UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Detección de necesidades del usuario a través de imágenes

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Colomer Bel, Marta

Tutor/a: Albert Albiol, Manuela

Director/a Experimental: Mestre Gascón, Antoni

CURSO ACADÉMICO: 2023/2024

Dedicatoria

Primero de todo me gustaría dedicar este trabajo a mis padres. Mis padres no son perfectos, como los de nadie, ni son mis amigos, porque como bien me dijo mi madre una vez "Quien tiene amigos en vez de padres, crece huérfano". Así que mis padres son y serán siempre mis padres, los que me han enseñado casi todo lo que sé de la vida y me han apoyado en todas las decisiones que he tomado. Me han cuidado mejor de lo que podría llegar a desear y siempre lo han dado todo por mí. Sin ellos yo no estaría aquí.

También dedicárselo a mis dos mejores amigos. A Mario, por estar siempre conmigo, haberme acompañado en todas mis aventuras, locuras e ideas. Por hacerme mejor cada vez que hablo con él y enseñarme que siempre hay solución y que las cosas siempre pueden ser y hacerse mejor. A Oihane, que no solo es mi mejor amiga y lo ha sido los últimos 10 años, sino que también es mi compañera de vida y la persona que más me ha cuidado y aguantado cuando nadie más ha podido. Me ha hecho ser mejor, trabajar más, tanto académicamente como personalmente, y me ha abierto los ojos a un mundo diferente, más colorido y alegre. A ellos dos porque no habría llegado tan lejos sin tenerlos a mi lado.

Y por último y más importante a mi hermano. No sé cómo es el mundo sin él y espero no saberlo nunca. Como Caronte guía a los espíritus a través del Estigia, mi hermano me guía a mi en la vida. Es trabajador, inteligente, determinado y un poco cabezota, pero es mi modelo a seguir, aquello a lo que aspiro ser algún día. Si no fuese por él no habría estudiado informática, ni habría terminado la carrera. Pero principalmente, sin su ayuda y guía, no habría entregado este trabajo. A ti, Jose, porque siempre has estado cuando te he necesitado.

Resumen

El Trabajo de Fin de Grado (TFG) se centra en el desarrollo de un componente software que tiene como objetivo principal detectar las necesidades de los usuarios a través del análisis de sus expresiones faciales. Este componente se especializa en reconocer cuándo un usuario requiere asistencia durante su interacción con una aplicación software. Para lograr esto, se emplearán técnicas de inteligencia artificial que permitirán el análisis en tiempo real de las imágenes faciales de los usuarios, identificando señales que indiquen confusión, frustración o cualquier otra necesidad de ayuda. La validación se realizará mediante experimentos prácticos con una aplicación concreta y diversos casos de uso, evaluando la efectividad del componente en situaciones reales.

Palabras clave: Componente software, detección de necesidades del usuario, análisis de imágenes faciales, reconocimiento de expresiones, inteligencia artificial, interacción usuario-aplicación**.**

Resum

El Treball de Fi de Grau (TFG) se centra en el desenvolupament d'un component software que té com a objectiu principal detectar les necessitats dels usuaris a través de l'anàlisi de les seues expressions facials. Este component s'especialitza a reconèixer quan un usuari requereix assistència durant la seua interacció amb una aplicació software. Per a aconseguir això, s'empraran tècniques d'intel·ligència artificial que permetran l'anàlisi en temps real de les imatges facials dels usuaris, identificant senyals que indiquen confusió, frustració o qualsevol altra necessitat d'ajuda. La validació es realitzarà mitjançant experiments pràctics amb una aplicació concreta i diversos casos d'ús, avaluant l'efectivitat del component en situacions reals.

Paraules clau: Component software, detecció de necessitats de l'usuari, anàlisi d'imatges facials, reconeixement d'expressions, intel·ligència artificial, interacció usuari-aplicació.

Abstract

The Final Degree Project (TFG) focuses on the development of a software component whose main objective is to detect the needs of users through the analysis of their facial expressions. This component specializes in recognizing when a user requires assistance during interaction with a software application. To achieve this, artificial intelligence techniques will be employed to allow real-time analysis of users' facial images, identifying signals that indicate confusion, frustration, or any other need for help. Validation will be performed through practical experiments with a concrete application and various use cases, evaluating the effectiveness of the component in real situations.

Keywords: software component, user needs detection, facial image analysis, expression recognition, artificial intelligence, user-application interaction**.**

Índice general

Tabla de figuras

1. Introducción

Las aplicaciones hoy en día se vuelven cada vez más complejas, más elaboradas y con más módulos. Estas evolucionan a un ritmo acelerado, añadiendo nuevas funcionalidades casi semanalmente y obligando al usuario a un aprendizaje rápido, casi fugaz, para no perder el ritmo que se le intenta marcar. Estas aplicaciones suelen estar formadas por dos partes. La primera siendo la lógica del servidor. Esta parte carga con la lógica de la aplicación, es la que determina qué hace la aplicación y cómo. Es la parte que se comunica con los servicios de los que hace uso la aplicación, así como la base de datos y API externas. Con el añadido, cada vez más común, de inteligencias artificiales (IA) a esta parte lógica, el uso de hasta la aplicación más simple se ha convertido en un ejercicio de deducción y paciencia para gran parte de los usuarios. La cantidad de atención requerida va en incremento con la cantidad de estímulos que se le ofrecen al usuario, al igual que el requerimiento de conocer o entender todos los conceptos informáticos y de la web que se le pide al usuario medio (cookies, firewall, add-ons, DNS, …).

El punto de unión entre estos sistemas y el usuario son las interfaces gráficas (GUI), la segunda parte que forma una aplicación. Estas son los medios o métodos a través de los cuales se comunica la acción, interacción o intención del usuario al sistema. Facilita la comunicación entre lo que quiere el usuario del sistema, ya que las acciones las realiza el propio usuario a través de la interacción con botones, ventanas o menús, en lugar de hacerle aprender, memorizar y usar comandos de texto. El uso de las GUI facilita la navegación y el uso de software y sistemas operativos haciendo uso de un aspecto intuitivo y atractivo, mejorando así la experiencia del usuario.

Respecto a estas interfaces gráficas, cada vez hay más investigación y diseño alrededor de la usabilidad y de facilitar el encontrar lo que buscas. Aunque la tendencia hacia el minimalismo, la inclusión cada vez más frecuente de animaciones de desplazamiento que inhabilitan cualquier otra acción, no ayudan a mejorar la accesibilidad y la facilidad de uso. Por otro lado, las opciones de mejora de la accesibilidad son, en la inmensa mayoría, inexistentes. En los casos en los que sí existen y facilitan el acceso universal al producto/web, quedan relegadas a las opciones de pie de página o son difíciles de encontrar. Por ejemplo, de las principales páginas de streaming de contenido (Disney+, Prime Video, …) solo Netflix cuenta con la opción de acceder al contenido que cuenta con audodescripción y se encuentra en el pie de página, que también es de difícil acceso al ser una página infinita que genera más contenido cuando llegas al final de esta. Ninguna página cuenta con una opción de agrandar los cárteles de los contenidos, con una descripción adecuada para indicar si contienen audiodescripción o un sistema de dicción por voz. Aunque las personas con estás dificultades suelen tener dispositivos adaptados y software de ayuda externo no deja de ser llamativo que las plataformas de streaming más grandes no sean las que más opciones de accesibilidad ofrezcan.

El desarrollo de estas opciones y facilidades requiere de un estudio previo, un buen diseño y una implementación que puede alargarse meses. Esto implica dinero, mucho dinero, y estas mismas empresas que empezaron y crecieron declarando que eran la competencia a la

televisión por el consumo por demanda, sin anuncios y ofreciendo contenido de gran calidad, han acabado recortando en calidad, aumentando en cantidad y añadiendo anuncios que solo puedes evitar si pagas más. Por lo que gastar dinero en funcionalidades y herramientas que solo unos pocos van a acabar usando no aporta beneficios, por mucho que para esos pocos sea necesario.

