PROTOTIPO DE INTELIGENCIA ARTIFICIAL PARA LECTURA DE SISTEMA BRAILLE PARA UNITRÓPICO

JULIO ALEJANDRO RIVERA RODRIGUEZ
JHOAN SEBASTIAN HERNANDEZ SANCHEZ

UNIVERSIDAD INTERNACIONAL DEL TRÓPICO AMERICANO-UNITRÓPICO
FACULTAD DE INGENIERÍAS
PROGRAMA DE INGENIERÍA DE SISTEMAS
YOPAL
2024

PROTOTIPO DE INTELIGENCIA ARTIFICIAL PARA LECTURA DE SISTEMA BRAILLE PARA UNITRÓPICO

JULIO ALEJANDRO RIVERA RODRIGUEZ
JHOAN SEBASTIAN HERNANDEZ SANCHEZ

Proyecto presentado para optar por el título de
INGENIERO DE SISTEMAS

FABIÁN ALBERTO AYALA LOZANO

UNIVERSIDAD INTERNACIONAL DEL TRÓPICO AMERICANO-UNITRÓPICO
FACULTAD DE INGENIERÍAS
PROGRAMA DE INGENIERÍA DE SISTEMAS
YOPAL
2024

Nota de aceptación:

Firma del presidente del jurado

Firma del jurado

Firma del jurado

Yopal, (fecha)

DEDICATORIA

Con profunda gratitud y reconocimiento, dedico este proyecto a mis padres, quienes han sido pilares fundamentales en mi trayectoria académica y personal. Su inquebrantable apoyo, aliento y valores inculcados han sido la brújula que me ha guiado hacia la consecución de mis metas. Agradezco infinitamente su amor incondicional, su confianza en mis capacidades y su constante motivación para alcanzar la excelencia.

Extiendo mi más sincero agradecimiento a mis profesores y mentores, quienes han compartido conmigo su invaluable conocimiento, sabiduría y experiencia. Sus valiosas enseñanzas, su acertada guía y su constante estímulo han sido esenciales para mi formación profesional e intelectual. Reconozco con profunda admiración su dedicación a la docencia y su compromiso con el aprendizaje de sus estudiantes.

A mis compañeros y amigos, quienes han formado parte de este camino académico, les agradezco profundamente su amistad, apoyo y compañerismo. Comparto con ustedes las satisfacciones alcanzadas y agradezco su presencia en los momentos de desafíos. Su solidaridad, colaboración y espíritu de equipo han enriquecido significativamente mi experiencia universitaria.

Con profunda satisfacción y sentido de logro, me dedico este proyecto a mí mismo, reconociendo el esfuerzo, la perseverancia y la disciplina que he invertido en su realización. Este trabajo representa un hito importante en mi trayectoria profesional y un paso significativo hacia el cumplimiento de mis aspiraciones.

AGRADECIMIENTOS

Al finalizar este proyecto de grado, me gustaría expresar mi más sincero agradecimiento a las siguientes personas e instituciones que han contribuido de manera significativa a su realización:

A mi director de grado, Fabián Alberto Ayala Lozano, por su invaluable guía, apoyo y paciencia durante todo el proceso de investigación. Sus valiosos consejos y sugerencias fueron fundamentales para la culminación exitosa de este trabajo.

A mis padres, por su amor incondicional, apoyo moral y constante motivación. Gracias por creer en mí y por sus sacrificios para que pudiera alcanzar mis metas.

A mi hermano, por su apoyo y aliento en los momentos difíciles. Su cariño y comprensión han sido pilares fundamentales en mi vida.

A los profesores, por sus valiosas enseñanzas y por haberme transmitido su pasión por el conocimiento.

A la Universidad Internacional del Trópico Americano, por brindarme la oportunidad de realizar mis estudios y por proporcionarme los recursos necesarios para desarrollar este proyecto.

A todas las personas que de una u otra manera han contribuido a este proyecto, les expreso mi más profunda gratitud.

Este logro no hubiera sido posible sin el apoyo y la colaboración de todos ustedes. Gracias por ser parte de este camino.

CONTENIDO

pág.

RESUMEN 11

ABSTRACT 12

INTRODUCCIÓN 13

1 PLANTEAMIENTO DEL PROBLEMA 14

1.1 FORMULACIÓN 15

2 JUSTIFICACIÓN 16

3 OBJETIVOS 17

3.1 OBJETIVO GENERAL. 17

3.2 OBJETIVOS ESPECÍFICOS. 17

4 ESTADO DEL ARTE 18

5 MARCO REFERENCIAL 19

5.1 MARCO TEÓRICO 19

5.1.1 Aplicación móvil. 19

5.1.2 AI (Inteligencia Artificial). 19

5.1.3 Braille. 19

5.1.4 Reconocimiento de imágenes. 19

5.1.5 Visión por ordenador 19

5.1.6 OCR (Reconocimiento Óptico de Caracteres). 20

5.1.7 Traducción. 20

5.1.8 Accesibilidad. 20

5.1.9 Interfaz de usuario (UI) y experiencia de usuario (UX). 20

5.1.10 Aprendizaje automático. 20

5.1.11 Aprendizaje profundo. 20

5.2 MARCO LEGAL 21

5.2.1 Ley 2265 de 2022. 21

5.2.2 Ley 1346 de 2009. 21
5.2.3 Ley 1680 de 2013. 21
5.2.4 Decreto 1377 de 2013. 22
5.3 CRONOGRAMA DE ACTIVIDADES 22
5.4 PRESUPUESTO 23
6 METODOLOGÍA....24
6.1 MODELO DE DESARROLLO 24
6.1.1 Red neuronal 25
6.2 MÉTRICAS....29
6.2.1 Red neuronal 29
6.3 VARIABLES 36
6.4 REQUERIMIENTOS....37
7 ENTORNO DE DESARROLLO 38
7.1 REQUISITOS PREVIOS 38
7.1.1 Visual Studio Code. 38
7.1.2 Hardware. 38
7.1.3 Python....38
7.1.4 Android. 38
8 RED NEURONAL....39
8.1 IMPLEMENTACIÓN DE RED NEURONAL SECUENCIAL....39
8.2 MODELADO DE ARQUITECTURA Y SERVICIOS....39
8.2.1 Requerimientos de librerías. 39
9 APLICACIÓN ANDROID 41
9.1 CAMERA FRAGMENT....41
9.1.1 Inicializar la cámara. 41
9.1.2 Configurar la detección de objetos. 41
9.1.3 Procesar los resultados de la detección. 42
9.1.4 Actualizar la interfaz de usuario (UI). 42
9.1.5 Manejar errores. 42

9.1.6 Control TTS (Text-To-Speech). 42
9.2 PERMISSIONS FRAGMENT 43
9.2.1 Definición de permisos. 43
9.2.2 Solicitud de permisos. 44
9.2.3 Respuesta del usuario. 44
9.2.4 Verificación de permisos. 44
9.2.5 Navegación al fragmento de la cámara. 44
9.2.6 Método `hasPermissions`. 44
9.3 MAIN ACTIVITY 45
9.3.1 Patrón de actividad única. 45
9.3.2 Enlace de la vista de layout. 46
9.3.3 Comportamiento del botón retroceso. 46
9.4 OBJECT DETECTOR HELPER 47
9.4.1 Paquete e importaciones. 47
9.4.2 Clase ObjectDetectorHelper. 48
9.4.3 Propiedades. 48
9.4.4 ObjectDetector. 48
9.4.5 Método. 48
9.4.6 Interfaz DetectorListener. 48
9.4.7 Constantes. 48
9.5 OVERLAY VIEW 50
9.5.1 Clase OverlayView. 50
9.5.2 Métodos. 50
9.5.3 Compañero del objeto. 50
10 INTERFAZ GRAFICA 52
BIBLIOGRAFÍA 56
ANEXOS 58

LISTA DE TABLAS

pág.

Tabla 1 - cronograma de actividades 22

Tabla 2 - presupuesto 23

LISTA DE ILUSTRACIONES

pág.

Ilustración 1 - Estructura modelo incremental 24
Ilustración 2 - Precisión red neuronal 29
Ilustración 3 - Pérdida red neuronal 30
Ilustración 4 - Entrenamiento red neuronal 31
Ilustración 5 - Evaluación modelo 32
Ilustración 6 - Visualización predicciones 33
Ilustración 7 - Precisión y perdida N° 1 34
Ilustración 8 - Prueba predicciones N°1 35
Ilustración 9 - Precisión y Pérdida N°2 35
Ilustración 10 - Prueba predicciones N°2 36
Ilustración 11 - Características Pc 38
Ilustración 12 - Python 3.9.13 39
Ilustración 13 - librerías 40
Ilustración 14 - Estructura de código 40
Ilustración 15 - Estructura de código 41
Ilustración 16 - CameraFragment 43
Ilustración 17 - PermissionsFragment 45
Ilustración 18 - MainActivity 47
Ilustración 19 - ObjectDetectorHelper 49
Ilustración 20 - OverlayView 51

RESUMEN

Este proyecto se enfoca en el desarrollo de una aplicación móvil destinada a facilitar el aprendizaje y la enseñanza del braille, dirigida especialmente a la comunidad de la Universidad Internacional del Trópico Americano. La aplicación, que combina inteligencia artificial, tecnología de cámara y funcionalidades de texto a voz, tiene como objetivo principal proporcionar una herramienta integral para el acceso al braille de manera eficiente y efectiva.

La funcionalidad central del software consiste en utilizar la cámara del dispositivo para capturar texto en braille, que luego es traducido al español y presentado al usuario a través de retroalimentación auditiva en tiempo real. Esta innovadora característica no solo permite a los usuarios leer el braille de forma rápida y precisa, sino que también les brinda la oportunidad de mejorar sus habilidades de lectura de manera autónoma.

Uno de los aspectos más destacados de esta aplicación es su enfoque en el fomento del aprendizaje independiente. Al proporcionar una interfaz fácil de usar y retroalimentación inmediata, se busca mejorar la motivación y el compromiso de los usuarios con el proceso de aprendizaje del braille. Además, al abordar los desafíos específicos asociados con la educación en braille, esta solución aspira a contribuir significativamente a la creación de un entorno de aprendizaje más inclusivo y accesible para todos.

En resumen, esta aplicación móvil representa un avance importante en el campo de la educación inclusiva, así como apoyo, pues al proporcionar a los usuarios las herramientas necesarias, se espera una significante mejora a la hora de practicar y mejorar sus habilidades de lectura en braille de manera autónoma y efectiva.

Palabras clave: Aplicación móvil, Inteligencia artificial, Tecnología de cámara, Traducción braille, Educación inclusiva.

ABSTRACT

This project focuses on the development of a mobile application aimed at facilitating the learning and teaching of Braille, especially for the community of the Universidad Internacional del Trópico Americano. The application, which combines artificial intelligence, camera technology and text-to-speech functionalities, has as its main objective to provide a comprehensive tool for accessing braille in an efficient and effective manner.

The core functionality of the software consists of using the device's camera to capture braille text, which is then translated into Spanish and presented to the user through real-time auditory feedback. This innovative feature not only allows users to read braille quickly and accurately, but also gives them the opportunity to improve their reading skills autonomously.

One of the highlights of this application is its focus on encouraging independent learning. By providing a user-friendly interface and immediate feedback, it seeks to enhance users' motivation and engagement with the braille learning process. Furthermore, by addressing the specific challenges associated with braille education, this solution aims to contribute significantly to the creation of a more inclusive and accessible learning environment for all.

In summary, this mobile application represents an important advancement in the field of inclusive education, as well as support, as by providing users with the necessary tools, a significant improvement is expected in practicing and improving their braille reading skills in an autonomous and effective manner.

Keywords: Mobile application, Artificial intelligence, Camera technology, Braille translation, Inclusive education.

INTRODUCCIÓN

El presente proyecto está destinado a apoyar la lectura en braille a través de la inteligencia artificial (IA). Este se centra en el desarrollo de un prototipo especializado de IA adaptado específicamente a la alfabetización en Braille, con el objetivo general de facilitar el aprendizaje independiente entre los estudiantes con discapacidad visual.

El objetivo principal de este proyecto es crear una aplicación de software que proporcione información auditiva en tiempo real, abordando los desafíos únicos asociados con la educación en Braille. Aprovechando el poder de la IA, esta solución pretende garantizar la facilidad de uso, mejorar la motivación y fomentar el compromiso entre los estudiantes con discapacidad visual. Se espera que el resultado final contribuya significativamente a un entorno educativo más integrador.

Las principales características de este innovador proyecto informático son la capacidad de traducción al braille, la información auditiva en tiempo real y el fomento de la alfabetización en braille. A través de estas características, la aplicación pretende ayudar a los estudiantes y a cualquier persona en proceso de aprendizaje para que practiquen y mejoren de forma autónoma sus habilidades de lectura en Braille. La aspiración final es crear una herramienta de aprendizaje que no sólo aborde los retos habituales a la hora de aprender braille, sino que también enriquezca su experiencia educativa.

El proyecto sienta las bases para explorar los distintos componentes del prototipo de IA, haciendo hincapié en su potencial para derribar barreras y crear nuevas oportunidades de aprendizaje autónomo. Las próximas secciones profundizarán en los aspectos técnicos del proyecto, su aplicación para abordar los retos propios de la enseñanza del braille y el impacto positivo previsto en la inclusividad de las experiencias educativas de los alumnos con discapacidad visual.

1 PLANTEAMIENTO DEL PROBLEMA

En los últimos 30 años se ha reducido la proporción de personas con discapacidad visual y ceguera en todo el mundo [1]. Según el Censo 2018, 1.948.332 personas con discapacidad visual habitan en Colombia, esto equivale al 4,1 por ciento de toda la población [2].

En Colombia, el español es el idioma oficial y lo habla la mayoría de la población. Las personas ciegas y deficientes visuales utilizan el braille para leer y escribir. En Colombia se utiliza el alfabeto Braille español, que es muy similar al Braille francés con la adición de la letra "ñ" y ligeras modificaciones de las letras acentuadas y algunas diferencias en la puntuación. El alfabeto Braille español completo se utiliza también para el gallego. La letra "ñ" se comparte con el braille vasco y el braille guaraní. La puntuación del braille gallego y vasco es la misma que la del braille español. Por lo tanto, una aplicación que facilite el aprendizaje del braille con traducción del español al braille podría ser útil para las personas ciegas y deficientes visuales de Colombia que están aprendiendo braille [3], [4]

El problema central del proyecto es la carencia de herramientas tecnológicas para asistir el aprendizaje del lenguaje braille, según investigación existente sobre el tema, la cual sugiere que hay poca evidencia que apoye el uso de la tecnología actual para la alfabetización en braille [5].

