tarea 4	2h
Día 24	Tarea 4	2h
Día 25	Tarea 4	2h
Total		**65h**

Fuente: Elaboración Propia

En la Tabla 4.22 se puede apreciar que el tiempo de desarrollo para las tareas del Sprint 3 generó nueve horas más de lo planificado. Sin embargo, se cumplió con el número de días establecidos. A continuación, en la Figura 4.38 se muestra la diferencia entre las horas planificadas y las horas desarrolladas en el Sprint 3.

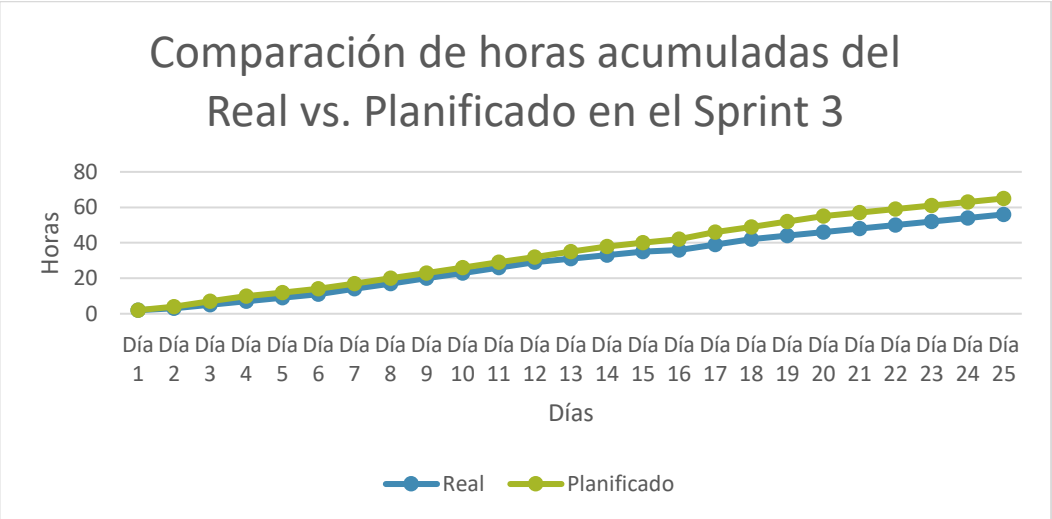


Figura 4.38 Comparación de horas acumuladas en el Sprint 3
Fuente: Elaboración propia.

4.2.3.4 Retrospectiva del *Sprint* 3

Con el *Sprint* 3 finalizado, se completaron todas las actividades planificadas para el desarrollo de la herramienta tecnológica. Existieron ciertas variaciones en el esfuerzo en horas planificado. Sin embargo, no hubo variación en el número de días.

4.3 Pruebas del desarrollo

Durante el desarrollo de cada Sprint se realizaron pruebas funcionales de tal manera que se pueda verificar el correcto funcionamiento de los módulos que integran la herramienta tecnológica. A continuación, se definieron escenarios de prueba dependiente de las actividades planificadas en cada *Sprint*.

4.3.1 Definición de pruebas

Para las pruebas de funcionalidad de la herramienta se seleccionó un usuario con las siguientes características, como se muestra en la Tabla 4.23. Se realizarán pruebas de funcionalidad en un ambiente que emule el interior de una tienda, debido a la cuarenta comunitaria obligatoria especificada en el marco de emergencia por pandemia sanitaria, en el artículo 4 del Decreto ejecutivo 1017 de 16 de marzo del 2020.

Tabla 4.23
Características del usuario para pruebas de funcionalidad

Características	Datos
Utiliza dispositivos tecnológicos	Si
Porcentaje de discapacidad visual	56 %
Motivo de uso de la herramienta tecnológica	Identificar los productos de manera autónoma

Fuente: Elaboración Propia

Para llevar a cabo las pruebas funcionales se definieron escenarios sólo del Sprint 3, debido a que el Sprint 1 y el Sprint 2 corresponde a configuraciones de la herramienta tecnológica y el usuario carece de los medios para realizar las pruebas de funcionalidad.

Acorde a las características indicadas en la Tabla 4.23, el usuario seleccionado permitió validar que la herramienta tecnológica funcione correctamente y brinde la autonomía esperada al momento de escoger e identificar los productos deseados.