El incremento de la complejidad de uso de estos sistemas y la falta de evolución y explotación en herramientas de ayuda nos empuja a buscar una ayuda o asistencia para los usuarios nuevos o menos expertos y para los grupos más vulnerables a este aumento de complejidad. Esta ayuda requiere de varios factores para saber cuándo se estresa el usuario, cuándo no sabe dónde buscar o cuándo está perdido. El sistema necesita saber cuándo mostrar la ayuda al usuario o qué busca el usuario. De la misma manera, ha de mostrar el mensaje de ayuda más apropiado. Para todo esto, nuestro sistema consta de: 1) un modelo de inferencia de emociones para reconocer el estado de ánimo del usuario y detectar frustración y 2) un registro de las veces que se ha entrado en cada ventana para determinar qué busca el usuario. Con esto se pretende mejorar la experiencia del nuevo usuario y del usuario medio, a la vez que proveer de una herramienta de accesibilidad y usabilidad a los usuarios que lo requieran.

¿Cómo diseñar ese sistema? ¿Qué elementos hacen falta? ¿Qué mensaje es el más apropiado? ¿Cuál es la mejor manera de ofrecer ayuda al usuario? Para empezar, como se ha mencionado en el párrafo anterior, se necesita un modelo de inferencia de emociones. Este ha de proporcionar la información en tiempo real de las emociones detectadas para el posterior análisis de estas. Además, se necesita modificar la aplicación donde se vaya a integrar el sistema para que analice la información procedente del modelo y muestre la ayuda si así se determina. Por último, se tiene que determinar qué mensaje de ayuda se va a mostrar y cómo.

En este TFG se va a desarrollar un sistema de detección de ayuda al usuario y que ofrezca la mejor ayuda, determinada por las acciones de este. Para ello se va a hacer uso de los factores anteriormente especificados integrados en una aplicación ya realizada (que no es el objeto de estudio en este trabajo). Las emociones del usuario se obtendrán de un modelo ya realizado, pero modificado para que permita comunicación con una aplicación externa. La aplicación será ligeramente modificada para mostrar la ayuda y recibir la información procedente del modelo.

A lo largo del documento se procederá a explicar cómo se ha realizado la implementación y comunicación de todas las partes, así como qué decisiones se han tomado hasta la determinación del mejor mensaje de ayuda posible. También se van a incluir un par de casos de uso para ejemplificar su utilidad y uso. Y finalmente esta memoria finalizará con un resumen de los resultados y conclusiones del trabajo realizado.

1.1 Motivación

La tecnología avanza a un ritmo acelerado y nos obliga a todos a intentar seguirle el ritmo que marca. Todos los que nos dedicamos a esto, que lo estudiamos, que lo usamos a diario a alto nivel, no tenemos problemas en su mayoría o nos cuesta menos ir al día. En cambio, el usuario medio, la gente de la tercera edad y cada vez más adultos tienen problemas para comprender el funcionamiento de muchos sistemas y para encontrar aquello que buscan en

internet o en alguna aplicación. Pero estos no son los únicos grupos que necesitan una ayuda extra a la hora de usar internet. Las personas con discapacidades visuales, de movilidad o intelectuales también pueden necesitar una ayuda a la hora de usar internet. Ofrecer una ayuda a aquel usuario que lo requiera o que la necesite y no sepa donde buscarla es primordial para mantener una alta accesibilidad al mundo del internet y la informática.

Es importante impartir una educación sobre internet y su uso a los niños y adolescentes, pero en general se olvida a los que más dificultad tienen (como son los grupos anteriormente comentados). Acercar a estos grupos al mundo de la tecnología de una manera accesible y de la forma correcta hará que se vean más incluidos en la sociedad de hoy en día, que le tengan menos miedo a la tecnología y que hasta ellos mismos se atrevan a experimentar y comiencen a aprender por su cuenta. Casi todo lo que nos rodea hace uso o implica uso de internet y cada vez más acciones que antes requerían de interacción con un empleado experto en la tarea están siendo modificadas para que sea el mismo usuario el que la realice a través de una aplicación o web. Estás medidas ayudan a aliviar las colas de espera en oficinas y las cargas de trabajo de los empleados, pero también aíslan y marginan a los que no tienen acceso o no saben hacer uso de la tecnología.

El objetivo es que con esta herramienta se haga un primer acercamiento al desarrollo de una herramienta de ayuda personalizada que salte a asistir al usuario cuando este lo necesite. Experimentando en una aplicación simple para probar su correcto funcionamiento se pretende determinar de qué forma y con qué mensaje se ofrece la mejor ayuda al usuario. El objetivo final de este trabajo es servir de inicio para el desarrollo de una aplicación de accesibilidad y ayuda al usuario adaptable a varios sistemas y abrir una puerta a la explotación de las herramientas y funcionalidades asistidas por IA.

1.2 Objetivos

El objetivo principal de este TFG es desarrollar una herramienta que identifique a partir de las emociones inferidas del usuario cuándo este necesita ayuda mientras está utilizando una aplicación. Esto permitirá a la aplicación adaptarse a las necesidades de los usuarios. Este objetivo se puede desglosar en varios subobjetivos:

- Seleccionar una técnica que permita a partir de imágenes de la cara de los usuarios inferir emociones.
- Diseñar e implementar una herramienta para identificar a partir de las emociones e interacciones del usuario con una aplicación cuando este no sabe cómo utilizar la misma.
- Extender una aplicación para que utilizando la herramienta construida se adapte a las necesidades del usuario.

1.3 Estructura de la memoria

Esta memoria presenta una estructura de 7 capítulos. El primero consta de la introducción ya presentada. Esta incluye un breve texto expresando la motivación detrás de la idea y una síntesis de los objetivos del trabajo y la memoria.

A continuación, en el segundo capítulo, se va a exponer la situación actual en el mundo de la tecnología y más concretamente de las IAs. Se hará una pequeña introducción a las IAs, qué son y su actual estado para finalizar explicando, más específicamente, el uso actual de las IAs en beneficio de la accesibilidad y facilitar la usabilidad al usuario.

Posteriormente, en el tercer capítulo se realizará la explicación de la arquitectura diseñada para el sistema. Esto incluirá una breve descripción de los componentes que forman parte del sistema y con que sistemas externos interactuarán. Además, se explicará que método se ha escogido para la conexión entre los diferentes componentes.

Seguidamente, en el capítulo 4, se procederá a explicar el modelo de reconocimiento de expresiones faciales y emociones escogido para el trabajo. Se comenzará con una explicación de qué son las emociones, como se manifiestan y como estas manifestaciones se pueden analizar para su estudio. Luego se procederá a explicar, brevemente, la variedad de IAs que analizan las emociones en base a las manifestaciones anteriormente explicadas. Y se finalizará con la explicación del modelo escogido, cómo ha sido entrenado y cómo funciona, y la aplicación escogida para la prueba de la herramienta al completo.

Una vez se ha explicado el modelo, en el capítulo 5 se expone el desarrollo principal del trabajo. En este capítulo nos vamos a introducir en el diseño del sistema, qué modificaciones se han realizado al modelo y a la aplicación utilizada. También se va a explicar el desarrollo de las diferentes herramientas utilizadas para que la ayuda se detecte y se muestre por pantalla. Ya llegando al final, se hará un desarrollo de la sucesión de decisiones que han llevado a determinar qué mensaje mostrar, cómo y cuándo hacerlo. Se procederá a explicar por qué se ha escogido y construido estos mensajes y cómo determina el sistema qué mensaje mostrar. Además, se va a determinar en qué momento se detecta que el usuario necesita ayuda.

Antes de finalizar la memoria, se desarrollarán un par de casos de uso para ejemplificar su utilidad y uso. Y finalmente, el trabajo terminará con las conclusiones, los trabajos futuros pensados para el proyecto y, por último, la bibliografía y los apéndices.

2. Estado del arte

La Inteligencia Artificial (IA), actualmente, es uno de los campos más dinámicos de la informática. Al igual que lo fue el Internet, el Bluetooth, el WiFi o las redes sociales, la inteligencia artificial es un tema de amplio debate. Los informativos se hacen eco de los avances más destacados en el mundo de la inteligencia artificial y se comenta en las cenas de Navidad.