La situación deseada del proyecto es crear un prototipo funcional de inteligencia artificial para la lectura del sistema Braille que pueda reconocer caracteres Braille y convertirlos en caracteres de lenguaje natural. El prototipo debe ser capaz de apoyar el aprendizaje en el lenguaje braille para aquellas personas en proceso de capacitación, para así enseñarle a aquellas con discapacidad visual, y en consecuencia, mejorar la competencia en la lectura del sistema Braille para las personas con discapacidad visual en la Universidad Internacional del Trópico Americano que utilizan el sistema, además de aumentar la accesibilidad a los documentos Braille, y proporcionar un método más eficiente y preciso de reconocimiento Braille utilizando la tecnología de inteligencia artificial.

El proyecto debe contribuir al campo de la tecnología de asistencia para personas con discapacidad visual y tener el potencial para el desarrollo de sistemas de inteligencia artificial más avanzados y sofisticados para la lectura del sistema Braille en el futuro. Asimismo, debe ser viable y rentable para la organización, y la finalización y aprobación del estudio de viabilidad debe desencadenar el inicio de la fase de planificación. En vista de ello, debe tener una meta y unos objetivos claros, y la fase de planificación debe afinar los objetivos del proyecto, identificar

las actividades específicas y los recursos necesarios para completar el proyecto, y establecer un calendario que tenga en cuenta los recursos necesarios para todas las tareas.

1.1 FORMULACIÓN

Pregunta de investigación: ¿Cómo puede una herramienta IA ayudar en la lectura de caracteres Braille para el aprendizaje de este?

2 JUSTIFICACIÓN

El Braille, el cual es una habilidad de alfabetización crucial, no está ampliamente disponible para el público en general en la universidad, es de ahí que surge la necesidad de desarrollar un prototipo de inteligencia artificial para un sistema de lectura Braille. Además, la limitada accesibilidad se atribuye a la escasez de instructores calificados en Braille y a la producción limitada de materiales en Braille, factores que a su vez se deben a la falta de concienciación y recursos asignados a la educación en Braille en los sistemas educativos convencionales [6]. El prototipo de inteligencia artificial busca superar estas barreras, ofreciendo una solución innovadora que puede hacer que el aprendizaje del Braille sea más eficiente y asequible para aquellos interesados en adquirir esta valiosa habilidad de alfabetización.

Además de los beneficios y la relevancia del proyecto, la aplicación se enfocará en mejorar la accesibilidad del aprendizaje del braille para estudiantes hispanohablantes, ofreciéndoles una herramienta en su idioma nativo. La precisión de la aplicación se destacará al proporcionar traducciones precisas del braille al español en tiempo real, lo que contribuirá a una enseñanza más efectiva [7].

Asimismo, la aplicación abordará la terminología especializada, asegurando la correcta traducción de la jerga del sector y las terminologías especializadas para el alumno, enriqueciendo así su comprensión del braille. Además, se destacará la transferencia de habilidades, permitiendo a los estudiantes ciegos transferir habilidades y conceptos del braille español al braille inglés sin contracciones, similar a cómo un estudiante vidente transfiere habilidades de la alfabetización impresa del español al inglés [8]. Este enfoque integral de la aplicación busca facilitar el aprendizaje del braille en español, con la posibilidad de transferir estas habilidades al braille en inglés.

En general, una aplicación que traduce braille en tiempo real al español para apoyar el aprendizaje de braille en la Universidad Internacional del Trópico Americano, Unitropico, tiene el potencial de hacer que el aprendizaje de braille sea más accesible, preciso y completo para los estudiantes de habla hispana.

3 OBJETIVOS

3.1 OBJETIVO GENERAL.

Desarrollar un prototipo de aplicación apoyado por inteligencia artificial para la lectura y traducción en tiempo real de caracteres Braille.

3.2 OBJETIVOS ESPECÍFICOS.

- Analizar los retos, capacidades y limitaciones asociados a la traducción braille en tiempo real.
- Diseñar una arquitectura de software para el prototipo, teniendo en cuenta los parámetros obtenidos del análisis.
- Codificar el prototipo con base en la metodología y arquitectura diseñada.
- Validar la precisión y fiabilidad del prototipo a partir de pruebas en distintas configuraciones y condiciones visuales.

4 ESTADO DEL ARTE

Actualmente, diversos estudios indican que aprender Braille puede representar un desafío, con limitado acceso a la enseñanza de Braille [9], [10], [11].Factores que influyen incluyen la disponibilidad de instructores capacitados, las diferencias entre Braille y la alfabetización impresa, así como los retos asociados con la memorización de símbolos y contracciones Braille [9], [12], [13].

En este contexto, se han desarrollado herramientas en línea para facilitar el aprendizaje de Braille, como WeCapable's Spanish to Braille Translator [14] que convierte texto en español a Braille. Aplicaciones como Braille Academy: Play & Learn [15] emplean métodos únicos para enseñar Braille Inglés Unificado (UEB). Además, el Duxbury Braille Translator para Windows [4] ofrece traducción al español, entre otros idiomas, para los grados 1 y 2 de Braille también ha contribuido significativamente a este esfuerzo.

Estudios sobre la maximización del acceso y la motivación para aprender Braille incluyen investigaciones con entrevistas a personas ciegas [3]. Además, el dispositivo mecatrónico diseñado para facilitar el aprendizaje del Braille, especialmente dirigido a niños [16] , es una contribución significativa.

En paralelo, se están llevando a cabo diversas iniciativas globales para promover la alfabetización en Braille y la accesibilidad, así como la Fundación AIR en Colombia, la cual trabaja para la inclusión de la comunidad con discapacidad, incluyendo niños ciegos, en la educación [17] y se están implementando prácticas de aprendizaje digital inclusivo para estudiantes con discapacidad visual. [18].

Diversas tecnologías y aplicaciones buscan facilitar la vida de personas con discapacidad visual, tales como Microsoft's Seeing AI el cual proporciona información del entorno a través de comentarios hablados, utilizando inteligencia artificial y tecnología de cámara [19], el proyecto BrailleTouch de Georgia Tech, que explora el uso de pantallas táctiles para facilitar la introducción de datos en Braille [20], KNFB Reader, que convierte texto impreso en voz alta o Braille utilizando reconocimiento óptico de caracteres (OCR), y finalmente Be My Eyes, que conecta usuarios ciegos con voluntarios videntes, fomentando el apoyo comunitario [21].

5 MARCO REFERENCIAL

5.1 MARCO TEÓRICO

5.1.1 Aplicación móvil. Una aplicación de software diseñada para ejecutarse en dispositivos móviles como smartphones y tablets [22]. Para la implementación exitosa de este proyecto, resulta imperativo incorporar una aplicación móvil. A través de esta aplicación, los usuarios podrán acceder a un servicio basado en redes neuronales, proporcionándoles una interfaz gráfica intuitiva que mejore significativamente su experiencia de uso.

5.1.2 AI (Inteligencia Artificial). La inteligencia artificial es la capacidad de un sistema para interpretar datos externos, aprender de ellos y utilizar ese conocimiento para lograr tareas y objetivos específicos mediante una adaptación flexible [23]. En el contexto de la app, la IA se utiliza para reconocer e interpretar caracteres braille.

5.1.3 Braille. Sistema de escritura táctil utilizado por personas con discapacidad visual. El sistema Braille consiste en la utilización de seis puntos en relieve, distribuidos de formas diferentes, que permiten expresar letras, números e incluso signos, convirtiéndolo en un sistema sumamente completo.[24]

5.1.4 Reconocimiento de imágenes. El reconocimiento de imágenes es un proceso realizado mediante el uso de software de inteligencia artificial capaz de identificar y analizar imágenes utilizando complejos algoritmos matemáticos. La inteligencia artificial es capaz de identificar, analizar y comparar las matrices de bits que componen una imagen digital, con el objetivo de realizar alguna acción basada en la información obtenida [25]. En la app, el reconocimiento de imágenes se aplica para interpretar caracteres Braille a partir de imágenes captadas por la cámara.

5.1.5 Visión por ordenador. Campo de estudio que permite a los ordenadores interpretar y tomar decisiones basadas en datos visuales del mundo. Utiliza técnicas de visión por ordenador para procesar imágenes y extraer información de ellas [26].

5.1.6 OCR (Reconocimiento Óptico de Caracteres). Tecnología que convierte diferentes tipos de documentos, como documentos en papel escaneados, PDF o imágenes capturadas por una cámara digital, en datos editables y que permiten realizar búsquedas [25]. En este caso, el OCR se utiliza para reconocer y convertir caracteres Braille.

5.1.7 Traducción. El proceso de convertir texto o voz de un idioma a otro. En el contexto de la aplicación, la traducción se refiere a la conversión de caracteres Braille a un formato que pueda ser entendido por los usuarios, como texto hablado o escrito.

5.1.8 Accesibilidad. El diseño de productos, dispositivos, servicios o entornos accesibles para personas con discapacidad. La aplicación pretende mejorar la accesibilidad de las personas con discapacidad visual proporcionando una herramienta para leer Braille.

5.1.9 Interfaz de usuario (UI) y experiencia de usuario (UX). UI se refiere a los elementos visuales a través de los cuales los usuarios interactúan con la aplicación, mientras que UX abarca la experiencia general que un usuario tiene con la aplicación. Ambas son cruciales para garantizar que la aplicación sea fácil de usar y eficaz [27].

5.1.10 Aprendizaje automático. Un subconjunto de la IA que implica el desarrollo de algoritmos y modelos estadísticos que permiten a los ordenadores mejorar su rendimiento en una tarea específica a lo largo del tiempo [28]. La app puede utilizar el aprendizaje automático para mejorar su capacidad de reconocer con precisión los caracteres braille.

5.1.11 Aprendizaje profundo. Un tipo de aprendizaje automático que implica redes neuronales con múltiples capas (redes neuronales profundas) [29]. El aprendizaje profundo es particularmente eficaz en tareas como el reconocimiento de imágenes y el análisis de patrones, que son relevantes para la funcionalidad de la aplicación.

5.2 MARCO LEGAL

El lenguaje braille, es una de las herramientas más valiosas, para las personas con discapacidad visual, es lo que les permite poder leer un libro, un artículo, o cualquier otro tipo de texto, por sí mismos, es fundamental para la inclusión social de las personas con discapacidad visual.

En este capítulo, nos encontramos con las siguientes leyes que componen nuestro marco legal el marco legal.

5.2.1 Ley 2265 de 2022. Esta ley, promulgada el 24 de mayo de 2022, establece los principios y lineamientos para la promoción de la inclusión social de las personas con discapacidad en Colombia. En el Artículo 1, se define la inclusión social como el proceso de asegurar a las personas con discapacidad el acceso pleno y en igualdad de condiciones a todos los derechos y oportunidades, en la vida social, económica, cultural, política y demás esferas de la vida [30].

La Ley 2265 de 2022, en su Artículo 21, establece la obligación del Estado de garantizar el acceso a la información y la comunicación, incluyendo el uso de tecnologías de la información y la comunicación (TIC) accesibles. De esta manera, se reconoce la importancia de las TIC como herramientas para la inclusión social de las personas con discapacidad [30].

5.2.2 Ley 1346 de 2009. Esta ley, promulgada el 3 de junio de 2009, define la discapacidad como una condición particular que surge de la interacción entre las deficiencias físicas o mentales de una persona y los obstáculos que presenta el entorno, que limitan o restringen la capacidad de dicha persona para realizar actividades de la vida diaria [31]. La Ley 1346 de 2009 establece los derechos de las personas con discapacidad, incluyendo el derecho a la educación, al trabajo, a la salud y a la participación en la vida social.

5.2.3 Ley 1680 de 2013. Esta ley, promulgada el 29 de mayo de 2013, regula el uso y la prestación de servicios de tecnologías de la información y la comunicación en Colombia. En el Artículo 1, se establece como objetivo principal de la ley promover el desarrollo y la utilización de las tecnologías de la información y las comunicaciones en Colombia, en condiciones de igualdad, equidad, calidad, eficiencia, universalidad y responsabilidad social [32].

5.2.4 Decreto 1377 de 2013. Este decreto, expedido el 27 de junio de 2013, reglamenta la Ley 1680 de 2013 y establece los lineamientos específicos para garantizar la accesibilidad de las TIC. El Decreto 1377 de 2013 define los requisitos que deben cumplir las páginas web, los softwares y las aplicaciones móviles para ser accesibles a las personas con discapacidad [33].

5.3 CRONOGRAMA DE ACTIVIDADES

Tabla 1 - cronograma de actividades

ACTIVIDADES		Duración (meses)						
		1	2	3	4	5	6	7
Planificación	Definición de objetivos y funcionalidades de la aplicación							
	Diseño de la interfaz de usuario							
	Selección de la arquitectura de la red neuronal							
	Recopilación de datos para entrenar la red neuronal							
Desarrollo	Implementación de la interfaz de usuario							
	Entrenamiento de la red neuronal							
	Integración de la red neuronal con la aplicación							
	Pruebas y correcciones de errores							
Pruebas	Pruebas con usuarios con discapacidad visual							

	Recolección de feedback y sugerencias							
	Realización de ajustes y mejoras de la aplicación							

Fuente: Elaboración propia.

