Desarrollo de una aplicación móvil para la generación de descripciones de imágenes para personas con discapacidad visual

Development of a mobile application for the generation of image descriptions for people with visual disabilities

Trabajo de Fin de Grado
Curso 2022–2023

Autores

Matías Amor Sanz
Alberto Chaves López
Víctor Ruiz Gea

Directores

Alberto Díaz Esteban
Raquel Hervás Ballesteros

Grado en Ingeniería Informática, de Computadores y de Software
Facultad de Informática
Universidad Complutense de Madrid

Desarrollo de una aplicación móvil para la generación de descripciones de imágenes para personas con discapacidad visual
Development of a mobile application for the generation of image descriptions for people with visual disabilities

Trabajo de Fin de Grado en Ingeniería Informática, de Computadores y de Software

Autores

Matías Amor Sanz
Alberto Chaves López
Víctor Ruiz Gea

Directores

Alberto Díaz Esteban
Raquel Hervás Ballesteros

Convocatoria: *Junio* 2023

Grado en Ingeniería Informática, de Computadores y de Software

Facultad de Informática

Universidad Complutense de Madrid

29 de mayo de 2023

Agradecimientos

Agradecemos a nuestros tutores Raquel y Alberto, por confiar en nosotros desde el primer momento y ayudarnos en todo lo posible. Damos las gracias a Margarita, Víctor Alberto y Gema por ofrecerse como colaboradores y motivarnos a sacar la aplicación adelante.

Resumen

Hoy en día, el uso de dispositivos móviles parece algo básico para la vida de cualquier persona. Es una herramienta que nos permite hacer múltiples tareas a través de sus aplicaciones. Además, es un medio de comunicación que permite relacionarse y comunicarse con alguien de forma remota e instantánea. Llamar, chatear, enviar documentos o imágenes a otras personas, son acciones que casi todos hacemos diariamente. Sin embargo, para las personas con una discapacidad visual, algo como lo mencionado anteriormente, les puede resultar muy complicado.

En este TFG nos hemos enfocado en el problema que tienen las personas invidentes en recibir y comprender las imágenes que reciben en su teléfono móvil. Investigando aplicaciones que existen para la descripción de imágenes, hemos podido ver que estas proporcionan descripciones de imágenes poco detalladas y no queda claro cómo es la foto o donde se encuentran los elementos dentro de la misma. Hablando y entrevistando a personas invidentes, nos dimos cuenta que estas personas tenían dificultades a la hora de comprender el contenido de una imagen.

Es por ello que en este trabajo hemos intentando resolver este problema. Hemos desarrollado una aplicación móvil que permite describir una imagen. Además permite navegar por ella y, haciendo *clicks* en la foto, proporciona información de qué elementos hay en la foto. El objetivo es ayudar a una persona invidente a saber como es la foto y a hacerse una imagen mental de esta.

El contenido del trabajo está ubicado en el siguiente repositorio: `https://github.com/Victorruizgea/TFG_App`

Palabras clave

Discapacidad visual, accesibilidad, aplicación móvil, descripción de imágenes.

Abstract

Nowadays, the use of mobile devices seems essential in anyone's life. It is a tool that allows us to perform multiple tasks through its applications. Furthermore, it is a means of communication that enables us to connect and interact with others remotely and instantly. Making phone calls, chatting, sending documents or images to others are actions that almost everyone does on a daily basis. However, for individuals with visual impairments, something as mentioned earlier can be very challenging.

In this TFG we have focused on the issue faced by visually impaired individuals in receiving and comprehending images on their mobile phones. Through our research on existing image description applications, we have observed that they provide vague descriptions of images, and it is unclear how the photo looks or where the elements are located within it. By speaking and interviewing visually impaired individuals, we realized that they encountered difficulties in understanding the content of an image.

That's why in this project, we have attempted to solve this problem. We have developed a mobile application that allows for image description. Additionally, it enables users to navigate through the image and, by clicking on specific areas, provides information about the elements present in the photo. This will assist visually impaired individuals in understanding how the photo appears and forming a mental image of it.

The content of the work can be seen at: `https://github.com/Victorruizgea/TFG_App`

Keywords

Visual impairment, accessibility, mobile application, image description.

Índice

Índice de figuras

Capítulo 1

Introducción

En este capítulo vamos a hacer una introducción sobre la motivación, objetivos y estructura de este Trabajo de Fin de Grado.

"La única discapacidad en la vida es una mala actitud"
— Scott Hamilton

1.1. Motivación

La visión es uno de los sentidos más importantes del ser humano, ya que nos permite percibir y procesar información del mundo que nos rodea. Llamamos discapacidad visual a cualquier tipo de alteración del sentido de la vista. Conocemos dos tipos de discapacidad visual. Por un lado, la deficiencia visual, que es una disminución significativa de la agudeza visual, pero permite ver la luz y orientarse hacia ella. Las personas que padecen esta deficiencia tienen un campo de visión funcional, pero reducido. La ceguera, en cambio, es la percepción mínima de la luz que impide su uso funcional.

El día a día de una persona invidente puede ser muy desafiante debido a la falta de visión. Sin embargo, la tecnología y la adaptación han hecho posible que las personas invidentes lleven una vida relativamente normal. Algunas personas utilizan bastones blancos para orientarse y perros guías para ayudarlos a desplazarse. También existen aplicaciones y dispositivos electrónicos especiales que les permiten realizar tareas cotidianas, como leer y navegar por la web. A pesar de estos avances, la falta de accesibilidad en la sociedad aún puede ser un obstáculo para las personas con discapacidad visual. Algunos de los desafíos diarios a los que se enfrentan son la movilidad, ya que puede ser difícil moverse y orientarse sin visión, lo que puede aumentar el riesgo de accidentes y lesiones; la accesibilidad, ya que la sociedad aún no está completamente adaptada a las necesidades de las personas invidentes, lo que puede dificultar el acceso a edificios, transporte público y otros servicios; y por último el empleo, debido a que las personas invidentes pueden tener dificultades para encontrar y mantener un trabajo debido a barreras de accesibilidad.

En la actualidad, la tecnología juega un papel fundamental en la vida de las personas, especialmente en la de aquellas con discapacidad visual. El reconocimiento

automático de imágenes puede tener un fuerte impacto en la vida de las personas invidentes, ya que les facilita la accesibilidad y la independencia, permitiéndoles una mejor percepción del entorno que les rodea. Una de las formas en las que esta tecnología puede ayudar a mejorar la vida de las personas invidentes es a través de aplicaciones que brinden descripciones detalladas y precisas de imágenes o fotografías. Esto les puede ser de gran utilidad, debido a que les permite acceder a información que de otra manera estaría fuera de su alcance.

1.2. Objetivos

Nuestro objetivo es intentar desarrollar una aplicación capaz de generar descripciones de imágenes, permitiendo así a las personas invidentes la posibilidad de obtener una idea mental de dicha imagen. Para conseguir este objetivo en nuestro TFG necesitamos cumplir una serie de objetivos específicos.

En primer lugar, queremos realizar un estudio con personas con discapacidad visual a las que potencialmente les podría ser útil nuestra aplicación. Para ello, será necesario determinar cuales son sus necesidades a la hora de acceder a la información de una imagen y manejar aplicaciones móviles.

Una vez hayamos identificado las necesidades y requisitos de nuestros usuarios, el siguiente paso consistirá en estudiar las técnicas de inteligencia artificial que están disponibles en la actualidad. De esta forma, podremos entender como implementar nuestras dos funcionalidades: generar una descripción y detectar objetos dentro de una imagen. En cuanto a la generación de descripciones, tendremos que investigar sobre la forma más eficiente de expresar todos los detalles existentes en una imagen.

Nuestra aplicación deberá estar conectada a un servidor que se encargue de realizar nuestras dos funcionalidades. Para poder generar descripciones y detectar objetos, también será importante saber cómo recoger y procesar estas imágenes. Por otro lado, nuestra aplicación móvil debe ser capaz de dar al usuario la oportunidad de escoger una imagen de su galería. En cuanto a su diseño, nuestro objetivo es intentar que sea lo más intuitiva y sencilla posible.

Finalmente, sería muy útil poder hacer una evaluación de nuestra aplicación con usuarios reales, para conocer su utilidad real e identificar posibles mejoras.

1.3. Estructura del documento

Este proyecto seguirá una estructura determinada por los siguientes capítulos:

- En los Capítulos 1 y 2 puede verse reflejada una introducción de nuestro proyecto, siendo el Capítulo 1 una introducción en español y el Capítulo 2 en inglés. En estos capítulos se habla de la motivación y los objetivos del proyecto.

- En el Capítulo 3 se profundiza en explicar aquellas aplicaciones y tecnologías en las que se sustenta nuestro trabajo. Por un lado, se describen las principales tecnologías de descripción de imágenes y detección de elementos dentro

de estas. Por otro lado, se revisan diferentes aplicaciones existentes para la descripción de imágenes y detección de elementos en imágenes.

- En el Capítulo 4 se resume y se transcribe las entrevistas que hemos realizado. Estas entrevistas nos han ayudado a capturar los requisitos que debería tener nuestra aplicación y los problemas que tienen las personas invidentes con las aplicaciones que existen en el mercado actualmente. Este capítulo constará de dos entrevistas, una primera realizada a un grupo de personas con discapacidad visual y una segunda a una persona experta en audiodescripción.

- En el Capítulo 5 se explica detalladamente el funcionamiento de la aplicación y su arquitectura interna. Es un capítulo extenso donde se ve cómo funciona la aplicación, las diferentes partes de esta y el funcionamiento de dichas partes.

- En el Capítulo 6 hablaremos sobre la evaluación final de nuestra aplicación. Se contará la manera en la que hemos realizado dicha evaluación y los resultados de esta.

- En los Capítulos 7 y 8 se elaboran unas conclusiones finales tras el análisis de nuestro trabajo y las ideas que tenemos para nuestro proyecto de cara al futuro, siendo el Capítulo 7 en español y el Capítulo 8 en inglés.

- Por último, en el Capítulo 9 contamos de forma extendida las aportaciones individuales que hemos hecho cada uno en el proyecto.

Capítulo 2

Introduction

In this chapter, we will provide an introduction to the motivation, objectives, and structure of this Bachelor's Thesis.

2.1. Motivation

Vision is one of the most important senses for human beings as it enables us to perceive and process information from the world around us. Visual impairment refers to any type of alteration in the sense of sight. There are two main types of visual impairment. Firstly, there is visual deficiency, which is a significant decrease in visual acuity but still allows for perception of light and orientation towards it. People with visual deficiency have a functional but reduced field of vision. On the other hand, blindness refers to minimal perception of light, preventing functional use of vision.

The daily life of a blind person can be very challenging due to the lack of vision. However, technology and adaptation have made it possible for blind individuals to lead relatively normal lives. Some people use white canes for orientation and guide dogs to assist with mobility. There are also special applications and electronic devices that enable them to perform daily tasks such as reading and navigating the web. Despite these advancements, the lack of accessibility in society can still be a barrier for people with visual disabilities. Some of the daily challenges they face include mobility difficulties, as it can be challenging to move around and orient oneself without vision, increasing the risk of accidents and injuries; accessibility issues, as society is not fully adapted to the needs of blind individuals, making it difficult to access buildings, public transportation, and other services; and lastly, employment challenges, as blind people may struggle to find and maintain jobs due to accessibility barriers.

Currently, technology plays a fundamental role in people's lives, especially those with visual disabilities. Automatic image recognition can have a significant impact on the lives of blind individuals by enhancing their accessibility and independence, allowing them to better perceive the surrounding environment. One way in which this technology can improve the lives of blind people is through applications that provide detailed and accurate descriptions of images or photographs. This can be

highly useful, as it allows them to access information that would otherwise be beyond their reach.

2.2. Objectives

Our objective is to develop an application capable of generating descriptions of images, thereby providing blind people with a mental idea of the content of those images. To achieve this goal in our Bachelor's Thesis, we need to fulfill a series of specific objectives.

Firstly, we aim to conduct a study with visually impaired individuals who could potentially benefit from our application. To do so, it will be necessary to determine their needs when accessing image information and using mobile applications.

Once we have identified the needs and requirements of our users, the next step will be to study the artificial intelligence techniques currently available. This will allow us to understand how to implement our two functionalities: generating descriptions and detecting objects within an image. Regarding description generation, we will need to research the most efficient way to express all the details present in an image.

Our application will need to be connected to a server that handles these two functionalities. In order to generate descriptions and detect objects, it will also be important to learn how to capture and process these images. Furthermore, our mobile application should give the user the opportunity to select an image from their gallery. As for its design, our goal is to make it as intuitive and simple as possible.

Finally, it would be very useful to evaluate our application with real users to assess its real-world utility and identify possible improvements.

2.3. Structure of the document

This project will follow a structure defined by the following chapters:

- Chapters 1 and 2 provide an introduction to our project, with Chapter 1 being an introduction in Spanish and Chapter 2 in English. These chapters discuss the motivation and objectives of the project.

- Chapter 3 delves into explaining the applications and technologies that underpin our work. On one hand, it describes the main image description technologies and object detection techniques. On the other hand, it reviews different existing applications for image description and object detection in images.

- Chapter 4 summarizes and transcribes the interviews we have conducted. These interviews have helped us capture the requirements that our application should have and the problems faced by visually impaired individuals with existing applications on the market. This chapter will consist of two interviews, one conducted with a group of visually impaired individuals and another with an expert in audio description.

- Chapter 5 explains in detail the functioning of the application and its internal architecture. It is an extensive chapter that showcases how the application works, the different components involved, and their functioning.
- Chapter 6 discusses the final evaluation of our application. It explains how we conducted the evaluation and presents the results obtained.
- Chapters 7 and 8 present final conclusions after analyzing our work and outline ideas for the future of our project, with Chapter 7 in Spanish and Chapter 8 in English.
- Finally, Chapter 9 provides an extended account of the individual contributions made by each team member in the project.

Capítulo 3

Estado de la Cuestión

En este capítulo profundizaremos en los trabajos relacionados con este proyecto. En la Sección 3.1 veremos las principales tecnologías que ayudan a detectar elementos en imágenes y describirlos. En la Sección 3.2 explicaremos diferentes aplicaciones de apoyo a personas ciegas para la descripción de imágenes que existen actualmente.

3.1. Tecnologías para la detección y descripción de imágenes

En esta sección se hablará de aquellas tecnologías en las que se sustenta este trabajo. Se explicarán diferentes aplicaciones y tecnologías que ayudan a la descripción de imágenes y detección de elementos en estas.

3.1.1. Descripción de imágenes y dataset

Las descripciones de imágenes son descripciones de texto que se utilizan para describir imágenes automáticamente y son especialmente útiles para personas que tienen una discapacidad visual. Incluyen información sobre el contenido y la composición de la imagen, como la descripción de los objetos, las personas, los colores y las texturas presentes en la imagen. Se utilizan en aplicaciones de accesibilidad y se integran en diferentes plataformas, como páginas web, aplicaciones móviles y programas de edición de imágenes. Los descriptores de imágenes se pueden agregar automáticamente a las imágenes mediante el uso de tecnologías de inteligencia artificial y aprendizaje automático, o se pueden agregar manualmente por los autores de contenido.

Es importante destacar que los descriptores de imágenes deben ser precisos y representativos del contenido de la imagen, y que deben ser proporcionados de manera apropiada y accesible para todos los usuarios. La calidad de los descriptores de imágenes puede afectar significativamente la accesibilidad de la información para las personas ciegas o con discapacidad visual, por lo que es importante asegurarse de que se proporcionen descriptores de alta calidad.

Las imágenes, con sus respectivas descripciones, se almacenan en lo que se conoce

como un dataset de descripción de imágenes. Estos *datasets* se utilizan comúnmente para entrenar y evaluar modelos de inteligencia artificial y aprendizaje automático en tareas de descripción de imágenes, como la generación de descripciones automáticas de imágenes. Suelen incluir una amplia variedad de imágenes, desde imágenes simples hasta imágenes complejas, y desde imágenes con un solo objeto hasta imágenes con múltiples objetos y escenas complejas. Las descripciones de texto incluidas en estos *datasets* suelen ser descripciones detalladas de la imagen, incluyendo información sobre los objetos, las personas, los colores y las texturas presentes en la misma.

Los *datasets* de descripción de imágenes son importantes porque permiten a los investigadores y desarrolladores entrenar y evaluar modelos de inteligencia artificial en tareas de descripción de imágenes, y mejorar así la accesibilidad de la información para las personas ciegas o con discapacidad visual.

Aunque hay una gran variedad de *datasets* como *Flickr*[1] o *ImageNet*[2], hemos utilizado el *dataset COCO(Common Objects in Context)*, un dataset con miles de imágenes con descripciones detalladas y ampliamente utilizado en la investigación y el desarrollo en el campo de la inteligencia artificial y el aprendizaje automático. A continuación, lo explicaremos más en detalle.

3.1.2. COCO

COCO[3] (Common Objetcts in Context) (Lin et al., 2015) es un conjunto de datos de subtítulos, segmentación y detección de objetos a gran escala publicado por *Microsoft*. Es una base de datos de imágenes de gran capacidad que se utiliza para entrenar modelos de visión por computadora. Fue desarrollada por el equipo de investigación en inteligencia artificial de *Facebook AI*. El conjunto de datos de COCO contiene conjuntos de datos de alta calidad, en su mayoría redes neuronales de última generación. Algunas de sus características son:

- Segmentación de objetos con anotaciones detalladas de instancias, es decir, cada objeto individual en una imagen está marcado mediante cuadrados, lo que permite a los modelos aprender la ubicación y la clase de los objetos en una imagen.
- Contiene más de 330.000 imágenes y más de 2,5 millones de anotaciones. Por ello, es uno de los *datasets* más grandes y diversificados que hay disponibles.
- Cada imagen de *COCO* está anotada con múltiples etiquetas de objetos, segmentaciones de instancias y descripciones de imágenes, lo que permite una variedad de tareas de procesamiento de imágenes y lenguaje natural.
- Las imágenes de *COCO* se capturaron en entornos naturales.
- Está disponible de forma gratuita para fines de investigación y no comerciales. Su licencia abierta permite a los investigadores utilizarlo de manera libre.

[1] `https://www.kaggle.com/datasets/hsankesara/flickr-image-dataset`
[2] `https://www.image-net.org/`
[3] `https://cocodataset.org/`

- Contiene más de 80 categorías de objetos, que van desde objetos cotidianos como personas, animales, etc, hasta objetos no tan comunes como instrumentos musicales, equipos deportivos, etc.

En *COCO* se pueden realizar tres tareas principalmente:

- Detección de objetos: se utiliza para entrenar y evaluar algoritmos de detección de objetos, con el objetivo de identificar y localizar objetos específicos en una imagen.
- Segmentación de objetos/instancias: permite identificar y separar objetos individuales de una imagen.
- Generación de descripciones de imágenes: se utiliza para entrenar modelos que generan descripciones de imágenes.

El formato de archivo utilizado en las anotaciones de *COCO* es *JSON*, como se puede ver en la figura 3.1 También puede tener listas o diccionarios anidados en su interior.

El objetivo de *COCO* es proporcionar una base de datos de imágenes variadas y realistas para el entrenamiento de modelos de detección de objetos, que puedan ser aplicados en una amplia gama de aplicaciones prácticas, desde la robótica hasta la seguridad y la vigilancia. *COCO* es una herramienta valiosa para la investigación y el desarrollo en el campo de la visión por computadora, y ha sido utilizada en una amplia gama de publicaciones y proyectos en la comunidad de investigación.

3.1.3. Roboflow

Roboflow[4] es una plataforma en línea que ofrece soluciones de automatización para el desarrollo de aplicaciones de visión por computadora. Ofrece herramientas que facilitan el procesamiento, anotación y organización de imágenes y vídeos para su uso en modelos de aprendizaje automático. *Roboflow* también brinda integraciones con diversos marcos de trabajo de visión por computadora, lo que permite un flujo de trabajo más eficiente y efectivo para los desarrolladores.

Algunas de sus características son:

- Carga de datos: los usuarios pueden cargar sus propios datasets de imágenes en *Roboflow* y etiquetar las imágenes para que se puedan utilizar en tareas de aprendizaje automático.
- Anotación de imágenes y vídeos: permite a los usuarios anotar y etiquetar imágenes y vídeos de forma eficiente, lo que es esencial para el entrenamiento de modelos de aprendizaje automático. En la figura 3.2 podemos ver cómo, gracias a *Roboflow*, se identifican los elementos en una imagen.
- Generación de archivos de entrenamiento: genera automáticamente archivos de entrenamiento y validación para el modelo de aprendizaje automático, utilizando la información de las etiquetas de las imágenes.