4.3.1.1 Sprint 3

Para validar el funcionamiento de la herramienta tecnológica se definió un escenario, como se muestra en la Figura 4.39, donde se detalla el diagrama de flujo para comprobar el funcionamiento de la identificación de productos y la navegación asistiva.

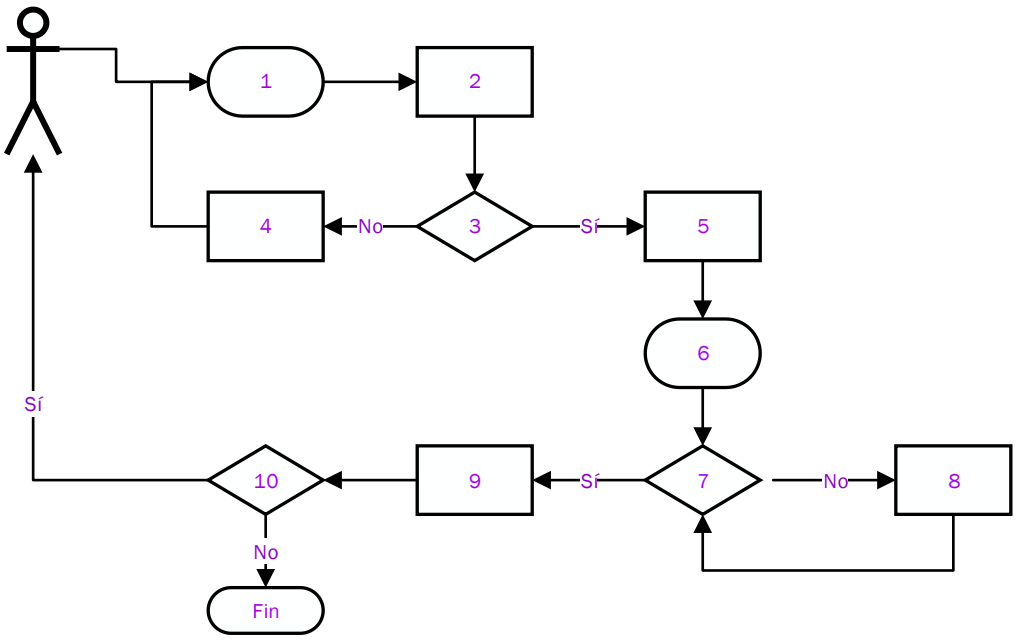


Figura 4.39 Diagrama de flujo del funcionamiento de la herramienta tecnológica
Fuente: Elaboración propia.

Eventos

1) La persona no vidente indica al asistente que producto desea buscar.
6) La persona no vidente toma el producto y solicita al asistente virtual que verifique si es el producto deseado.

Procesos

2) La herramienta procesa la solicitud.
4) Si no hay stock, el asistente virtual le indica al usuario que busque otro producto.
5) Si hay stock, el asistente virtual guía a la persona no vidente hacia la ubicación del producto mediante la identificación de señales ubicadas en los pasillos del supermercado.
8) Si la herramienta no identifica el producto, el asistente virtual solicita a la persona cambiar la orientación del producto escogido.
9) El asistente virtual le indica que es el producto correcto.

Decisiones

3) La herramienta valida si existe stock del producto.
7) La herramienta valida el producto escogido por la persona no vidente.
10) El asistente virtual pregunta si desea buscar otro producto.

4.3.2 Resultado de pruebas

4.3.2.1 Sprint 3

Los resultados del escenario de prueba definido para corroborar el funcionamiento de la herramienta tecnología. A continuación, se visualizan los procesos para:

Identificación de productos

Para validar el proceso de identificación se utilizó el producto aceite de la marca Girasol, donde el usuario solicitó al asistente virtual por medio del comando de voz “Validar” que detecte si es el producto escogido es el correcto, como se puede visualizar en la Figura 4.40.

Figura 4.40 Usuario validando un producto.
Fuente: Elaboración propia.

Navegación asistiva

Para validar el proceso de navegación asistiva se estableció una ruta señalizada para que el usuario pueda movilizarse hasta la señal de color (Amarillo / Naranja) que indique que en esa ubicación de encuentra el producto aceite girasol. Si la herramienta detecte una flecha de otro color únicamente indicará al usuario que continúe avanzando, como se puede visualizar en la Figura 4.41.