5.4 PRESUPUESTO

Tabla 2 - presupuesto

Costos de personal			
Desarrollador	$ 1,600,000	7 meses	$ 11,200,000
Desarrollador	$ 1,600,000	7 meses	$ 11,200,000
Diseñador gráfico	$ 1,400,000	1 meses	$ 1,400,000
		Subtotal	$ 23,800,000
Costos organizacionales			
Servicios públicos	$ 100,000	7 meses	$ 700,000
Internet	$ 80,000	7 meses	$ 560,000
Alquiler de equipos de computo	$ 600,000	7 meses	$ 4,200,000
		Subtotal	$ 5,460,000
Total			$ 29,260,000

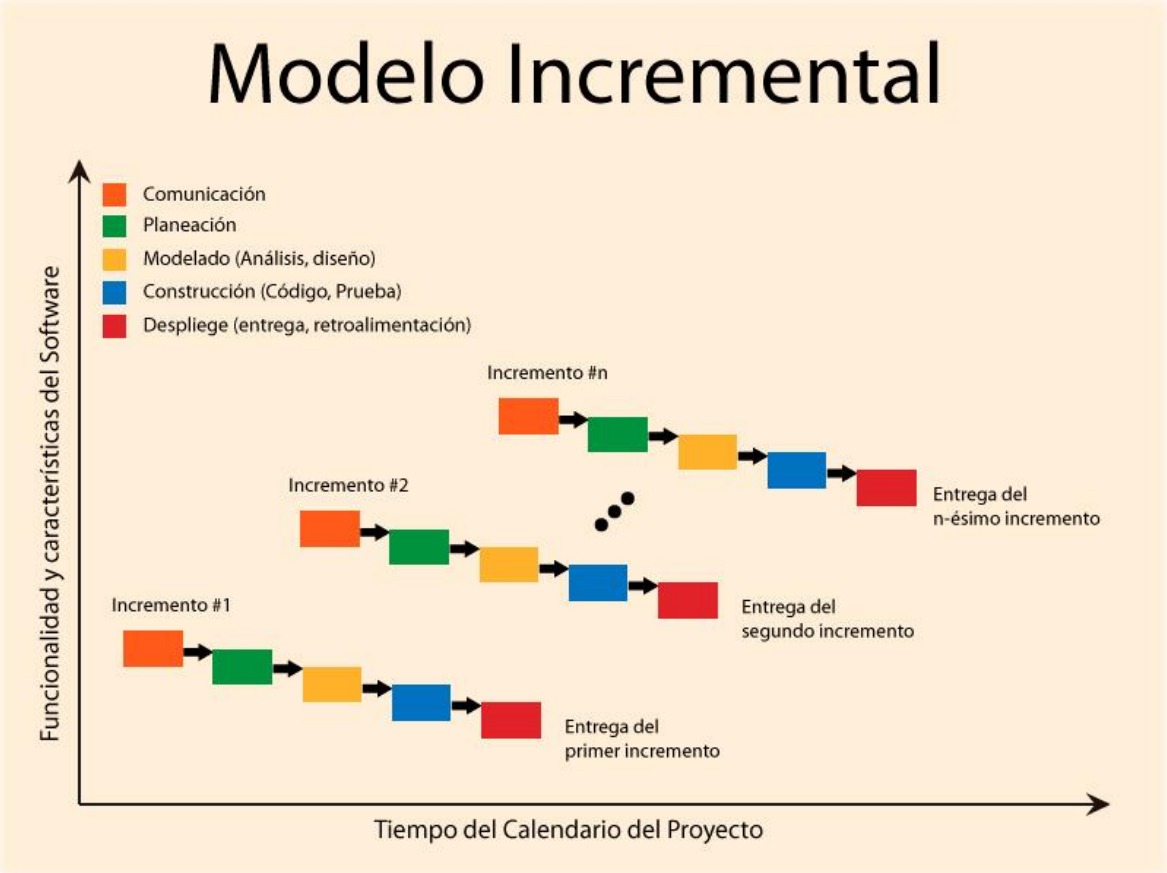
Fuente: Elaboración propia.

6 METODOLOGÍA

6.1 MODELO DE DESARROLLO

El modelo incremental es una metodología de desarrollo de software que puede utilizarse para construir sistemas grandes y complejos. Se basa en la idea de ir añadiendo nuevas características, o "incrementos", a un sistema existente en lugar de construirlo todo desde cero a la vez. El modelo incremental es un proceso iterativo que ayuda a identificar y corregir defectos en las primeras fases del proceso de desarrollo [34].

Ilustración 1 - Estructura modelo incremental



Tomado de: (isw-udistrital.blogspot.com)

En base a esta metodología, se dividió el desarrollo del proyecto en las siguientes fases para cada incremento:

- Análisis de requisitos (uno por modelo)
- Diseño
- Implementación
- Pruebas

- Integración
- Entrega

Fundamentalmente, el modelo de desarrollo consistió en tres etapas, la primera enfocada al desarrollo de la red neuronal, la segunda a la aplicación Android, y la tercera a la optimización del producto final.

6.1.1 Red neuronal

6.1.1.1 Fase 1: Desarrollo del modelo de red neuronal convolucional (CNN).

1. Análisis de requisitos
    - Para desarrollar un modelo de red neuronal para Android que reconozca los caracteres Braille, se consideraron los requisitos básicos de una red neuronal convolucional estándar.

    - Recopilación de datos: Es necesario recopilar un conjunto de datos de caracteres Braille. Lo ideal es que este conjunto de datos proceda de usuarios con discapacidad visual que utilicen dispositivos con pantalla táctil. El conjunto de datos debe incluir diferentes caracteres Braille en diferentes idiomas.

    - Preprocesamiento: Los datos recogidos deben preprocesarse para eliminar cualquier ruido o incoherencia. Esto puede incluir técnicas como la segmentación de imágenes, la mejora del contraste y la reducción del ruido.

    - Extracción de características: Las características relevantes deben extraerse de los datos preprocesados. Pueden incluir la posición y disposición de los puntos Braille, así como cualquier otro patrón o característica distintiva.

    - Selección del modelo: Debe seleccionarse una arquitectura de red neuronal adecuada para la tarea. Las redes neuronales convolucionales (CNN) han demostrado su eficacia en el reconocimiento de Braille. También pueden utilizarse técnicas de aprendizaje por transferencia para mejorar el rendimiento del modelo.

    - Entrenamiento y evaluación: El modelo seleccionado debe entrenarse en el conjunto de datos preprocesados y evaluarse utilizando métricas como la exactitud, la precisión, la recuperación y la puntuación F1. También puede ser necesario ajustar los hiperparámetros del modelo para optimizar su rendimiento.

- Integración en Android: Una vez entrenado y evaluado el modelo, es necesario integrarlo en una aplicación Android. Esta aplicación debe ser capaz de capturar imágenes de caracteres Braille, preprocesar las imágenes, extraer características y alimentarlas a la red neuronal para su reconocimiento.

- Interfaz de usuario: La aplicación debe tener una interfaz de usuario que permita a los usuarios con discapacidad visual escanear y reconocer fácilmente los caracteres Braille. La aplicación debe proporcionar una respuesta clara y audible al usuario.

- Accesibilidad: Asegúrese de que la aplicación es accesible para usuarios con discapacidad visual. Esto puede lograrse siguiendo las directrices de accesibilidad y las mejores prácticas.

- Pruebas y validación: La aplicación debe probarse a fondo con usuarios con discapacidad visual para garantizar que satisface sus necesidades y es fácil de usar. El rendimiento de la aplicación debe validarse utilizando escenarios reales y comentarios de los usuarios.

2. Diseño:
   - Definir la arquitectura de la red neuronal convolucional, incluyendo el número y tipo de capas, funciones de activación y optimizadores.
   - Diseñar la estructura de entrada y salida de la red para el reconocimiento de caracteres Braille.

3. Implementación:
   - Codificación del diseño del modelo en Python utilizando Tensor Flow, incluyendo la construcción de las capas y la definición de las operaciones de entrenamiento.
   - Configuración de los hiperparámetros del modelo y de la función de pérdida.

4. Pruebas:
   - Evaluar el modelo en un conjunto de datos de validación para medir su rendimiento inicial.
   - Ajustar los hiperparámetros y la arquitectura en función de los resultados de las pruebas.

5. Integración:
    - Integrar el modelo CNN en un entorno de desarrollo móvil Android para prepararlo para su uso en la aplicación móvil.

6. Entrega:
    - Preparación de una documentación detallada del modelo CNN, incluyendo su estructura, parámetros y uso.
    - Empaquetar el modelo en un formato compatible con Android para su distribución.

6.1.1.2 Fase 2: Desarrollo de la aplicación móvil Android.

1. Análisis de requisitos:
    - Acceso a la cámara: La aplicación necesita acceso a la cámara del dispositivo para capturar imágenes de caracteres Braille.
    - Modelo CNN: Se requiere un modelo de red neuronal convolucional (CNN) entrenado para reconocer caracteres Braille. Este modelo debe ser capaz de procesar imágenes del tamaño adecuado y mostrar el carácter Braille correspondiente.
    - Texto a voz (TTS): La aplicación debe incluir un motor TTS para leer en voz alta los caracteres Braille reconocidos.
    - Interfaz de usuario: Una interfaz fácil de usar que permita a los usuarios capturar imágenes y ver los caracteres Braille reconocidos y la salida TTS correspondiente.
    - Permisos: La aplicación debe solicitar los permisos necesarios, como acceso a la cámara y permisos de almacenamiento.
    - Compatibilidad: La aplicación debe ser compatible con una amplia gama de dispositivos Android y admitir la versión 23 de la API de Android (Android 6.0 Marshmallow) o superior.
    - TensorFlow Lite: La aplicación debe ser compatible con TensorFlow Lite para implementar y ejecutar el modelo CNN en dispositivos Android.

2. Diseño:
    - Diseño de la interfaz de usuario (UI) de la aplicación, incluyendo pantallas de captura de imágenes, resultados de reconocimiento y ajustes.
    - Diseño de la estructura y flujo de navegación de la aplicación.
3. Implementación:
    - Desarrollo de la aplicación móvil en Android Studio utilizando Java o Kotlin.
    - Integración de la funcionalidad de captura de imágenes y comunicación con el modelo CNN.
4. Pruebas:

- Pruebas unitarias para asegurar el correcto funcionamiento de cada componente de la aplicación.
- Pruebas de integración para verificar la interacción entre la aplicación y el modelo CNN.

5. Integración:
   - Integración de la aplicación con otros servicios y APIs, si es necesario (por ejemplo, servicios de texto a voz para la retroalimentación audible).

6. Despliegue:
   - Preparar la aplicación para su distribución a través de Google Play Store u otro canal de distribución pertinente.
   - Ponerla a disposición de los usuarios para su descarga y uso.

6.1.1.3 Fase 3: Desarrollo de la aplicación móvil Android

1. Análisis avanzado de requisitos:
   - Evaluación detallada de los requisitos existentes y de la funcionalidad actual del producto para identificar áreas de mejora y optimización.
   - Investigar nuevas tecnologías y técnicas que puedan mejorar la eficiencia, precisión y experiencia de usuario del producto final.

2. Diseño de mejoras y optimizaciones:
   - Diseñar mejoras en la arquitectura del modelo de red neuronal convolucional (CNN) y en la interfaz de usuario de la aplicación móvil para optimizar el rendimiento y la usabilidad.
   - Identificar áreas específicas de optimización, como la reducción del tiempo de procesamiento, la mejora de la precisión del reconocimiento y la optimización del consumo de recursos del dispositivo.

3. Implementar las mejoras y optimizaciones:
   - Implementar las mejoras diseñadas en el modelo CNN y la aplicación móvil utilizando las últimas técnicas y herramientas disponibles para mejorar la eficiencia y el rendimiento.
   - Incorporar técnicas de optimización de código y procesamiento de imágenes para reducir el tiempo de respuesta y la carga del dispositivo.

4. Pruebas avanzadas de rendimiento:
   - Realizar pruebas exhaustivas de rendimiento para evaluar el impacto de las mejoras en la velocidad, precisión y estabilidad del producto final.

- Utilizar herramientas de análisis del rendimiento para identificar cuellos de botella y áreas de mejora.

5. Integración final y despliegue:
   - Integrar las mejoras en el producto final, garantizando una transición fluida y sin fisuras desde la versión anterior.
   - Despliegue del producto final optimizado, teniendo cuidado de configurar y ajustar el entorno de producción para garantizar un rendimiento óptimo.

6. Seguimiento continuo y retroalimentación:
   - Establecer un proceso de supervisión continua para controlar el rendimiento del producto en producción y recabar opiniones de los usuarios.
   - Utilizar los comentarios de los usuarios y los datos de análisis para identificar nuevas oportunidades de optimización y orientar futuras iteraciones de desarrollo.

6.2 MÉTRICAS

6.2.1 Red neuronal

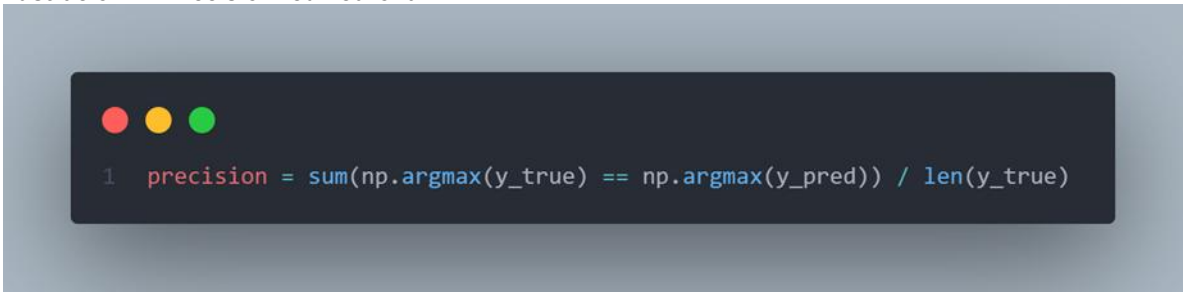
6.2.1.1 Métricas empleadas

En este apartado se describen las métricas utilizadas para evaluar y optimizar el rendimiento de la red neuronal convolucional durante el proceso de entrenamiento. La selección de estas métricas se basó en su relevancia para la tarea de clasificación de imágenes y su capacidad para proporcionar información útil sobre el comportamiento del modelo.

6.2.1.2 Precisión

La precisión se definió como la proporción de predicciones correctas realizadas por el modelo. Esta métrica fundamental se calcula mediante la siguiente fórmula:

Ilustración 2 - Precisión red neuronal



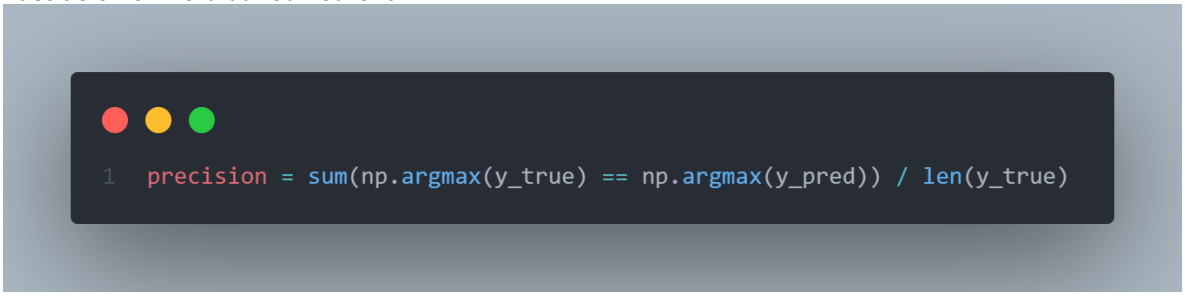
Tomado de: Elaboración propia

donde `y_true` y `y_pred` representan las etiquetas reales y las predicciones del modelo, respectivamente. La precisión proporciona una indicación directa del rendimiento del modelo y se considera una métrica esencial para la evaluación de su efectividad.