[4] `https://roboflow.com/`

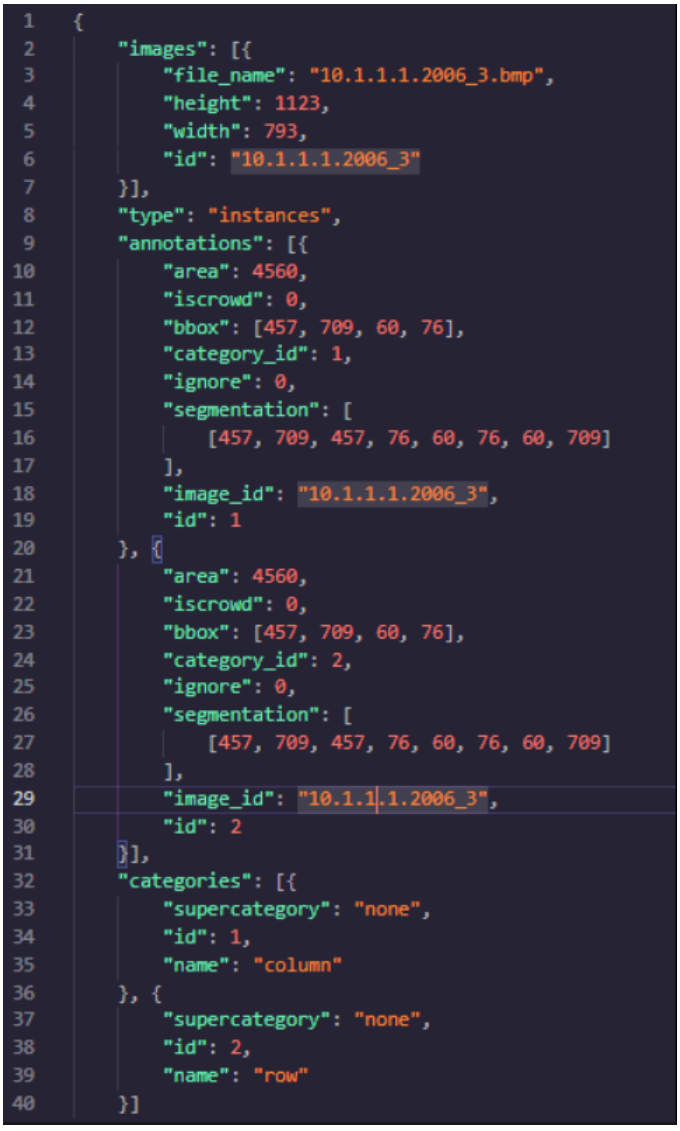

Figura 3.1: Ejemplo JSON de COCO

- Procesamiento de imágenes: ofrece herramientas para procesar imágenes y vídeos, como redimensionarlos, rotarlos y convertirlos a diferentes formatos.

- Entrenamiento del modelo: permite a los usuarios entrenar sus modelos de aprendizaje automático en la nube, utilizando los archivos de entrenamiento generados previamente.

- Integraciones con marcos de trabajo de visión por computadora: brinda integraciones con marcos de trabajo como *TensorFlow, PyTorch* y *Caffe*, lo que permite a los desarrolladores importar fácilmente sus datos anotados y utilizarlos en sus modelos.

- Evaluación del modelo: proporciona herramientas de evaluación para ayudar a los usuarios a evaluar el rendimiento de sus modelos y compararlos con modelos similares.

- Almacenamiento y organización de datos: ofrece una plataforma en línea para almacenar y organizar sus datos de entrenamiento, lo que permite un acceso más fácil y eficiente a ellos.

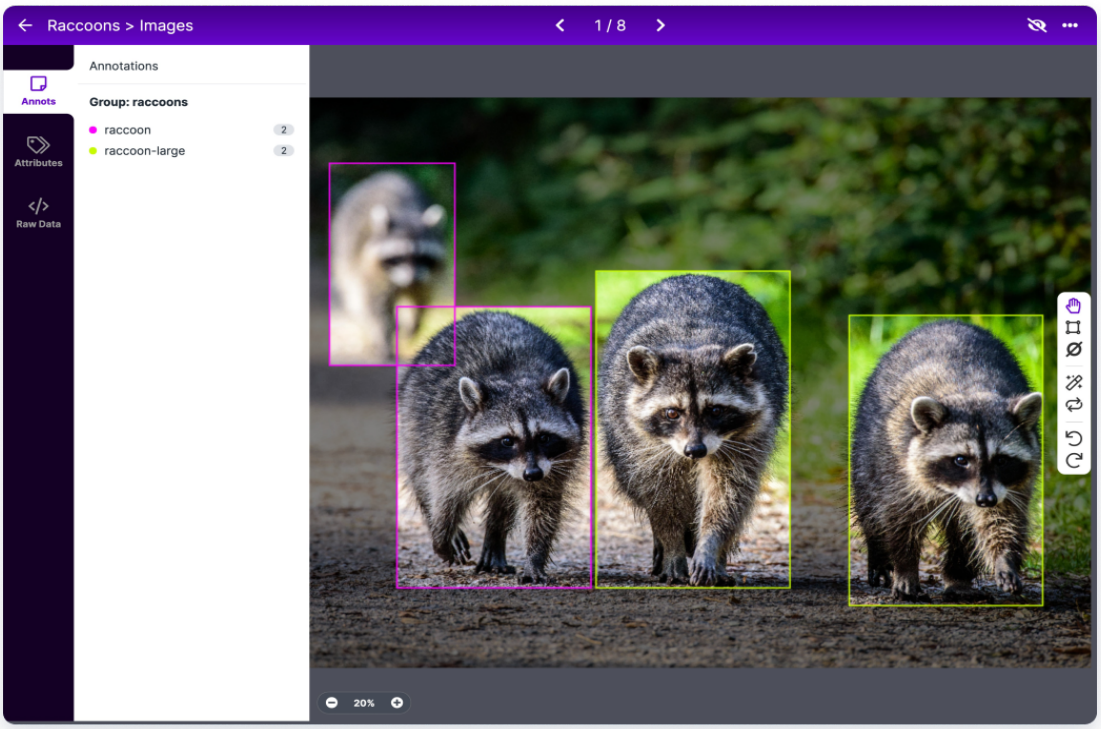

Figura 3.2: Se puede observar como, gracias a Roboflow, se identifican a los mapaches, incluso estando difuminada su figura.

3.1.4. Hugging Face

Hugging Face[5] es una empresa cuya labor es la de desarrollar herramientas y plataformas de procesamiento de lenguaje natural basadas en inteligencia artificial. Su función principal es la de crear modelos de aprendizaje automático para tareas como la comprensión de lenguaje natural, la generación de lenguaje natural, traducción automática,... Cabe destacar que a parte de liderar la revolución de la inteligencia artificial en el procesamiento del lenguaje natural, está consiguiendo que utilizar las tecnologías de *NLP* sea fácil y accesible.

Hugging Face Hub[6] es la plataforma donde se encuentra todos los modelos, conjuntos de datos y demostraciones de *Hugging Face*, todas de código abierto. Aquí es donde las personas pueden trabajar unos con otros en sus flujos de trabajo de aprendizaje automático. Funciona como espacio abierto para que cualquier persona comparta, explore, descubra y experimente con el aprendizaje automático de código abierto. En cuanto a sus modelos, para animar a promover el uso y desarrollo de

[5] `https://keepcoding.io/blog/que-es-hugging-face/`
[6] `https://huggingface.co/docs/hub/index`

estos cada repositorio está equipado de tarjetas donde se informa al usuario a como implementar cada modelo. Además, cuentan con una interfaz integrada que permite utilizar el modelo de manera intuitiva. Existen modelos con diversidad de funcionalidades como: clasificar imágenes, convertir audio a texto, generar una imagen a partir de texto,... En cuanto a los *datasets*, esta plataforma sobrepasa los 5000 *datasets* en más de 100 idiomas. Todos los *datasets* van acompañados una amplia documentación que permite al usuario explorar los datos directamente en su navegador. La biblioteca *"datasets"* permite interactuar al usuario mediante técnicas de programación con los conjuntos de datos, para que pueda usar fácilmente conjuntos de datos del *Hub* en sus proyectos. Finalmente esta plataforma consta de una sección que sirve para alojar las aplicaciones de demostración de los usuarios llamada *"Spaces"*. Permite mostrar sus proyectos y trabajar en colaboración de otras personas.

En el ejemplo de uso 3.3 aparece la interfaz de cada modelo nombrada anteriormente donde utilizamos un modelo llamado *runwayml/stable-diffusion-v1-5* (Rombach et al., 2022) cuya función es crear una imagen en base a un texto escrito por el usuario.

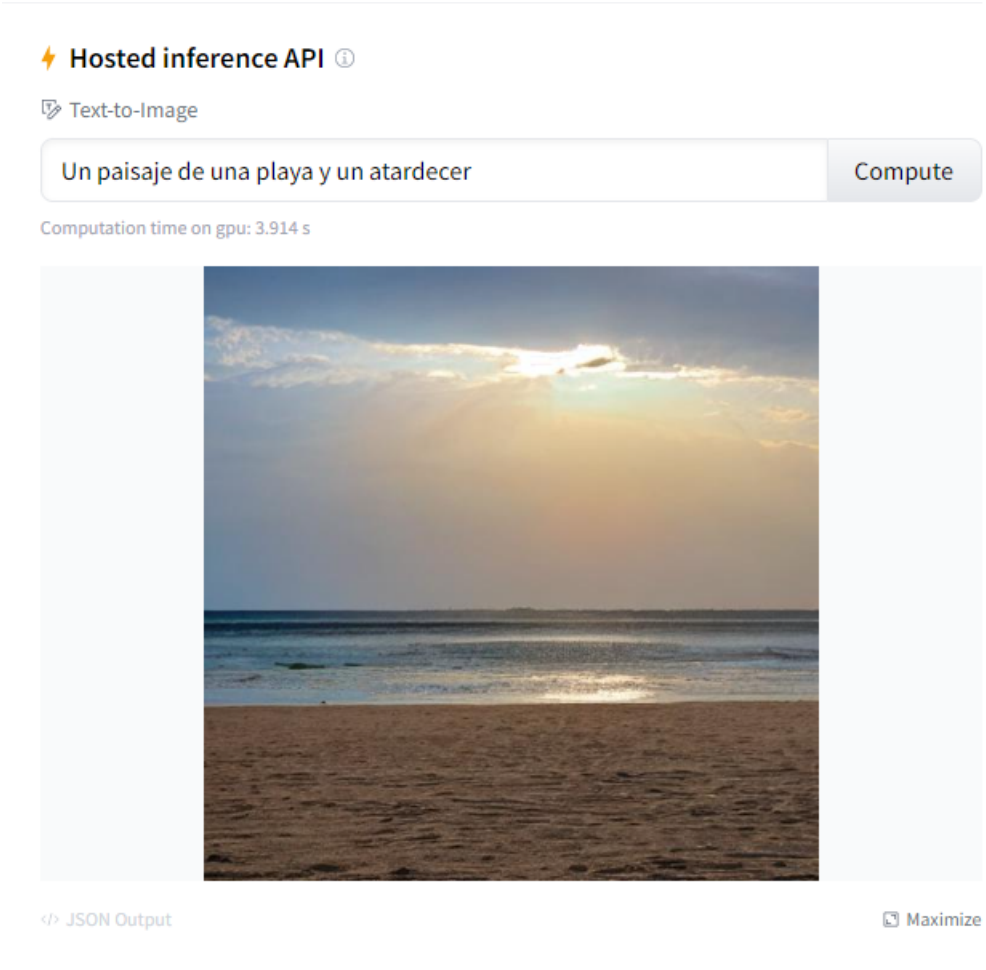

Figura 3.3: Ejemplo de uso de Hugging Face

3.1.5. YOLO

YOLO[7] *(You Only Look Once)* (Redmon et al., 2016) es un algoritmo de detección de objetos en imágenes y vídeos desarrollado por Joseph Redmon, Santosh Divvala, Ross Girshick y Ali Farhadi en 2016 y que se ha convertido en uno de los algoritmos más eficientes y populares para la detección de objetos en tiempo real. Su principal característica es que usa una única red neuronal para reconocer objetos y su clase en una imagen, lo que hace que sea más rápido y eficiente en comparación con otros sistemas o algoritmos de detección. Además es capaz de detectar múltiples objetos en una sola imagen, incluso cuando se superponen, pudiendo detectar objetos de diferentes formas y tamaños.

El funcionamiento de *YOLO* se basa en la división de una imagen en una cuadrícula, prediciendo cajas delimitadoras y clases de objetos en cada cuadrícula, como se puede ver en la figura 3.4. Para ello usa una red neuronal convolucional que permite procesar una imagen y dividirla en cuadrículas. Para cada cuadrícula se predice un conjunto de cajas delimitadoras, que contienen la ubicación y el tamaño de los objetos que están dentro de estas, asignándoles una puntuación de confianza a cada caja. Luego, se aplica un umbral para seleccionar las cajas con alta probabilidad de tener un objeto y se eliminan aquellas con baja confianza. Por último, se asigna una etiqueta a cada caja, utilizando la clasificación, en función del objeto que contiene.

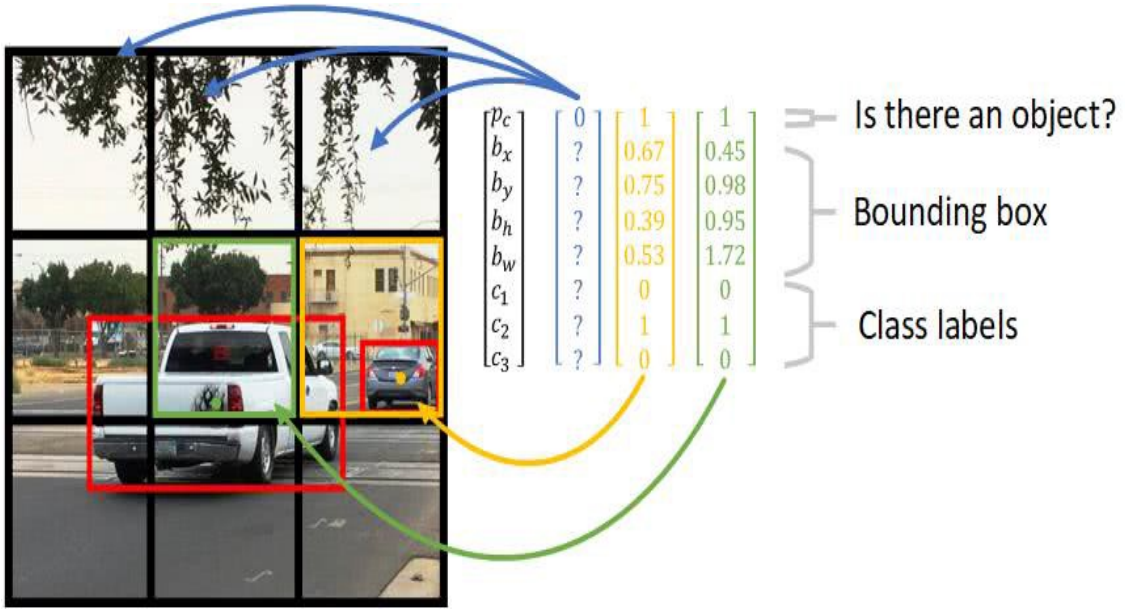

Figura 3.4: YOLO divide una imagen en celdas e identifica los distintos objetos dentro de cada una de ellas. Primero comprueba si hay un objeto dentro de la celda, y en caso de haberlo, identifica su posición, tamaño y el tipo de objeto.

Las principales ventajas de YOLO son:

- Es un sistema muy rápido por lo que es la principal herramienta para la detección de objetos a tiempo real. Esto es debido a que no requiere un pipeline demasiado complejo.

[7] `https://pjreddie.com/darknet/yolo/`

- A diferencia de otras herramientas, este sistema utiliza la imagen entera y no solo una parte de esta. Esto hace que se limiten los errores al reconocer las clases de objetos que hay en la imagen
- Tiene una precisión elevada, debido a que se usa una única red neuronal para la detección, pudiendo entrenarla de principio a fin.

YOLO es ideal para aplicaciones en tiempo real, como en sistemas de seguridad, vehículos autónomos y realidad aumentada. Además, su precisión en la detección de objetos es muy alta, superando a muchos otros algoritmos de detección de objetos en términos de velocidad y precisión.

3.2. Aplicaciones de reconocimiento de imágenes

A continuación, vamos a explicar algunos ejemplos de las aplicaciones que ayudan a las personas invidentes a reconocer imágenes en su día a día.

3.2.1. Seeing AI

Seeing AI[8] (Gil et al., 2022) es una aplicación desarrollada por *Microsoft* que utiliza tecnologías de inteligencia artificial, la cámara del móvil y algoritmos de visión para analizar el entorno del usuario y describírselo en voz alta. Es compatible con iOS y permite al usuario identificar objetos, animales y personas, indicando a qué distancia se encuentran, reconociendo a las personas que hayamos guardado previamente utilizando algoritmos de reconocimiento facial e incluso describir su estado de ánimo. Además, la aplicación también puede leer en voz alta documentos, etiquetas y monedas. En la figura 3.5 se puede ver distintos ejemplos de uso de la aplicación. Todos los menús, botones e información están en inglés, aunque se puede cambiar el idioma al español, así como predefinir el tipo de moneda. La precisión del reconocimiento se puede ver afectada por el pulso del usuario, la orientación del documento y la distancia al mismo. Para acceder a las imágenes del dispositivo, contiene una opción que permite acceder a la galería fotográfica del dispositivo y reconocer el contenido de la fotografía, ya sea un texto o una escena. Tiene otra opción de feedback que permite ponerse en contacto con los desarrolladores mediante el envío de un correo electrónico con el objetivo de proporcionar sugerencias o comunicar cualquier tipo de incidencia.

Seeing AI es una herramienta muy útil para las personas con discapacidad visual, ya que les permite explorar y comprender su entorno de una manera más accesible. Además, la aplicación también es útil para personas con daltonismo o cualquier otro tipo de discapacidad visual que les impida ver las imágenes claramente.

3.2.2. Facebook ATT

Facebook es una de las redes sociales más conocidas y utilizadas en el mundo, y por ello, siempre intentará atraer al mayor público posible. Una de las funciones

[8] `https://apps.apple.com/es/app/seeing-ai/id999062298`

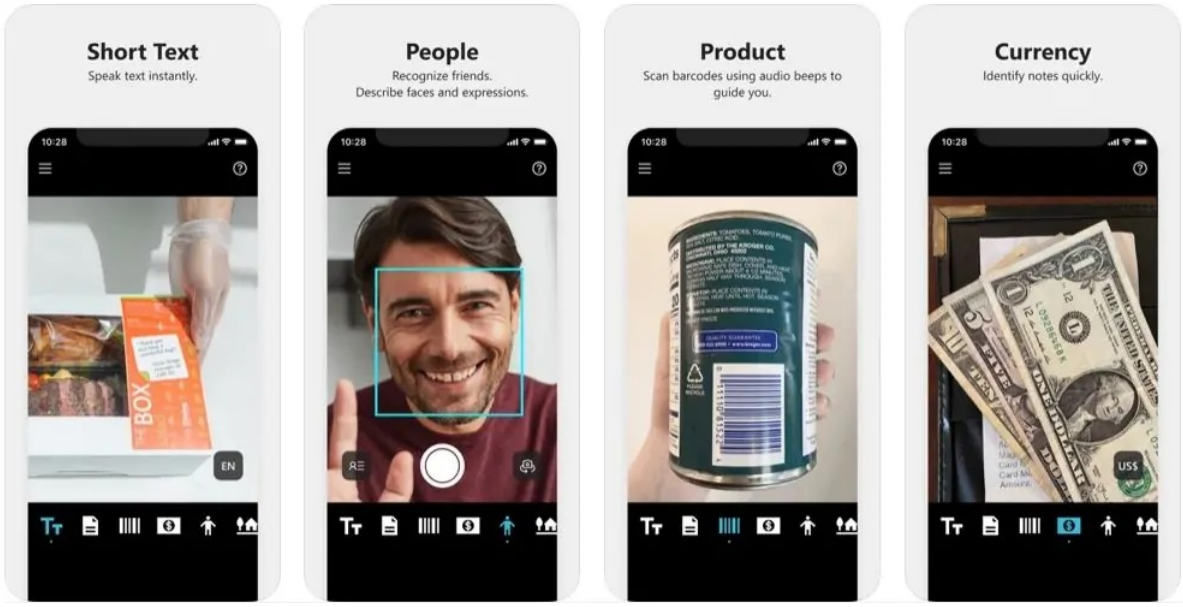

Figura 3.5: En el ejemplo se puede ver las distintas posibilidades que te ofrece Seeing AI.

más importantes de esta es subir fotos a tu muro y compartirla con tus seguidores. A las personas con discapacidad visual se les facilita el reconocimiento de imágenes a través de unas etiquetas establecidas por el usuario que sube la foto. Sin embargo, la mayoría de los usuarios no usan este texto alternativo, lo que llevó a *Facebook* a implementar una tecnología que genera descripciones de dichas fotos, pudiendo hacer descripciones detalladas e identificar acciones, personas, animales, personas...