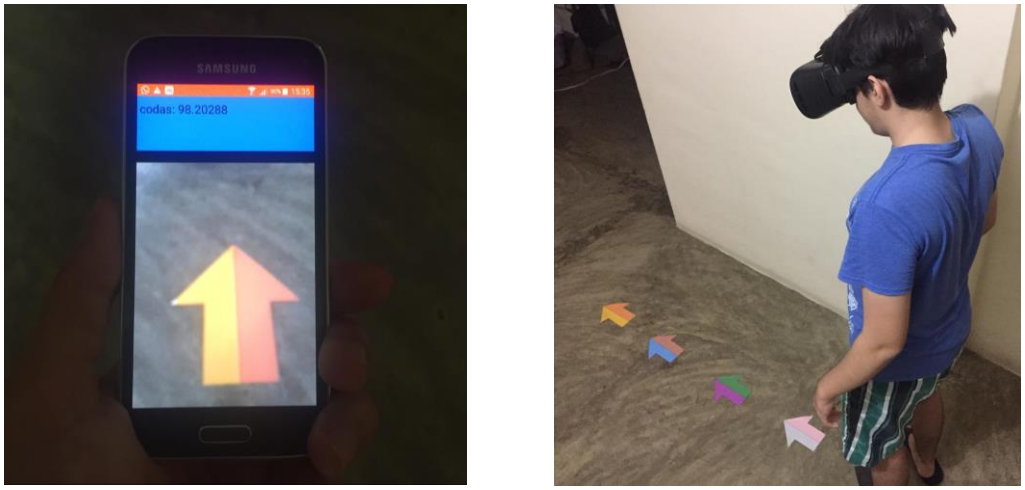


Figura 4.41 Usuario siguiendo la ruta hasta el producto solicitado.
Fuente: Elaboración propia.

Adicional, en la Figura 4.41 se puede observar que la herramienta tecnológica identifica el tipo de señal acorde al producto deseado.

De las pruebas realizadas por el usuario con la herramienta tecnológica, surgieron los siguientes comentarios de mejora:

- Incrementar el volumen de voz del asistente.
- Diseñar un soporte para gafas más ligero y con un aspecto más amigable.

4.4 Resumen de pruebas

A partir de las pruebas realizadas al *Sprint 3* de la herramienta tecnológica, permitieron verificar el correcto funcionamiento del módulo de identificación de productos y el módulo de navegación asistiva. La persona fue capaz de ubicar e identificar productos mediante las directrices por voz del asistente virtual. Adicional, el usuario sugirió ciertas mejoras que facilitarán el uso y la interacción con la herramienta tecnológica durante el proceso de compra.

CAPÍTULO 5 CONCLUSIONES Y RECOMENDACIONES

En el presente trabajo se construyó una herramienta tecnológica que implementa el modelo seleccionado, *MobileNet,* el cual permite la ubicación e identificación de productos de primera necesidad. Esta herramienta fue probada por una persona en la que se comprobó que su funcionamiento contribuye a la inclusión de este grupo vulnerable acorde a la Constitución vigente en el Ecuador; asistiéndolos en el proceso de compra dentro de un supermercado.

Para cumplir con el primer objetivo planteado, se realizó un análisis comparativo de los conceptos y características de dos modelos de clasificación de imágenes, en la que se determinó que *MobileNet* es un modelo apropiado para implementarse en dispositivos móviles ya que está optimizado para obtener un alto rendimiento para la clasificación de imágenes. Adicional, su porcentaje de precisión supera el 90% para la detección de imágenes.

Para llevar a cabo el segundo objetivo y en base a los resultados obtenidos se concluye que el modelo *MobileNet* genera archivos que requieren poco espacio de almacenamiento en el dispositivo, el consumo de memoria RAM es 12% menor que el modelo *InceptionV3*. Además, se contrastó que el porcentaje promedio de precisión en la detección de productos es 7% mayor en dispositivos móviles y su rapidez al identificarlos es 20 segundos menor que el otro modelo. Por tanto, se integró el modelo *MobileNet* en el desarrollo de los módulos de identificación de productos y navegación asistiva.

Para lograr el tercer objetivo, se utilizó *Scrum* como marco de desarrollo ágil, donde se definieron las actividades de cada *sprint*. En el *Sprint* 1, se realizaron las configuraciones de la interfaz de la administración de comandos para el asistente virtual. En el *Sprint* 2 y 3 se integraron los módulos de identificación de productos y navegación asistiva donde existieron ciertas variaciones en el esfuerzo en horas planificado. Finalmente, a partir de las pruebas funcionales, la persona aportó con

comentarios que permitan mejorar el funcionamiento de la herramienta tecnológica.