Pero, ¿qué es la IA? La Comisión Europea la define como los sistemas software diseñado por humanos que, dado un objetivo complejo, actúan en la dimensión física o digital percibiendo el entorno a través de la recolección de datos, interpretándolos, estén estructurados o no, o razonando sobre el conocimiento, procesando la información derivada de los datos y decidiendo cuales son las mejores acciones para lograr el objetivo dado[1]. Amazon la define como el campo de la ciencia de la computación dedicado a la resolución de problemas cognitivos asociados comúnmente a la inteligencia humana, como el aprendizaje, la creación y el reconocimiento de imágenes[2]. Pero la más cercana y simple es la que dio John McCarthy en 1956 cuando acuñó el concepto con el nombre que le damos hoy en día: “hacer que una máquina se comporte de formas que serían llamadas inteligentes si un ser humano hiciera eso”[3]. Esta definición, siendo la menos técnica es la más acertada. Nos da la libertad de considerar qué es inteligente y qué no; de aceptar que tanto la IA, como nosotros mismos, podemos cometer errores y que eso no se considera inteligente, pero sigue siendo parte del sistema y de nosotros mismos; de que el proceso de aprendizaje se basa en encontrar el mejor camino de acción probando y fallando tantas veces como se necesite hasta determinar qué es lo correcto e inteligente.

Hoy en día contamos con varias opciones si pensamos en inteligencias artificiales. Para nuestro día a día tenemos los asistentes virtuales. El primer acercamiento a lo que tenemos hoy fue Siri en 2011. Este asistente virtual nos ayudaba a realizar ciertas acciones en nuestro móvil sin ni siquiera tocarlo y a buscar cosas por internet cuando lo necesitábamos. Aunque anteriormente se había experimentado con la idea en IBM durante la década de los 90, no se había visto y comercializado al público nada parecido a Siri. Posteriormente, en 2014, apareció Alexa en escena. Este asistente virtual estaba integrado en su propio dispositivo, Echo, y podía comprar online, controlar las luces y los electrodomésticos de la casa y tenía la opción de personalizar respuestas y crear extensiones para su funcionamiento y adaptación a otros dispositivos[4].

Lo último en esta tendencia y aquello que más se está viendo como IA, son las generativas. Estas IAs se definen como la técnica computacional capaz de generar contenido aparentemente nuevo y significativo, así como texto, imágenes o audio, a partir de datos de entrenamiento[5]. Toda máquina de IA se debe de entrenar y para ese entrenamiento se necesitan referencias. Esas referencias se deben obtener de alguna base de datos y, ¿qué es más accesible que el internet para obtener estas referencias? El problema con esta práctica es que muchas de las referencias extraídas para entrenar las máquinas están protegidas por derechos de autor y a los desarrolladores de estas no parece importarles. Se ha demostrado en diversas

ocasiones que se hacen uso de esas referencias protegidas para entrenar las máquinas sin pedir permiso a su creador o sin pagar los derechos. Pero este problema no es tan simple de atajar ya que conseguir que una IA desaprenda algo es casi imposible, ya que lo volvería a aprender en algún punto y retener el aprendizaje de una máquina es mucho más costoso[6]. Unos de los casos más sonados últimamente son los de Meta y Adobe. Ambos han incluido en sus términos de servicio artículos y secciones para obtener derechos sobre lo que publican (en caso de Meta) o guardan en la nube (en caso de Adobe) sus usuarios y entrenar sus máquinas a costa de estos. Aunque a diferencia de Meta, Adobe sí ha dado un paso atrás cuando los usuarios se han quejado y ha aclarado que es para regular el contenido ilegal[7]. En cambio, Meta ha dado la opción de solicitar que se te excluya del proceso, pero tienes que saber dónde está para encontrarlo[8]. Aunque finalmente el proyecto ha sido paralizado por la Autoridad de Protección de Datos de Irlanda (DPC, según sus siglas en inglés). Esta autoridad pidió formalmente a la compañía que paralizase el proyecto y ahora ambos trabajarán junto a la Unión Europea para traer el proyecto a Europa, pero siguiendo las legislaciones correspondientes[9].

Por otra parte, dejan ver la vulnerabilidad a la que se expone la sociedad con el avance de estas tecnologías y la falta de regulación sobre estas. La Unión Europea propuso la primera ley referente a las IAs, se ha aprobado después de muchos trámites, pero todavía no ha entrado en vigor, ya que las leyes europeas tardan 2 años en entrar en vigor completo. De todas formas, esta ley no es perfecta. Cómo toda ley tiene sus dilemas morales y éticos. En este caso, la ley excluye las utilidades militares y a los cuerpos de seguridad de la ley. Es decir, se pueden usar escáneres biométricos para realizar reconocimientos y seguimientos si así se considera necesario. En cambio, se regulan los sistemas que sirven para categorizar en función de religión, comportamiento o raza, limita la categorización de CV por IA para las entrevistas de trabajo y prohíbe los sistemas para expandir o crear bases de datos faciales[10]. El Gobierno de España, durante su presidencia en el Consejo de la Unión Europea, fue el impulsor de esta ley y viéndose aprobada han propuesto una estrategia como iniciativa para mantener a España en la vanguardia tecnológica[11].

Estos son solo los ejemplos más destacados de IA, los más usados por los usuarios y algunos de los problemas o regulaciones alrededor de este mundo. Existen muchos otros tipos de IA, pero en este trabajo nos vamos a centrar exclusivamente en el reconocimiento e identificación de emociones.

2.1 Uso de IA para asistencia

Uno de los usos más comunes que se le están dado a las IAs hoy en día, relativo a las asistencias, es en el campo de la medicina. Ya en 2016 el 86% de las organizaciones proveedoras de servicios de asistencia médica utilizaban IA. Este dato ha ido en aumento y se espera que para 2025 los sistemas puedan responder de forma independiente a cuestiones específicas de los pacientes. Más concretamente, las aplicaciones se centran en el apoyo de toma de decisiones clínicas y el análisis de imágenes para diagnóstico. Los primeros sistemas ayudan al médico a diseñar tratamientos y el segundo busca lesiones u otros hallazgos en las diferentes pruebas con imágenes. Estos sistemas implican una detección precoz y un diagnóstico más rápido de diferentes enfermedades y podrían usarse para observar los síntomas y su evolución y

alertar a los médicos en caso de detectar una anomalía o una señal de riesgo. También mejoran la eficiencia de ensayos clínicos y acelera el desarrollo de nuevos fármacos[12]. La parte quirúrgica de la medicina no se queda atrás. Ya se han desarrollado y probado robots para realizar operaciones simples. Aunque parece que de momento no se ha llegado al punto de que un robot lo haga todo, aquí en Valencia ya se ha realizado la primera operación asistida con un robot en España. La utilización de un robot redujo los riesgos, los efectos secundarios y el tiempo de recuperación y aumentó la precisión[13]. En general, los beneficios asociados a esta inclusión superan con creces a los detrimentos.

Por otra parte, se están realizando muchas investigaciones en lo relativo a la inclusión de las IAs en el mundo de la educación. Tiene un gran potencial para acelerar el proceso de realización y desarrollo de los objetivos globales en torno a la educación facilitando el acceso, automatizando los procesos de gestión y optimizando los métodos que mejoran los resultados en el aprendizaje. En relación con los *chatbots*, son herramientas que pueden ejercer de profesor o tutor en entornos que es necesario atender a preguntas y consultas de los estudiantes. La inclusión de IAs en la educación también nos permite mejorar los tratamientos especializados a las personas con necesidades especiales, ya sean de atención o educacionales. Pero también plantea otro gran reto como es la privacidad y seguridad de la información. La IA se basa en el análisis y procesamiento de grandes cantidades de datos y, si se usa para dar un tratamiento especializado, se requiere de mucha información y datos personales lo que podría poner en peligro los derechos de las personas y sus vidas privadas. Aquí se entra en esa encrucijada que tantas veces aparece cuando se avanza con la tecnología: ¿Qué se está dispuesto a sacrificar para avanzar? En este caso, con la educación, se nos está pidiendo que sacrifiquemos toda privacidad de las primeras etapas de la vida para avanzar en metodologías educativas, escolarización y calidad de enseñanza. Este sacrificio nos permitiría diseñar diferentes metodologías que englobasen a todo el mundo en la enseñanza, permitiendo a los profesores escoger el mejor método para cada caso[14].

En el campo de la asistencia por IA hay muchísimas más aplicaciones que no requieren de tanta precisión o sacrificio personal. Una de ellas es la programación. Un *chatbot* nos permite realizar un código básico y simple que nos sirva de base para un desarrollo más complejo y profundo o nos puede ayudar a encontrar y entender un error en el código que aparentemente no tiene sentido. Otra aplicación sería en el mundo de las humanidades. Se han desarrollado varias herramientas que dado un *prompt* generan una historia de la longitud que se solicite o que te traducen textos mejor que los traductores online comunes (genera un producto de mejor o peor calidad, pero hacen el trabajo). Estas últimas herramientas traen a la orden otro debate: ¿Qué permisividad se le da a los textos, códigos y traducciones generadas por IA? ¿Qué valor artístico se les puede dar? ¿No es el arte algo humano? Hay gente que ha estudiado una carrera, se ha especializado, ha practicado y realizado diversos proyectos, que han tenido que especializarse en una o varias lenguas para poder trabajar, pero llega una máquina que hace lo mismo (de menor calidad), por menos dinero y a veces con resultados casi indiferenciables del arte humano[15].