Esta tecnología se denomina Texto Alternativo Automático (AAT[9]). El texto alternativo es una descripción que se añade a una imagen para ayudar a las personas ciegas o con visión reducida, como se puede ver en la figura 3.6. Si la persona que ha subido o publicado la imagen no incluye ningún texto alternativo, la tecnología de texto alternativo automático de *Facebook* utiliza visión e inteligencia artificial para crear automáticamente una descripción de la imagen. Es posible que el texto alternativo automático no esté siempre completo, pero el usuario que subió dicha imagen puede editarlo. En los últimos años esta tecnología ha mejorado mucho, llegando a indicar donde se ubican los distintos elementos que aparecen en la imagen. Además, dice de quien es la foto, cuándo se tomó y cuántos comentarios y *likes* tiene dicha foto. Se empezó tomando los textos generados por humanos, pero pasó a entrenar a una *IA* que permita hacer dichas descripciones, aunque daba problemas al reconocer algunos objetos. La versión final se convirtió en un modelo entrenado con datos supervisados parcialmente.

Algunas de las características de las descripciones que ofrece el AAT, de las cuales tomaremos en cuenta para nuestra aplicación, son:

- Descripciones breves y en pocas líneas, explicando solo los elementos más importantes de la imagen.

[9] `https://www.facebook.com/help/216219865403298`

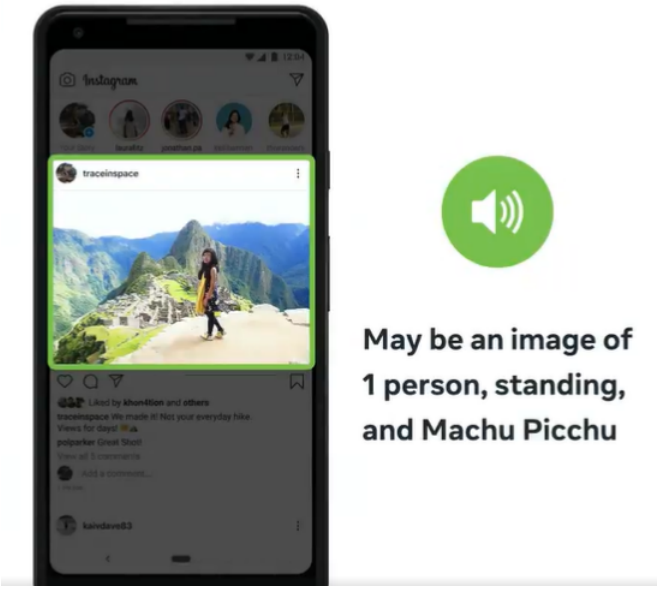

Figura 3.6: Descripción de la imagen producida por Facebook ATT.

- Opción de una descripción más extensa para aquellos casos en los que queramos un análisis de la imagen más detallada.
- Evitar descripciones innecesarias, como objetos de fondo, descripción de colores, descripciones poéticas o evitar signos de puntuación, para no hacer un texto excesivamente largo.
- El tamaño de los elementos es relativo a la distancia donde se hace dicha fotografía, por lo que se evitarán palabras como grande, pequeño, largo...

La descripción detallada de la imagen puede contener lo siguiente:

- Descripción del usuario: texto alternativo que ha escrito la persona que ha subido o publicado la imagen.
- Descripción generada por *Facebook*: texto alternativo generado automáticamente por *Facebook*.
- Texto en la imagen: cualquier texto que haya en la imagen, leído de arriba abajo y de izquierda a derecha.
- Información de posición: el lugar donde se encuentran los objetos en la imagen.
- Información de tamaño: se usa el tamaño de un objeto para determinar el enfoque de una imagen.
- Elementos por categoría: los objetos se clasifican por categorías como personas, plantas y objetos.

3.2.3. Be My Eyes

Be My Eyes[10] es una aplicación diseñada para ayudar a personas con discapacidad visual. La aplicación conecta a personas ciegas o con baja visión con voluntarios que pueden ayudar a resolver problemas cotidianos a través de una llamada de vídeo en vivo.

Los usuarios con discapacidad visual pueden solicitar ayuda de un voluntario en cualquier momento. La aplicación se encarga de encontrar un voluntario disponible para la llamada, y la llamada se realiza a través de la cámara del teléfono del voluntario. El voluntario puede ver lo que está sucediendo en el entorno del usuario con discapacidad visual y proporcionar información verbal sobre lo que está viendo.

Algunos ejemplos de cómo los voluntarios pueden ayudar incluyen leer etiquetas de productos, ayudar a encontrar objetos perdidos, describir imágenes en un sitio web, y ayudar a llenar formularios en línea.

Be My Eyes es una aplicación gratuita y está disponible para descargar en los dispositivos *iOS* y *Android*. La aplicación se ha convertido en una herramienta valiosa para las personas con discapacidad visual, y ha ayudado a muchas personas a realizar tareas diarias con más independencia y confianza.

3.2.4. TapTapSee

TapTapSee[11] es una aplicación móvil diseñada para personas ciegas o con baja visión. Funciona capturando imágenes con la cámara del dispositivo, como se puede ver en la figura 3.7, y utilizando tecnología de reconocimiento de imágenes para describir lo que está en la imagen. La aplicación puede describir objetos, personas, animales, texto y más.

Para utilizar *TapTapSee* simplemente hay que apuntar y hacer clic en la cámara para capturar una imagen. La aplicación luego proporciona una descripción verbal de lo que está en la imagen, lo que permite a la persona escuchar una descripción de su entorno y tomar decisiones informadas. La descripción vocal incluye información sobre el tipo de objeto, su posición en la imagen, su forma y otros detalles relevantes. Es una herramienta útil para mejorar la accesibilidad y la inclusión para personas ciegas o con baja visión, ya que les permite obtener información valiosa sobre su entorno sin depender de la ayuda de otras personas. La precisión de la descripción depende de la calidad de la tecnología de reconocimiento de imágenes.

[10] `https://www.bemyeyes.com/language/spanish`

[11] `https://play.google.com/store/apps/details?id=com.msearcher.taptapsee.android&hl=es&gl=US&pli=1`

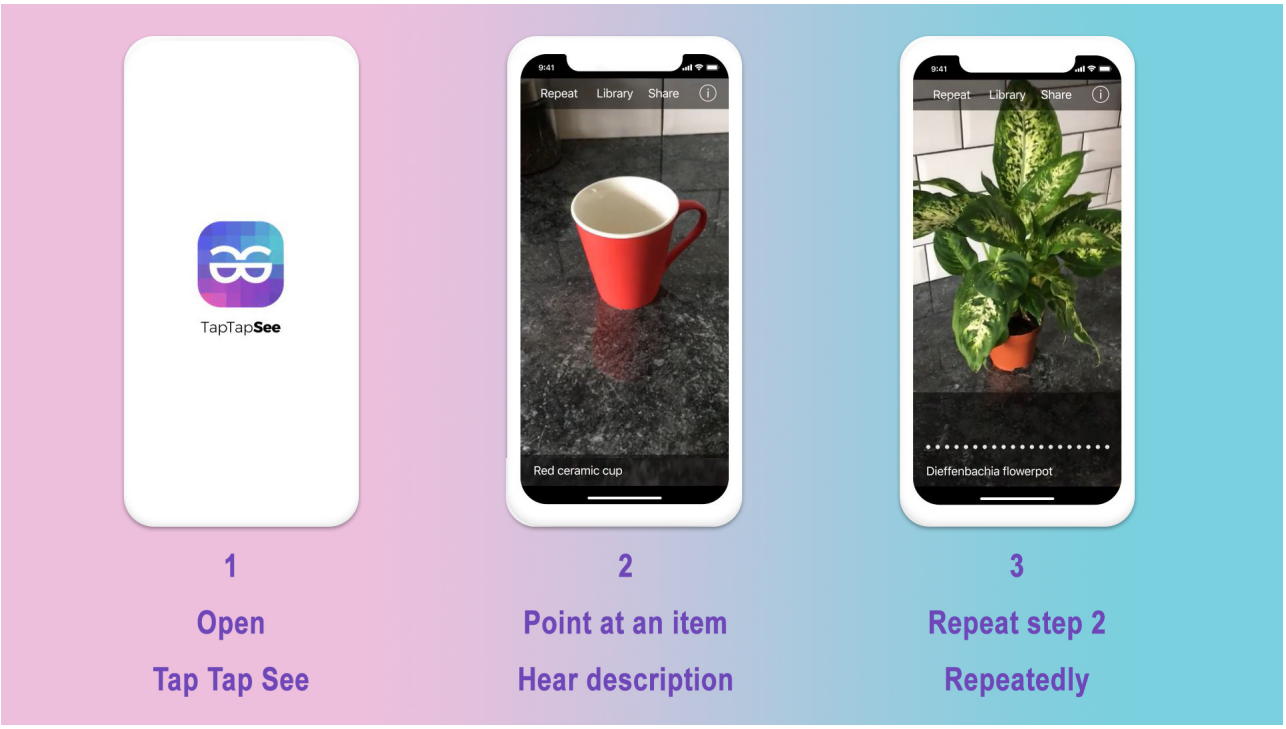

Figura 3.7: Captura de una imagen a través de TapTapSee.

Capítulo 4

Captura de requisitos

Nuestro trabajo está enfocado en el uso de dispositivos móviles en personas invidentes, por lo que conocer sus hábitos dentro de estas tecnologías, sus problemas y sus preferencias nos ayudará a entender y plasmar unos buenos requisitos para nuestra aplicación. Para ello, hemos realizado dos entrevistas diferentes. Una primera donde entrevistamos a tres personas reales con discapacidad visual (Sección 4.1) y una segunda donde entrevistamos a una experta en audiodescripción (Sección 4.2). Tras las entrevistas y un estudio del resultado de estas, hemos podido obtener una serie de requisitos que explicaremos en la última sección de este capítulo (Sección 4.3).

4.1. Entrevista a personas invidentes

La primera entrevista que realizamos fue a tres usuarios con discapacidad visual para conocer mejor los requisitos necesarios para la aplicación. Esta entrevista fue presencial en la Facultad de Informática de la Universidad Complutense de Madrid el día 29 de octubre del 2022. Fue una entrevista de aproximadamente dos horas, en la que uno de nosotros hacía las preguntas y los entrevistados intentaban contestarnos de la forma más clara y extensa posible. Estas respuesta intentaban poder ayudarnos a encontrar e identificar todos los requisitos que estas personas creían que eran importantes en las aplicaciones que ellos utilizaban y que debería tener la nuestra. Con el consentimiento de los participantes, la entrevista fue grabada para su futuro análisis.

Los personas entrevistadas fueron:

- Víctor Alberto Lorenzo (VA), tiene 40 años y es investigador social en asuntos de accesibilidad y discapacidad en la UNED.
- Gema Prieto (G), tiene 51 años, estudió empresariales y actualmente no tiene empleo.
- Margarita Burgueño (M), tiene 51 años, estudió historia y actualmente no tiene empleo.

4.1.1. Transcripción de la entrevista

A continuación hemos incluido una transcripción completa de la entrevista realizada.

1.¿Vive solo o convive con otra persona? ¿Padece una ceguera total o parcial? ¿Su ceguera es de nacimiento? Si no, ¿a qué edad perdió la vista?

- *VA: Padece ceguera total, solo percibe la luz.*
- *G: Perdió la vista hace 5 años. Padece ceguera total y solo percibe la luz.*
- *M: Perdió la vista hace 7 años. Tiene un resto visual de un 10 por ciento en un ojo y en otro únicamente percibe la luz y la oscuridad.*

2.¿Qué sistema operativo tiene su móvil (*iOS* o *Android*)? ¿Qué versión del software tiene su móvil?

- *VA: Utiliza un iPhone con sistema operativo iOS porque para él, son los móviles más accesibles para las personas invidentes. VoiceOver.*
- *G: Utiliza un iPhone con sistema operativo iOS, nunca ha utilizado un móvil con sistema operativo Android*
- *M: Antes de perder la vista utilizaba un móvil con sistema operativo Android y después se cambió a iOS porque le resulta más práctico.*

  Los tres utilizan *VoiceOver* que es un lector de pantalla que les describe toda acción de sus teléfonos.

3.¿Suele utilizar el móvil de forma frecuente en su día? ¿Para qué utiliza el móvil? ¿Qué problemas se encuentra?

- *VA: Utiliza su móvil fundamentalmente para la comunicación y las aplicaciones que más usa son WhatsApp, redes sociales como Twitter,Facebook, Linkedin. También utiliza aplicaciones específicas para el uso del transporte público. También utiliza Seeing AI para detectar escenas. También utiliza el móvil para lectura de libros, a través de una aplicación de la ONCE llamada Gestor ONCE libros digitales. Encuentra descripciones muy escasas en las imágenes que le envían.*
- *G: Utiliza poco el móvil, únicamente usa Whatsapp y lee algún correo electrónico. Alguna vez ha leído el periódico mediante el teléfono móvil.*
- *M: Utiliza mucho el móvil. Lo utiliza para ver su correo electrónico, WhatsApp, redes sociales como Twitter. También lee el periódico y escucha audio libros a través de él. Un problema con el que convive mucho es con que los botones de las aplicaciones están mal etiquetados, no se describe bien cual es la función de ellos. Descripciones de las personas con muy pocos detalles.*

4.¿Cómo utilizan su teléfono móvil?

- *Deslizar el dedo hacia la derecha para cambiar de aplicación y hacia la izquierda vuelve a la anterior aplicación.*
- *Tocar dos veces en la pantalla para abrir una aplicación.*
- *Tres dedos hacia arriba cerrar una aplicación.*
- *Mediante interfaz de voz.*

5.¿El no poder ver fotos o imágenes te supone un problema en tu día a día? De ser así, ¿cómo lo solventas?

- *VA: Depende de las descripciones que le da otra persona o de las descripciones de su móvil. Recalca de nuevo que las descripciones de las imágenes que obtiene de su móvil no son nada detalladas y que no le sirven de mucho.*
- *M: Como primera opción utiliza una lupa y si no lo consigue, pide ayuda a alguien de su familia.*

6.¿Qué necesita saber de una imagen?

- *VA: Cuantos más detalles mejor, quiere saber el máximo de detalle de una imagen. Primero una descripción general y luego una descripción más en detalle.*
- *G: Como mínimo les gustaría que la descripción les ubique, les permita tener una idea de la imagen.*
- *M: Le gustaría que la descripción le permitiera reconocer si hay algún familiar en la imagen y quien es.*

7.¿De dónde provienen las imágenes que recibís?

- *Los tres coinciden en que reciben las imágenes por WhatsApp, páginas web y correo.*

8.¿Tienes que lidiar con algún tipo de imagen en tu trabajo? si es así, ¿con qué tipo de imagen?¿cómo lo hace?

- *VA: Sí que tiene lidiar con imágenes, pero estas imágenes están ya etiquetadas por quien se las manda.*

9.¿Qué cosas cambiarías de las aplicaciones que conoces?¿Cómo te gustaría que fuese la aplicación que te describe las imágenes?

- *VA: Cambiaría que la imagen se describiera automáticamente. Le gustaría que a medida que vas moviendo el dedo por la imagen, en la pantalla del móvil, le fuese diciendo que va encontrando y se lo vaya describiendo. Tiene que estar diseñada para todo tipo de personas. Botones etiquetados.*
- *G: La aplicación no describe los colores correctamente. Le gustaría que la aplicación pusiera énfasis en aquello para lo que esté diseñada y que reconociera los colores básicos.*

- *M: Tiene que ser accesible, es decir, que una persona ciega sea capaz de poder usarla sin dificultad de más. Le gustaría que reconociese colores, sobre todo para la ropa. Que tenga un menú sencillo, sin demasiadas opciones.*

10.¿Cómo os gustaría que fuera la aplicación propuesta?

- *VA: Botones etiquetados y que diga qué hace cada acción. Estaría muy bien un pequeño tutorial que sea claro y útil que explique cómo usar la aplicación.*

- *M: Con un menú de uso sencillo, en el que no haya 17 opciones por las que tengas que pasar. Un registro sencillo en el que la contraseña no sea difícil y no obligue a poner caracteres especiales ni contraseñas muy largas en el que de tiempo a hacer el registro ya que nosotros tenemos que teclear y escuchar que hemos tecleado.*

11.¿En qué situaciones del día a día podría ayudaros la aplicación?

- *VA: Para cualquier foto que se mande por Whatsapp o en una página web.*

- *M: Para ser independiente. El no tener que preguntar a alguien cercano qué hay en la foto o preguntarse:¿Quién es?¿Qué es esto?¿Qué estoy viendo? Recalca que frustra mucho tener que preguntar.*

12.¿Pregunta o alguna observación de lo que hemos hablado?

- *VA:Nos pide tener en cuenta todos los perfiles de usuario. Puede ser que para algunos tenga mucha información y para otros tenga poca.*

- *G: Si necesitamos que prueben la aplicación o tenemos alguna pregunta o cualquier cosa nos dice que no dudemos en decírselo.*

- *M: Dice que está encantada de ser los primeros en probarlo. Recalca que cuando terminemos nuestras carreras, siempre pensemos en los usuarios con discapacidad visual para cualquier tipo de proyecto que hagamos. Que si algo pone que es accesible que sea de verdad, que no sea solo para que te pongan el tick de accesibilidad.*

4.2. Entrevista a experta en audiodescripción

En esta sección hablaremos sobre la entrevista realizada a la persona experta en audiodescripción. Su nombre es Mar, trabaja como docente en la universidad de Córdoba, y además impartió un curso de audiodescripción en la propia universidad. La entrevista se hizo el 1 de diciembre de 2022 y el método de entrevista fue similar a la anterior. Fue una entrevista que duró una hora en la que intentamos profundizar en la audiodescripción y cómo actualmente se desarrolla este tipo de descripciones en imágenes.

4.2.1. Transcripción de la entrevista

En esta subsección, incluiremos la transcripción completa de la entrevista a la experta.

1.¿En qué consiste tu trabajo y dicho curso?

- *Mar ha trabajado como descriptora audiovisual. En el curso, trabajaban la audio descripción en inglés y en español, explicando las partes más técnicas como por ejemplo, elaborar un guión. Enseñaba cómo elaborar una descripción correcta, sin ser demasiado escueta ni demasiado larga.*

2.¿Cómo definirías la audiodescripción a personas que nunca han tratado con ella?

- *Es un servicio de apoyo para la audiencia, para percibir toda la información visual. Crear conciencia de lo importante que es la información audiovisual.*

3.¿Qué pasos sigues para realizar una buena audiodescripción?¿Cómo sabes que es una buena descripción?

- *Analizar el texto audiovisual (todos los canales de comunicación), ya que hay información que es evidente y no es necesaria para no hacer la descripción redundante. Para Mar, una buena descripción tiene que ser directa, clara y a nivel léxico tiene que ser rica, no sirve utilizar un lenguaje básico.*

4.¿A qué detalles de las imágenes le das más importancia?¿Cuáles ignoras más?

- *Además de expresiones faciales, colores, posición de las personas en la imagen, paisajes, etc, hay que tener en cuenta factores culturales, sin sacar conclusiones ya que eso lo deberá hacer el usuario. Lo más importante es ser objetivo a la hora de elaborar una descripción.*

5.¿Con qué tipo de problemas te puedes encontrar al realizar una audio descripción?¿Cómo se resuelven?

- *El léxico es uno de los principales problemas, sobre todo cuando se tratan de estados de ánimo. Es irremediable que se pierda información.*

6.¿Tenéis algún tipo de soporte donde las personas invidentes os puedan dar un feedback?

- *No tienen ningún tipo de feedback por parte de personas invidentes.*

7.¿Una aplicación así podría resultar útil en tu trabajo?

- *Sería útil sin lugar a dudas.*

8.¿Cómo crees que sería mejor hacerla?

- *Habría que hacer descripciones que sean breves, claras y que aporten toda la información que se percibe. Sería muy interesante tener feedback de varias personas para tener diferentes perspectivas, y feedback de los propios usuarios.*

4.3. Conclusiones y requisitos

Tras hacer las entrevistas y realizar un estudio de los datos obtenidos, podemos sacar una serie de conclusiones y requisitos o funcionalidades que necesita nuestra aplicación.