El cuarto objetivo se completó mediante pruebas de funcionalidad de la herramienta tecnológica. Las mismas se realizaron empleando diseños de escenarios de pruebas descritos en la sección 4.3.1 las cuales resultaron exitosas y sirvieron para corroborar que cada una de las actividades planificadas en los *Sprints* se desarrollaron apropiadamente. En lo referente a las pruebas funcionales de la herramienta tecnológica realizadas con personas en un ambiente que emuló el interior de una tienda, las personas indicaron que la herramienta es de gran ayuda al momento de identificar y encontrar productos, ya que ofrece autonomía en el proceso de compra.

Entre las contribuciones de este trabajo de investigación se realizó una revisión sistemática de los conceptos de la inteligencia artificial, su relación con el aprendizaje profundo y sus marcos de trabajo para la detección de imágenes. A partir de esta revisión se comparó de forma teórica y cuantitativa los modelos *InceptionV3* y *MobileNet* para su implementación en dispositivos móviles. Adicional, se desarrolló una herramienta tecnológica que emplea *MobileNet* para la identificación de productos y navegación asistiva.

La herramienta tecnológica se limitó únicamente a la identificación de productos de primera necesidad y a la navegación asistiva mediante un sistema de flechas establecidas. Finalmente se proponen como recomendaciones para trabajos futuros considerar una conexión directa entre la herramienta y el sistema de inventario del supermercado para identificar en tiempo real el stock de productos. También generar un conjunto de datos más extenso para que el modelo pueda identificar una gama de productos más completa. Diseñar un soporte para gafas más ligero y con un aspecto más amigable. Desarrollar un proceso que mejore el módulo de navegación asistiva para identificar la ruta más optima al producto.

REFERENCIAS

Ahmed, F., Mahmud, S., & Yeasin, M. (2019). RNN and CNN for Way-Finding and Obstacle Avoidance for Visually Impaired. *2019 2nd International Conference on Data Intelligence and Security (ICDIS)*, 225-228.

Allen, G., & Murphy, M. (2011). *Beginning Android 4.* New York: Apress.

Allen, G., & Owens, M. (2010). *The Definitive Guide to SQLite.* Apress.

Ameen, S., & Vadera, S. (2017). A convolutional neural network to classify American Sign Language fingerspelling from depth and colour images. *Expert Systems*.

Apache Software Foundation. (2017). *MXNET*. Obtenido de https://mxnet.apache.org/

Avishka, I., Kumarawadu, K., Kudagama, A., Weerathunga, M., & Thelijjagoda, S. (2019). Mobile App to Support People with Dyslexia and Dysgraphia. *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)*, 1-6.

Bankar, J., & Gavai, N. (2018). Convolutional Neural Network based Inceptionv3 Model for Animal Classification. *International Journal of Advanced Research in Computer and Communication Engineering*.

Bauer, Z., Dominguez, A., Cruz, E., Gomez-Donoso, F., Orts-Escolano, S., & Cazorla, M. (2019). Enhancing perception for the visually impaired with deep learning techniques and low-cost wearable sensors. *Pattern Recognition Letters*.

Binhua, T., Zixiang, P., Kang, Y., & Asif, K. (2019). Recent Advances of Deep Learning in Bioinformatics and Computational Biology. *Integration of Multisource Heterogenous Omics Information in Cancer*, 214.

Bullinger, H.-J. (2009). *Technology Guide.* Berlin: Springer.

Burbach, L., Halbach, P., Plettenberg, N., Nakayama, J., Ziefle, M., & Calero, A. (2019). "Hey, Siri", "Ok, Google", "Alexa". Acceptance-Relevant Factors of Virtual Voice-Assistants. *2019 IEEE International Professional Communication Conference (ProComm)*, 101-111.

Caffe. (2015). *Caffe*. Obtenido de http://caffe.berkeleyvision.org/

Chougrad, H., Zouaki, H., & Alheyane, O. (2014). Convolutional Neural Networks for Breast Cancer Screening: Transfer Learning with Exponential Decay. *ArXiv*.

Daroya, R., Peralta, D., & Naval, P. (2018). Alphabet Sign Language Image Classification Using Deep Learning. *TENCON 2018 - 2018 IEEE Region 10 Conference*, 0646-0650.