6.2.1.3 Pérdida

La función de pérdida utilizada en este trabajo fue la entropía cruzada categórica, la cual mide el error promedio del modelo al clasificar las imágenes. Esta métrica se calcula mediante la siguiente función:

Ilustración 3 - Pérdida red neuronal



Tomado de: Elaboración propia

La minimización de la pérdida durante el entrenamiento es crucial para mejorar el rendimiento del modelo en la tarea de clasificación.

6.2.1.4 Visualización del Historial de Entrenamiento

Para comprender mejor la evolución del rendimiento del modelo a lo largo del entrenamiento, se graficaron las curvas de precisión y pérdida en función de las épocas. Esto permitió observar la tendencia del modelo a mejorar su precisión y reducir su error con el paso del tiempo.

Ilustración 4 - Entrenamiento red neuronal



```python
# Importa matplotlib para crear las gráficas
import matplotlib.pyplot as plt

# Extrae las métricas del historial de entrenamiento
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']

# Crea las gráficas de precisión y pérdida
plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
plt.plot(acc, label='Precisión de Entrenamiento')
plt.plot(val_acc, label='Precisión de Validación')
plt.title('Precisión del Modelo')
plt.xlabel('Época')
plt.ylabel('Precisión')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(loss, label='Pérdida de Entrenamiento')
plt.plot(val_loss, label='Pérdida de Validación')
plt.title('Pérdida del Modelo')
plt.xlabel('Época')
plt.ylabel('Pérdida')
plt.legend()

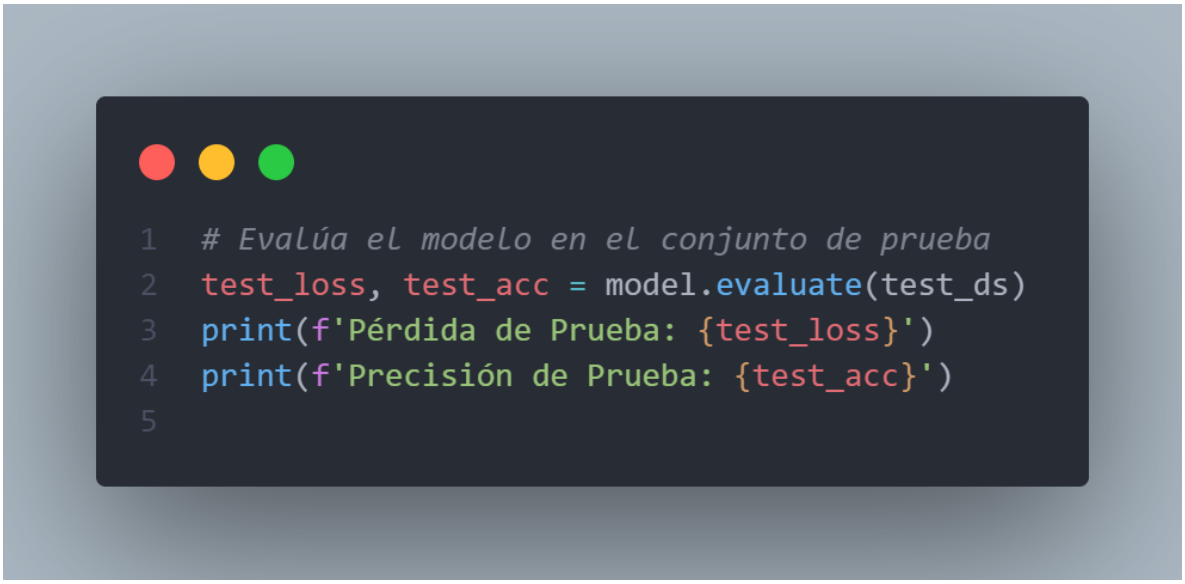
plt.tight_layout()
plt.show()
```

Tomado de: Elaboración propia

6.2.1.5 Evaluación en el Conjunto de Prueba

Para evaluar la capacidad del modelo para generalizar a datos nuevos y no vistos durante el entrenamiento, se calculó la precisión en un conjunto de prueba independiente. Esta métrica proporcionó una medida más realista del rendimiento del modelo en escenarios reales.

Ilustración 5 - Evaluación modelo



```
# Evalúa el modelo en el conjunto de prueba
test_loss, test_acc = model.evaluate(test_ds)
print(f'Pérdida de Prueba: {test_loss}')
print(f'Precisión de Prueba: {test_acc}')

```

Tomado de: Elaboración propia

6.2.1.6 Visualización de Predicciones

Para obtener una comprensión cualitativa del rendimiento del modelo, se visualizaron ejemplos de predicciones junto con las etiquetas reales de las imágenes. Esto permitió identificar posibles errores o sesgos del modelo y evaluar su capacidad para clasificar correctamente las imágenes.

Ilustración 6 - Visualización predicciones



```python
# Define una función para visualizar predicciones
def visualizar_predicciones(modelo, conjunto_datos_prueba, num_muestras=5):
    plt.figure(figsize=(15, 5))
    for i, (imágenes, etiquetas) in enumerate(conjunto_datos_prueba.take(num_muestras)):
        predicciones = modelo.predict(imágenes)
        clase_predicha = tf.argmax(predicciones, axis=1).numpy()
        clase_real = tf.argmax(etiquetas, axis=1).numpy()
        plt.subplot(1, num_muestras, i + 1)
        plt.imshow(imágenes[0])
        plt.title(f'Predicho: {clase_predicha[0]}, Real: {clase_real[0]}')
        plt.axis('off')
    plt.show()

# Visualiza predicciones con el conjunto de datos de prueba
visualizar_predicciones(modelo, test_ds)
```

Tomado de: Elaboración propia

6.2.1.7 Optimización del modelo

Se ha demostrado que el rendimiento de la red neuronal mejora con el tiempo a través de una serie de iteraciones. Esta mejora se ha logrado mediante el desarrollo y perfeccionamiento continuos de la arquitectura de la red, las capas, los hiperparámetros y otros factores. Cada sesión de entrenamiento se ha seguido y evaluado meticulosamente, lo que ha permitido comprender mejor el comportamiento del modelo y mejorar su capacidad de predicción.

La arquitectura de la red se ha sometido a varias iteraciones, incorporando técnicas y capas novedosas para mejorar su capacidad de aprender patrones complejos a partir de los datos. La arquitectura actual emplea una combinación de capas convolucionales separables, capas de agrupamiento máximo, agrupamiento máximo global y capas densamente conectadas, que culminan en una capa de salida softmax para la clasificación multiclase.

6.2.1.8 Ajuste de hiperparámetros:

La optimización de hiperparámetros como la tasa de aprendizaje, el tamaño del lote, la intensidad de la regularización y las funciones de activación ha sido fundamental para mejorar la generalización y el rendimiento del modelo en datos no observados. Mediante la experimentación y validación sistemáticas, se han identificado las configuraciones óptimas de los hiperparámetros, lo que ha contribuido a mejorar la convergencia del entrenamiento y a reducir el sobreajuste.

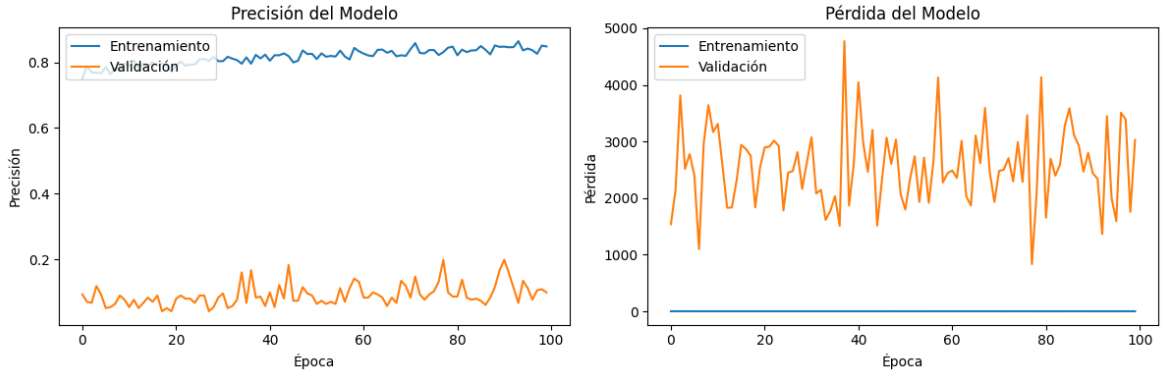
6.2.1.9 Estrategias de aumento de datos:
Se han integrado en el proceso de formación técnicas de aumento de datos, como la rotación y el cizallamiento, para aumentar la diversidad del conjunto de datos de formación. Este aumento ayuda al modelo a generalizar mejor las variaciones en los datos de entrada, mejorando así su robustez y reduciendo el riesgo de sobreajuste.

A lo largo del proceso de entrenamiento, se ha realizado un seguimiento exhaustivo de las métricas de entrenamiento y validación para evaluar el progreso del modelo e identificar posibles áreas de mejora. Las representaciones gráficas del historial de entrenamiento, incluidas las curvas de precisión y pérdida, han sido fundamentales para visualizar la trayectoria de rendimiento del modelo y orientar las iteraciones posteriores.

6.2.1.10 Cuantificación para optimizar el despliegue:
Como preparación para su despliegue en entornos con recursos limitados, como los dispositivos móviles o los dispositivos periféricos, el modelo se ha cuantificado con el fin de reducir su huella de memoria y su complejidad computacional, preservando al mismo tiempo la precisión de la inferencia. Esto se ha logrado aprovechando técnicas como la conversión y optimización de TensorFlow Lite, que han permitido transformar el modelo en una representación compacta y eficiente sin comprometer su rendimiento predictivo.

Primera Versión
Precisión y perdida del modelo por época durante procesos de entrenamiento y validación

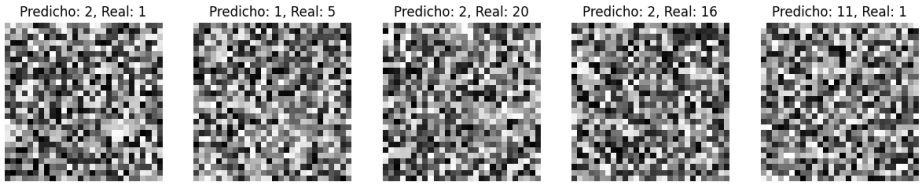
Ilustración 7 - Precisión y perdida N° 1



Tomado de: Elaboración propia

Pruebas de precisión con predicciones usando muestras de datos de prueba:

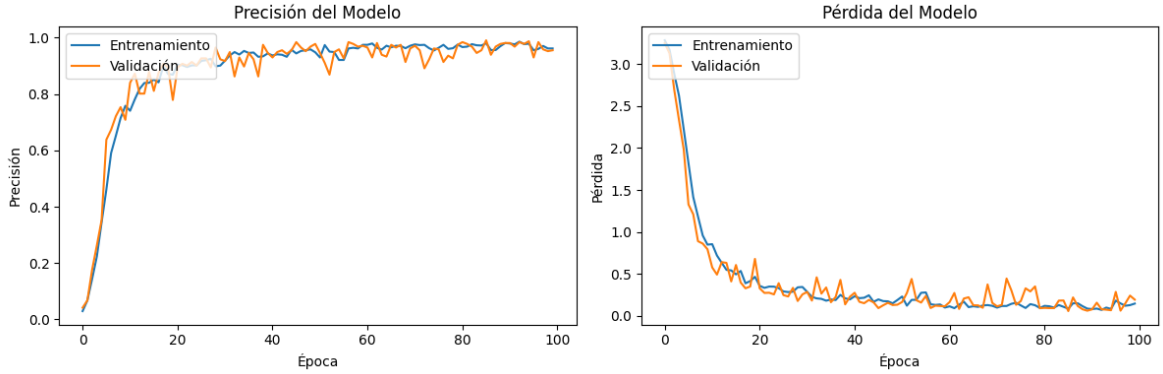
Ilustración 8 - Prueba predicciones N°1



Tomado de: Elaboración propia

Versión Final
Precisión y perdida del modelo por época durante procesos de entrenamiento y validación.

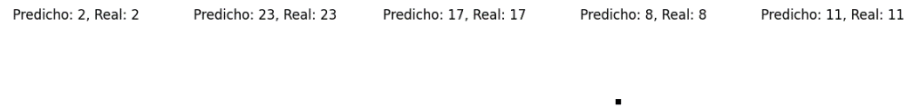
Ilustración 9 - Precisión y Pérdida N°2



Tomado de: Elaboración propia

Pruebas de precisión con predicciones usando muestras de datos de prueba:

Ilustración 10 - Prueba predicciones N°2



Tomado de: Elaboración propia

6.3 VARIABLES

El proyecto también depende de variables comunes al desarrollo de un proyecto de software, tales como:

- Tiempo: la cantidad de tiempo necesaria para completar el proyecto, incluido el tiempo necesario para la investigación, el desarrollo, las pruebas y la revisión.
- Coste: La cantidad de dinero necesaria para desarrollar y mantener la aplicación, incluidos los costes de diseño, desarrollo, pruebas, marketing y alojamiento.
- Calidad: El nivel de calidad que se espera de la aplicación: facilidad de uso, eficacia, capacidad de respuesta, alto rendimiento y seguridad.
- Alcance: Los requisitos especificados para el resultado final, incluida la definición de lo que se supone que debe lograr el proyecto y una descripción específica de lo que debe ser o lograr el resultado final.
- Recursos: Las habilidades, herramientas y equipos necesarios para completar el proyecto, incluida la selección de la plataforma adecuada, la elección de la tecnología y la disponibilidad de las habilidades adecuadas en el momento oportuno.
- Necesidades de los usuarios: Las necesidades y preferencias del público objetivo, incluido el contexto en el que se utilizará la app, el tipo de dispositivo y sistema operativo que utilizan los usuarios, y el tipo de datos y utilidades que necesitan los usuarios.

Además, en el contexto del proyecto, algunas posibles variables son la precisión y la velocidad de la traducción al braille, la compatibilidad con diferentes sistemas braille, la facilidad de uso y la accesibilidad para los usuarios con discapacidad visual, y la eficacia de la aplicación para apoyar el aprendizaje del braille en la Universidad Internacional del Trópico Americano.