En primer lugar, hemos podido ver que el sistema operativo que más utilizan las personas con discapacidad visual es *iOS*, debido a que este sistema de accesibilidad está mejor preparado y da más facilidades que otros sistemas operativos.

Uno de los mayores problemas que tienen dichas personas en la mayoría de aplicaciones es que estas no contienen buenas descripciones en los botones (etiquetas), por lo que les resulta sumamente costoso navegar por dichas aplicaciones, confundiéndose muchas veces en acciones y teniendo que volver y probar con otra opción.

Por otro lado, con el tema de la descripción de imágenes, hemos comprobado que las descripciones actuales de imágenes son demasiado básicas y no son claras, provocando que apenas se utilicen. Hablando con la experta y con las personas invidentes, hemos llegado a la conclusión de que estas deben contener la información suficiente para que la persona pueda hacerse una idea clara de qué contiene la foto, pero sin saturarla de información. Deben ser descripciones claras pero con nivel léxico alto, para intentar describir de forma más precisa los detalles.

Teniendo en cuenta todo lo anterior, listamos a continuación una serie de requisitos clave que debe tener nuestra aplicación:

- Un método para importar una imagen en una aplicación desde la galería.
- Botones en la aplicación con buenas etiquetas para que a la hora de la narración se sepa perfectamente qué botón estamos pulsando.
- Una inteligencia artificial entrenada que obteniendo una imagen devuelva una descripción de esta.
- Una forma en la que al tocar la foto, la aplicación recoja las coordenadas del toque y devuelva qué elemento se encuentra en dichas coordenadas.
- Un detector de colores que ayude a las personas invidentes a reconocer el color de la ropa.
- Un detector de caras conocidas con el que pueda detectar a las personas de una imagen, pudiendo etiquetar dichas personas de sus contactos.
- Incluir un lector de pantalla por voz que narre a la persona lo que está pasando en la aplicación. Esto incluye la narración de la descripción, la información de los elementos de la imagen al navegar en ella o los botones de la aplicación.

Capítulo 5

Implementación

En este capítulo vamos a hablar de la estructura del cliente y del servidor de la aplicación, de sus respectivas implementaciones y de la manera en la que se comunican. En la Sección 5.1 explicaremos cómo es la arquitectura del sistema de nuestro proyecto. En la Sección 5.2 hablaremos detalladamente del servidor, mientras que en la Sección 5.3 hablaremos de la aplicación.

5.1. Arquitectura general del sistema

En esta sección vamos a explicar brevemente cómo es la arquitectura interna de nuestro proyecto. En la figura 5.1 podemos ver un esquema visual de cómo es la arquitectura. La arquitectura se divide en:

- Cliente: puede elegir la imagen que será enviada al servidor. Cuando recibe la respuesta, el cliente procesa el *JSON* recibido y lo transforma en texto para que sea mostrado al usuario en forma de audio a través del altavoz del dispositivo. Además, tras recibir el *JSON* con la información de los elementos de la imagen del servidor, el usuario podrá tocar por la imagen. Internamente se valora si en dichas coordenadas existe un elemento en la imagen.

- Servidor: recibe la información del cliente (una imagen), la procesa internamente y devuelve la información de la imagen en forma de *JSON*. Dentro del servidor encontramos tres servicios diferentes:

    - Generador de descripciones: se encarga de recibir la imagen y devolver una descripción de esta. Este servicio manda la imagen a un modelo ya entrenado para la descripción de imágenes y devuelve un *JSON* con la descripción de la imagen.
    - Detector de entidades en imágenes: es el encargado de recibir la imagen y devolver un *JSON* con la información de todos los elementos de la imagen. El servicio está conectado a un modelo ya entrenado para la detección de elementos en imágenes, el cual envía a dicho modelo la imagen y este devuelve el *JSON* con toda la información.

- Traductor de texto a español: las dos APIs anteriores devuelven un *JSON* con la información en inglés. Esta *API* se encarga de recibir los *JSON* de dichas APIs y traducirlos al español. La respuesta también se hace en forma de *JSON*.

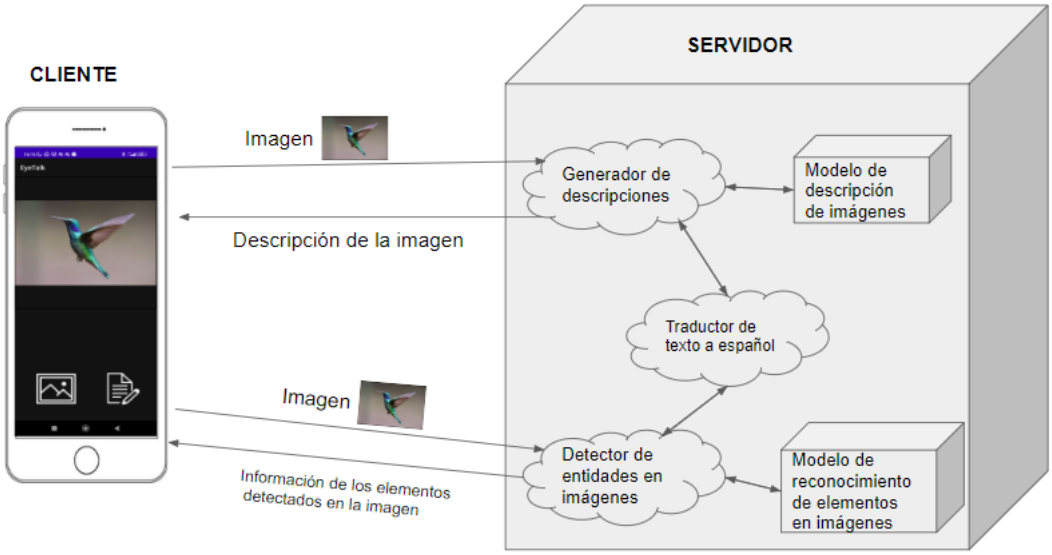

Figura 5.1: Representación de la arquitectura del proyecto

5.2. Servidor

En este proyecto vamos a utilizar un servidor escrito en *Python* llamado *Flask*[1]. *Flask* es un framework web que permite crear programas que se ejecutarán en el servidor. La razón por la que usamos un servidor es la necesidad de un componente que actúe de intermediario entre la aplicación móvil y ciertas APIs que son fundamentales para poder ofrecer el servicio que queremos a nuestros usuarios. En nuestro servidor hemos desarrollado dos funcionalidades, cada una de ellas asociada a un *endpoint* que conecta con el cliente. Estas funcionalidades son:

- Descripción de una imagen: Esta funcionalidad se encarga de crear una descripción completa de una imagen. Esta descripción se lleva a cabo gracias al uso de un modelo de entrenamiento que es capaz de extraer las características más relevantes de la imagen como la presencia de objetos,colores y patrones.
- Detección de entidades en una imagen: Esta funcionalidad se encarga de detectar todos los objetos encontrados en una imagen, además de las coordenadas donde se encuentran dichos objetos.

Para poder trabajar con las imágenes enviadas por el cliente ha sido necesario utilizar una biblioteca llamada *request*. Con esta biblioteca hemos podido cargar,

[1] `https://flask.palletsprojects.com/en/2.3.x/`

guardar y obtener la ruta de dichas imágenes, lo que es necesario para procesarlas correctamente y utilizarlas junto con los modelos de entrenamiento correspondientes. Todos los modelos de entrenamiento que utilizamos se encuentran cargados dentro del servidor para optimizar el tiempo de respuesta de este.

Cada una de las funcionalidades son explicadas en las siguientes subsecciones.

5.2.1. Descripción de una imagen

Esta funcionalidad, que se encarga de generar descripciones de imágenes se basa en el procesamiento de imágenes y el uso de modelos de lenguaje natural para producir una descripción textual de lo que aparece en la imagen. Para su implementación hemos tenido que investigar sobre modelos de lenguaje natural y hacer una exhaustiva búsqueda de dónde encontrarlos y cómo usarlos correctamente.

En esta búsqueda uno de los recursos más interesantes que descubrimos fue la plataforma de *Hugging Face*[2] (Jain, 2022). *Hugging Face* es una compañía que desarrolla software y herramientas para el procesamiento del lenguaje natural. Uno de sus productos más conocidos es una biblioteca de aprendizaje automático de código abierto *Transformers*, que permite a los desarrolladores utilizar y entrenar modelos de lenguaje natural pre-entrenados. *Hugging Face* también ofrece una plataforma en línea llamada *Hugging Face Hub* [3], donde los desarrolladores pueden compartir y colaborar en modelos de PLN y descargar modelos pre-entrenados para su uso en aplicaciones y proyectos.

Para el desarrollo de nuestro descriptor de imágenes, hemos escogido un modelo llamado *Salesforce/blip-image-captioning-large* (Li et al., 2022).Tras haber realizado diversas pruebas, hemos elegido este modelo ya que la descripción generada cumplía los requisitos extraídos en las entrevistas a nuestros usuarios, como se puede ver en la figura 5.2, donde se genera una descripción de una imagen de una persona caminando por la playa al atardecer, o en la figura 5.3 donde se genera una descripción de un entrecot con patatas.

Una vez elegido el modelo, investigamos cómo utilizar correctamente el modelo. Cada modelo de *Hugging Face* viene con su propia documentación, la cual es muy útil para poder implementarlo en un servidor. Inicialmente probamos el modelo en *Google Colaborate* que es una plataforma que permite hacer pruebas de *scripts* de una manera más sencilla y rápida. Para trabajar con cualquier modelo de esta plataforma es necesario instalar la biblioteca de *Transformers*.

En concreto para utilizar este modelo utilizamos dos clases de esta biblioteca llamadas *BlipProcessor* y *BlipForConditionalGeneration*. Su función es cargar el modelo pre-entrenado y obtener una instancia del mismo para generar una descripción a partir de una imagen. La respuesta de este modelo es un *JSONObject* compuesto por una clave-valor donde la clave es *generated_ text* y el valor es la descripción. Por ejemplo, para la figura 5.2 la respuesta del modelo será:

```
{
    "generated_text": "there is a man walking on the beach at
```

[2] https://huggingface.co/

[3] https://huggingface.co/docs/hub/index

```
        sunset"
}
```

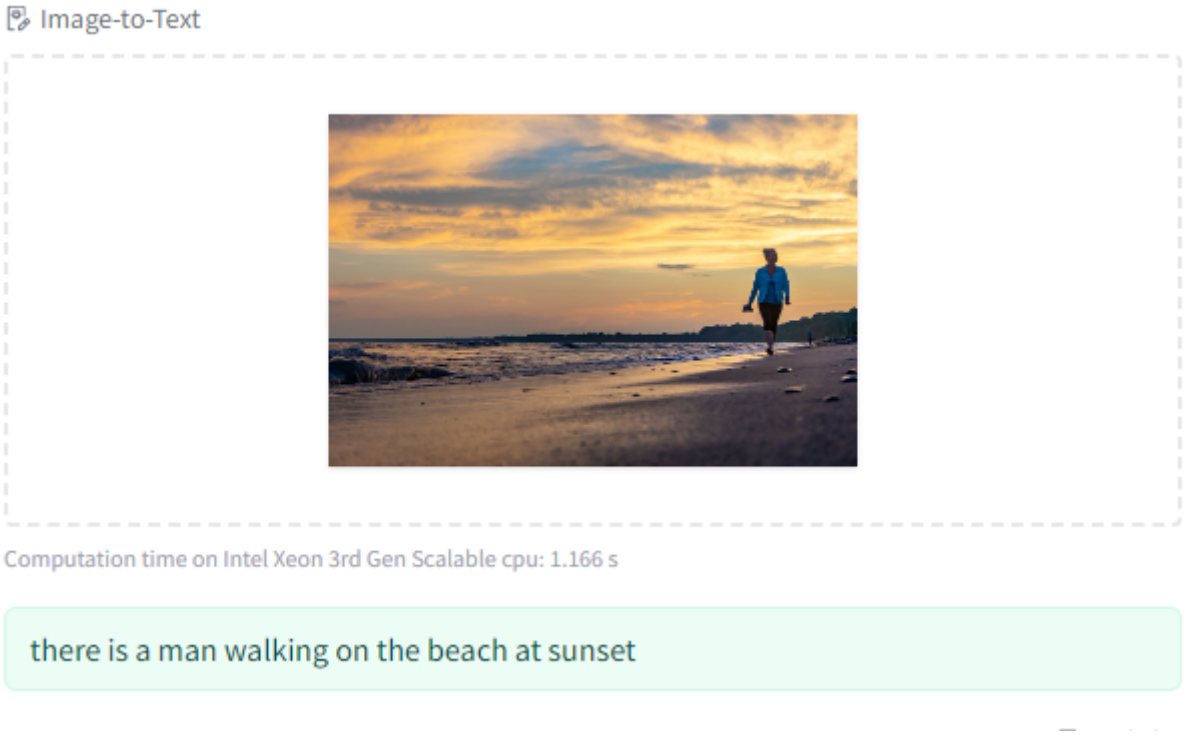

Figura 5.2: Ejemplo de uso del modelo *Salesforce/blip-image-captioning-large*.

5.2.2. Detección de entidades en la imagen

Esta funcionalidad tiene como objetivo detectar todos los objetos presentes en una imagen. Para cada objeto necesitamos un modelo que nos reconozca qué clase de objeto es y sus coordenadas en la misma imagen. Para ello, utilizamos dos enfoques diferentes. El primer enfoque consistió en utilizar una plataforma llamada *RoboFlow* que ofrece una variedad de modelos de detección de objetos. El segundo enfoque fue utilizar los modelos de detección de objetos de la plataforma *Hugging Face*, que nos proporcionaron resultados más precisos.

5.2.2.1. Detección de objetos con RoboFlow

Hemos elegido *RoboFlow* como nuestra primera opción porque se ha posicionado como una plataforma líder en el entrenamiento y despliegue de modelos de detección de objetos.

Nuestro primer paso en esta plataforma fue buscar un modelo que fuese capaz de reconocer un único objeto en una imagen. Escogimos un modelo capaz de detectar entre un perro, un gato o un pájaro. Este modelo se llama *dogs-cats-and-birds*. En la figura 5.4 evaluamos este modelo en la página web de *RoboFlow* con una imagen de

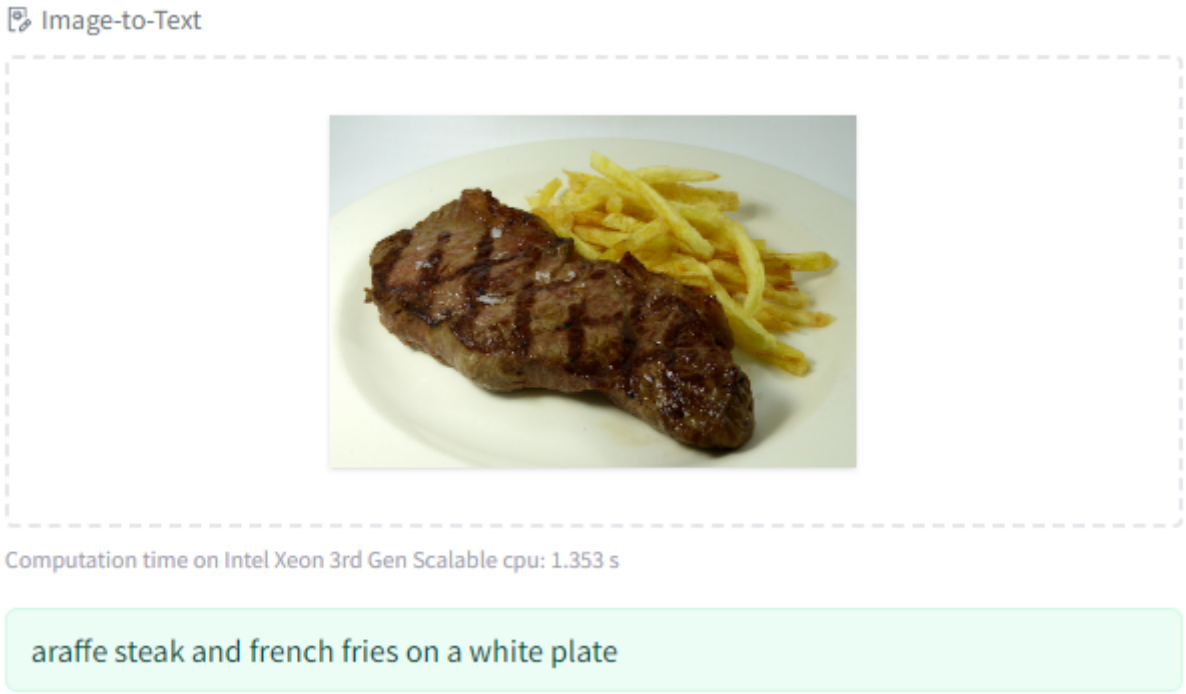

Figura 5.3: Ejemplo de uso del modelo *Salesforce/blip-image-captioning-large*.

un perro, y se puede verificar que identifica el perro de manera adecuada al marcar un cuadrado a su alrededor.

Este modelo tiene como salida un objeto *JSON* con la siguiente forma:

```
{
    {'predictions': [{'x': 320.0, 'y': 211.0, 'width': 640.0,
       'height': 422.0, 'confidence': 0.9894294738769531, '
       class': 'Dog'}], 'image': {'width': '640', 'height':
       '422'}}

}
```

RoboFlow, además de tener una gran cantidad de modelos tiene la ventaja de que proporciona mucha documentación sobre cómo implementar dichos modelos en tu código. Aprovechando esta ventaja, utilizamos *Google Colaborate* para probar dicho código y comprobar su correcto funcionamiento. Para poder utilizar estos modelos de detección de objetos es necesario instalar un paquete llamado *roboflow* que contiene todo lo necesario para usar sus modelos.

Nuestro siguiente objetivo consistió en buscar un modelo que tuviera la capacidad de detectar todos los objetos presentes en una imagen. Para ello, llevamos a cabo una exhaustiva búsqueda en la plataforma de RoboFlow donde encontramos un modelo que es capaz de detectar todos los perros o gatos presentes en una imagen. Este modelo se llama *cats_dogs_and_wild_animals*. Como se puede ver en el ejemplo de funcionamiento 5.5 reconoce todos los elementos de la imagen. Este modelo tiene como salida un *JSONObject* compuesto de un *JSONArray* con el nombre de la clase y

coordenadas de cada objeto en la imagen. Se puede ver aquí la salida correspondiente a este ejemplo:

```
{
  {'predictions': [{'x': 123, 'y': 278, 'width': 109, 'height
     ': 96, 'confidence': 0.8431802988052368, 'class': 'cat-
     British_Shorthair'}, {'x': 731, 'y': 162, 'width': 118, '
     height': 117, 'confidence': 0.8351427316665649, 'class':
     'cat-Ragdoll'}, {'x': 516, 'y': 108, 'width': 154, '
     height': 187, 'confidence': 0.8154898881912231, 'class':
     'dog-boxer'}, {'x': 124, 'y': 276, 'width': 104, 'height
     ': 96, 'confidence': 0.48035940527915955, 'class': 'cat-
     Russian_Blue'}], 'image': {'width': '1000', 'height':
     '484'}}
}
```

La finalidad de esta funcionalidad es detectar todo tipo de objetos en una imagen, para ello la mejor opción es encontrar un modelo que utilice un dataset de *COCO*. Sin embargo, dentro de *RoboFlow* no encontramos ningún modelo acorde con nuestras necesidades. Por lo que tuvimos que buscar otras alternativas como la plataforma de *Hugging Face*.