Las opiniones y respuestas a los debates morales aquí planteados dependen de cada uno, pero lo que se puede sacar de esto es que no se han desarrollado IAs para la mejora de la accesibilidad y usabilidad de las aplicaciones. Se pueden usar IAs para programar, para que diseñe una página, para que dé las pautas de una mejor accesibilidad y usabilidad, pero, al fin y al cabo, todo depende de los programadores, y de que se cumpla con nuestra parte del trabajo. Sin una gran inversión de una gran empresa tecnológica o una buena financiación en

investigación, no se espera en un futuro cercano un desarrollo de este tipo de herramientas. Esto obliga a muchas personas a sentirse marginadas y excluidas de los avances tecnológicos cada vez más constantes y frecuentes. Cada cambio de interfaz y cada actualización que añade o cambia algunas funcionalidades suponen un gran cambio para los usuarios frecuentes con algún tipo de discapacidad visual o intelectual que ya se habían acostumbrado a la versión anterior. Por eso el desarrollo de estas herramientas es necesario, porque no todos tienen las mismas facilidades, pero si merecen las mismas oportunidades. Facilitar la adaptación a un cambio o educar y enseñar al usuario en un entorno ayudaría a que muchos más usuarios se sientan cómodos usando la tecnología y que fuesen más independientes a la hora de moverse por este mundo cada vez más lleno de tecnología.

3. Casos de uso

A continuación, se van a ejemplificar un par de casos de uso para determinar la utilidad y el uso de la aplicación. Estos casos de prueba van a intentar reproducir lo más fielmente la realidad. Se creará un perfil de una persona qué más utilidad le pueda encontrar al sistema y se ejemplificará qué pasará antes, durante y después de que se muestre la ayuda. Siguiendo este modelo se reproducirán varios casos con diferentes perfiles, de diferente edad y situación que muestran una gran dificultad a la hora de manejar aplicaciones web. Para ambos casos descritos se puede hacer uso de este diagrama:

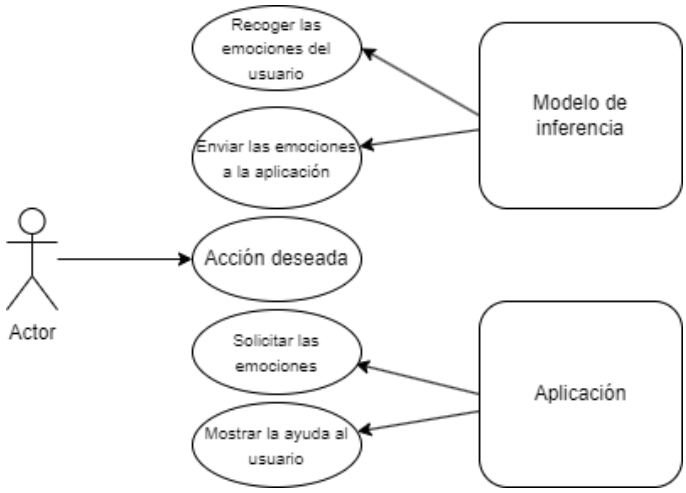

Fig. 1 Diagrama de Caso de uso

3.1 Primer caso de uso

Con el primer caso de uso vamos a ejemplificar un trámite bancario online con una persona de la tercera edad. Estos trámites cada vez se hacen más por internet, y más en zonas rurales donde están desapareciendo las sucursales bancarias. Esta tendencia margina a aquellos que no tienen acceso a los recursos tecnológicos o a una educación sobre estos. Con este caso de uso se pretende ejemplificar la utilidad de esta herramienta para facilitar y educar sobre estos trámites a las personas que más lo necesitan.

El actor principal de este caso de uso es nuestra persona de la tercera edad, a la que a partir de ahora nos referiremos como “usuario no-experto”. Este usuario no-experto no ha recibido nunca una educación sobre tecnología ni sobre su uso, ya que vive en una zona rural donde recientemente han cerrado la sucursal bancaria. El usuario no-experto ha realizado siempre el pago de las facturas desde esa sucursal ahora cerrada. Antes de cerrar le advirtieron que a partir de ese momento debería proceder al pago de las facturas desde un terminal del

banco, cosa que nuestro usuario no-experto no ha realizado nunca, aunque sí que ha usado los cajeros para sacar dinero. Nuestros actores secundarios son nuestra herramienta de inferencia de emociones y ayuda al usuario y la aplicación del cajero a la que se le ha integrado la herramienta para estos casos.

El usuario no-experto ya tiene facturas que pagar y se ha acercado al cajero a efectuar su pago. Cuando llega procede a introducir la cartilla del banco que usa siempre e introduce el PIN, como hace cuando va a sacar dinero. Ahora tiene delante el menú de opciones que le muestra sus acciones más frecuentes o favoritas (donde está la opción de sacar dinero), otras opciones, como ingresar dinero o realizar una transferencia, y un botón de "Más acciones". Al no ver la opción que busca en la pantalla principal, va al menú de "Más acciones". Una vez en este menú se desplaza usando las flechas que se encuentran en el lateral para visualizar la lista al completo. Se desplaza varias veces por el mismo menú de arriba abajo porque no encuentra la opción que busca. Entra un par de veces en la opción de "Transferencia entre cuentas" pensando que tiene que ser esta opción, pero el mismo sistema le describe que es entre cuentas propias. Después de volver arriba del todo del menú 3 veces el usuario no-experto se empieza a frustrar y frunce el ceño. En ese momento, el sistema le muestra un mensaje: "Parece que no encuentra lo que busca – Si desea pagar facturas debe ir a la opción 'Pago de servicios'". Ahora el usuario no-experto va directo a buscar la opción que le ha indicado el sistema y procede al pago de las facturas sin más impedimento.

En este caso, la herramienta ha realizado su función aclarando un concepto que con su nombre específico o técnico puede ser confusa y poco clara y que el usuario no-experto no ha encontrado por su desconocimiento del tecnicismo y ha requerido de ayuda del sistema para que se lo aclarase. Una vez el sistema ha mostrado la ayuda y el usuario no-experto ha comprendido el tecnicismo, ha realizado el resto de la gestión sin problema ni inconvenientes.

3.2 Segundo caso de uso

Este caso de uso pretende ejemplificar la primera aproximación de una persona adulta mayor a las redes sociales. Desde su creación su uso ha ido incrementándose y cada vez más personas tienen un perfil en una de ellas que le permite mantener el contacto con sus amistades o reconectar con viejos amigos. Pero si eres una persona nueva en este mundo puede dar miedo una primera aproximación ya que hay muchos conceptos y palabras nuevas que aprender. Con este caso se pretende ejemplificar como la herramienta permite un acercamiento amistoso a una nueva tecnología y que el usuario aprenda a su ritmo.

El segundo caso se trata de una persona con muy poco contacto con la tecnología que desea mantener más contacto con sus amistades a través de las redes sociales. A esta persona la llamaremos "nuevo usuario" a partir de ahora Al ser el único de su grupo de amigos que no tiene un perfil, le han convencido de que se haga uno y se lo han creado. Esto fue en compañía de sus amigos, pero ahora está solo en casa y no recuerda lo que le explicaron sus amigos, ya que es su primer contacto con las redes sociales. Como actores secundarios mantenemos la herramienta y la aplicación que la ha integrado en este caso, la red social.