Dosi, S., Sambare, S., Singh, S., Lokhande, N., & Garware, B. (2018). Android Application for Object Recognition Using Neural Networks for the Visually Impaired. *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, 1-6.

Duarte, K., Cecílio, J., & Furtado, P. (2014). Overview of assistive technologies for the blind: Navigation and shopping. *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, 1929-1934.

Ecuador TV. (2019, Noviembre 7). *Ecuador TV*. Retrieved from https://www.ecuadortv.ec/noticias/tecnologia/speakliz-app-ecuatoriana-discapacidad-

Fatih, E., & Galip, A. (2017). Data classification with deep learning using Tensorflow. *2017 International Conference on Computer Science and Engineering (UBMK)*.

Fawzy Gad, A. (2019). *Practical Computer Vision Applications Using Deep Learning with CNNs: With Detailed Examples in Python Using TensorFlow and Kivy.* Apress.

Feiler, J. (2015). Introducing SQLite for Mobile Developers. Apress.

Fernandez, W., Xian, Y., & Tian, Y. (2017). Image-Based Barcode Detection and Recognition to Assist Visually Impaired Persons. *2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems*, 1241-1245.

Forte, C., Luciano, S., Pazini, C., & Marengoni, M. (2013). EyeBuy: Using augmented reality in accessible interfaces for consumers with visual acuity reduction. *2013 1st Workshop on Virtual and Augmented Assistive Technology (VAAT)*, 1-6.

Gavilán, I. (2019). *La carrerar digital.* ExLibric.

Google. (2018). *Google Cloud*. Obtenido de https://cloud.google.com/speech

Google. (2018). *Tensorflow*. Obtenido de https://www.tensorflow.org/

Google LLC. (2020). *GitHub*. Obtenido de https://github.com/tensorflow/hub

Gramajo, M., Ballejos, L., & Ale, M. (2018). Software Requirements Engineering through Machine Learning Techniques: A Literature Review. *2018 IEEE Biennial Congress of Argentina (ARGENCON)*, 1-7.

Guan, Q., Wan, X., Lu, H., Ping, B., Li, D., Wang, L., . . . Xiang, J. (2019). Deep convolutional neural network Inception-v3 model for differential diagnosing of lymph node in cytological images: a pilot study. *Annals of translational medicine*.

Guzman, A. (2019). Voices in and of the machine: Source orientation toward mobile virtual assistants. *Computers in Human Behavior*, 343-350.

He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J., & Li, M. (2018). Bag of Tricks for Image Classification with Convolutional Neural Networks.

Hersh, M., & Johnson, M. (2008). *Assistive technology for visually impaired and blind people.* Lóndres: Springer-Verlag.

Hong-Yen, C., & Chung-Yen, S. (2018). An Enhanced Hybrid MobileNet. *2018 9th International Conference on Awareness Science and Technology (iCAST)*, 308-312.

Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., . . . Hartwig, A. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.

Jabeen, F., Muhammad, A., & Martinez Enriquez, A. (2015). Feed forward neural network training based interactive shopping for blind. *2015 12th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, 1-6.

Keras. (2018). *Keras Documentation*. Retrieved from https://keras.io/

Kulyukin, V., & Kutiyanawala, A. (2010). Demo: ShopMobile II: Eyes-free supermarket grocery shopping for visually impaired mobile phone users. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, 31-32.

Kulyukin, V., Gharpure, C., & Nicholson, J. (2005). RoboCart: toward robot-assisted navigation of grocery stores by the visually impaired. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2845-2850.

Lanigan, P., Paulos, A., Williams, A., Rossi, D., & Narasimhan, P. (2006). Trinetra: Assistive Technologies for Grocery Shopping for the Blind. *2006 10th IEEE International Symposium on Wearable Computers*, 147-148.

Lei, H., Ganjeizadeh, F., Jayachandran, P., & Ozcan, P. (2017). A statistical analysis of the effects of Scrum and Kanban on software development projects. *Robotics and Computer-Integrated Manufacturing*, 59-67.

Li, H., Habesab, M., Wolk, D., & Fana, Y. (2019). A deep learning model for early prediction of Alzheimer's disease dementia based on hippocampal magnetic resonance imaging data. *Alzheimer's & Dementia*, 1059-1070.