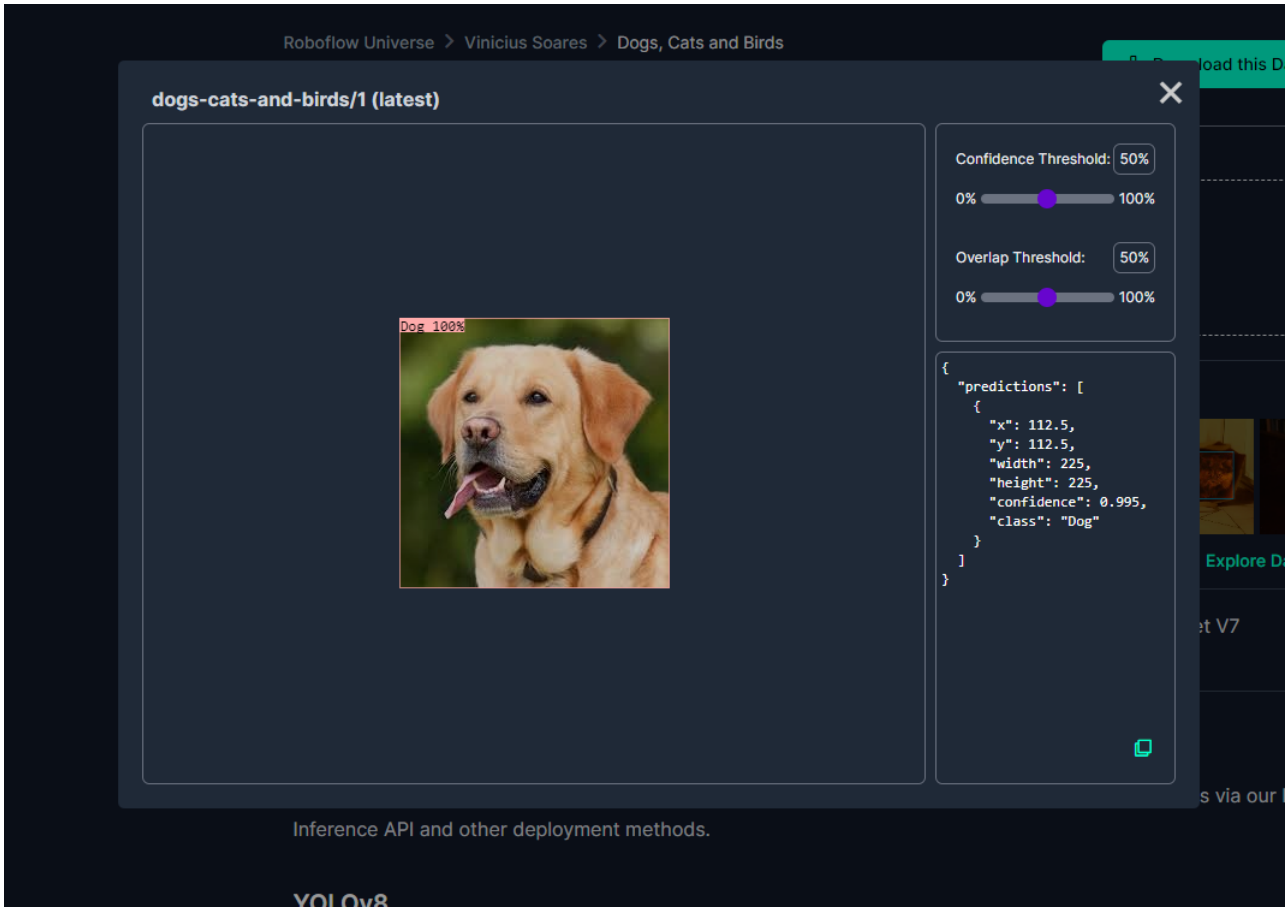

Figura 5.4: Ejemplo donde el modelo seleccionado reconoce un objeto en una imagen.

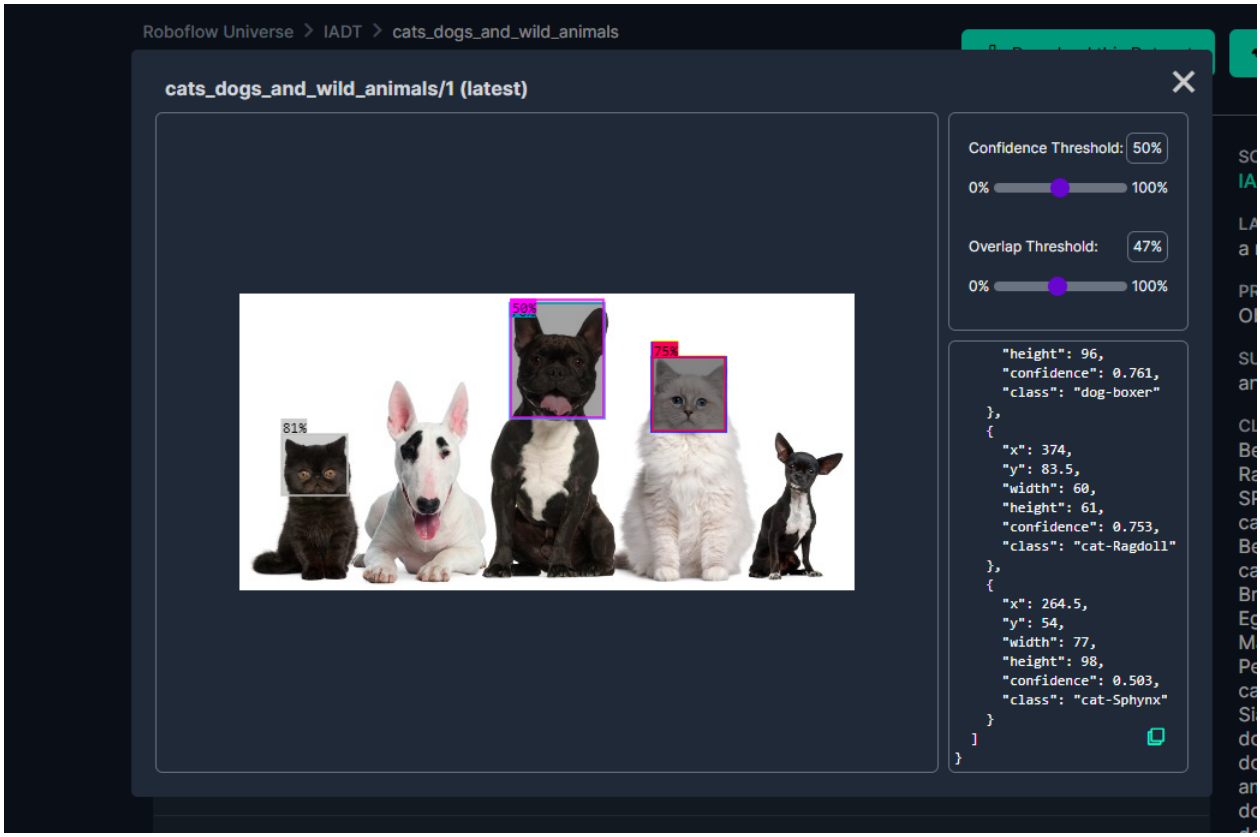

Figura 5.5: Ejemplo donde el modelo seleccionado reconoce varios objetos en una imagen.

5.2.2.2. Detección de objetos con Hugging Face

Dentro de *Hugging Face*, en la sección llamada *Hugging Face Hub*, encontramos numerosos modelos de detección de objetos. Entre ellos encontramos un modelo que fue entrenado con el conjunto de datos *COCO* 2017, el cual consiste en un conjunto de 118k/5k imágenes anotadas para entrenamiento/validación, respectivamente.

Gracias a este modelo llamado *facebook/detr-resnet-101* (Carion et al., 2020) pudimos ampliar la funcionalidad a la detección de todo tipo de objetos. Como ya hemos comentado en la anterior funcionalidad, para el uso de cualquier modelo de esta plataforma es necesario tener instalada la biblioteca *Transformers*. En la figura 5.6 podemos observar un ejemplo de uso del modelo en la web de *Hugging Face* donde es capaz de reconocer distintos objetos marcándolos con un cuadrado. Una vez hubimos probado con distintas imágenes, escogimos este modelo como nuestro modelo final en esta funcionalidad. El resultado de este modelo está compuesto por este *JSONObject* por cada elemento:

```
{
  {
    "score": 0.9979230761528015,
    "label": "bus",
    "box": {
      "xmin": 373,
```

```
      "ymin": 199,
      "xmax": 463,
      "ymax": 267
    }
}
```

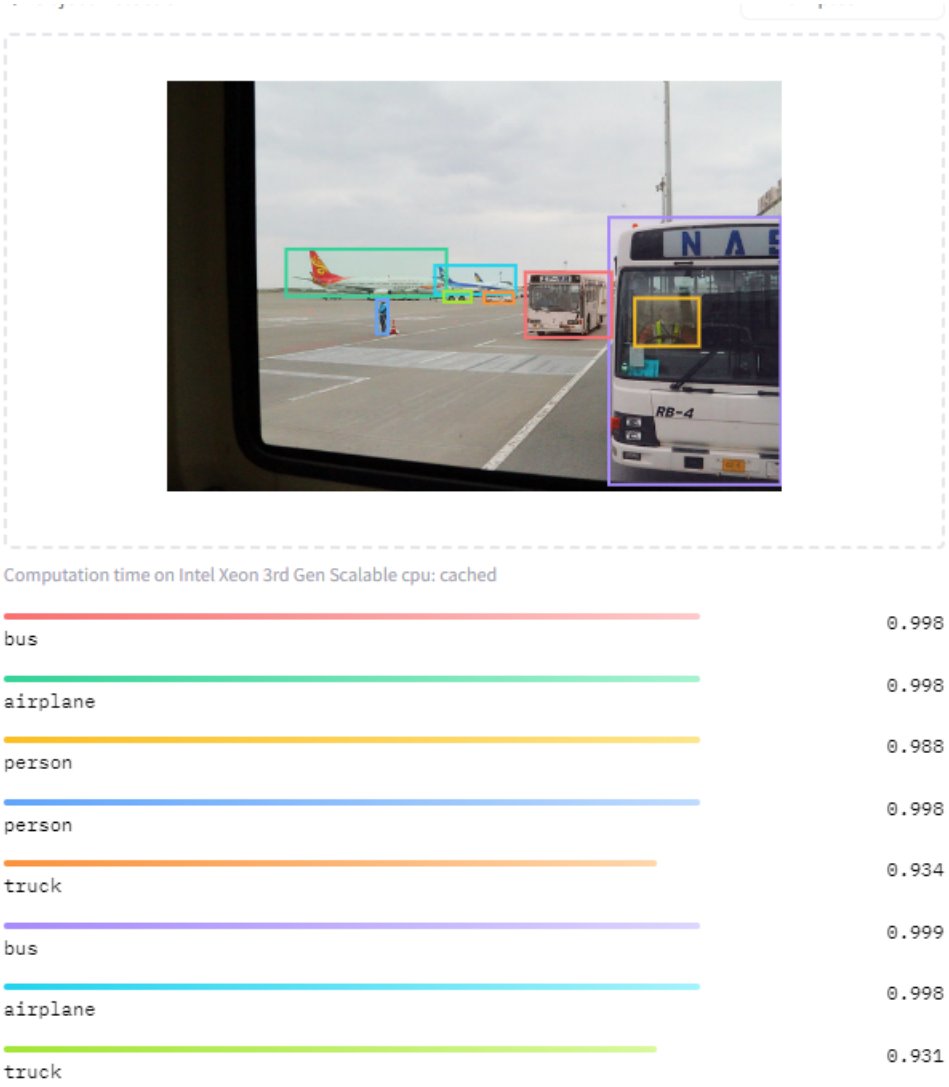

Figura 5.6: Ejemplo de uso de modelo en la pagina de Hugging Face.

5.2.3. Traducción de las salidas de los modelos

Dado que las salidas de todos los modelos utilizados en la implementación de las funcionalidades del servidor se encuentran en inglés, hemos tenido que buscar un modelo de traducción para satisfacer las necesidades de nuestro público objetivo, que requieren tanto la descripción, como la detección de objetos, en español.

Para poder usar este modelo, utilizamos la biblioteca *Transformers* de Hugging Face para crear un objeto *pipeline* que se encarga de la traducción automática de

texto en inglés al español, utilizando un modelo pre-entrenado específico y un límite máximo de longitud de secuencia.

Se utiliza la función *pipeline* de la biblioteca *Transformers* para crear un objeto que realiza la tarea de traducción. En este caso, se especifica que la tarea es la traducción del idioma inglés al español, utilizando el modelo *Helsinki-NLP/opus-mt-en-es* como argumento.

Una vez creado este objeto lo podemos utilizar para traducir en ambas funcionalidades.

5.3. Aplicación móvil

A la hora de desarrollar la aplicación móvil, tomamos la decisión de desarrollarla en *Android*, a través de *Android Studio* y usando Java como lenguaje de programación. A pesar de que *iOS* posee más herramientas y está más preparado para personas con discapacidad, hemos elegido programar en *Android* ya que no tenemos la posibilidad de programar en *iOS* por la falta de herramientas para ello.

La aplicación, sencilla y funcional, tiene las siguientes características:

- Una interfaz sencilla, intuitiva y fácil de manejar. En esta interfaz es muy importante que los botones y acciones estén muy bien etiquetados y sean fácilmente accesibles. Las personas invidentes, gracias al lector por voz, pueden navegar cómodamente por la aplicación.
- Una conexión a un servidor remoto, para enviar la foto elegida y las coordenadas de la foto señaladas por el usuario.
- Las voces que se encargan de reproducir el significado de los distintos botones de la aplicación.

A continuación describimos en detalle cada una de estas partes de la aplicación.

5.3.1. Interfaz de la aplicación

Como se puede ver en la figura 5.7, una vez abierta la aplicación, encontramos una interfaz bastante sencilla. Una vez abierta la aplicación podemos encontrar dos botones: uno para abrir la galería de imágenes para cargar la imagen y otro para generar una descripción. Los dos botones son de un tamaño que le permite al usuario acceder a ellos sin mayor dificultad. Una vez insertada la imagen, como se puede ver en la figura 5.8, la aplicación informará al usuario a través de un sistema de voz que la imagen ha sido cargada correctamente y que la descripción ha sido generada. Una vez generada la descripción, se escuchará la descripción de la imagen. En el caso de que el usuario deseara volver a escuchar la descripción, tendría que pulsar el botón correspondiente. Por ejemplo, el usuario escuchará: “Una fotografía de un colibrí volando en el aire con sus alas extendidas.” Si el usuario pulsara encima del colibrí, escuchará “ave”.

A la hora de desarrollar los botones, hemos tenido en cuenta los requisitos que nos proporcionaron las personas que fueron entrevistadas. Los botones son sencillos

de seleccionar, y cada uno de ellos tiene su función bien marcada. Las etiquetas de los botones son claras, para evitar cualquier tipo de confusión. Una vez abierta la imagen, además de poder pulsar los dos botones disponibles, el usuario puede pulsar en la imagen para que la aplicación le indique, mediante el lector de voz, qué objeto está seleccionando.

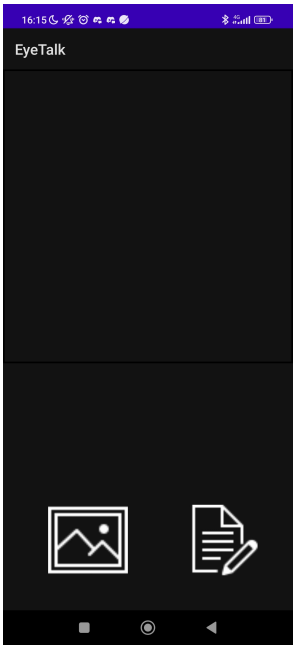

Figura 5.7: Pantalla incial de la aplicación.

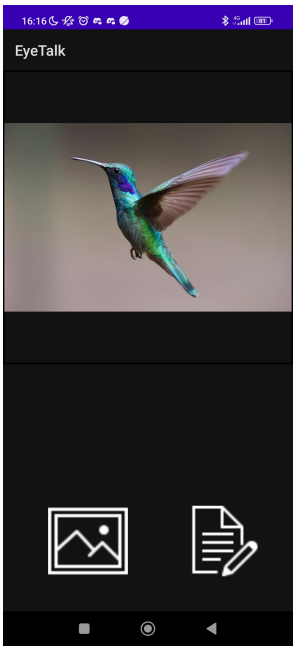

Figura 5.8: Pantalla de la aplicación una vez insertada la imagen.

5.3.2. Lector de voz

Utilizamos la tecnología *TextToSpeech* que nos proporciona *Android Studio* para escuchar la descripción de la imagen. Es una herramienta de accesibilidad que permite convertir el texto en voz. Su funcionamiento se divide en dos etapas:

- Preprocesamiento: Durante esta etapa, se realiza una revisión del texto para identificar los elementos lingüísticos y gramaticales, y se aplican reglas de pronunciación y entonación para producir una representación fonética del texto.
- Síntesis de voz: Se utiliza una base de datos de sonidos pregrabados para producir la voz sintetizada.

También se puede configurar para ajustar la voz, el idioma y la velocidad de lectura. La voz y el idioma se pueden seleccionar de una lista de voces e idiomas disponibles.

5.3.3. Conexión con el servidor e implementación

La comunicación entre la aplicación y el servidor, se realiza a través de peticiones *HTTP* tipo *POST*. Con una única petición obtenemos los datos de los objetos de la imagen y su descripción. Una vez obtenidos estos datos, los almacenamos en la aplicación. De esta manera, el usuario tendrá acceso a ellas, sin necesidad de elevar el número de llamadas al servidor. Esto reduce la sobrecarga de datos y el tiempo de respuesta. Dentro de la aplicación podemos encontrar dos clases:

- *ObjectDetector.java*: Esta clase va a ser la encargada de enviar la imagen al servidor. A través de esta clase, obtenemos la descripción y los datos necesarios de la imagen.
- *Descriptor.java*: Esta clase se encargará de gestionar la descripción de la imagen. Ajustará el idioma de la descripción, para que se escuche en castellano, y la velocidad de reproducción.

En la figura 5.9 se puede observar un diagrama de secuencia el cual explica cómo es la conexión entre el cliente y el servidor y nos sirve como representación del comportamiento del sistema.

Como podemos observar, el cliente crea una clase principal llamada *MainActivity* donde se encuentra la mayoría de la lógica de nuestra aplicación. Tras abrir la galería y elegir una foto, se crea un objeto de clase *ObjectDetector*. Este objeto tiene un método *execute*. Este método conecta con el servidor, carga la imagen que hay dentro de la llamada *execute* y se la envía a un modelo de reconocimiento de elementos en imágenes. Esto devuelve un *JSONList* con la información de los elementos que se han detectado en la imagen pero este está en inglés, por lo que se le envía a un traductor que devuelve dicho *JSONList* en español. Por último el *JSONList* es devuelto al cliente.

Tras recibir la información de los elementos de la imagen, el cliente crea un objeto de la clase *Descriptor*. Este objeto tiene un método *execute* que le envía al servidor la imagen. El servidor le pasa dicha imagen a un modelo de descripción de

imágenes que devuelve un *JSON* con la descripción en inglés. Este *JSON* se envía a un traductor para traducirlo al español y se le devuelve al cliente.

Si tocamos el botón de descripción, el cliente obtendrá del objeto de clase *Descriptor* un texto de la descripción que sera leído por el altavoz del teléfono.

Por otro lado, si se toca en la imagen, el cliente obtendrá la información de los elementos que se encuentran en la imagen atreves del objeto de clase *ObjectDetector*. Internamente el cliente comprueba si en las coordenadas que se ha tocado en la imagen existe un elemento y si es así, coge el texto del elemento y lo narra por el altavoz del teléfono.

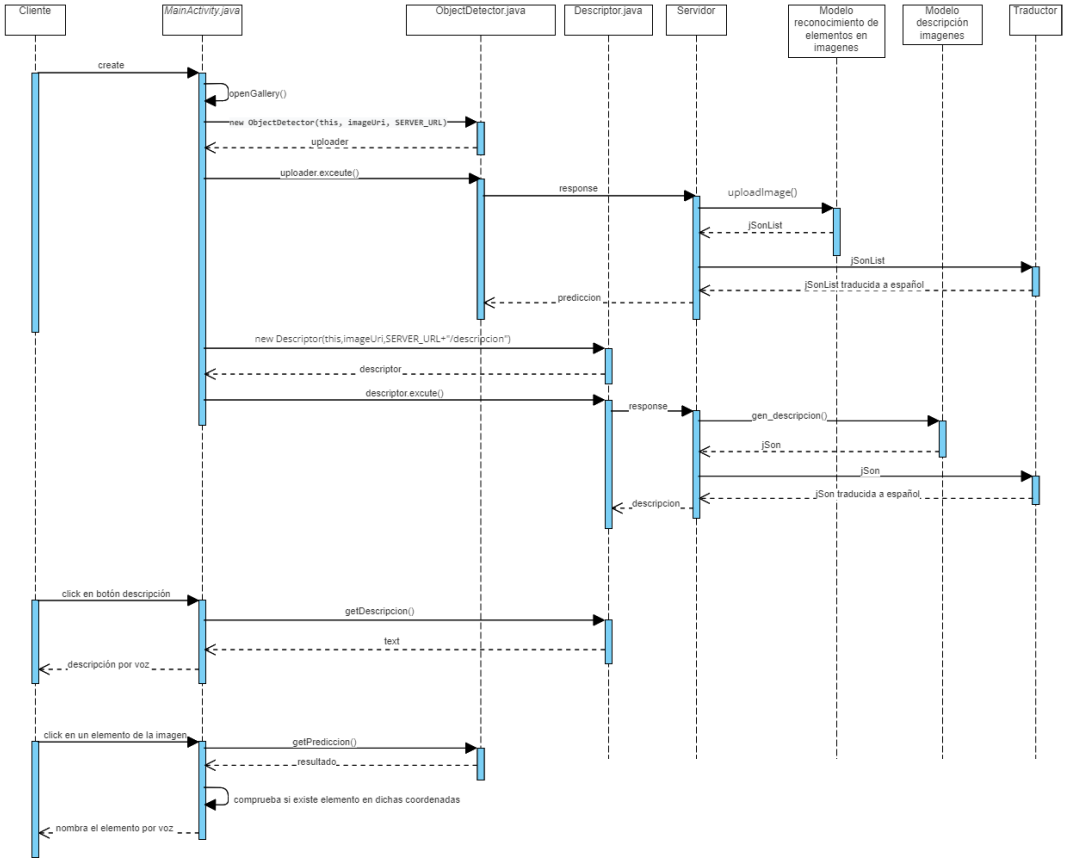

Figura 5.9: Diagrama de secuencia del funcionamiento de la aplicación.