6.4 REQUERIMIENTOS

REQURIMIENTOS PROTOTIPO		
Requerimiento	Descripción	Actividad
Cámara	La aplicación usa la cámara del dispositivo para la lectura del sistema de escritura braille.	El dispositivo debe pedir permiso para usar la cámara. Solamente requiere este permiso la primera vez que se use la aplicación.
Interfaz	La aplicación cuenta con una interfaz gráfica intuitiva.	La interfaz de diseño se llevó a cabo priorizando la simplicidad, con el objetivo de crear una experiencia intuitiva para el usuario.
Procesamiento datos	La aplicación tiene la capacidad de traducir lo registrado con la cámara	La aplicación usa una red neuronal, para el procesamiento de los datos registrados con la cámara.
Salida de traducción	La aplicación muestra de manera escrita y por medio de audio la traducción realizada por la red neuronal	La aplicación muestra de manera escrita en un recuadro la traducción realizada por la red neuronal, y también reproduce la traducción en audio.

7 ENTORNO DE DESARROLLO

El desarrollo de esta aplicación se usó el editor de código Visual Studio Code, el cual es uno de los editores de código más populares desarrollado por Microsoft.

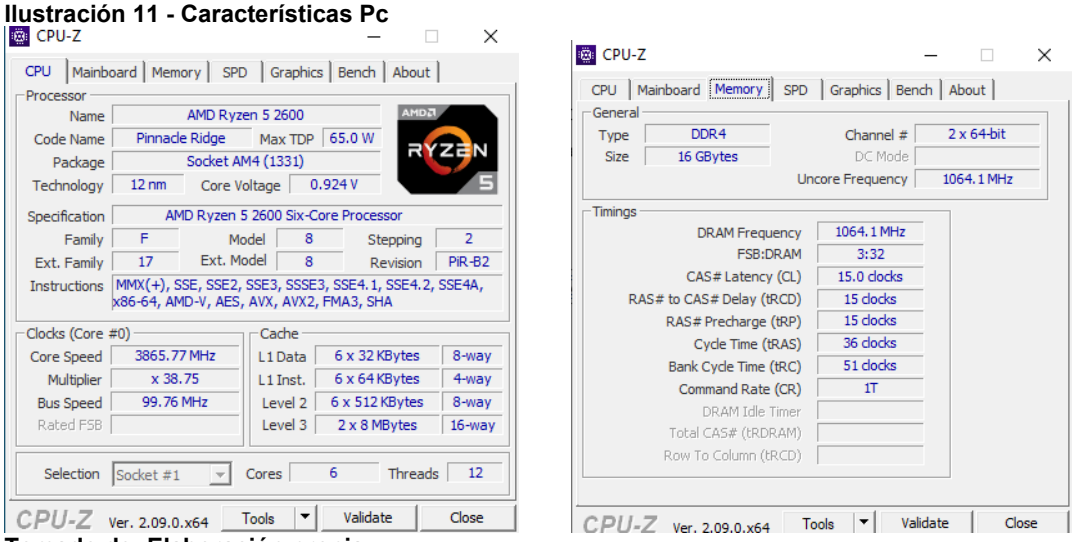
7.1 REQUISITOS PREVIOS

7.1.1 Visual Studio Code. Está disponible para Windows, Linux y macOS

7.1.2 Hardware. En este proyecto se usa un equipo con las siguientes características

- Procesador AMD Ryzen 5
- 16 GB de memoria RAM
- 1.5 TB de almacenamiento

Ilustración 11 - Características Pc



Tomado de: Elaboración propia

7.1.3 Python. la version que se usó en este proyecto es 3.9.13

7.1.4 Android. Android studio iguana 2023.2.1

8 RED NEURONAL

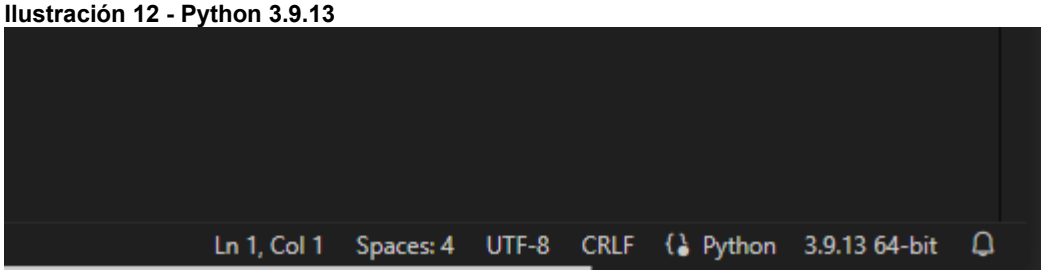
8.1 IMPLEMENTACIÓN DE RED NEURONAL SECUENCIAL

Este proyecto propone utilizar redes neuronales convolucionales (CNNs) para la traducción del lenguaje braille. Las CNNs son más adecuadas para el procesamiento de imágenes, como las representaciones del braille, y han demostrado ser muy eficientes en tareas de reconocimiento de patrones y clasificación, además, ofrecen ventajas como la captura de patrones espaciales, la consideración del contexto local, la robustez a variaciones y la eficiencia computacional, haciéndolas una opción prometedora para la traducción del braille.

8.2 MODELADO DE ARQUITECTURA Y SERVICIOS

Lenguaje de programación que uso en el proyecto en la parte de la red neuronal se trabaja con Python en su versión 3.9.13, cómo se observa en la siguiente imagen

Ilustración 12 - Python 3.9.13



Tomado de: Elaboración propia

8.2.1 Requerimientos de librerías. Para llevar a cabo la implementación de modelos de machine learning y el desarrollo del prototipo, fue esencial contar con un conjunto de librerías, que nos proporcionan las herramientas necesarias para la realización de tareas específicas. Estas librerías no solo facilitaron la construcción y entrenamiento de los modelos, sino que también el procesamiento de los datos. A continuación, se detallan las librerías usadas.

Ilustración 13 - librerías

```python
# Tomado de https://www.kaggle.com/code/faiyazabdullah/braille-image-classifier-using-cnn

# Importar TensorFlow para construir y entrenar el modelo
# Requiere pip install tensorflow==2.10.0
import tensorflow as tf

# Importar capas de Keras para la arquitectura del modelo
from keras import layers

# Importar Path para trabajar con rutas de archivos
from pathlib import Path

# Especificar la ruta al conjunto de datos de imágenes
data_dir = Path("images")

# Importar backend de Keras, matplotlib y otras utilidades
from keras import backend as K
from keras.regularizers import l2
import matplotlib.pyplot as plt

# Definir dimensiones de imagen y tamaño de lote
image_size = (28, 28)
batch_size = 32
```

Tomado de: Elaboración propia

Estructura de código para generación de datos

Ilustración 14 - Estructura de código

```python
# Definir dimensiones de imagen y tamaño de lote
image_size = (28, 28)
batch_size = 32
image_height = 28
image_width = 28
num_classes = 26  # Número de clases a predecir

# Crear un generador de datos para los datos de entrenamiento con aumento
# GENERADOR COMPLEJO
# train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
#     rescale=1./255,  # Normalizar los valores de los píxeles a 0-1
#     rotation_range=20,  # Girar aleatoriamente las imágenes hasta 20 grados
#     zoom_range=0.2,  # Ampliar aleatoriamente las imágenes hasta un 20%
#     horizontal_flip=True  # Voltear aleatoriamente las imágenes horizontalmente
# )

# GENERADOR KAGGLE
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(rotation_range=20,
                             shear_range=10,
                             validation_split=0.2)

# Cargar imágenes de entrenamiento desde el directorio utilizando el generador de datos
train_ds = train_datagen.flow_from_directory(
    data_dir,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='categorical',  # Etiquetas codificadas en uno a uno
    subset='training',  # Usar solo el subconjunto de entrenamiento
    seed=456  # Establecer una semilla para la reproducibilidad
)

# Cargar imágenes de validación directamente desde el directorio
validation_ds = tf.keras.preprocessing.image_dataset_from_directory(
    data_dir,
    subset='validation',  # Usar solo el subconjunto de validación
    validation_split=0.2,  # Usar el 20% de los datos para la validación
    seed=456,  # Misma semilla para la consistencia
    labels="inferred",  # Inferir etiquetas de la estructura del directorio
    label_mode='categorical',  # Etiquetas codificadas en uno a uno
    image_size=image_size,
    batch_size=batch_size
)
```

Tomado de: Elaboración propia

Ilustración 15 - Estructura de código

```
#MODELO KAGGLE
model = tf.keras.Sequential([
    layers.SeparableConv2D(64,(3,3),activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.SeparableConv2D(128,(3,3),activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.SeparableConv2D(256,(2,2),activation='relu'),
    layers.GlobalMaxPooling2D(),
    layers.Dense(256),
    layers.LeakyReLU(),
    layers.Dense(64,kernel_regularizer=l2(2e-4)),
    layers.LeakyReLU(),
    layers.Dense(26,activation='softmax'),
    ])

# Compilar el modelo
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Entrenar el modelo en el conjunto de datos de entrenamiento, validando en el conjunto de datos de validación
# model.fit(train_ds, epochs=100, validation_data=validation_ds)

def plot_training_history(history):
    plt.figure(figsize=(12, 4))
    # Plotear valores de precisión de entrenamiento y validación
    plt.subplot(1, 2, 1)
    plt.plot(history.history['accuracy'])
    plt.plot(history.history['val_accuracy'])
    plt.title('Precisión del Modelo')
    plt.xlabel('Época')
    plt.ylabel('Precisión')
    plt.legend(['Entrenamiento', 'Validación'], loc='upper left')
    # Plotear valores de pérdida de entrenamiento y validación
    plt.subplot(1, 2, 2)
    plt.plot(history.history['loss'])
    plt.plot(history.history['val_loss'])
    plt.title('Pérdida del Modelo')
    plt.xlabel('Época')
    plt.ylabel('Pérdida')
    plt.legend(['Entrenamiento', 'Validación'], loc='upper left')
    plt.tight_layout()
    plt.show()
```

Tomado de: Elaboración propia

9 APLICACIÓN ANDROID

9.1 CAMERA FRAGMENT

El archivo CameraFragment.kt define la funcionalidad principal de la aplicación. Es responsable de:

9.1.1 Inicializar la cámara.

- Solicita los permisos necesarios para acceder a la cámara del dispositivo.
- Utiliza la librería ProcessCameraX para configurar la cámara y los casos de uso (vista previa y análisis de imagen).
- Define un analizador de imágenes que procesa cada frame recibido de la cámara para detectar objetos en tiempo real.

9.1.2 Configurar la detección de objetos.

- Crea una instancia de la clase `ObjectDetectorHelper` que se encarga de la detección de objetos utilizando TensorFlow Lite.
- Establece los parámetros de detección como el número máximo de objetos a detectar, el umbral de confianza y el número de hilos de procesamiento.

9.1.3 Procesar los resultados de la detección.

- El analizador de imágenes llama a la función `detectObjects` cada vez que recibe un nuevo frame de la cámara.
- Esta función a su vez llama a la clase `ObjectDetectorHelper` para procesar el frame y detectar objetos.
- Los resultados de la detección (lista de objetos detectados) se envían de vuelta al `CameraFragment` a través del callback `onResults`

9.1.4 Actualizar la interfaz de usuario (UI).

- El método `onResults` se encarga de actualizar la UI en el hilo principal de la aplicación.
- Muestra el tiempo de inferencia (tiempo que tarda el modelo en detectar objetos) y la lista de objetos detectados con su etiqueta.
- Utiliza la clase `OverlayView` para dibujar cuadros delimitadores alrededor de los objetos detectados en la vista previa de la cámara.

9.1.5 Manejar errores.

- El callback `onError` se llama si ocurre algún error durante la detección de objetos.
- Muestra un mensaje de error en la pantalla.

9.1.6 Control TTS (Text-To-Speech).

- Inicializa la funcionalidad Text-To-Speech para convertir texto a voz.
- Cuando se presiona el botón de lectura (captureButton), se llama a la función `captureButtonPressed` para obtener las predicciones y hablarlas en voz alta si están disponibles.

Ilustración 16 - CameraFragment

README.md | build.gradle.kts (:app) | MainActivity.kt | CameraFragment.kt

```
19
20    import ...
48
      Julio Alejandro Rivera Rodriguez
49    class CameraFragment : Fragment(), ObjectDetectorHelper.DetectorListener {
50
51        private val TAG = "ObjectDetection"
52
53        private var _fragmentCameraBinding: FragmentCameraBinding? = null
54
          Julio Alejandro Rivera Rodriguez
55        private val fragmentCameraBinding
56            get() = _fragmentCameraBinding!!
57
58        private lateinit var objectDetectorHelper: ObjectDetectorHelper
59        private lateinit var bitmapBuffer: Bitmap
60        private var preview: Preview? = null
61        private var imageAnalyzer: ImageAnalysis? = null
62        private var camera: Camera? = null
63        private var cameraProvider: ProcessCameraProvider? = null
64
65        /** Blocking camera operations are performed using this executor */
66        private lateinit var cameraExecutor: ExecutorService
67
          Julio Alejandro Rivera Rodriguez
68        override fun onResume() {...}
77
          Julio Alejandro Rivera Rodriguez
78        override fun onDestroyView() {...}
85
          Julio Alejandro Rivera Rodriguez
86        override fun onCreateView(
87            inflater: LayoutInflater,
88            container: ViewGroup?,
89            savedInstanceState: Bundle?
90        ): View {...}
95
          Julio Alejandro Rivera Rodriguez
96        @SuppressLint("MissingPermission")
97        override fun onViewCreated(view: View, savedInstanceState: Bundle?) {...}
116
          Julio Alejandro Rivera Rodriguez
```

Tomado de: Elaboración propia

9.2 PERMISSIONS FRAGMENT

El archivo PermissionsFragment.kt se encarga de solicitar los permisos necesarios para que la aplicación funcione. En este caso, el único permiso requerido es el acceso a la cámara (Manifest.permission.CAMERA).

9.2.1 Definición de permisos.

- Se define una constante `PERMISSIONS_REQUIRED` que contiene un arreglo con el permiso necesario (cámara).

9.2.2 Solicitud de permisos.

- El fragmento utiliza el registro `registerForActivityResult` para lanzar un diálogo de solicitud de permiso.
- Cuando el usuario responde al diálogo, el callback `requestPermissionLauncher` se ejecuta.

9.2.3 Respuesta del usuario.

- Si el usuario concede el permiso, se muestra un mensaje de confirmación y se llama al método `navigateToCamera` para navegar al fragmento de la cámara (CameraFragment).
- Si el usuario deniega el permiso, se muestra un mensaje indicando que el permiso ha sido denegado.

9.2.4 Verificación de permisos.

- En el método `onCreate`, se verifica si el permiso de cámara ya está concedido utilizando `ContextCompat.checkSelfPermission`.
- Si el permiso está concedido, se navega directamente al fragmento de la cámara.
- Si el permiso no está concedido, se solicita el permiso utilizando `requestPermissionLauncher`.