El nuevo usuario quiere buscar y añadir a sus amigos a un viejo amigo de la escuela. Procede a desbloquear su smartphone y abrir la red social. Una vez dentro ve la lupa que sus

amigos le indicaron que sirve para buscar a la gente y la pulsa. Ya en la pestaña del buscador escribe el nombre de su antiguo compañero de clase. Le aparecen varios resultados y va navegando entre ellos buscando a la persona que le interesa. Viendo las fotos del 4to usuario se da cuenta de que ha encontrado a la persona que buscaba. Busca la opción de añadir a sus amigos, va arriba y abajo dentro del perfil del usuario y entra y sale varias veces de él, pero no da con el botón. Cuando ya va 3 veces que entra y sale del perfil, la aplicación le muestra un mensaje: "¿Estás un poco perdido? – Si quieres añadir a alguien a tus amigos, prueba a darle al botón de 'Seguir' de su perfil – Si quieres chatear con él, dale al botón que contiene el icono de un sobre". En ese momento el nuevo usuario recuerda que sus amigos se lo explicaron, pero que ahora, buscándolo, le había parecido extraño "seguir" a alguien. Cierra el mensaje de la aplicación y le da al botón de "Seguir".

La herramienta, en este caso, ha realizado su función en una red social, enseñando a un nuevo usuario qué acción realizar para añadir a alguien a sus amigos o seguir. Ahora el nuevo usuario ya sabe que cuando quieres añadir a alguien a tus amigos le tienes que dar a "Seguir". La próxima vez es posible que no necesite el mensaje y que sepa hacerlo por sí mismo. Además, en este caso, al haber más de una posibilidad de confusión sobre qué busca el usuario, se han incluido dos explicaciones para que sea cual sea la necesidad se cubra y el usuario tenga una solución.

4. Diseño de la solución

A continuación, se va a explicar la arquitectura del sistema ideado y el porqué de cada componente que forma parte. Además, se hará una descripción de los componentes ideados para añadir a la aplicación de prueba y mostrar la ayuda. Y finalmente, se hará una descripción de la conexión que se realiza entre el modelo de inferencia y la aplicación cliente.

4.1 Diseño

El sistema está compuesto e ideado para integrar 2 componentes principales. El primero es el modelo de inferencia de emociones a partir de expresiones faciales y el segundo es una aplicación cliente que interactúa con este modelo. El modelo de inferencia obtiene la información a procesar de una webcam perteneciente al ordenador o móvil que hace uso de la aplicación. Es una parte vital para este sistema, ya que es la fuente principal y única de información para obtener la inferencia sobre las expresiones faciales. Además, el modelo debe de proveer al segundo componente de la información de los sentimientos inferidos del usuario, por lo tanto, ha de existir una comunicación entre ambos. Esta comunicación se especificará más adelante. Nos quedaría tal que así el sistema:

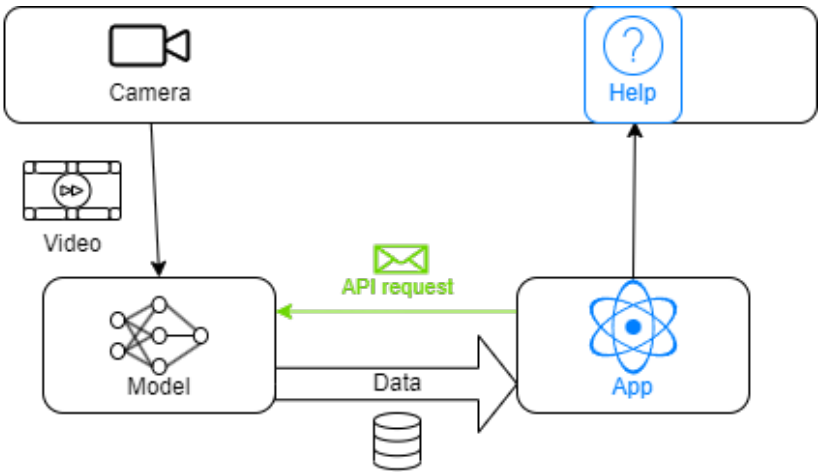

Fig. 2 Diagrama de la arquitectura del sistema

Como se puede observar, es un sistema relativamente sencillo, pero bien conectado. Al diagrama se ha añadido la ayuda que manda la aplicación al usuario en caso de detectar que la necesita en base a los datos obtenidos del modelo. También se ha dibujado la petición de la aplicación cliente al modelo de inferencia, dado que estos estarán conectados a través de un *end-point* y la aplicación deberá de realizar peticiones periódicas para obtener los datos de las

emociones. El modelo, como se puede observar en el diagrama, responde con los datos que tenga en ese momento.

A la hora de mostrar la ayuda se valoran 2 opciones. Hay que considerar que es lo que menos importuna al usuario y la forma más acoplada al estilo de la aplicación. La primera es un modal de React que se abra cuando la aplicación detecta que el usuario requiere de ayuda. Un modal es un componente de interfaz de usuario que se utiliza para mostrar contenido emergente sombre el contenido principal de la aplicación. Es muy útil a la hora de añadir información en formularios o mensajes de confirmación. Se presenta como una ventana emergente perteneciente a la aplicación, pero no ocupa toda la pantalla, solo una porción, y ensombrece el fondo para mandar toda la atención del usuario al modal. Este componente se puede cerrar haciendo clic en cualquier parte de la pantalla o si el desarrollador así lo desea, en un botón específico. Este modal contiene un título y el cuerpo del mensaje, que consta de un texto explicativo. Está opción puede llegar a ser una manera muy invasiva de ofrecer la ayuda, ya que no es seguro que el usuario la requiere según las predicciones y puede que el usuario solo esté navegando por la aplicación y la aparición del modal le importune, le genere frustración y llegue a dejar de usar la aplicación por ello.

Debido a esto se piensa una segunda opción, un pop-up. Un pop-up es un componente de interfaz gráfica parecido al modal, pero en vez de hacer uso de la pantalla completa solo aparece en la zona designada. Puede cerrarse manualmente, que con el tiempo desaparezca o ambas. Aporta información extra sobre lo que hace el sistema o para confirmar alguna acción del usuario. Son propósito que requieren de manera temporal la atención del usuario. En nuestro caso, se ha diseñado para que aparezca en la esquina inferior izquierda que muestre el mensaje de ayuda, se pueda cerrar y si no, desaparezca al cabo de un tiempo. Está segunda opción es menos invasiva y agresiva y ofrece al usuario la opción de ignorarla en caso de no necesitarla.

4.2 Conexiones

A la hora de comunicar ambas partes se nos presentan varias opciones. Cada una tiene sus ventajas e inconvenientes, por lo que hay que escoger la más apropiada para este caso.

En primer lugar, tenemos las APIs. Estás pueden ser RESTful (Rest), SOAP o GraphQL. La primera suele ser la opción más común dado que utiliza los protocolos HTTP[1] y JSON[2] para el formato de los datos. SOAP utiliza XML[3] para el formato de los datos, pero se usa para aplicaciones que requieren acciones complejas y un nivel adicional de seguridad. Por último, GraphQL, es una alternativa a Rest ya que permite especificar exactamente qué datos se necesitan, optimizando así el rendimiento, pero, es más complejo de configurar y requiere de un buen desarrollo para evitar problemas de rendimiento y seguridad a causa de consultas mal formuladas u optimizadas.

Otro de los métodos más comunes para conectar aplicaciones y compartir datos es la mensajería asíncrona. Este método es el modelo de mensajería asíncrona Pub/Sub, que permite la comunicación entre sistemas de manera que el emisor envía el mensaje y el consumidor o

[1] Protocolo para la comunicación y transferencia de información entre cliente y servidor de manera estandarizada
[2] Formato de texto que extiende de JavaScript que tiene como objetivo el almacenamiento e intercambio de datos
[3] Lenguaje de marcado que define una estructura para almacenar y transportar datos

consumidores lo reciben sin que el emisor conozca a los suscriptores. Algunas de las herramientas que permiten este tipo de comunicación son Apache Kafka, RabbitMQ o AWS SNS, pero estas herramientas son demasiado complejas para el desarrollo que se ha propuesto.

Teniendo todo esto en cuenta, se ha optado por la primera opción. Más en concreto en una API Rest, dado que no se requiere de mucha configuración y es fácil de implementar. Esto permitirá la creación de un único *end-point* de comunicación entre el modelo de inferencia y la aplicación cliente, asegurando una integración fluida entre ambos componentes. Para el correcto funcionamiento de este *end-point*, se ha de tener en cuenta que los datos se han de transformar a JSON antes de enviarse. Una vez se hayan enviado, la aplicación cliente es responsable de realizar las acciones correspondientes para su almacenamiento o comprobación.