Capítulo 6

Evaluación

En este capítulo vamos a explicar cómo se ha realizado la evaluación de nuestra aplicación y las conclusiones que hemos obtenido de esta. Primero contaremos en la Sección 6.1 el diseño de la evaluación. En la Sección 6.2 hablaremos del desarrollo de la evaluación y en la Sección 6.3 el cuestionario SUS. Por último, en la Sección 6.4 se expondrá las conclusiones que hemos obtenido de la evaluación.

6.1. Diseño de la evaluación

Hemos realizado una evaluación a las mismas personas que se prestaron para responder a nuestras preguntas en la entrevista (Sección 4.1). Esta reunión fue presencial en la Facultad de Informática de la Universidad Complutense de Madrid y se grabó para poder revisar posteriormente los comentarios de los evaluadores con más detenimiento. Nuestros usuarios son Víctor Alberto como primer usuario, Marga como segundo usuario y Gema como tercer usuario. Van a probar el funcionamiento de nuestra aplicación donde cada uno cargará tres imágenes, escucharán su descripción y navegarán por la imagen para identificar cada objeto. Dentro de estas imágenes haremos una distinción por niveles (bajo, medio y alto), donde la diferencia entre cada nivel será la complejidad de su descripción y el número de objetos en la imagen.

Las imágenes elegidas para esta evaluación son las mostradas en la figura 6.1.

Después de la ejecución de cada imagen les realizaremos algunas preguntas para saber qué han podido percibir gracias a la navegación y descripción. Con estas preguntas queremos saber su opinión acerca de la descripción generada por la aplicación y que descripción serían capaces de hacer nuestros usuarios tras dicha ejecución.

Tras la ejecución de todas las imágenes, nuestros usuarios finalmente responderán a un cuestionario de usabilidad SUS con las siguientes preguntas donde tendrán que seleccionar una puntuación entre el 1 y el 5 dependiendo de lo de acuerdo que están con la afirmación o negación:

- Me gustaría usar esta aplicación frecuentemente.
- Considero que la aplicación es innecesariamente compleja.
- Considero que la aplicación es fácil de usar.

Figura 6.1: Imágenes utilizadas para la evaluación

- Considero necesario el apoyo de personal experto para poder utilizar esta aplicación.
- Considero que las funciones de la aplicación están bien integradas.
- Considero que la aplicación presenta muchas contradicciones.
- Imagino que la mayoría de las personas aprenderían a usar esta aplicación rápidamente.
- Considero que el uso de esta aplicación es tedioso.
- Me sentí muy confiado/a al usar la aplicación.
- Necesité saber bastantes cosas antes de poder empezar a usar esta aplicación.

Esta evaluación nos ayudara a encontrar tanto puntos positivos como puntos negativos dentro nuestra aplicación.

6.2. Desarrollo de la evaluación

En esta sección vamos a explicar el desarrollo de la evaluación que se realizó a las personas con discapacidad visual con las que tuvimos la reunión. Como se ha explicado en la Sección 6.1, los usuarios probaron la aplicación, enseñándoles el

funcionamiento de esta en una serie de imágenes previamente elegidas y ordenadas por niveles (figura 6.1). Por cada imagen de cada usuario será una subsección. Cada subsección se muestra la imagen seleccionada, la descripción generada por la aplicación, los elementos detectados por esta, la descripción del usuario y finalmente las valoración del usuario después de obtener la información.

6.2.1. Primera imagen del primer usuario (figura 6.2)

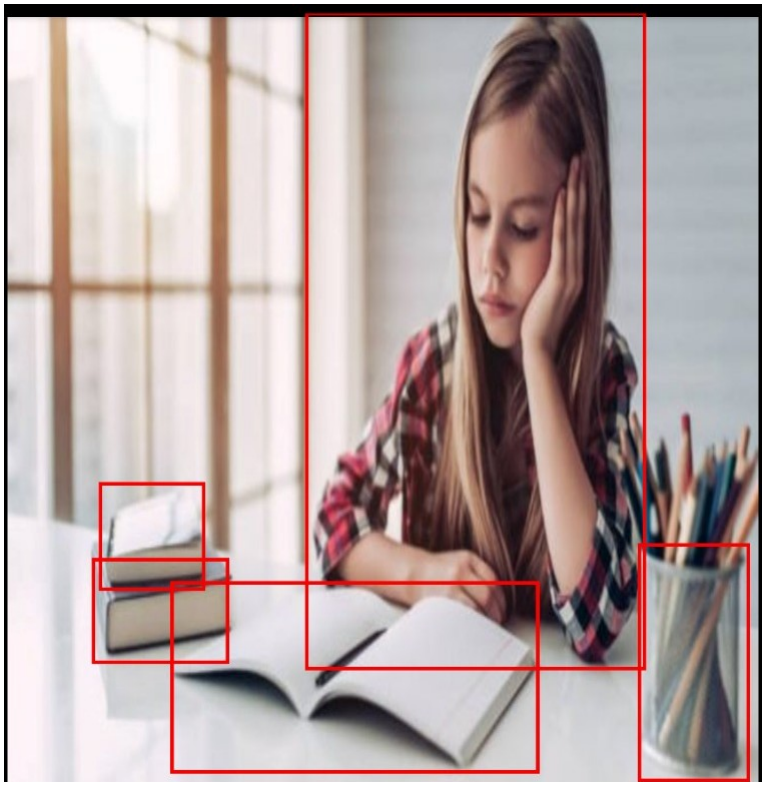

Figura 6.2: Primera imagen de Víctor Alberto.

- **Descripción:** Una fotografía de una chica sentada en una mesa con un libro y lápices.
- **Elementos detectados por la aplicación:**
  - Libro
  - Libro
  - Libro
  - Persona
  - Taza
- **Descripción del usuario:** Una chica en una mesa que está desayunando.

- **Respuesta del usuario:** Víctor Alberto dice que le ha ayudado a comprender la imagen pero que podría haberle ayudado más ya que no sabe donde están los objetos. Ha sabido detectar los objetos pero no donde están. No sabe si están en una mesa o si están en el suelo. Además no sabe que está haciendo la chica, si por ejemplo está estudiando o está desayunando.

6.2.2. Primera imagen del segundo usuario (figura 6.3)

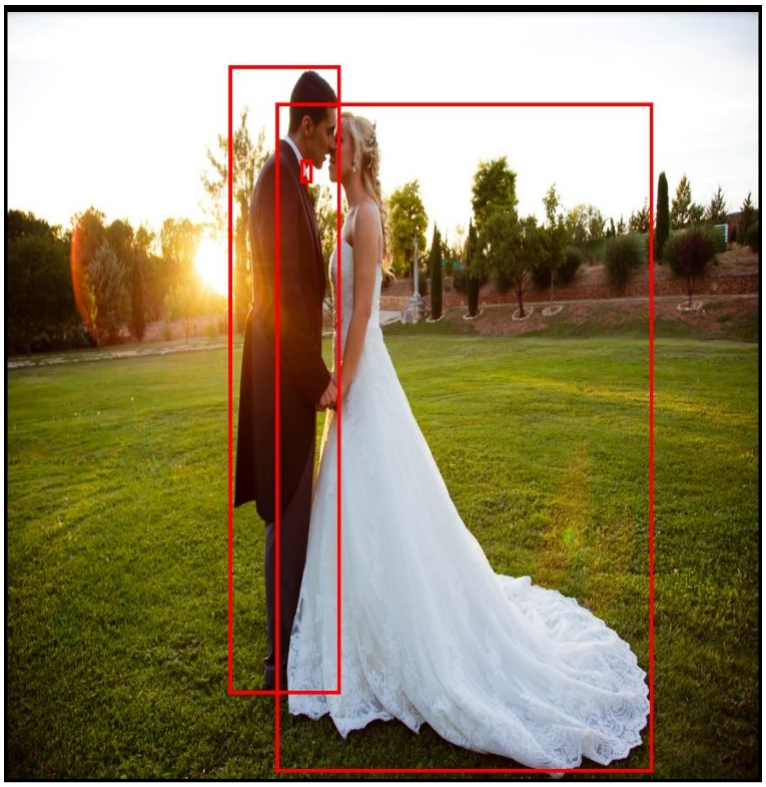

Figura 6.3: Primera imagen de Margarita.

- **Descripción:** Una fotografía de una novia y un novio besándose en un campo.
- **Elementos detectados por la aplicación:**
  - Persona
  - Persona
- **Descripción del usuario:** Una novia muy grande que le está dando un beso a su marido el día de su boda en un campo.
- **Respuesta del usuario:** Margarita dice que hay una persona muy grande que ocupa la mayoría de la imagen. La aplicación únicamente dice persona, por ello no puede identificar quién es quién ni tampoco si hay más personas en la fotografía.

6.2.3. Primera imagen del tercer usuario (figura 6.4)

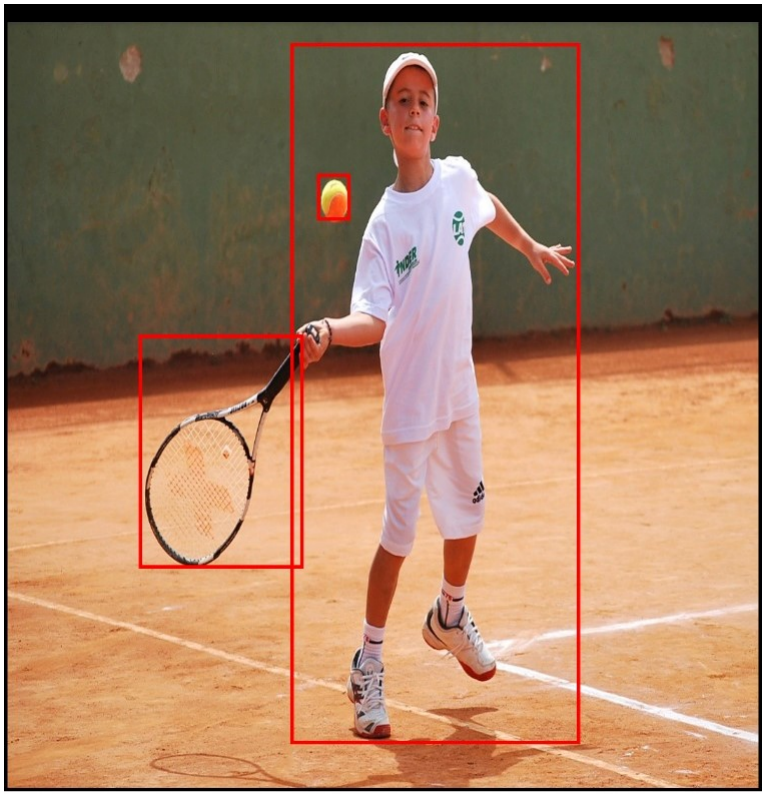

Figura 6.4: Primera imagen de Gema.

- **Descripción:** Una fotografía de un joven golpeando una pelota de tenis con una raqueta de tenis.
- **Elementos detectados por la aplicación:**
  - Raqueta de tenis.
  - Pelota deportiva.
  - Pelota.
- **Descripción del usuario:** Un niño dándole a una pelota con una raqueta de tenis con la derecha.
- **Respuesta del usuario:** Gema dice que la descripción está muy bien pero no dice donde está. Si está en un campo de tenis, en un parque o en un frontón. Describe la figura central pero no describe el fondo.

6.2.4. Segunda imagen del primer usuario (figura 6.5)

- **Descripción:** Una fotografía de una mujer sentada en una mesa con un portátil.

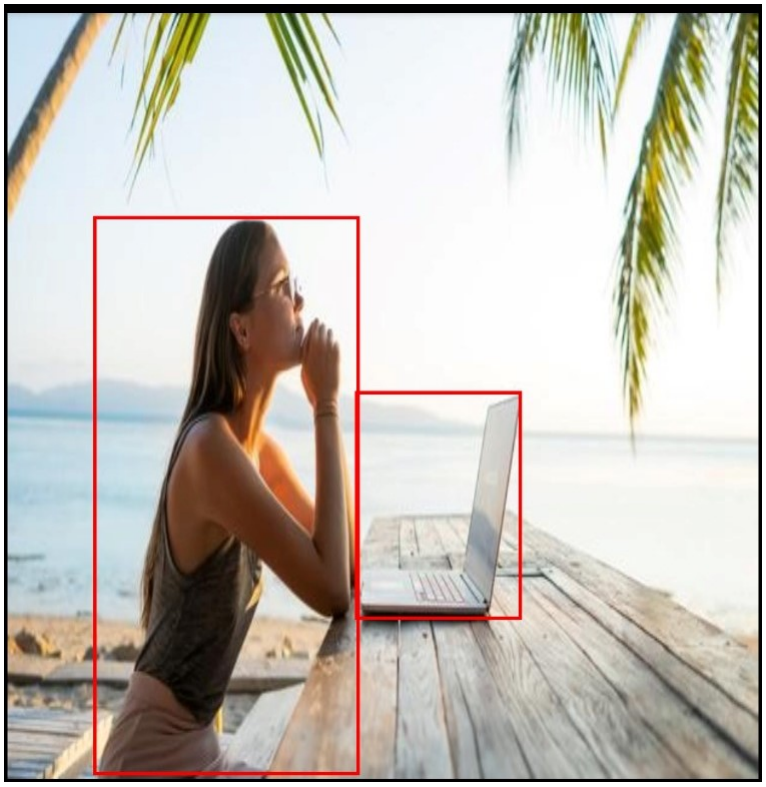

Figura 6.5: Segunda imagen de Víctor Alberto.

- **Elementos detectados por la aplicación:**
  - Persona
  - Portátil
- **Descripción del usuario:** Una chica de perfil que está sentada con un ordenador portátil que está sobre la mesa.
- **Respuesta del usuario:** Víctor Alberto dice que la chica está de perfil porque se lo ha imaginado así. No detecta la mesa y hay problemas cuando los objetos están muy juntos. Sigue sin saber que está haciendo la chica. Si está jugando o está trabajando. Tampoco sabe si está en casa, al aire libre o en una oficina. Le falta contexto.

6.2.5. Segunda imagen del segundo usuario (figura 6.6)

- **Descripción:** Una fotografía de dos vacas en un campo con un cielo azul.
- **Elementos detectados por la aplicación:**
  - Vaca
  - Vaca
- **Descripción del usuario:** Dos vacas paciendo en el campo.

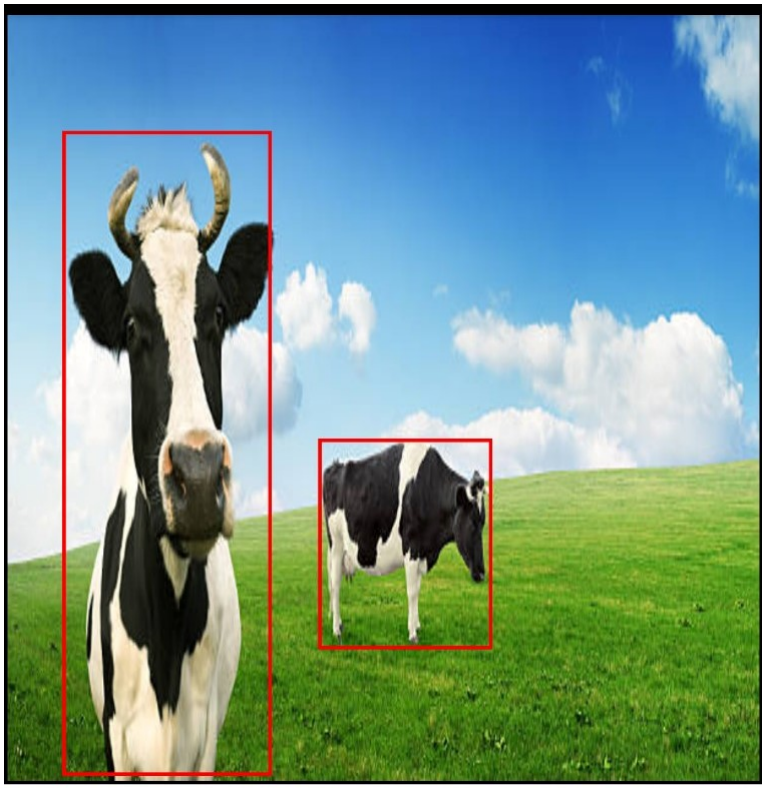

Figura 6.6: Segunda imagen de Margarita.

- **Respuesta del usuario:** Margarita comenta que no ha sabido identificar las dos vacas debido a que la aplicación solo dice vaca y están muy juntas, por lo que solo reconoce a una. Echa en falta distinguir entre una vaca y otra. Echa en falta una descripción del entorno, como el color del campo.

6.2.6. Segunda imagen del tercer usuario (figura 6.7)

- **Descripción:** Una fotografía de una niña leyendo un libro en una biblioteca.
- **Elementos detectados por la aplicación:**
  - Persona
  - Libro
- **Descripción del usuario:** Una persona en horizontal con un libro muy grande.
- **Respuesta del usuario:** Gema agradece que en esta descripción si explica donde se encuentra la niña. Dice que el libro es muy grande, debido a que este ocupa gran parte de la imagen. Al ocupar tanto, se hace a la idea de que la fotografía es en primer plano.

Figura 6.7: Segunda imagen para Gema.

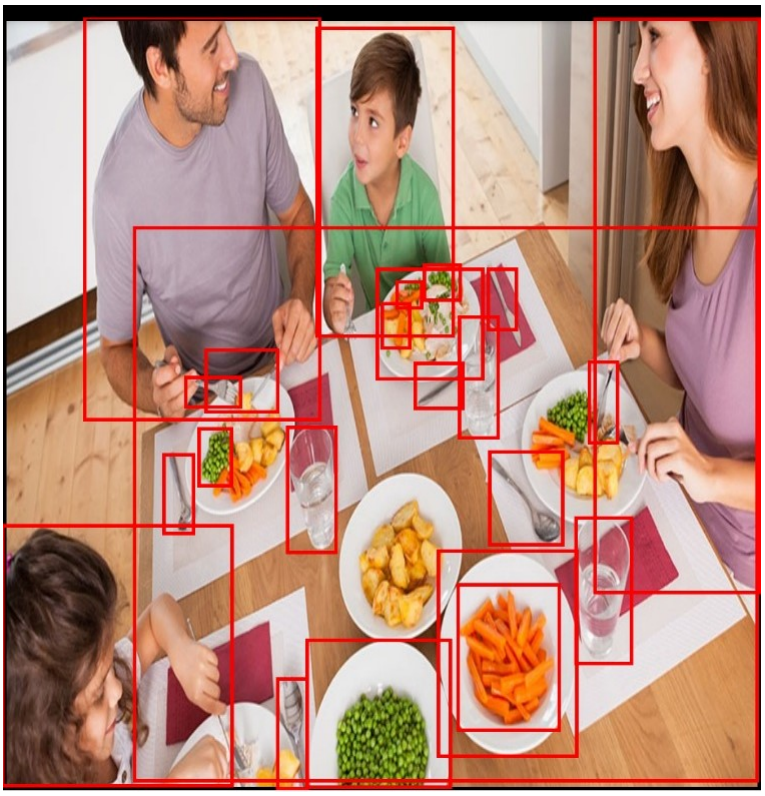

Figura 6.8: Tercera imagen de Víctor Alberto.

6.2.7. Tercera imagen del primer usuario (figura 6.8)

- **Descripción:** Una fotografía de una familia sentada en una mesa cenando.
- **Elementos detectados por la aplicación:**
  - Mesa de comedor
  - Tazón
  - Tazón
  - Tazón
  - Tazón
  - Taza
  - Taza
  - Taza
  - Taza
  - Cuchillo
  - Cuchillo
  - Tenedor
  - Tenedor
  - Guisantes
  - Persona
  - Persona
  - Persona
  - Persona
- **Descripción del usuario:** Una familia que está alrededor de la mesa con el cuchillo, el tenedor y el tazón desayunando.
- **Respuesta del usuario:** Víctor Alberto ha deducido que la familia está desayunando debido a que en vez de decir plato, la aplicación ha dicho tazón. No sabe la cantidad de personas que hay, ya que al tocar cualquier persona, solo te dice "persona".