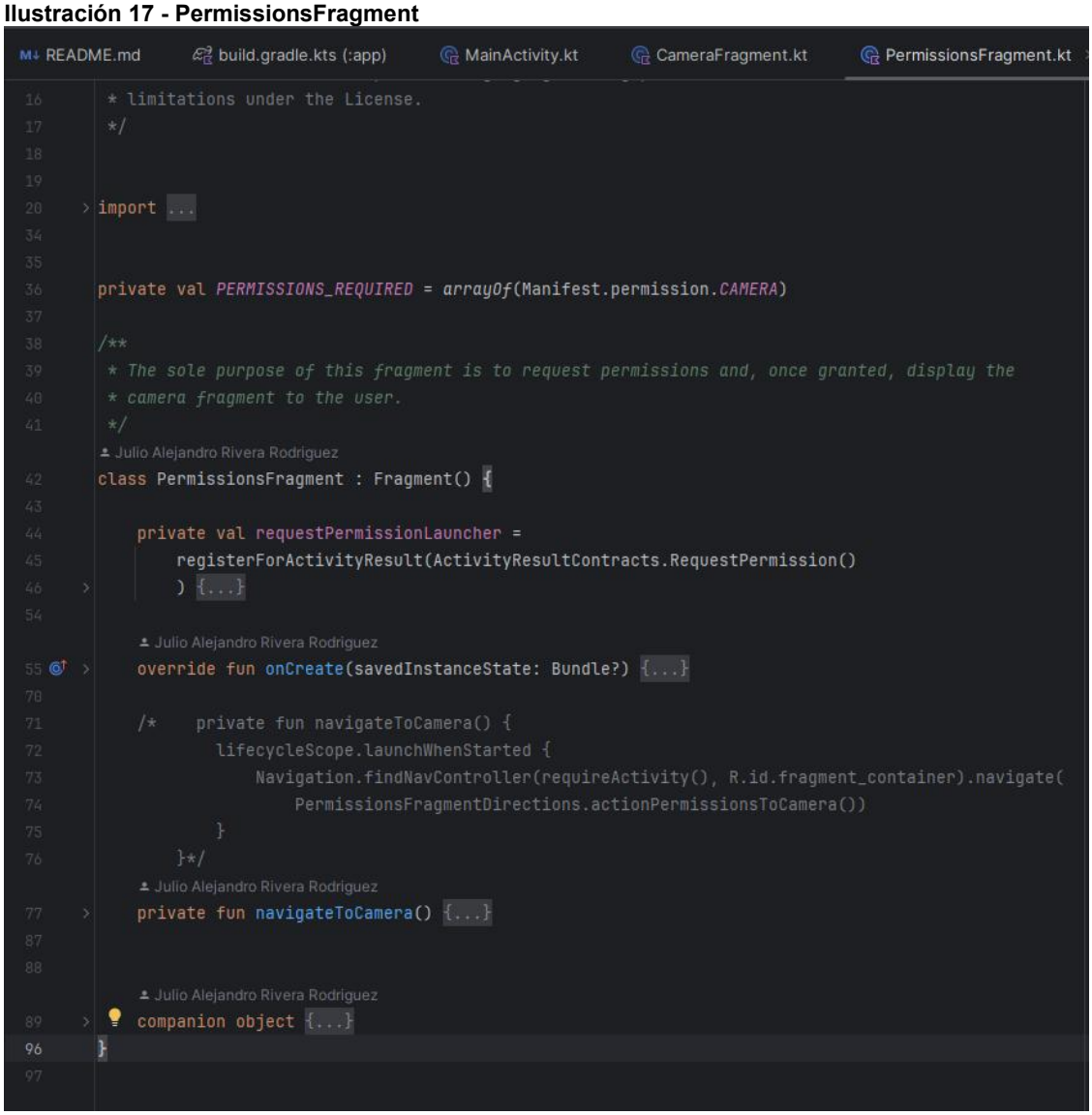
9.2.5 Navegación al fragmento de la cámara.

- El método `navigateToCamera` utiliza `lifecycleScope` para lanzar un coroutine dentro del ciclo de vida del fragmento.
- Se utiliza `repeatOnLifecycle` para repetir la lógica de navegación solo cuando el fragmento se encuentre en estado `STARTED`.
- Dentro del bloque `repeatOnLifecycle`, se utiliza `Navigation.findNavController` para navegar a la acción `actionPermissionsToCamera` definida en el archivo de navegación.

9.2.6 Método `hasPermissions`.

- Este método estático (companion object) permite verificar desde cualquier parte de la aplicación si todos los permisos requeridos están concedidos.
- Recibe un contexto como parámetro e itera sobre el arreglo `PERMISSIONS_REQUIRED` para verificar cada permiso utilizando `ContextCompat.checkSelfPermission`.

Ilustración 17 - PermissionsFragment



Tomado de: Elaboración propia

En resumen, el fragmento PermissionsFragment garantiza que la aplicación tenga los permisos necesarios antes de navegar al fragmento CameraFragment, que es el responsable de la detección de objetos en tiempo real.

9.3 MAIN ACTIVITY

El archivo MainActivity.kt es el punto de entrada principal de la aplicación.

9.3.1 Patrón de actividad única.

- La aplicación sigue el patrón de actividad única, lo que significa que solo tiene una actividad principal (MainActivity).

- Toda la funcionalidad de la aplicación se implementa utilizando fragmentos.

9.3.2 Enlace de la vista de layout.

- En el método `onCreate`, se utiliza `ActivityMainBinding` para inflar el archivo de layout `activity_main.xml`.
- La propiedad `activityMainBinding` proporciona acceso a los elementos del layout.
- Se establece la vista raíz del layout (`activityMainBinding.root`) como contenido de la actividad.

9.3.3 Comportamiento del botón retroceso.

- El método `onBackPressed` se encarga de manejar la acción del botón retroceso del dispositivo.
- En Android Q (API nivel 29) se implementa una solución temporal para un problema conocido de fuga de memoria.
- Si la versión de Android es Q, se llama a `finishAfterTransition` para finalizar la actividad después de la animación de transición.
- Para otras versiones de Android, se llama al método `super.onBackPressed` del padre para manejar el comportamiento predeterminado.

Ilustración 18 - MainActivity

README.md | build.gradle.kts (:app) | MainActivity.kt

```
28      * functionality is implemented in the form of fragments.
29      */
        Julio Alejandro Rivera Rodriguez
30      class MainActivity : AppCompatActivity() {
31
32          private lateinit var activityMainBinding: ActivityMainBinding
33
            Julio Alejandro Rivera Rodriguez
34          override fun onCreate(savedInstanceState: Bundle?) {
35              super.onCreate(savedInstanceState)
36              activityMainBinding = ActivityMainBinding.inflate(layoutInflater)
37              setContentView(activityMainBinding.root)
38          }
39
            Julio Alejandro Rivera Rodriguez
40          override fun onBackPressed() {
41              if (Build.VERSION.SDK_INT == Build.VERSION_CODES.Q) {
42                  // Workaround for Android Q memory leak issue in IRequestFinishCallback$Stub.
43                  // (https://issuetracker.google.com/issues/139738913)
44                  finishAfterTransition()
45              } else {
46                  super.onBackPressed()
47              }
48          }
49      }
50
```

Tomado de: Elaboración propia

En resumen, MainActivity es una actividad simple que infla el layout principal y delega toda la funcionalidad de la aplicación a los fragmentos. Es un ejemplo del patrón de actividad única, donde la lógica de la UI se implementa en fragmentos independientes y reutilizables.

9.4 OBJECT DETECTOR HELPER

ObjectDetectorHelper.kt es una clase Kotlin que actúa como un ayudante para el detector de objetos en una aplicación Android que utiliza TensorFlow Lite para la detección de objetos en tiempo real. Su funcionalidad se desglosa de la siguiente forma:

9.4.1 Paquete e importaciones.

- El código comienza con el paquete `com.example.deteccionbraille` y las importaciones necesarias, incluidas las clases de Android y las clases de TensorFlow Lite.

9.4.2 Clase ObjectDetectorHelper.

- Esta clase encapsula la lógica para configurar y utilizar un detector de objetos.
- Los parámetros del constructor incluyen el umbral de confianza, el número de hilos, el número máximo de resultados, el delegado actual, el modelo actual, el contexto de la aplicación y un objeto de escucha para eventos de detección.

9.4.3 Propiedades.

- Las propiedades como `threshold`, `numThreads`, `maxResults`, `currentDelegate`, `currentModel`, `context` y `objectDetectorListener` son inicializadas en el constructor de la clase.

9.4.4 ObjectDetector.

- Se declara una variable `objectDetector` que se utilizará para realizar la detección de objetos. Esta variable se inicializa en el método `setupObjectDetector()`.

9.4.5 Método.

- `clearObjectDetector()`: Este método se utiliza para anular la referencia al detector de objetos, permitiendo que sea recogido por el recolector de basura.
- `setupObjectDetector()`: Configura el detector de objetos utilizando las opciones actuales como el umbral de puntuación, el número de hilos y el tipo de delegado (CPU, GPU, NNAPI).
- `detect(Bitmap image, int imageRotation)`: Realiza la detección de objetos en la imagen dada. Utiliza un preprocesador para ajustar la imagen y luego pasa la imagen al detector de objetos. Después de la detección, notifica al `objectDetectorListener` con los resultados.

9.4.6 Interfaz DetectorListener.

- Define métodos que serán llamados para manejar errores y resultados de detección.

9.4.7 Constantes.

- La clase contiene constantes estáticas que definen los diferentes tipos de delegados y modelos de detección disponibles.

Ilustración 19 - ObjectDetectorHelper



Tomado de: Elaboración propia

En resumen, esta clase encapsula la lógica para configurar y utilizar un detector de objetos en una aplicación Android, permitiendo la detección en tiempo real de objetos en imágenes utilizando TensorFlow Lite.

9.5 OVERLAY VIEW

OverlayView.kt es una clase Kotlin que actúa como una vista superpuesta en la aplicación Android. Esta vista se utiliza para dibujar cuadros delimitadores alrededor de los objetos detectados y mostrar etiquetas de texto asociadas con esos objetos.

9.5.1 Clase OverlayView.

- Esta clase extiende la clase `View` de Android y se utiliza para dibujar cuadros delimitadores y etiquetas de texto en la pantalla.
- Contiene una lista de detecciones (`results`) y los pintores (`boxPaint`, `textBackgroundPaint`, `textPaint`) utilizados para dibujar los elementos en la vista.
- La propiedad `scaleFactor` se utiliza para escalar los cuadros delimitadores según las dimensiones de la vista y las dimensiones de la imagen original.

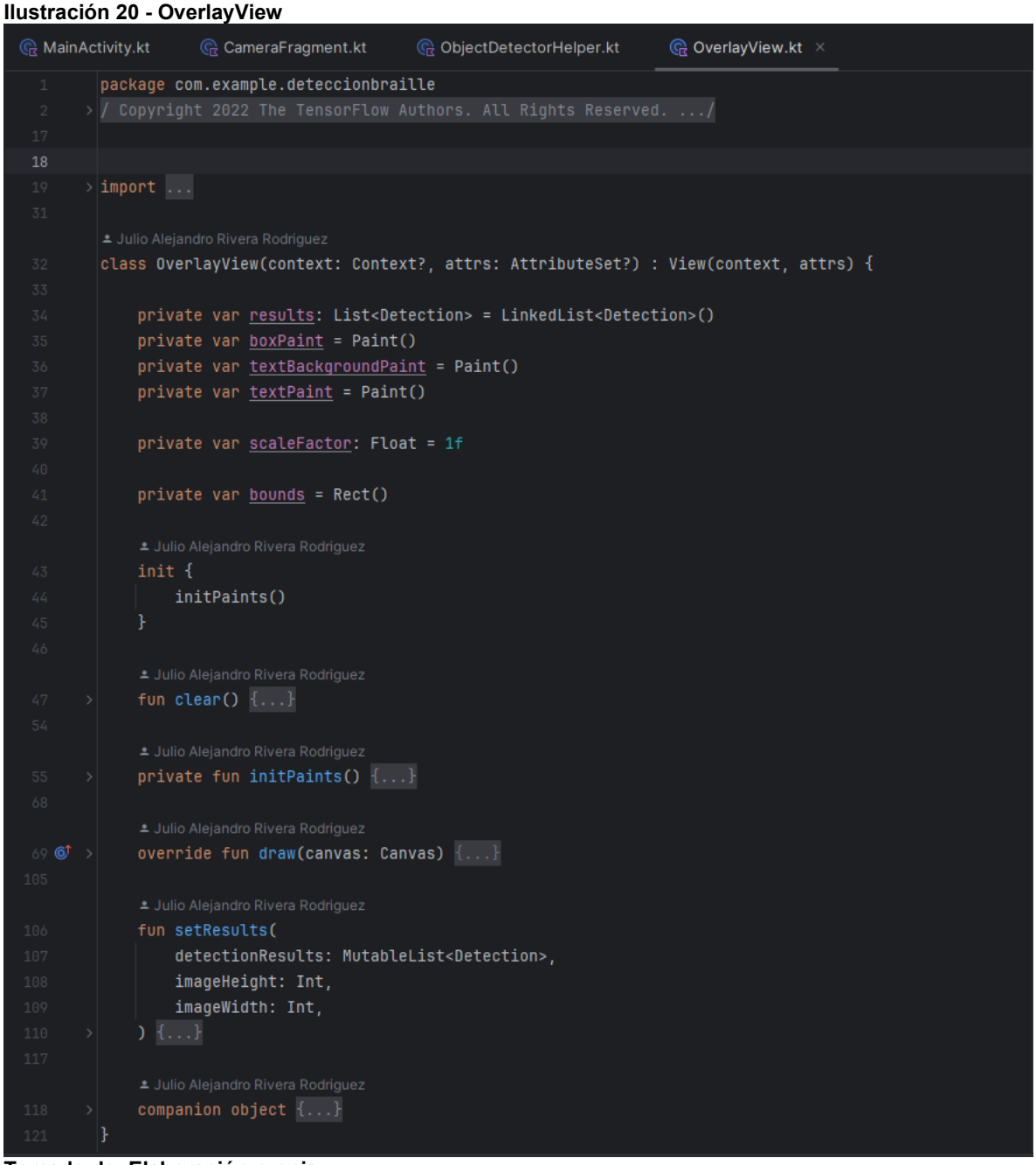
9.5.2 Métodos.