5. Tecnologías y puesta a punto

Las emociones son condiciones dinámicas, cognitivas y fisiológicas a experiencias, pensamientos o interacciones sociales. El reconocimiento de estas se ha demostrado crucial para muchas áreas, así como el marketing, la atención sanitaria o la seguridad. En lo relativo a la atención sanitaria se han hecho grandes investigaciones sobre trastornos neurológicos como el Parkinson o la esquizofrenia. También son clave para detectar el dolor o la depresión e incluso sirven para diagnosticar autismo o trastorno de déficit de atención e hiperactividad (TDAH). Donde también han demostrado una gran importancia es en la interacción humano-máquina, por lo tanto, perfeccionar y automatizar la detección de emociones es necesario para un futuro no muy lejano donde las interacciones con máquinas sean constantes y la curva de aprendizaje aumente en los que más ayuda necesitan[16].

Para la detección de emociones en estudios e investigaciones se puede recurrir a 3 opciones. La primera son los cuestionarios. Con ellos se pretende que la persona recapacite sobre sus sentimientos y desarrolle su inteligencia emocional sobre ellos. La segunda opción son las señales fisiológicas. Detectar y estudiar estas señales es más complicado que cualquiera de las otras opciones, pero también es el método más fiable y común para recoger información. Estas señales son las que activa nuestro cerebro y cuerpo ante un estímulo, es decir, la emoción más física. Un electroencefalograma, un electrocardiograma, la medición de la temperatura de la piel y el movimiento de los ojos (*eye-tracking*), así como la medición de la respiración son pruebas que se usan para recoger esas señales fisiológicas y estudiarlas para determinar las emociones. Como se puede comprobar, el equipo necesario para realizar estas pruebas no está al alcance de todos, por lo que estas señales solo se miden en investigaciones. Y por último, las señales que se van a utilizar en este trabajo, las señales físicas. Estas son las reacciones casi inevitables y visibles a las emociones. Incluyen expresiones faciales, variación del tono y la velocidad del habla, gestos y lenguaje corporal.[16] Todas estas señales son casi automáticas ya que son reacciones a los cambios fisiológicos anteriormente comentados, por lo tanto, de fácil recopilación y estudio. Este trabajo se centrará solo en las expresiones faciales.

En lo referente al mundo de las IAs de reconocimiento de emociones, se han realizado varios modelos en base a diferentes señales, tanto fisiológicas como físicas. Dentro de la categoría de señales fisiológicas tenemos los modelos en base a electrocardiogramas o el movimiento de los ojos (*eye-tracking*). Los conjuntos de datos usados para entrenar los modelos son privados dada la sensibilidad de estos, aunque este hecho complica el desarrollo de este tipo de aplicaciones. Con señales físicas, se han realizado modelos en base a el tono y la velocidad del habla y las expresiones faciales. Estos conjuntos de datos sí son en su mayoría públicos.[16] El conjunto de datos escogido para este trabajo es FER2013, proveniente de una competición de Kaggle llamada "*Challenges in Representation Learning: Facial Expression Recognition Challenge*"[17].

5.1 Modelo escogido para este trabajo

Para el desarrollo de este trabajo se ha escogido un modelo IA de reconocimiento de expresiones faciales simple ya realizado. Este modelo hace uso de un detector de rostros Haar Cascade [4] implementado en OpenCV [5] y un clasificador de emociones implementado en TensorFlow[6]. Como otros detectores o reconocedores de emociones, este no es más que otro detector de objetos especializado en rostros, a partir de los cuales infiere la emoción expresada. Los pasos que sigue el reconocedor original son:

1. Hace una captura del vídeo de entrada
2. Lo pasa por el detector de rostros
3. Realiza las predicciones de las emociones del rostro detectado
4. Crea un gráfico de barras para mostrar la distribución de probabilidad
5. En base a las coordenadas producidas por el detector y la predicción del clasificador, dibuja los resultados en la imagen original

Para el entrenamiento y la ejecución del modelo se instala Anaconda, una distribución de Python para simplificar la gestión de paquetes y la implementación de entornos. Para la creación del entorno del modelo se hace uso de un archivo YAML[7] (apéndice A.1). Al hacer uso de la librería *tensorflow-gpu* el ordenador ha de contar con una tarjeta gráfica y tener instalados los drivers de Nvidia y CUDA Toolkit[8]. Al inicio del desarrollo del trabajo se trató de instalar en un ordenador de unos 7 años con una GeForce 1050 y Ubuntu 22. Pero debido a la antigüedad de la gráfica y la versión actual de los drivers y el software para Linux, fue imposible instalar lo necesario para el correcto funcionamiento del sistema. Por lo tanto, se optó por usar el mismo ordenador, pero con Windows. En este caso sí se pudo instalar todo sin problema y se pudo comenzar con el proyecto. Una vez creado el entorno en Anaconda se procedió a la construcción del modelo.

El modelo consta de 3 archivos principales. El primero declara las emociones sobre las que se va a inferir y los colores asociados a estas en BGR. Por ejemplo, 'enfadado' es (0, 0, 255). Hay que aclarar que los datos con los que se va a entrenar la máquina tienen 7 emociones, pero para este modelo solo se han declarado 6, juntando las 2 primeras en 1 (asco y molestia en enfado). Todos los datos vienen en un archivo CSV[9] en el que las emociones van asociadas a un entero que la identifica y una lista de 2304 números correspondientes a los píxeles de la imagen. Para descargar estos datos se necesita una cuenta en la web mencionada con anterioridad.

El siguiente archivo es *train_model.py* que como bien dice el nombre, es el que se encarga de entrenar el modelo y como tal hace uso de 16 importaciones de TensorFlow-Keras. Comienza creando la función que construye la arquitectura de la red compuesta por dos bloques

[4] Método de detección de objetos basado en características, comúnmente para identificar rostros
[5] Librería de código abierto para procesamiento de imágenes y visión por computadora
[6] Librería de código abierto para el aprendizaje automático y la inteligencia artificial
[7] Formato de serialización de datos legible por humanos
[8] Conjunto de herramientas de NVIDIA para la computación paralela en GPU
[9] *Comma Separated Values*, formato de archivo utilizado para almacenar datos tabulares

de 32 filtros, dos de 64 filtros y dos de 128 filtros, seguidos de tres capas densas. Para evitar el *overfitting* se añaden varias capas de *Dropout()*. La siguiente función carga los datos para entrenamiento, validación y pruebas a partir del archivo CSV. Primero define las listas de las imágenes y etiquetas de cada subconjunto. Luego abre el CSV en modo lectura para iterar sobre cada una de sus líneas y luego modifica las etiquetas, acorde a la modificación comentada con anterioridad, para que los datos y las inferencias sigan teniendo sentido. Seguidamente recorre la columna '*pixels*' para convertirla en una matriz 48x48 y la columna '*Usage*' para determinar a qué subconjunto pertenece la imagen. Y para finalizar la función convierte las listas en arrays de NumPy[10], aplica codificación *one-hot* a las etiquetas y devuelve el conjunto de datos. El archivo finaliza con la construcción de la ejecución. Primero recoge los datos y los carga, luego genera la red neuronal y, por último, define el *callback* para guardar *checkpoints* de los mejores modelos encontrados. El procesamiento será de 128 imágenes a la vez y durante 300 epochs.

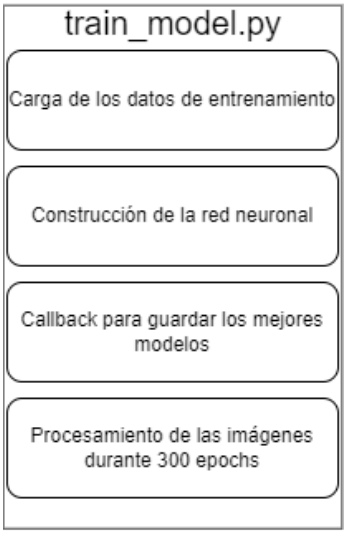

Fig. 3 Diagrama de ejecución de train_model

El último archivo es *recognize.py*. Este es el que se encarga de analizar la imagen de entrada y mostrar los datos de la inferencia por pantalla junto a la imagen que se está analizando (en nuestro caso, la webcam). Primero está la función que va a mostrar por pantalla las probabilidades de las predicciones en un gráfico de barras y la función que dibuja sobre la cara el rectángulo del color correspondiente a la emoción. La última función auxiliar de este archivo es la función que asocia la imagen a una emoción. A continuación, está la declaración de las *flags* de entrada. Después se carga el mejor modelo, el vídeo de entrada y el detector. Finalmente, itera sobre cada fotograma del vídeo y lo redimensiona para que el procesamiento sea más rápido. También genera un lienzo sobre el que se dibujará el gráfico de barras de las emociones. Pasa el detector sobre una copia en escala de grises del fotograma actual redimensionado y si hay detecciones se queda con la que ocupe mayor área para disminuir la posibilidad de falsos positivos. Se ejecuta el clasificador de emociones sobre la región de interés asociada al rostro detectado y ensambla el gráfico de barras con la distribución de la detección. Finalmente dibuja el cuadrado sobre la cara detectada y clasificada con el color correspondiente a la emoción predominante y muestra el resultado por pantalla[18].