6.2.8. Tercera imagen del segundo usuario (figura 6.9)

- **Descripción:** Una fotografía de dos policías de pie junto a un coche.
- **Elementos detectados por la aplicación:**
  - Persona
  - Persona
  - Coche

Figura 6.9: Tercera imagen de Margarita.

- **Descripción del usuario:** Es un coche de policía con dos policías de pie, uno a cada lado del coche.
- **Respuesta del usuario:** Margarita da una descripción perfecta de la imagen. Dice que con la descripción ha identificado que las personas son policías y que gracias a navegar por la foto, ha identificado que el coche está en medio de las dos personas. Margarita sigue recalcando que le falta información del entorno.

6.2.9. Tercera imagen del tercer usuario (figura 6.10)

- **Descripción:** Una fotografía de un hombre sentado en un sofá con un perro.
- **Elementos detectados por la aplicación:**
  - Persona
  - Perro
  - Sofá
- **Descripción del usuario:** Una persona de lado sentado en un sofá con un perro pequeño.
- **Respuesta del usuario:** Gema detecta el perro como un perro pequeño aunque sea más grande, pero acierta en que ambos están sentados de lado.

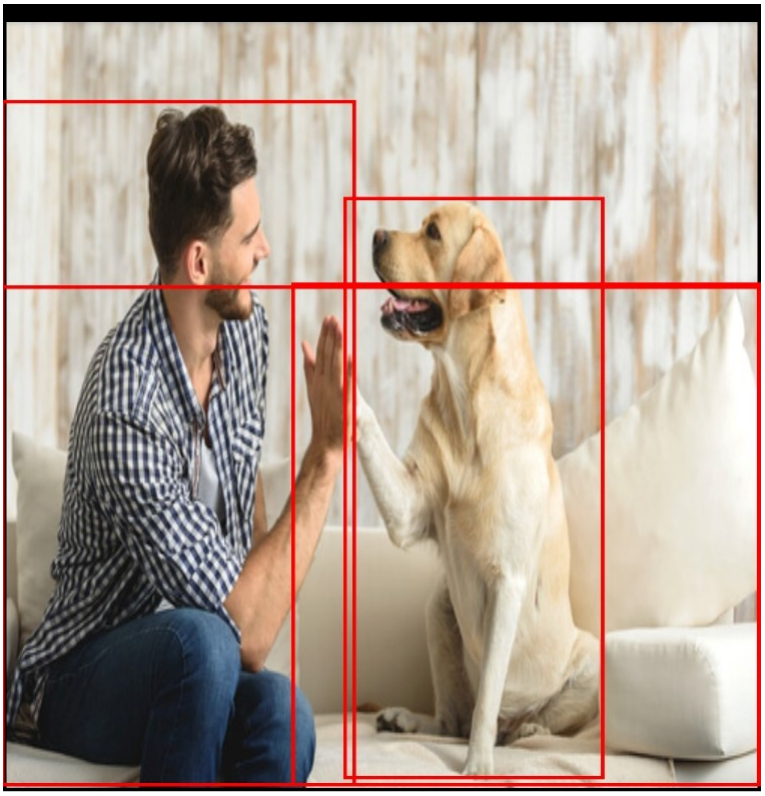

Figura 6.10: Tercera imagen de Gema.

6.3. Cuestionario SUS

El cuestionario SUS [1] es una herramienta que sirve para evaluar la usabilidad de de nuestra aplicación. Estos resultados nos ayudan a obtener información sobre la experiencia de nuestros usuarios utilizando nuestra aplicación y en nuestro caso, a conocer si nuestra interfaz realmente es accesible para personas con discapacidad visual. Estos son los resultados obtenidos:

[1] https://www.uifrommars.com/como-medir-usabilidad-que-es-sus/

	Evaluadores		
Cuestionario SUS	Alberto	Gema	Marga
Me gustaría usar esta aplicación frecuentemente	3	4	4
Considero que la aplicación es innecesariamente compleja	3	1	2
Considero que la aplicación es fácil de usar	4	4	4
Considero necesario el apoyo de personal experto para poder utilizar esta aplicación	2	1	2
Considero que las funciones de la aplicación están bien integradas	3	3	4
Considero que la aplicación presenta muchas contradicciones	4	4	2
Imagino que la mayoría de las personas aprenderían a usar esta aplicación rápidamente	4	4	4
Considero que el uso de esta aplicación es tedioso	3	1	2
Me sentí muy confiado/a al usar la aplicación	4	4	4
Necesité saber bastantes cosas antes de poder empezar a usar esta aplicación	2	2	2

Puntuación de cuestionario SUS:

- Víctor Alberto:

  Respuestas enunciados impares: (3 + 4 + 3 + 4 + 4) = 18 – 5 = 13

  Respuestas enunciados pares: (3 + 2 + 4 + 3 + 2) = 25 – 14 = 11

  Cálculo del SUS: (13 + 11) * 2,5 = 60

- Gema:

  Respuestas enunciados impares: (4 + 4 + 3 + 4 + 4) = 19 – 5 = 14

  Respuestas enunciados pares: (1 + 1 + 4 + 1 + 2) = 25 – 9 = 16

  Cálculo del SUS: (14 + 16) * 2,5 = 75

- Marga:

  Respuestas enunciados impares: (4 + 4 + 4 + 4 + 4) = 20 – 5 = 15

  Respuestas enunciados pares: (2 + 2 + 2 + 2 + 2) = 25 – 10 = 15

  Cálculo del SUS: (15 + 15) * 2,5 = 75

Cálculo final: (60 + 75 + 75) / 3 = 70

El rango de puntuaciones del cuestionario SUS va desde 0 hasta 100. Nuestro resultado ha sido de 70, lo que nos indica una buena usabilidad percibida por parte de los usuarios.

6.4. Conclusiones de evaluación

Tras haber escuchado a los tres usuarios, hemos obtenido una serie de puntos positivos y negativos de nuestra aplicación. Los puntos positivos son los siguientes:

- En términos generales, les ha resultado una aplicación sencilla. Nos comentaron que únicamente con los dos botones era suficiente para ellos.
- No necesitan de gran ayuda extra para usar la aplicación.
- La navegación por la imagen a partir del tacto les ha resultado útil. Les ha permitido, en algunas de las imágenes, tener una idea mental de la posición de los objetos. Por ejemplo, en la imagen 6.9 Marga fue capaz de situar a los policías y al coche.
- Por lo general, las descripciones generadas por nuestra aplicación proporcionan suficiente información. Las descripciones también dan información de las acciones. Por ejemplo, en la imagen 6.4 la descripción te informa de que es un niño jugando a la pelota.
- No necesitan un conocimiento previo complejo para usar la aplicación. Simplemente con una pequeña introducción de la aplicación les fue suficiente.

Los puntos negativos son:

- Algunos objetos mencionados en la descripción no son localizables navegando por la imagen. Por ejemplo, en la imagen 6.6 la descripción te menciona el cielo azul y el campo. Sin embargo, en ningún momento se localiza el cielo o el campo cuando navegan por la imagen.
- No hay diferenciación entre personas, especies de animales u objetos. Cuando en las imágenes hay más de una persona, más de un animal de la misma especie o más de un objeto igual, el identificador es el mismo. Por ejemplo, en la imagen 6.3 tanto el novio como la novia tienen el identificador de "persona".
- Falta de detalle cuando navegan por la imagen. Además de lo comentado en el punto anterior, los objetos que aparecen únicamente tienen un identificador asignado, es decir, no hay una pequeña descripción de estos que les permita a los usuarios tener un idea más detallada.
- Poca información del entorno. Resaltaron la importancia de conocer en todas las imágenes el entorno. Por ejemplo en la imagen 6.5, la descripción no da ningún detalle del lugar en el que está la persona.
- Si el usuario navega fuera de la imagen, no tiene información de que no está navegando en ella. Muchas veces tocaban en zonas de la pantalla donde no había nada. Al no haber un mensaje que les informara de que en esa zona no se encontraba la imagen, se desorientaban y perdían el tiempo.

- Cuando dos objetos se superponen, se pierde información de uno de ellos. Por ejemplo, en la imagen 6.10 es muy complicado localizar el sofá a partir de la navegación, ya que estaban superpuestos el perro y la persona con el sofá.
- El tiempo de carga de las imágenes es bastante elevado. Nos dimos cuenta de que tenían que esperar demasiado para escuchar la descripción y empezar a navegar por la imagen.
- En la imagen 6.8, se identifican los guisantes como brócoli. Aunque solo sucediera con esa imagen de las que probamos, nos lleva a pensar que es posible que suceda en otras imágenes en las que no hemos probado.
- Tiene problemas de compatibilidad con la funcionalidad *TalkBack* de *Android*. Tuvimos que desactivarlo ya que *TalkBack* impedía que se escuchase la descripción.

Capítulo 7

Conclusiones y Trabajo Futuro

En este capítulo se recogen las conclusiones obtenidas tras la realización y el análisis del proyecto, y se proponen algunas tareas de trabajo futuro.

7.1. Conclusiones

Nuestro propósito con este proyecto era desarrollar una aplicación móvil capaz de generar descripciones y de identificar objetos en una imagen. Para conseguirlo, hemos intentado cumplir de la mejor manera todos los objetivos que nos planteamos al inicio de este proyecto.

Entendimos que para que esta aplicación fuera realmente útil, lo más importante era centrarnos en lo que necesitaba el usuario. Para ello, entrevistamos a tres personas con discapacidad visual, les mostramos nuestra idea de proyecto y les hicimos una serie de preguntas relacionadas con el manejo de imágenes en su día a día. Con esto pudimos recoger ideas y algunos requisitos que debería tener una buena aplicación de este estilo. Considerando que uno de nuestros elementos fundamentales es la descripción de una imagen nos pusimos en contacto con una experta en audiodescripción para que nos explicase cuales son los elementos indispensables en una descripción. Con esto finalizamos con la captura de requisitos de nuestros usuarios.

La parte más tediosa de este proyecto ha ido relacionada con la investigación de técnicas de inteligencia artificial. Al no tener ninguna experiencia previa en este mundo, lo primero que hicimos fue tratar de entender qué es un dataset, qué es un modelo de entrenamiento y para qué sirven. Tras este proceso de entendimiento, pasamos al proceso de exploración donde encontramos plataformas muy útiles donde poder experimentar con modelos de entrenamiento de detección de objetos y de descripción de imágenes.

Tras estudiar y experimentar con dichas técnicas, tuvimos que aprender conceptos básicos del lenguaje de Python ya que nuestro servidor está implementado mediante código Python. Para conseguir integrar los modelos de entrenamiento en nuestro servidor decidimos utilizar la herramienta de Google Colaborate para comprobar su correcto funcionamiento.

Para conseguir que la interfaz de nuestra aplicación sea sencilla y accesible, ini-

cialmente observamos las interfaces de las aplicaciones descritas anteriormente y tras la evaluación, conseguimos información sobre algunas mejoras que podrían implementarse en un futuro.

7.2. Trabajo Futuro

Una vez finalizado el desarrollo de nuestra aplicación, hemos observado una serie de aspectos que se podrían mejorar:

- Desarrollar una versión de esta aplicación para dispositivos iOS, ya que la mayoría de las personas invidentes utilizan dispositivos móviles con este sistema operativo.
- Añadir una funcionalidad que permita reconocer objetos y colores a tiempo real a través de la cámara del móvil. De esta manera los usuarios podrían elegir con más precisión la ropa que van a vestir o comprar.
- Tras numerosas pruebas, hemos comprobado que el tiempo de espera del servidor a la hora de cargar la imagen, generar la descripción y detectar los objetos es de 25 segundos aproximadamente. Una posible mejora se conseguiría utilizando modelos más rápidos u optimizando el código del servidor.
- Añadir la posibilidad de seleccionar el idioma y la velocidad de reproducción de la descripción y del nombre del objeto.

Tras haber tenido la evaluación de la aplicación, como hemos explicado anteriormente, hemos obtenido una serie de posibles mejoras por parte de los usuarios que probaron la aplicación. Dichas mejoras son las siguientes:

- En lugar de seleccionar la imagen a través de la galería, añadir una funcionalidad que les permita cambiar de imagen deslizando hacia la derecha o izquierda o deslizando hacia arriba o abajo.
- Emitir un mensaje de aviso cuando el usuario toca en una zona de la pantalla donde no hay imagen. Esto les permite una mayor agilidad a la hora de navegar por ella.
- Diferenciación de personas, animales y objetos. Si en una imagen hay varios animales de una misma especie, varias personas o varios objetos iguales, asignarles un identificador distinto para cada uno de ellos.
- Situar los botones en las esquinas inferiores, ya que la mayoría de las aplicaciones que utilizan tienen los botones situados en esas posiciones y les es más fácil utilizar las esquinas como referencia.
- Descripciones más detalladas cuando navegan por la imagen. Es importante que los elementos de la imagen tengan, además de un identificador, un pequeña descripción que les ayude a tener una idea más clara.

- Mejorar la aplicación para que reconozca los entornos tanto en la descripción como en la navegación de la imagen.
- Implementar una funcionalidad que te dicte los objetos existentes en la imagen de izquierda a derecha o de arriba a abajo.

Capítulo 8

Conclusions and Future Work

In this chapter, the conclusions obtained from the execution and analysis of the project are presented, and some tasks for future work are proposed.

8.1. Conclusions

Our purpose with this project was to develop a mobile application capable of generating descriptions and identifying objects in an image. To achieve this, we have tried to fulfill all the objectives we set at the beginning of this project in the best possible way.

We understood that in order for this application to be truly useful, the most important thing was to focus on what the user needed. To do this, we interviewed three visually impaired individuals, showed them our project idea, and asked them a series of questions related to handling images in their daily lives. This allowed us to gather ideas and some requirements that a good application of this kind should have. Considering that one of our fundamental elements is image description, we contacted an expert in audio description to explain to us what essential elements a description should include. With this, we concluded the collection of user requirements.

The most tedious part of this project has been related to researching artificial intelligence techniques. Since we had no previous experience in this field, the first thing we did was try to understand what a dataset is, what a training model is, and what they are used for. After this process of understanding, we moved on to the exploration phase where we found very useful platforms to experiment with object detection and image description training models.

After studying and experimenting with these techniques, we had to learn basic concepts of the Python language since our server is implemented using Python code. To integrate the training models into our server, we decided to use the Google Colaboratory tool to verify their proper functioning.

To make the interface of our application simple and accessible, we initially observed the interfaces of previously described applications and, through evaluation, obtained information about some improvements that could be implemented in the future.

8.2. Future Work

Once the development of our application was completed, we identified several aspects that could be improved:

- Develop a version of this application for iOS devices, as the majority of visually impaired individuals use mobile devices with this operating system.
- Add functionality that allows real-time object and color recognition through the mobile camera. This way, users could more accurately choose the clothes they are going to wear or buy.
- After numerous tests, we have found that the server's waiting time for image loading, description generation, and object detection is approximately 25 seconds. One possible improvement would be to use faster models or optimize the server code.
- Add the ability to select the language and playback speed of the description and object name.

After evaluating the application, as explained earlier, we received a series of possible improvements from the users who tested the application. The suggested improvements are as follows:

- Instead of selecting the gallery button and then selecting the image, have a swipe system from right to left or from top to bottom to switch images.
- Display a warning message when the user taps on an area of the screen where there is no image. This allows them to navigate more efficiently.
- Differentiate between people, animals, and objects. When the user selects, for example, one person and then selects another, there should be a distinction between them. If one of the people is a contact stored on their phone, it would be very useful for the application to recognize that specific person.
- Place the buttons in the lower corners, as most applications they use have buttons positioned in those locations, making it easier for them to use the corners as reference points.
- Provide more detailed descriptions when navigating through the image. It is important for the elements in the image to have not only an identifier but also a brief description to help users have a clearer understanding.
- Improve the application to recognize environments both in the description and image navigation.
- Implement functionality that dictates the objects in the image from left to right or top to bottom.

Capítulo 9

Contribuciones Personales

En este capítulo se contará con detalle el trabajado realizado por cada integrante de este proyecto. En la Sección 9.1 se detalla el trabajo realizado por Matías Amor Sanz, en la Sección 9.2 el de Alberto Chaves López y por último, en la Sección 9.3 el de Víctor Ruiz Gea.

9.1. Matías Amor Sanz

En el momento de decidir qué Trabajo de Fin de Grado elegir para finalizar mis estudios en ingeniería de computadores, buscaba un TFG que me permitiera ayudar a personas que lo necesitan. Además, que me permitiera realizarlo en grupo. Por ello, antes de elegir el tema, hablamos los tres para confirmar que íbamos a realizar el TFG juntos. Una vez acordado que lo íbamos a hacer grupal, observamos todas las posibilidades que teníamos. Sabíamos que queríamos realizar una aplicación, sobretodo, una aplicación que ayudase a los demás. Por ello, terminamos eligiendo este TFG. Nos pusimos de objetivo realizar una aplicación que fuera útil para, en este caso, las personas con discapacidad visual.

Nos faltaban conocimientos básicos de *dataset* y descripciones de imágenes, por lo que los tutores nos indicaron por dónde debíamos empezar. Los tres comenzamos buscando información general para, posteriormente, ponerla en común. Además, hice un pequeño curso de *Python*, ya que en la carrera no había hecho nada de *Python* y era un lenguaje que íbamos a necesitar posteriormente. Busqué información básica de *dataset* que nos fueran a ser útiles. Raquel y Alberto nos comentaron la funcionalidad del *dataset COCO*, por lo que procedimos a informarnos sobre este. A continuación, buscamos sobre aplicaciones similares, para tener una idea inicial de nuestra aplicación y poder tener un punto de partida en la implementación.

Una vez supimos el entorno de trabajo junto con una idea general de *dataset*, descripciones de imágenes y aplicaciones similares, nuestros tutores nos indicaron que el siguiente paso era capturar los requisitos necesarios para la aplicación. Raquel y Alberto consiguieron que tres personas pertenecientes a la ONCE se ofrecieran para realizarles una entrevista. El objetivo de esta entrevista era capturar unos requisitos y obtener toda la información posible sobre su uso diario del teléfono móvil. Preparamos una serie de preguntas junto con el material necesario para realizar la

entrevista, ya que era necesario grabarla para posteriormente transcribirla. Durante la entrevista, obtuvimos unos requisitos relacionados con la interfaz, las funcionalidades, el modo de uso, etc. También nos comentaron que les gustaría en un futuro poder probar la aplicación, por lo que les tuvimos en cuenta para la evaluación. Una vez finalizada la entrevista, la transcribimos entre los tres.

Hicimos una lista con los requisitos marcados por los usuarios, y uno de los puntos importantes era que utilizaban dispositivos con sistema operativo *iOS*, ya que les ofrecía una mayor accesibilidad que *Android*. Tuvimos el problema de que ninguno de nosotros disponía de un ordenador compatible para la programación en *iOS*. Por mi parte, procedí a descargarme una máquina virtual, para poder programar en *iOS*, pero tuve muchos problemas de compatibilidad. Por ello, decidimos realizarla en *Android*, utilizando *Android Studio*.

Raquel y Alberto consiguieron que tuviéramos una nueva entrevista, esta vez con una experta en audiodescripciones. Al igual que en la primer entrevista, la grabamos para posteriormente transcribirla. Era importante conseguir información referente a las descripciones. A partir de esta entrevista, conseguimos entender qué es una buen descripción y como elaborarla.

A partir de este momento, ya comenzamos con el desarrollo de nuestra aplicación. Estas han sido mis contribuciones en la implementación:

- Servidor: Para realizar nuestra aplicación necesitábamos crear un servidor local, por lo que comenzamos investigando qué es un servidor *flask*. Lo primero que necesitábamos era instalarnos una serie de paquetes que permitieran que el servidor funcionase correctamente. A continuación, vimos estructuras básicas programadas en *Python* para poder tener una primera *API* funcional. Tras conseguir esta pequeña *API*, comencé a informarme sobre cómo configurar el servidor para que reciba una imagen desde la aplicación. Implementé un pequeño código que conseguía almacenar una imagen recibida de la aplicación para posteriormente almacenarla en una carpeta del ordenador local. De esta manera, conseguíamos saber si la imagen se estaba mandando correctamente. Además, inicialmente teníamos la idea de que las coordenadas del usuario al navegar por la pantalla, se pasaran al servidor desde la aplicación. Por ello, implementé una función dentro del servidor que recibía estas coordenadas y las mostraba en la consola. Finalmente, esta idea la desechamos por lo que adaptamos el código para que las coordenadas no se pasasen desde la aplicación cómo hemos explicado en la memoria.