- `initPaints()`: Inicializa los pintores utilizados para dibujar los cuadros delimitadores y el texto.
- `draw (canvas: Canvas) `: Este método se invoca para dibujar los elementos en la vista. Itera sobre la lista de resultados de detección y dibuja un cuadro delimitador y un texto asociado para cada detección.
- `setResults (detectionResults: MutableList<Detection>, imageHeight: Int, imageWidth: Int) `: Este método se utiliza para establecer los resultados de la detección de objetos. Calcula el factor de escala basado en las dimensiones de la vista y la imagen original.
- `clear () `: Restablece los pintores y la vista, invalidando cualquier dibujo previo.

9.5.3 Compañero del objeto.

- Contiene una constante `BOUNDING_RECT_TEXT_PADDING` que representa el relleno alrededor del texto en el cuadro delimitador.

Ilustración 20 - OverlayView



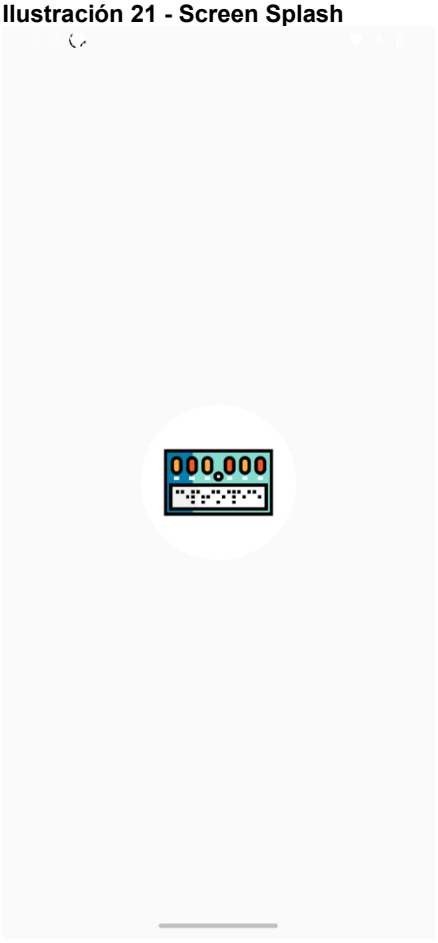
Tomado de: Elaboración propia

En resumen, esta clase se utiliza para visualizar los resultados de la detección de objetos en una vista superpuesta en la pantalla de la aplicación Android. Los resultados incluyen cuadros delimitadores alrededor de los objetos detectados y etiquetas de texto asociadas con esos objetos.

10 INTERFAZ GRAFICA

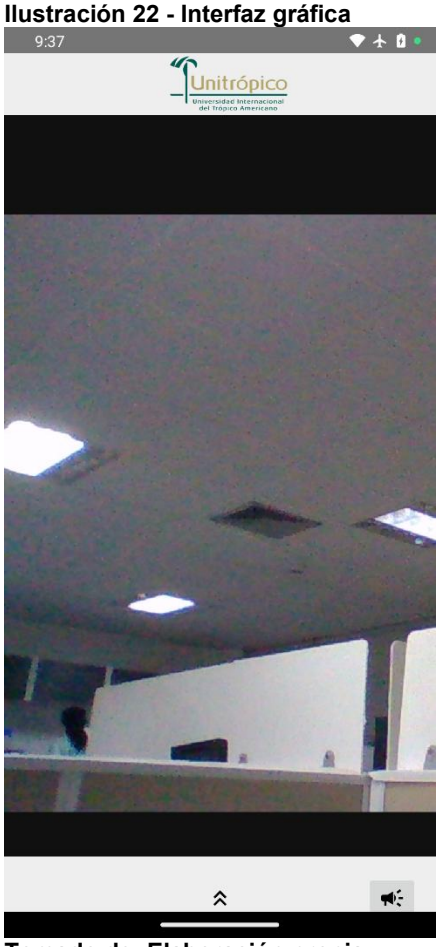
Se realizó diseño de interfaz gráfica con Android Studio Iguana 2023.2.1, el diseño se basa en una sola pantalla, en donde el usuario entrará directamente con la cámara del dispositivo para poder realizar la traducción, esto para tener una interacción más intuitiva del usuario.

Ilustración 21 - Screen Splash



Tomado de: Elaboración propia

Ilustración 22 - Interfaz gráfica



Tomado de: Elaboración propia

CONCLUSIONES

En conclusión, el desarrollo del modelo de red neuronal para detectar caracteres Braille y la creación de la aplicación Android que lo acompaña representan un avance significativo hacia la ayuda a las personas con discapacidad visual. La integración con éxito de algoritmos de aprendizaje automático en un dispositivo portátil ofrece una vía prometedora para mejorar la accesibilidad y la independencia de los usuarios. La capacidad de detección en tiempo real, junto con la opción de anunciar las predicciones de forma audible, proporciona una valiosa herramienta para navegar por entornos en los que predomina la señalización en Braille. Sin embargo, es esencial reconocer las limitaciones actuales de la aplicación, incluida su falta de optimización para usuarios con discapacidad visual y la incapacidad para detectar palabras completas a partir de los caracteres detectados.

Se ha llevado a cabo un análisis riguroso de los retos y capacidades inherentes a la traducción braille en tiempo real. Aunque el proceso puso de manifiesto ciertas limitaciones, como la detección de un carácter cada vez, también subrayó el potencial de innovación en este campo.

El diseño de la arquitectura de software del prototipo se basó en la información obtenida en la fase de análisis. A pesar de las dificultades encontradas, sobre todo para garantizar la compatibilidad con las plataformas Android, se desarrolló con éxito una arquitectura robusta que sirve de base para la traducción braille en tiempo real.

Partiendo de la metodología y la arquitectura meticulosamente diseñadas, la fase de codificación supuso la transformación de los conceptos teóricos en una realidad tangible. Aunque se observaron algunos retrasos, sobre todo en la capacidad de respuesta del botón de texto a voz (TTS), la funcionalidad general del prototipo se mantiene intacta y ofrece perspectivas prometedoras de perfeccionamiento.

La fase final del proyecto fue la de validación. El prototipo se sometió a pruebas rigurosas en distintas configuraciones y condiciones visuales. A pesar de algunas dificultades, el prototipo demostró un excelente rendimiento, lo que confirma su viabilidad y sienta las bases para futuras mejoras.

RECOMENDACIONES

Para abordar las limitaciones mencionadas y mejorar aún más la utilidad de la aplicación, se pueden explorar varias vías de investigación futura. En primer lugar, debería ser prioritario mejorar las funciones de accesibilidad de la aplicación. Esto podría implicar la incorporación de funciones como la navegación controlada por voz, la información táctil y la compatibilidad con lectores de pantalla para garantizar que los usuarios con discapacidad visual puedan utilizar plenamente las funcionalidades de la aplicación. Además, ampliar el alcance de la detección para que reconozca palabras o frases completas a partir de los caracteres detectados mejoraría significativamente la usabilidad de la aplicación y su viabilidad en situaciones reales. Por otra parte, la optimización del modelo de red neuronal para detectar múltiples caracteres simultáneamente e interpretar con precisión su disposición espacial podría mejorar enormemente la eficiencia y eficacia de la aplicación. Esto puede implicar el perfeccionamiento de los datos de entrenamiento, el ajuste de la arquitectura del modelo y la aplicación de técnicas avanzadas como el aprendizaje secuencia a secuencia o los mecanismos de atención. Además, la realización de estudios de usuarios y la solicitud de opiniones a personas con discapacidad visual serían muy valiosas para identificar problemas específicos de usabilidad y áreas de mejora. La colaboración con expertos en accesibilidad y organizaciones dedicadas a ayudar a personas con discapacidad visual podría aportar valiosas ideas y orientaciones a lo largo del proceso de desarrollo.

En conclusión, poniendo en práctica las recomendaciones mencionadas y perfeccionando continuamente el diseño y la funcionalidad de la aplicación, es posible crear una herramienta más inclusiva y eficaz para mejorar la accesibilidad y la independencia de las personas con discapacidad visual.

BIBLIOGRAFÍA

[1] P. Ackland, S. Resnikoff, y R. Bourne, “World blindness and visual impairment: despite many successes, the problem is growing”, *Community Eye Health*, vol. 30, núm. 100, p. 71, 2017, Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: /pmc/articles/PMC5820628/

[2] C. Parra Dussan, “Los ciegos en el Censo 2018”, *Blog INCI*, núm. 106, 2020, Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: https://www.inci.gov.co/blog/los-ciegos-en-el-censo-2018

[3] J. Guerreiro, D. Gonçalves, D. Marques, T. Guerreiro, H. Nicolau, y K. Montague, “The today and tomorrow of Braille learning”, *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2013*, 2013, doi: 10.1145/2513383.2513415.

[4] New England Low Vision and Blindness, “Duxbury Braille Translator for Windows, Single User Software License - Blindness Software”. New England Low Vision and Blindness, Worcester, 1995. Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: https://nelowvision.com/product/duxbury-braille-translator-for-windows/

[5] E. R. Hoskin, M. K. Coyne, M. J. White, S. C. D. Dobri, T. C. Davies, y S. D. Pinder, “Effectiveness of technology for braille literacy education for children: a systematic review”, *Disabil Rehabil Assist Technol*, 2022, doi: 10.1080/17483107.2022.2070676.

[6] Barry Hampshire, *La práctica del braille: el braille como medio de comunicación*. Editorial de la Unesco, 1981.

[7] Patrick, “5 Ways to Effectively Use Braille Translation”, Accredited Language Services. Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: https://www.accreditedlanguage.com/translation/5-ways-to-effectively-use-braille-translation/

[8] S. Baltisberger, “Spanish Braille and English Language Learners”, Paths to Literacy. Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: https://www.pathstoliteracy.org/spanish-braille-and-english-language-learners/

[9] Institute of Education Sciences, “Inside IES Research | World Braille Day: Research on Teaching Braille to Students with Visual Impairments”, NCSER. Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: https://ies.ed.gov/blogs/research/post/world-braille-day-research-on-teaching-braille-to-students-with-visual-impairments

[10] S. Garg, “Further Challenges Faced By Visually Impaired While Acquiring Education”, Linkedin Pulse. Consultado: el 19 de septiembre de 2023. [En línea]. Disponible en: https://www.linkedin.com/pulse/further-challenges-faced-visually-impaired-while-acquiring-garg/

[11] Jernigan Institute, “The Braille Literacy Crisis in America Facing the Truth, Reversing the Trend, Empowering the Blind”, Baltimore, mar. 2009.

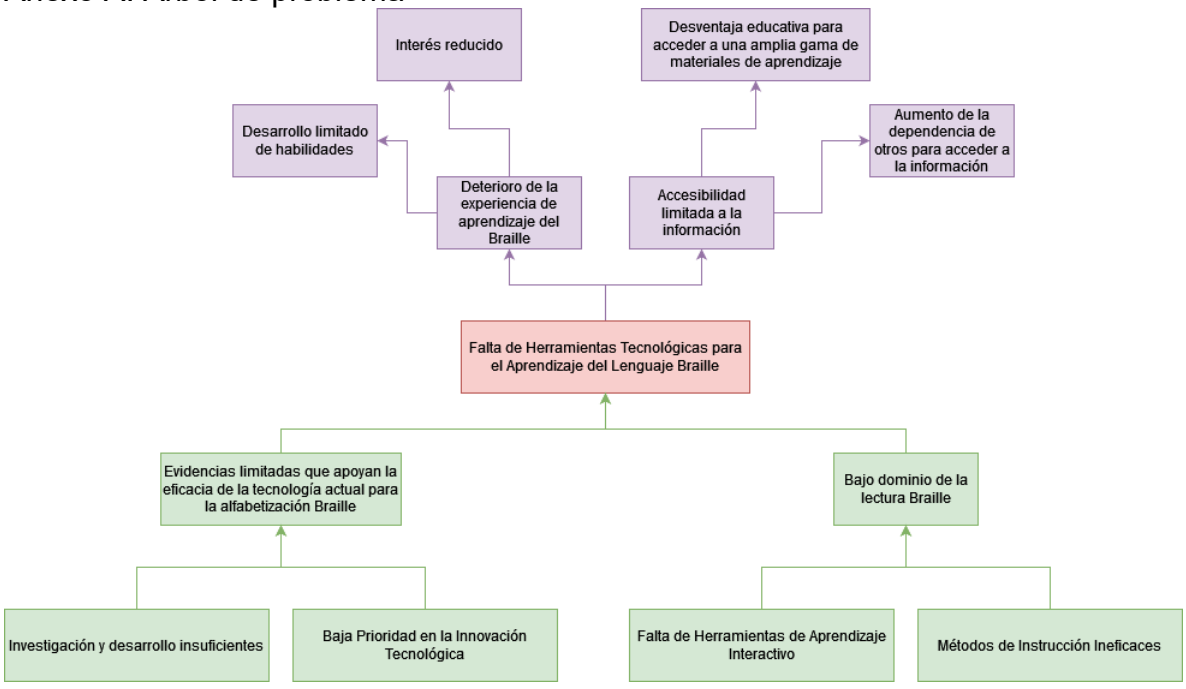
[12] V. Argyropoulos y V. Papadimitriou, “Braille Reading Accuracy of Students Who Are Visually Impaired: The Effects of Gender, Age at Vision Loss, and

Level of Education", *CEU*, 2015, Consultado: el 19 de septiembre de 2023. [En línea]. Disponible en: http://jvib.org/CEUs.
[13] M. A. Swenson, "Beginning with Braille: Challenges and Choices", National Federation of the Blind. Consultado: el 19 de septiembre de 2023. [En línea]. Disponible en: https://nfb.org/sites/default/files/images/nfb/publications/bm/bm13/bm1302/bm130206.htm
[14] WeCapable, "Spanish to Braille Translator". Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: https://wecapable.com/braille-translator/spanish-to-braille-translator/
[15] X. Dong, "Braille Academy: Play & Learn". Dong Digital, pp. 0–0, 2021. Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: https://apps.apple.com/us/app/braille-academy-play-learn/id1561530189
[16] K. Duarte-Barón, J. X. Pabón, R. Claros, y J. J. Gil, "Design and construction of a device for facilitating the learning of Braille literacy system", *Ingeniería y competitividad*, vol. 18, núm. 1, pp. 77–90, 2016, doi: 10.25100/iyc.v18i1.2179.
[17] M. Calvo, "It Began in Colombia", The AIR Foundation. Consultado: el 12 de septiembre de 2023. [En línea]. Disponible en: http://www.accessibilityisaright.org/blog_it_began_in_colombia
[18] U. I. for I. T. in E. UNESCO IIEP, "COVID 19, technology-based education and disability: the case of Colombia", *UNESDOC*, 2021, Consultado: el 19 de septiembre de 2023. [En línea]. Disponible en: https://unesdoc.unesco.org/ark:/48223/pf0000377773
[19] Microsoft, "Seeing AI App". 2017. Consultado: el 12 de noviembre de 2023. [En línea]. Disponible en: https://www.microsoft.com/en-us/ai/seeing-ai
[20] A. Webster, "BrailleTouch brings 'eyes-free' texting to the visually impaired - The Verge", *The Verge*, feb. 2012, Consultado: el 12 de noviembre de 2023. [En línea]. Disponible en: https://www.theverge.com/2012/2/17/2805109/apple-ios-braille-touch-georgia-tech-eyes-free-texting
[21] H. Jørgen Wiberg, "Be My Eyes". 2015. Consultado: el 12 de noviembre de 2023. [En línea]. Disponible en: https://www.bemyeyes.com/
[22] Luis Herazo, "anincubator", ¿QUÉ ES UNA APLICACIÓN MÓVIL?
[23] Amazon Web Services, "Amazon Web Services, Inc.", ¿Qué es la inteligencia artificial? - Explicación de la inteligencia artificial (IA) - AWS.
[24] Julián Pérez Porto, "Definicion.de", Braille - Qué es, definición, historia y características.
[25] Hyland.com, "Hyland.com", What is Optical Character Recognition (OCR) Technology?
[26] Stefan Platzer, "Konfuzio", ¿Qué es la visión por ordenador? La percepción mediante IA explicada de forma sencilla.
[27] Mauricio García, "Medium", Interfaz de usuario (UI), Experiencia de usuario (UX) y un poco más allá….
[28] Colaborador de TechTarget, "computerweekly.es", Aprendizaje automático o automatizado (machine learning).