[10] Librería de Python para cálculos numéricos y operaciones con matrices y arrays

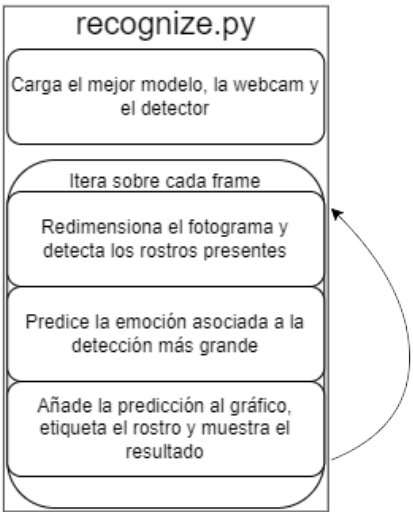

Fig. 4 Diagrama de ejecución de recognize.py

5.2 Aplicación escogida para este trabajo

Para probar el modelo se ha escogido una aplicación simple. Se trata de una aplicación realizada en la asignatura de Proyecto de Ingeniería de Software (PIN) durante este curso académico. La aplicación se llama Beta y es una red social ideada para extraer canciones de Spotify acordes a el estado de ánimo que introduce el usuario en el momento en el que inicia la aplicación. De estas recomendaciones, el usuario escoge la que más se asocia a su emoción actual o la representa y se publica en un *feed* junto al resto de canciones de sus amigos.

Beta fue un proyecto conformado por 5 estudiantes de Ingeniería Informática y 2 de Bellas Artes siendo estos: Shadi Awad, Alejandro Camps, Sergio Figueroa, Mario Luque e Inés Arcas y Pablo Meca (yo misma formé parte del equipo de desarrolladores). El proyecto, como parte de la asignatura de PIN, se presentó en la Feria de Proyectos que la ETSInf organiza en diciembre. Esta feria está patrocinada por grandes empresas como NTT Data o HP, y fueron estas empresas las que dieron el premio a Beta de "Proyecto Destacado por las Empresas Patrocinadoras".

La aplicación está construida con las siguientes tecnologías:

5.2.1 Ionic React

Ionic es un *framework* para el desarrollo de aplicaciones móviles, web y de escritorio multiplataforma. Esto implica que, con un buen desarrollo y una buena adaptación a los diferentes medios, con el mismo código puedes tener la aplicación en todos los dispositivos que desees. Esto aporta gran versatilidad a la aplicación y que el desarrollo multiplataforma sea

paralelo al desarrollo principal de la aplicación. También facilita la creación de interfaces atractivas y funcionales y proporciona herramientas para el desarrollo rápido y la implementación de aplicaciones para varios sistemas operativos de móviles.

Ionic trabaja con otras tecnologías web como HTML, CSS y JavaScript, aunque en nuestro caso usamos TypeScript para el tipado de datos y la orientación a objetos. Este *framework* aporta componentes de interfaz prediseñados y permite trabajar con otros *frameworks* gráficos como Angular, React o Vue.js. En nuestro caso se escogió React ya que es el más rápido y fácil de aprender, el más común y un miembro del equipo ya lo había usado con anterioridad.

React es una librería de código abierto para el desarrollo de interfaces basadas en componentes en JavaScript. React se puede usar para desarrollar las aplicaciones conocidas como de página única (un sitio web que cabe en una sola página, como si fuera una aplicación de escritorio), aplicaciones móviles o aplicaciones web *server-rendered* (generan dinámicamente páginas HTML en base a scripts y peticiones de los usuarios). Al igual que Ionic, se puede realizar el desarrollo completo de la aplicación en TypeScript ya que la mayoría de las librerías están implementadas para ambos.

5.2.2 Firebase

Firebase es una plataforma de desarrollo de aplicaciones que ofrece a su vez un conjunto de servicios de *backend* en la nube, diseñados para simplificar el proceso de creación y gestión de aplicaciones. Proporciona soluciones para alojar base de datos no relacionales y servicios de autenticación que permite a los desarrolladores implementar seguridad y manejo de usuarios de manera eficiente añadiendo la opción de inicio de sesión a través de otras aplicaciones. Además, incluye emuladores para pruebas en desarrollo, lo que facilita la integración y el despliegue continuo. También ofrece herramientas avanzadas para el análisis de datos de la aplicación, ofreciendo información detallada sobre el uso de la misma, las consultas realizadas a la base de datos, el rendimiento general y la estabilidad del sistema. Estas herramientas permiten a los desarrolladores construir aplicaciones de gran calidad, monitorearlas y optimizar su funcionamiento al desplegarlas y alojarlas en el mismo Firebase.

Firebase ofrece dos maneras de almacenar datos: Firestore y Realtime. Firestore permite el almacenaje de datos en documentos, que a su vez se agrupan en colecciones. Los documentos pueden contener datos estructurados en pares clave-valor y soportan tipos complejos como arrays y mapas anidados. Ofrece también capacidades de consultas que permiten filtrar, ordenar y combinar datos al igual que operaciones por lote, que permite modificar múltiples campos de manera atómica y segura. Por otra parte, Realtime permite la sincronización instantánea entre todos los clientes conectados y almacena dichos datos en formato JSON y los estructura de manera jerárquica, lo que permite a los desarrolladores organizar y acceder a los datos mediante rutas URL. Ambos están diseñados para escalar automáticamente y para guardar datos en la caché local en caso de desconexión y que así, la aplicación siga funcionando si problemas. También siguen un sistema de reglas basado en JSON, que permite definir permisos detallados para leer y escribir datos en función de la autenticación u otros criterios específicos como fechas o roles. En el caso de Beta se escogió Firestore por el uso de colecciones y documentos y las diferentes opciones de consultas.

Del resto de servicios y herramientas que ofrece Firebase también se usó el *Authenticator*, los emuladores y el servicio de alojamiento. El *Authenticator* simplificó la implementación de un sistema de usuarios, el inicio de sesión y el registro. De esta manera, cuando el usuario inicia sesión, el usuario y su id, se quedan almacenados tanto en la aplicación como en Firebase, lo que simplifica también los permisos para realizar ciertas acciones. Respecto a los emuladores, se usaron principalmente para las pruebas en desarrollo. Estos se ejecutan en local y replican aquellos servicios que se requieran, desde *Authenticator*, hasta Firestore. Las pruebas en local de esta manera eran mucho más realistas que si no se llega a replicar el entorno y evitó muchos problemas que se habrían desarrollado durante la exposición en la feria. Y el servicio de alojamiento aportó un lugar donde desplegar la aplicación para la feria y para usos futuros como entrevistas de trabajo.

5.2.3 React Redux Toolkit

Durante el desarrollo de la aplicación se hizo uso de varias librerías de React que aportan funcionalidades, componentes y elementos estéticos. La librería usada para la gestión de estados dentro de la aplicación fue React Redux Toolkit. Esta librería fue diseñada con el propósito de hacer el desarrollo de *reducers*[11] más rápido y eficiente. Es compatible con TypeScript y, aunque añada muchas configuraciones para su uso, permite el uso de middlewares para extender funcionalidades. React Redux Toolkit permite la gestión tanto del estado local como del estado global, según las necesidades de la aplicación.

5.3 Otras tecnologías

A parte de las tecnologías y aplicaciones ya mencionadas, también se ha hecho uso de otras tecnologías durante el desarrollo. Estas tecnologías se han usado tanto para leer y desarrollar código fuente como para almacenar los diferentes proyectos.

5.3.1 Visual Studio Code como IDE

Como entorno de desarrollo integrado (IDE) se ha optado por Visual Studio Code, o VS Code de ahora en adelante para simplificar. Es un editor de código fuente gratuito y multiplataforma que combina simplicidad y flexibilidad. La principal razón tras la elección de este IDE es la familiaridad de la autora con el mismo, pero también las funcionalidades que ofrece y aporta al proyecto.