Lilit , H., Lumsden, J., O'Sullivan, D., & Bartlett, H. (2013). Mobile assistive technologies for the visually impaired. *Survey of Ophthalmology, 58*(6), 513-528. Obtenido de http://www.sciencedirect.com/science/article/pii/S0039625712002512

Liu, J.-W., Ho, C.-Y., Chang, J., & Chia-An Tsai, J. (2019). The role of Sprint planning and feedback in game development projects: Implications for game quality. *Journal of Systems and Software*, 79-91.

Lopez, W., Vieira, S., Garcia-Dias, R., & Mechelli, A. (2019). *Convolutional neural networks.* Elsevier.

López-de-Ipiña, D., Lorido, T., & López, U. (2011). Indoor Navigation and Product Recognition for Blind People Assisted Shopping.

Machado, E., Carrillo, I., Saldana, D., Chen, F., & Chen, L. (2019). An Assistive Augmented Reality-based Smartglasses Solution for Individuals with Autism Spectrum Disorder. *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress*, 245-249.

McClure, N. (2017). *Tensorflow machine learning cookbook.* Packt Publishing.

Melissa, C., & Reith, R. (25 de Octubre de 2019). *International Data Corporation*. Obtenido de https://www.idc.com/promo/smartphone-market-share/os

Michelucci, U. (2018). *Training Neural Networks.* Apress.

Ministerio de Salud Pública. (2019, Octubre). *Consejo Nacional para la Igualdad de Discapacidades*. Retrieved from https://www.consejodiscapacidades.gob.ec/estadisticas-de-discapacidad/

MINTEL/JLV. (2016). *Ministerio de Telecomunicaciones y de la Sociedad de la Información*. Retrieved from https://www.telecomunicaciones.gob.ec/handeyes-una-muestra-de-como-las-tecnologias-cambian-vidas/

Mitchell, M. (2019). *Artificial Intelligence: A Guide for Thinking Humans.*

Mocanu, B., Tapu, R., & Zaharia, T. (2019). Design of a CNN Face Recognition System Dedicated to Blinds. *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 1-2.

Monika, K., Moohana Priya, P., Jayashree, C., Mohanavalli, S., Sasirekha, S., & Joe Louis Paul, I. (2019). Smart Eye for Visually Impaired-An aid to help the blind people. *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, 1-5.

Moolayil, J. (2019). *Learn Keras for Deep Neural Networks.* Apress.

Mulfari, D. (2018). A TensorFlow-based Assistive Technology System for Users with Visual Impairments. *In Proceedings of the Internet of Accessible Things*, 1–2.

Mulfari, D., Palla, A., & Fanucci, L. (2017). Embedded Systems and TensorFlow Frameworks as Assistive Technology Solutions. *Studies In Health Technology And Informatics*, 396-400.

Muñoz Moreno, J. (2019). Herramienta de software para reconocimiento de objetos como ayuda a los procesos de mercadeo institucional.

Naway, A., & Li, Y. (2018). A Review on The Use of Deep Learning in Android Malware Detection. *International Journal of Computer Science and Mobile Computing*, 42-58.

Nayak, A., & Dutta, K. (2017). Impacts of machine learning and artificial intelligence on mankind. *2017 International Conference on Intelligent Computing and Control (I2C2)*, 1-3.

NET Foundation and Contributors. (2019). *Infer.NET*. Obtenido de https://dotnet.github.io/infer/default.html

Nicholson, J., Kulyukin, V., & Coster, D. (2009). ShopTalk: Independent Blind Shopping Through Verbal Route Directions and Barcode Scans. *The Open Rehabilitation Journal*, 11-23.

Nitin, G., Yashashree, J., Seema, T., & Rashmi, B. (2017). MobileNets for flower classification using TensorFlow. *2017 International Conference on Big Data, IoT and Data Science (BID)*, 154-158.

Organización de las Naciones Unidas. (2015). *Objetivos de Desarrollo Sostenible*. Obtenido de https://www.un.org/sustainabledevelopment/es/inequality/

Organización Mundial de la Salud. (9 de Junio de 2011). *Organización Mundial de la Salud*. Obtenido de https://www.who.int/dg/speeches/2011/disability_20110609/es/

Organización Mundial de la Salud. (2018). *Organización Mundial de la Salud*. Obtenido de https://www.who.int/es/news-room/fact-sheets/detail/blindness-and-visual-impairment

Orozco, M., & Corazón, R. (2019). Vehicular Detection and Classification for Intelligent Transportation System: A Deep Learning Approach Using Faster R-CNN Model. *International Journal of Simulation -- Systems, Science & Technology*, 11.1-11.7.