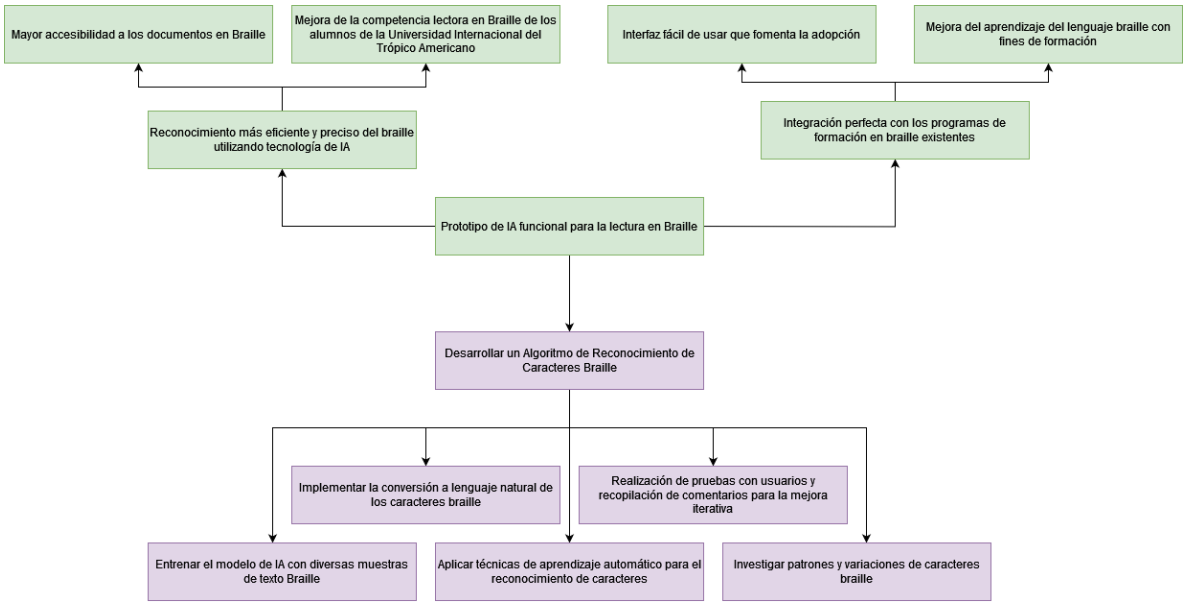
[29] Oracle.com, “Oracle”, ¿Sabe qué es el aprendizaje profundo?
[30] Congreso de Colombia, “Ley 2265 de 2022”, http://www.secretariasenado.gov.co/senado/basedoc/ley_2265_2022.html#1.
[31] Congreso de Colombia, “Ley 1346 de 2009”, https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=37150.
[32] Congreso de Colombia, “Ley 1680 de 2013”, http://www.secretariasenado.gov.co/senado/basedoc/ley_1680_2013.html.
[33] Presidente de la República de Colombia, “Decreto 1377 de 2013”, https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=53646.
[34] Plutora, “Incremental Model”, Incremental Model: What It Is and How to Implement It.
[35] G. Latif, G. Ben Brahim, S. E. Abdelhamid, R. Alghazo, G. Alhabib, y K. Alnujaidi, “Learning at Your Fingertips: An Innovative IoT-Based AI-Powered Braille Learning System”, *Applied System Innovation 2023, Vol. 6, Page 91*, vol. 6, núm. 5, p. 91, oct. 2023, doi: 10.3390/ASI6050091.
[36] “Model Validation”. Consultado: el 16 de abril de 2024. [En línea]. Disponible en: https://c3.ai/glossary/data-science/model-validation/

ANEXOS

Anexo A. Árbol de problema



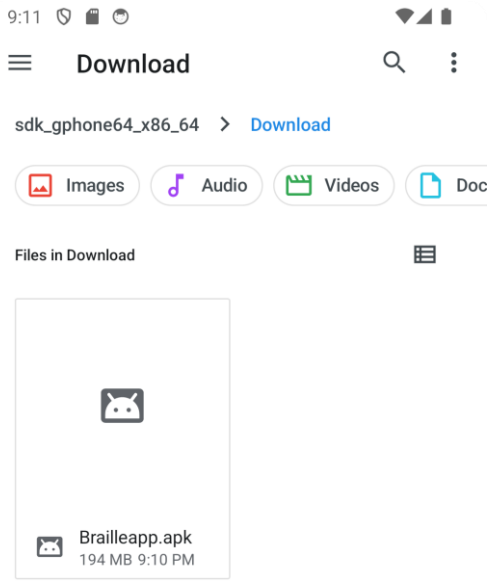
Anexo B. Árbol de objetivo

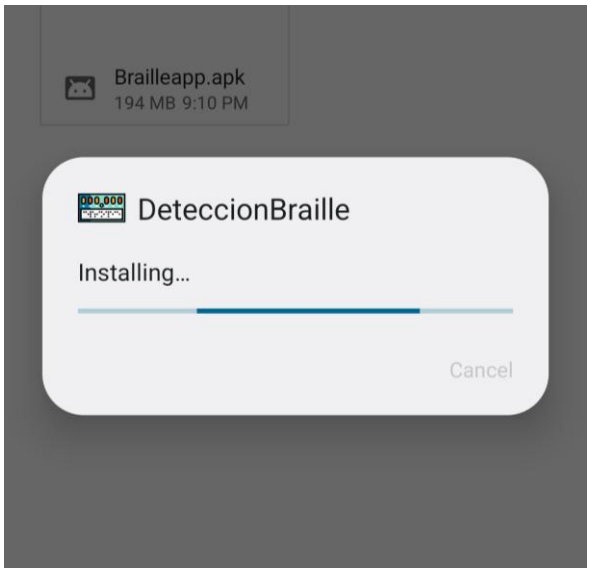

Mayor accesibilidad a los documentos en Braille
Mejora de la competencia lectora en Braille de los alumnos de la Universidad Internacional del Trópico Americano
Interfaz fácil de usar que fomenta la adopción
Mejora del aprendizaje del lenguaje braille con fines de formación
Reconocimiento más eficiente y preciso del braille utilizando tecnología de IA
Integración perfecta con los programas de formación en braille existentes
Prototipo de IA funcional para la lectura en Braille
Desarrollar un Algoritmo de Reconocimiento de Caracteres Braille
Implementar la conversión a lenguaje natural de los caracteres braille
Realización de pruebas con usuarios y recopilación de comentarios para la mejora iterativa
Entrenar el modelo de IA con diversas muestras de texto Braille
Aplicar técnicas de aprendizaje automático para el reconocimiento de caracteres
Investigar patrones y variaciones de caracteres braille

MANUAL DE USUARIO

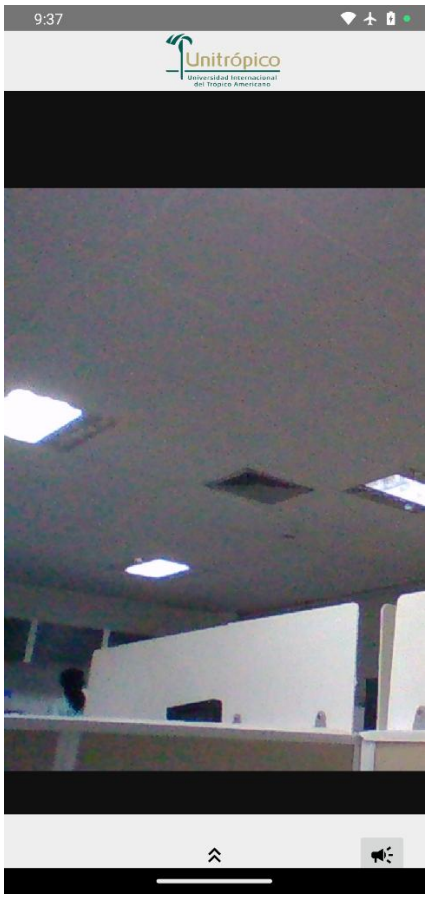
Visión general de la aplicación:
Esta aplicación Android utiliza TensorFlow para fines de inferencia de aprendizaje automático. Emplea varios componentes, como una vista previa de la cámara, una interfaz de hoja inferior para el ajuste de la configuración y superposiciones para la interacción del usuario.

1. **Cómo empezar:**
   - Instala la app desde Google Play Store o descarga el archivo APK desde la fuente proporcionada.





   - Abre la aplicación pulsando sobre su icono en el cajón de aplicaciones de tu dispositivo.



2. Interfaz principal:

Al iniciar la app, te encontrarás con los siguientes componentes:

- Vista previa de la cámara:
    - Esta sección muestra la imagen en directo capturada por la cámara de tu dispositivo.
    - Puedes ver imágenes en tiempo real a través de esta vista previa.

- Vista superpuesta:
    - La vista superpuesta proporciona comentarios visuales o información adicional superpuesta sobre la vista previa de la cámara.
    - Esta superposición mejora la interacción del usuario y proporciona información relevante.

- Interfaz de hoja inferior:
    - Situada en la parte inferior de la pantalla, la interfaz de la hoja inferior ofrece varios ajustes y controles para la app.
    - Puedes acceder y ajustar diferentes parámetros relacionados con la inferencia de aprendizaje automático y la funcionalidad de la cámara.

3. Uso de la hoja inferior:

La interfaz de la hoja inferior contiene varios parámetros ajustables:

- Botón Leer predicciones:
    - Al tocar este botón, la app lee en voz alta las predicciones o resultados generados por el modelo de aprendizaje automático.

- Tiempo de inferencia:
    - Indica el tiempo que tarda el modelo de aprendizaje automático en procesar una inferencia.
    - Este parámetro proporciona información sobre el rendimiento del modelo.

- Umbral de confianza:
    - Ajusta el umbral de confianza para las predicciones realizadas por el modelo de aprendizaje automático.
    - Puede aumentar o disminuir este umbral para filtrar las predicciones en función de su nivel de confianza.

- Resultados máximos:
    - Ajusta el número máximo de resultados o predicciones que muestra la app.
    - Ajusta este parámetro en función de tus preferencias o necesidades.

- Número de hilos:
    - Configura el número de hilos utilizados por el modelo de aprendizaje automático para la inferencia.
    - Aumentar el número de hilos puede mejorar el rendimiento, pero también puede consumir más recursos del dispositivo.

- Selección de delegado:
    - Permite seleccionar el delegado para la inferencia del aprendizaje automático.
    - Diferentes delegados ofrecen diferentes características de rendimiento y utilización de recursos.

- Selección del modelo:
    - Le permite elegir el modelo de aprendizaje automático utilizado para la inferencia.

    - Seleccione el modelo que mejor se adapte a sus necesidades o a la tarea específica en cuestión.

- Información adicional:
    - Asegúrate de que los permisos de cámara están concedidos a la app para habilitar la funcionalidad de previsualización de la cámara.
    - Algunas funciones pueden requerir una conexión activa a Internet para las actualizaciones del modelo o funciones adicionales.

4. **Solución de problemas:**
    - Si encuentras algún problema o error mientras utilizas la app, prueba a reiniciar la app o a reiniciar tu dispositivo.
    - Asegúrate de que tu dispositivo cumple los requisitos mínimos del sistema especificados para la aplicación.

5. **Comentarios y asistencia:**

Para cualquier comentario, sugerencia o asistencia técnica, ponte en contacto con nuestro equipo de asistencia en [ingsistemas@unitropico.edu.co].

6. **Condiciones de uso:**

Al utilizar esta aplicación, aceptas cumplir los términos y condiciones descritos en el Acuerdo de licencia de usuario final (CLUF). Lea atentamente el CLUF antes de utilizar la aplicación.

7. **Política de privacidad:**

Nos comprometemos a proteger su privacidad. Nuestra Política de privacidad describe cómo recopilamos, utilizamos y divulgamos su información personal. Consulte la Política de privacidad para obtener más información.

8. **Información sobre derechos de autor:**

Derechos de autor © [2024] Universidad Internacional del Trópico Americano (UNITROPICO). Todos los derechos reservados.

Esta aplicación es un proyecto de grado desarrollado para la Universidad Internacional del Trópico Americano (UNITROPICO) y es propiedad de dicha institución. La aplicación puede contener componentes sujetos a derechos de autor de terceros, cuyos derechos están reservados a sus respectivos propietarios. Para obtener más información sobre los derechos de autor de los componentes individuales, consulte los avisos correspondientes.

Información sobre la licencia:
La aplicación está licenciada bajo la Licencia Apache, Versión 2.0. Puede obtener una copia de la licencia en http://www.apache.org/licenses/LICENSE-2.0.

Información sobre la versión:
Versión de la aplicación: 1.0.0
Última actualización: 20 de marzo de 2024

Para conocer las últimas actualizaciones y anuncios, visite nuestro sitio web o síganos en las redes sociales:

- Correo electrónico: ingsistemas@unitropico.edu.co
- Teléfono: 3102041973
- Sitio web: https://www.unitropico.edu.co/
- Instagram: https://www.instagram.com/unitropicooficial/
- Facebook: https://www.facebook.com/unitropico.edu.co/
- Twitter: https://twitter.com/unitropico/

Gracias por utilizar nuestra aplicación.