  A continuación investigué el funcionamiento de *Roboflow*. Después de probar numerosos modelos, nos dimos cuenta que con *Roboflow* no iba a ser suficiente para realizar nuestra aplicación por lo que comenzamos con la búsqueda de otras alternativas como *Hugging Face*. Con *Hugging Face* teníamos más posibilidades ya que los modelos cumplían con nuestras funcionalidades principales. Además, estos modelos venían con una documentación propia sobre su implementación, lo que nos facilitó mucho la codificación en el servidor. También fue necesario un modelo que tradujera al castellano las salidas de los modelos usados, ya que de manera predeterminada están en inglés.

  Una vez conseguimos tener un servidor local, intentamos usar un servidor

externo para poder realizar la evaluación. La universidad nos proporcionó un contenedor para ello, sin embargo, tras haber realizado numerosos intetnos, teníamos fallos de conexión que no se conseguían solucionar. Debido a ello, terminamos decidiendo utilizar el servidor local para la evaluación posterior.

- Aplicación: Lo primero en lo me centré fue en conseguir que la aplicación se conectara con el servidor. Investigué sobre el funcionamiento de las peticiones y la distintas maneras en la que se puede conectar con un servidor. Tras tener claro las librerías necesarias para realizar peticiones con el servidor, investigué la manera de implementar un código que te permitiese mandar una imagen al servidor a partir de una imagen seleccionada desde la galería. Descubrí que par poder realizar dicha tarea, era necesario tener un código que trabajase en un hilo secundario para evitar conflictos. Tras esto implementé un código que ejecutaba la tarea en un hilo secundario al principal y mandaba una imagen al servidor.

  Posteriormente, implementé un código que mandase al servidor las coordenadas del usuario al tocar en una zona de la pantalla. Para ello, utilicé la misma lógica de hilos que en el código de envío de la imagen. Además, mostraba por consola dichas coordenadas para poder realizar pruebas.

  Un vez implementados los modelos en el servidor, decidimos hacer cambios dentro de la aplicación para mejorar la eficiencia. Desechamos el código que mandaba las coordenadas al servidor ya que, con los modelos implementados, es más útil obtener una lista de objetos desde el servidor. Por ello, la clase que se encargaba de mandar la imagen, también recibía los objetos de esta. A partir de esta lista de objetos almacenadas en la aplicación, únicamente teníamos que implementar un código que, dependiendo de la zona de la pantalla donde tocase, devuelva el objeto de dicha zona. Sin embargo, esto supuso un problema, ya que el tamaño de la imagen que llegaba desde el servidor variaba respecto al tamaño de la imagen que se ajustaba a la pantalla. Para ello, fue necesario una redimensión de las coordenadas de cada objeto.

  Finalmente integramos una lector de voz que narrase tanto la descripción como los objetos que detectaba la aplicación.

Finalmente, obtuvimos una aplicación funcional y nuestros tutores se pusieron en contacto con los tres usuarios que nos dieron los requisitos para hacer una evaluación de la aplicación. Para esta evaluación decidimos buscar tres imágenes cada uno. Estas imágenes tenían que ser variadas y tener un grado de dificultad (fácil, medio y difícil). Además, pensamos una serie de preguntas que nos permitieran saber cómo de útil y accesible les ha resultado la aplicación. Estas preguntas también nos permitían obtener información sobre mejoras y funcionalidades futuras.

9.2. Alberto Chaves López

Cuando llegó el momento de tomar una decisión sobre el tema de mi TFG para concluir mis estudios en ingeniería de software, quería encontrar un TFG que además de investigación pudiera desarrollar algún tipo de aplicación o software para

demostrar lo aprendido todos estos años. Antes de elegir que TFG escoger, ya sabía que iba a hacerlo con Víctor y Matías, ya que cada uno estudiaba una rama diferente de la informática, pudiendo dar diferentes puntos de vista y aportar diferente conocimiento al proyecto. Estuvimos barajando diferentes TFGs de aplicaciones móviles hasta que nos decantamos por este debido a que no solo tenia lo que buscábamos personalmente, si no que era un proyecto que nos llamó la atención. No solo íbamos a elaborar una aplicación, si no que teníamos la posibilidad de intentar ayudar a personas con discapacidad visual mediante esta aplicación.

Una vez elegido el proyecto, tuvimos la primera reunión con los directores del proyecto, Alberto y Raquel. A partir de aquí, tuvimos una reunión con ellos cada dos o tres semanas. En esta primera reunión nos explicaron las bases del proyecto, su objetivo y el conocimiento básico que deberíamos aprender para empezar a desarrollar la aplicación. Nosotros no teníamos que hacer una aplicación desde cero creando *datasets* nuevos ni entrenando modelos de descripción de imágenes. Nuestro objetivo era crear una aplicación juntando todos estos elementos que ya existían. Tener conocimiento sobre *Python*, los *datasets* y la descripción de imágenes fue nuestra primera tarea. En la carrera no he tenido ninguna asignatura que me enseñara *Python*, por lo que busqué pequeños tutoriales para aprender las bases de la programación en *Python*, además de hacer un breve curso de este. Por otro lado investigué sobre los *datasets* y los diferentes *datasets* que existían actualmente. Los directores nos dijeron que nos centráramos en el *dataset COCO*, por lo que empecé a estudiar su funcionamiento y cómo podíamos implementarlo en nuestra aplicación. Para la descripción de imágenes estuve informándome de lo que existía en el mercado y probando aplicaciones que describían imágenes en mi móvil.

Una vez asentados los conocimientos básicos e identificados los objetivos del proyecto, teníamos que identificar los requisitos de nuestra aplicación. Para ello, Alberto y Raquel consiguieron que tres personas pertenecientes a la ONCE con discapacidad visual se reunieran con nosotros para poder entrevistarles. Para preparar dicha entrevista, nos reunimos varias veces para concretar cómo íbamos a hacerla y elaborar una serie de preguntas que nos ayudaran a conseguir estos requisitos. El día de la entrevista, nos reunimos con ellos en la Facultad de Informática de la Universidad Complutense de Madrid. Los directores del proyecto también asistieron y nos proporcionaron el equipo necesario para hacer dicha entrevista. Cada miembro de este proyecto tomó un rol diferente en esta entrevista. El mío fue grabarla para su futura transcripción y apuntar ideas y detalles que surgían en esta. Esto nos ayudó a entender las dificultades que tienen estas personas con los dispositivos móviles, cómo los utilizan y qué debería tener una buena aplicación para la descripción de imágenes. Aparte de esta entrevista, los directores nos consiguieron una entrevista con una experta en audiodescripción. Esta entrevista fue más corta que la anterior y fue telemática. En esta profundizamos en la audiodescripción y cómo actualmente se desarrolla este tipo de descripciones en imágenes.

Tras las entrevistas y obtener los requisitos que debería tener nuestra aplicación, nos dimos cuenta que estas personas optaban por utilizar móviles con el sistema operativo *iOS* en vez *Android* debido a su mejor accesibilidad. Es por ello que decidimos empezar a desarrollar una aplicación para *iOS*. Pero nos surgió nuestro primer problema. El desarrollo de aplicaciones para *iOS* está pensado para desarrollarlas en

un entorno *iOS*, no *Windows* que era el que teníamos. Probé varias formas como fue descargarse una máquina virtual pero era inconsistente. Cada acción que probaba iba muy lento por lo que tuvimos que decidir hacer una aplicación para *Android*. Lo bueno de hacerlo para *Android* es que yo ya había hecho aplicaciones por mi cuenta, por lo que el uso de *Android Studio* ya me era familiar.

Para la implementación de la aplicación voy a dividir en dos partes mi contribución:

- Servidor: Gracias al curso de *Python* que hice, tenia el conocimiento para empezar un servidor local. Decidimos hacer un servidor *flask*, por lo que me tocó informarme de cómo crearlo, cómo meter paquetes y cómo usarlo. La misión principal en este momento era crear un servidor que pudiera conectarse a la aplicación. La universidad nos proporcionó un contenedor en la nube para poder tener un servidor remoto. Estuve instalando los paquetes necesarios que teníamos en el servidor local, pero nos surgieron muchos errores y no nos dejaba instalar ciertos paquetes. Poco después el contenedor se cayó, por lo que tuvimos que hacer el servidor únicamente en local.

  Una vez conectado el servidor con la aplicación, investigué modelos ya entrenados que detectaran elementos y pudieran hacer una descripción de una imagen. Fue entonces cuando surgió *Roboflow*. Estuve probando diferentes modelos dentro de *Roboflow*, como fue un modelo para la detección de perros y gatos o otro modelo para la detección de piezas de ajedrez en imágenes. Pero vimos que los modelos de *Roboflow* se nos quedaban cortos ya que no encontramos ninguno que abarcara todos los elementos de una imagen. Encontramos entonces *Hugging Face*, una página donde había una infinidad de modelos ya entrenados. Además de haber un gran repertorio de modelos de descripción de imágenes y detección de objetos, cada modelo tenia una documentación donde se veía reflejado cómo implementarlo en *Python* y una pequeña prueba funcional de cómo funcionaba el modelo. Esto hizo que pudiera probar de una forma más sencilla diferentes modelos hasta poder encontrar uno que se acercará a nuestro modelo ideal. Entre todos encontramos uno para la detección de imágenes y otro para la descripción de imágenes. Pero la respuesta de estos estaba en inglés. La solución la encontró Víctor, encontrando otro modelo que lo tradujera.

- Aplicación: Como ya tenía experiencia en *Android Studio*, lo primero que hice es una aplicación muy simple en la que pudieses elegir una foto en la galería y guardarla en un objeto para posteriormente poder enviárselo al servidor.

  Una vez creado el servidor local, estuve mirando cómo conectarlo al servidor y que al hacerlo, el servidor devolviera algún tipo de mensaje. Una vez lograda la conexión, procedimos a enviarle una imagen para que el servidor pudiera procesarla en los diferentes modelos que hay dentro de este.

  El servidor nos devolvía una descripción acorde a lo que buscábamos. Ahora teníamos el problema de plantear la detección de elementos. Conseguimos implementar una función en la aplicación que devolvía las coordenadas de la pantalla donde se tocaba. Tras muchas pruebas tocando la pantalla, tuve que

averiguar cómo asignaba las coordenadas en *Android Studio*, y conseguir que el punto de inicio de coordenadas no fuera la parte superior izquierda del móvil, si no que fuera la de la imagen. Esto es debido a que el modelo escogido para la detección de imágenes cogía las coordenadas desde ese punto de la imagen.

Al principio probamos en enviarle las coordenadas al servidor y que fuera este el que devolviera qué objeto había en dicha imagen. Pero nos dimos cuenta que era mejor y más sencillo que al cargar la foto en el servidor, y que este nos devolviera un *JSONList* con la información de los elementos de la imagen, diciéndonos el tipo de objeto y sus coordenadas. Aquí surgió el principal problema que hemos tenido dentro de la aplicación. Al ser una aplicación que se probaría en diferentes móviles, tuvimos que poner un tamaño fijo a las imágenes, pero el servidor nos devolvía las coordenadas de los elementos con el tamaño original de la imagen. Al final conseguimos arreglarlo redimensionando las coordenadas de cada objeto según su tamaño.

Por último, integramos un lector de voz que pudiese narrar la descripción y los elementos de la imagen. Además, me encargué de etiquetar los botones para que el lector pudiera narrar las acciones. Ayudé también a la detección de errores en la aplicación, como fue el error en la conexión con el servidor.

Al estudiar ingeniería de software tenia más conocimiento en elaborar diagramas. Para hacer más visual la conexión entre las diferentes partes de la aplicación hice un dibujo representativo de la arquitectura de la aplicación (figura 5.1). Decidí hacer un diagrama de secuencia para ayudar a entender la conexión cliente-servidor en el que podemos ver cómo viajan los datos entre las diferentes clases y cómo intercambia información al ejecutarse la aplicación. Esto lo podemos ver en la figura 5.9.

Una vez terminada la aplicación, gracias a nuestros directores de proyecto, conseguimos tener otra reunión con las personas invidentes con las que tuvimos la entrevista previamente. Decidimos que para poder demostrar el funcionamiento de la aplicación, cada usuario la probaría con tres fotografías diferentes. Por lo que individualmente tuve que buscar una imagen simple con pocos elementos, una un poco más compleja y otra con muchos elementos para dificultar la descripción. Una vez identificadas las imágenes, nos reunimos los participantes para planificar cómo íbamos a enseñar dichas imágenes e idear una serie de preguntas para la evaluación. Una vez empezada la evaluación, cada uno tomó un rol. En mi caso, mi función fue grabar la entrevista con el material que me proporciono la facultad y coger todas las notas e ideas posibles de lo que decían los usuarios. Esto nos ha ayudado a elaborar el desarrollo de la evaluación y las conclusiones de esta.

Por último, la investigación y redacción de esta memoria la hemos hecho siempre a partes iguales. Todas las semanas hacíamos una reunión donde intentábamos repartirnos el trabajo equitativamente y tenerlo hecho para la siguiente reunión. Las primeras semanas estuvimos utilizando un tablero de *Trello*, pero nos dimos cuenta que al ser tan pocas personas en el proyecto era más eficiente hablarlo por un grupo de *Whatsapp* y tener toda la documentación que encontrábamos en un *Google Drive*.

9.3. Víctor Ruiz Gea

A la hora de determinar qué tema abordar en mi Trabajo de Fin de Grado para finalizar mis estudios en ingeniería informática, busque proyectos que me motivasen y tuviesen como objetivo ayudar de alguna forma a personas que lo necesitasen. Mi idea principal fue realizar mi TFG de forma grupal, por lo que me puse en contacto con mis compañeros Alberto y Matías. Tras leer detalladamente todos los proyectos ofertados, nos llamó mucho la atención nuestro TFG, ya que nos brindaba la posibilidad de intentar ayudar a personas con discapacidad visual, por lo que decidimos centrarnos en él.

Una vez decidido, nos pusimos en contacto con nuestros tutores Raquel y Alberto. Ellos nos explicaron cuál era el propósito del proyecto y los objetivos que cumplir. También nos recomendaron empezar a investigar sobre las tecnologías para la detección y descripción de imágenes.

Para poder entender cuál era el problema principal que presentaba nuestro proyecto, en este caso la dificultad de las personas con discapacidad visual para reconocer imágenes, empecé a leer artículos de cómo es su día a día. Esto me ayudo para ver una vista general de las necesidades que presentaban los usuarios a los que nos queremos dirigir.

Dentro de la investigación sobre las tecnologías para la detección y descripción de imágenes, tuve que buscar respuestas a preguntas como, qué es un *dataset*, qué es *deep learning* o para qué sirve el *captioning* de imágenes. Además también nos dimos cuenta que el lenguaje utilizado para este campo es *Python*. Al no tener ningún tipo de conocimiento previo tuve que realizar un curso básico proporcionado por nuestros tutores. A continuación, buscamos aplicaciones similares para ver su funcionamiento, observando cosas que podríamos mejorar y así, conseguir una idea principal de nuestra aplicación.

Raquel y Alberto nos informaron de que teníamos la oportunidad de entrevistar a tres personas con discapacidad visual y a una experta en audiodescripción. Por lo que nos reunimos en grupo para decidir las preguntas necesarias para conseguir captar todos los requisitos de nuestros usuarios y, por otro lado, las preguntas para captar información sobre cuáles son las características de una buena descripción de una imagen.

Uno de los requisitos principales fue que la aplicación estuviese desarrollada para *iOS*. Al no tener a nuestra disposición ningún ordenador para desarrollar aplicaciones para *iOS* exploramos opciones para desarrollar mediante *Windows*. Encontré una maquina virtual que lograba arrancar, pero no era capaz de mantener el entorno de programación a un ritmo que nos permitiese desarrollar la aplicación. Por lo tanto, tuvimos que desarrollar la aplicación para *Android*.

Dentro de la implementación del proyecto he dividido mi contribución en dos partes:

- Servidor: Inicialmente, tuve que investigar sobre qué es un servidor *flask* y cómo se implementa. Con la estructura de este tipo de servidor clara, empezamos a buscar cómo procesar imágenes para poder usarlas correctamente. A continuación, tuve que investigar dentro de plataformas como *Roboflow* y

Hugging Face para encontrar modelos de entrenamiento que pudiesen detectar objetos y generar descripciones de una imagen. Mi investigación dentro de *Roboflow* se basó en ir probando diferentes modelos de detección de objetos con diferentes imágenes para poder comparar las respuestas de cada uno y elegir el modelo más eficiente. Como se comenta en la arquitectura de esta memoria, con los modelos de entrenamiento de *Roboflow* no conseguíamos abarcar todo tipo de objetos existentes. Esto me obligo a buscar otras alternativas. Dentro de *Hugging Face* realice el mismo procedimiento que en *Roboflow* encontrando dos modelos que se acercaban a lo que queríamos conseguir con nuestras dos funcionalidades, un modelo para la generación de descripciones y otro para la detección de todo tipo de objetos.

El siguiente paso era integrar estos dos modelos en nuestro servidor. Me resultó mucho más sencillo gracias a que en la plataforma de *Hugging Face* cada modelo viene con su propia documentación de cómo implementarlo mediante lenguaje *Python*. Una vez entendida la forma de implementarlo, utilicé *Google Colaborate* para probar su correcto funcionamiento. Además me sirvió para saber que librerías debía instalar en el propio servidor para poder utilizar dichos modelos.

Tras integrar los modelos en el servidor, me di cuenta que sus salidas estaban en inglés, por lo que tuve que hacer una búsqueda dentro de *Hugging Face* de un modelo de entrenamiento que fuese capaz de traducir de inglés a español y conectarlo a las salidas de los anteriores modelos.

Finalmente, dentro de cada *endpoint* tuve que guardar la información necesaria de la salida del modelo en un *JSONObject* para enviársela a nuestra aplicación por cada petición realizada.

- Aplicación: Ayudé a mis compañeros Matías y Alberto a conseguir nuestro primer paso que fue conseguir que la aplicación se comunicase con nuestro servidor y que pudiese cargar imágenes desde la galería del dispositivo móvil. Una vez establecida la conexión entre el servidor y la aplicación, procesé las respuestas del servidor en la aplicación para cada imagen enviada. Para que el usuario pudiese tocar la pantalla y que la aplicación reconociera que objeto es, utilicé la lista de objetos devuelta por el servidor e implementé una función que dependiendo de las coordenadas donde tocase el usuario te devolviese el nombre del objeto. Para esta funcionalidad tuvimos un gran problema con la diferencia de tamaño entre la imagen enviada al servidor y el espacio donde se iba a mostrar la imagen en la aplicación. Ya que las posiciones de los objetos que detectaba el servidor se veían distorsionadas por el cambio de tamaño. Para solucionarlo tuve que redimensionar todas las coordenadas de cada objeto según la relación entre sus tamaños. Integramos un lector de voz para que narrase la descripción y el objeto pulsado por el usuario. En cuanto al diseño de esta aplicación, me encargué de introducir un cuadro donde se muestra la imagen y dos figuras que actúan como botones, uno para cargar la imagen y otro para escuchar de nuevo la descripción generada.

Una vez terminada la implementación del proyecto, nuestros tutores Raquel y

Alberto se pusieron en contacto con las personas con discapacidad visual que entrevistamos para poder hacer una evaluación de nuestra aplicación. Nos reunimos para hacer el diseño de esta evaluación donde tuvimos que buscar cada uno 3 imágenes válidas para que utilizasen nuestros usuarios y pensar preguntas para comprobar cuanta información de la imagen han sido capaces de obtener.

Bibliografía

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A. y Zagoruyko, S. End-to-end object detection with transformers. *CoRR*, vol. abs/2005.12872, 2020.

Gil, L. E. O., Guzmán, F. V., Zayas, E. V., Ortega, I. G. y Guzmán, J. F. A. Evaluación de la usabilidad y accesibilidad de las aplicaciones lookout y seeing ai para dispositivos móviles en personas con discapacidad visual: Un estudio descriptivo (usability and accessibility evaluation of lookout and seeing ai applications for mobile devices in people with visual impairment: A descriptive study). *Pistas Educativas*, vol. 44(143), 2022.

Jain, S. M. Hugging face. En *Introduction to Transformers for NLP: With the Hugging Face Library and Models to Solve Problems*, páginas 51–67. Springer, 2022.

Li, J., Li, D., Xiong, C. y Hoi, S. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. 2022.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L. y Dollár, P. 2015.

Redmon, J., Divvala, S., Girshick, R. y Farhadi, A. You only look once: Unified, real-time object detection. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 779–788. 2016.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P. y Ommer, B. High-resolution image synthesis with latent diffusion models. En *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 10684–10695. 2022.