Paluszek, M., & Thomas, S. (2017). *MATLAB Machine Learning.* Apress.

Pattanayak, S. (2017). *Introduction to Deep-Learning Concepts and TensorFlow.* Apress.

Peña, A., Bonet, I., Manzur, D., Góngora, M., & Caraffini, F. (2019). Validation of convolutional layers in deep learning models to identify patterns in multispectral images. *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, (págs. 1-6). Coimbra.

Pintado, D., Sanchez, V., Adarve, E., Mata, M., Gogebakan, Z., Cabuk, B., . . . Oh, P. (2019). Deep Learning Based Shopping Assistant For The Visually Impaired. *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 1-6.

Pytorch. (2018). *Pytorch*. Obtenido de https://pytorch.org/

Qinlong, G., Xingmei, C., Weiwei, T., & Minghai, Y. (2010). Study and application of SQLite embedded database system based on Windows cE. *The 2nd International Conference on Information Science and Engineering*, 6920-6923.

Reddy, S. (2019). A survey of big data and machine learning. *International Journal of Electrical & Computer Engineering*, 575-580.

Senabre, E. (2019). Adapting the scrum framework for agile project management in science: case study of a distributed research initiative. *Heliyon*.

Sewak, M. (2019). *Deep Reinforcement Learning.* Springer Singapore.

Shehieb, W., Osama Nasri, M., Mohammed, N., Debsi, O., & Arshad, K. (2018). Intelligent Hearing System using Assistive Technology for Hearing-Impaired Patients. *018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference*, 725-729.

Shrivastava, N., Saxena, A., Kumar, Y., Ratn Shah, R., Mahata, D., & Stent, A. (2019). MobiVSR: A Visual Speech Recognition Solution for Mobile Devices.

Smith, A., Keane, A., Dumesic, J., Huber, G., & Zavala, V. (2019). A machine learning framework for the analysis and prediction of catalytic activity from experimental data. *Applied Catalysis B: Environmental*.

Smith, A., Keane, A., Dumesic, J., Willis Huber, G., & Zavala, V. (2019). A Machine Learning Framework for the Analysis and Prediction of Catalytic Activity from Experimental Dat. *Applied Catalysis B: Environmental* .

Srivastava, A., Sengupta, S., Kang, S.-J., Kant, K., Khan, M., Ali, A., . . . Brown, D. (2019). Deep Learning for Detecting Diseases in Gastrointestinal Biopsy Images. *Systems and Information Engineering Design Symposium (SIEDS)*, 1-4.

Sze, V., Chen, Y.-H., Yang, T.-J., & Emer, J. (2017). Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *Proceedings of the IEEE*, 2295 - 2329.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). Rethinking the Inception Architecture for Computer Vision. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* , 2818–2826.

Theano. (2018). *Theano*. Obtenido de http://deeplearning.net/software/theano/

Toro-Hernández, M. L., Kankipati, P., Goldberg, M., Contepomi, S., Rodrigues, D., & Bray, N. (2019). Appropriate Assistive Technology for Developing Countries. *Physical Medicine and Rehabilitation Clinics of North America, 30*(4), 847-865.

Vijayasarathy, L., & Butler, C. (2016). Choice of Software Development Methodologies: Do Organizational, Project, and Team Characteristics Matter? *IEEE Software*, 86-94.

Wang, C., Chen, D., Hao, L., Liu, X., Zeng, Y., Chen, J., & Zhang , G. (2019). Pulmonary Image Classification Based on Inception-v3 Transfer Learning Model. *IEEE Access*, 146533-146541.

Wannipa, S.-L., Wiphada, W., & Pattara, A. (2019). Convolutional Neural Networks Using MobileNet for Skin Lesion Classification. *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 242-247.

Xiaoling, X., Cui, X., & Bing, N. (2017). Inception-v3 for Flower Classification. *2017 2nd International Conference on Image, Vision and Computing*.

Zientara, P., Lee, S., Smith, G., Brenner, R., Itti, L., Rosson, M., . . . Narayanan, V. (2017). Third Eye: A Shopping Assistant for the Visually Impaired. *Computer, 50*(2), 16-24.