VS Code permite añadir puntos de parada en el código, que pausan la ejecución del código en un punto en concreto para observar los valores de las variables en ese instante y facilitar la depuración. Esto, junto a los logs, permite una fácil depuración de código y la detección de errores.

Por otro lado, VS Code tiene integrado un mercado de extensiones que desarrollan los usuarios mismos e incluso algunas empresas (como Azure o GitHub). Estas extensiones aportan funcionalidades extra, facilitan la escritura de código fuente con *snippets*[12] o modifican el

[11] Fragmento de código reutilizable o plantilla que facilita la escritura

aspecto del IDE para hacerlo más estéticamente agradable y adaptar el tema de color a un lenguaje concreto y facilitar la lectura. En este caso se ha hecho uso de varias extensiones que cierran y completan automáticamente etiquetas de tipo HTML, de extensiones que aportan *snippets* para el desarrollo y del *debugger*[13] de Python.

5.3.2 GitHub para el control de versiones de los proyectos

GitHub es una plataforma de almacenamiento de proyectos y para el desarrollo colaborativo de software, basada en Git. Esta plataforma permite mantener uno o varios repositorios en la nube para que uno o varios autores puedan colaborar en el desarrollo de un proyecto de una manera controlada. Ofrece la posibilidad de revertir posibles cambios destructivos y el trabajo paralelo en ramas. Al no estar almacenado de manera local en el dispositivo donde se está trabajando, permite el trabajo desde cualquier lugar en distintas máquinas, característica sumamente útil durante el desarrollo de este proyecto.

Git es un software de control de versiones para la eficiencia, la compatibilidad y la confiabilidad del mantenimiento de versiones de aplicaciones con una gran cantidad de archivos de código fuente. El propósito es llevar un registro de los cambios realizados en el repositorio en una máquina y coordinar el trabajo de varios colaboradores sobre archivos compartidos. Está funcionalidad también ha jugado un papel crítico en el desarrollo de la aplicación.

[13] Herramienta utilizada para identificar y corregir errores

6. Desarrollo

A lo largo del proyecto se han desarrollado nuevas funciones y componentes, se han modificado los elementos ya construidos y se han encontrado errores que se han corregido. En las siguientes páginas se va a exponer todas las modificaciones y correcciones realizadas, así como los desarrollos que se han llevado a cabo. También se hablará de las decisiones de diseño tomadas para los mensajes de ayuda y sobre qué límite poner para mostrar o no el mensaje de ayuda, es decir como se ha determinado que el usuario necesita ayuda.

Durante el entrenamiento y la construcción del modelo se realizaron ciertas modificaciones para que este funcionase como se requería. Primero, se eliminaron las *flags* de compilación. Las que se habían determinado en el modelo original eran: '-m' para determinar el modelo (opcional), '-d' para determinar el detector de rostros (requerida) y '-v' para determinar la fuente de entrada de video (requerida). La primera al ser opcional no se modificó ni se borró, pero las dos restantes sí. Para el detector de rostros se puso a mano la dirección del archivo "*haarcascade_frontalface_default.xml*", al igual que la entrada de vídeo se dejó solo en webcam (introducir un 0). Esto facilita la ejecución del modelo y el no tener que recordar qué *flags* usar o donde están los archivos. Finalmente, se ejecutó el entrenamiento del modelo, el cual duró 28h, y se procedió a comprobar que funcionaba correctamente junto con la webcam y que las predicciones eran fiables. En la aplicación no se han realizado grandes cambios visuales. Sí se han añadido elementos y componentes que se expondrán detalladamente a lo largo de esta sección.

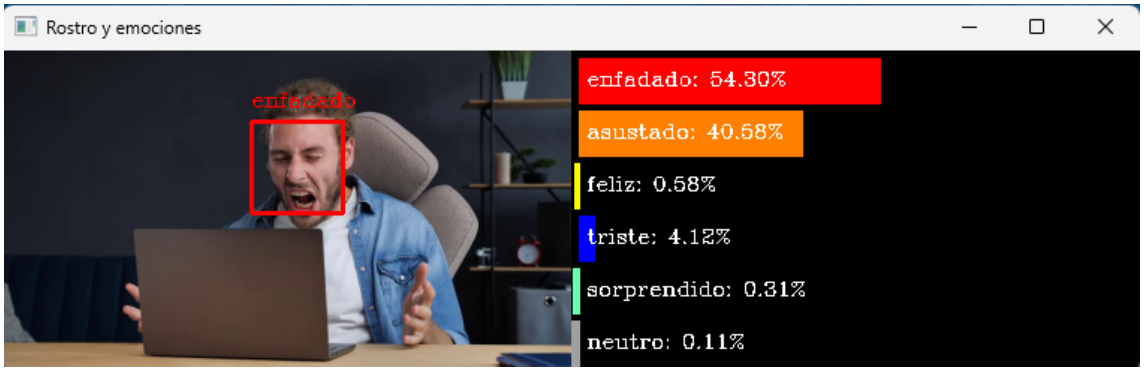

Fig. 5 Ejemplo de detección enfadado

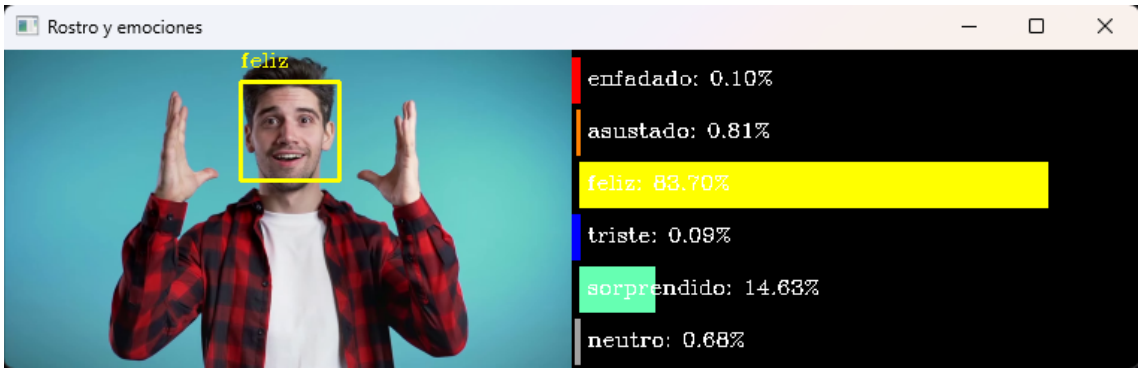

Fig. 6 Ejemplo de detección feliz

Fig. 7 Ejemplo de detección sorprendido

6.1 Implementación del diseño

La parte a implementar, del diseño que se ha comentado anteriormente, son los componentes de la aplicación cliente para mostrar el mensaje de ayuda, ya que la parte de la conexión entre los dos componentes se explica en la próxima sección y las ayudas que manda la aplicación cliente en la posterior.

El primer componente pensado es un modal de React (apéndice A.1). Como ya se ha explicado anteriormente un modal es un componente de interfaz gráfica que muestra contenido emergente perteneciente a la aplicación y que aporta información extra. Esta ha sido la primera opción para el diseño del componente que muestra el mensaje de ayuda, ya que independientemente de la atención prestada o la conciencia de entorno del usuario, el componente bloquea la acción sobre el resto de la aplicación. Pero como también se ha explicado con anterioridad este componente se puede cerrar con facilidad y sin que el usuario llegue a leer el mensaje. Por lo tanto, se añadió un botón, en la esquina superior derecha del mismo componente, para que independientemente de donde haga clic el usuario solo se cierre dándole clic al botón de cierre. El componente se desarrolló, pero el uso de este componente podría llegar a ser molesto y frustrante para los usuarios expertos que usen de la aplicación y que hagan saltar el mensaje de ayuda sin necesitarlo. Por lo tanto, se comenzó a desarrollar la segunda opción.

La segunda opción explicada es un *pop-up* (apéndice A.2). Este componente emergente temporal es la opción menos invasiva de las dos, por lo que se propone como alternativa a la primera. Al igual que el modal, aporta información extra, pero en este caso puede ser temporal y cerrarse al cabo de un tiempo determinado, aunque también se ha añadido la opción de que se cierre manualmente con otro botón, también en la esquina superior derecha del componente. En este caso, al ser un componente emergente temporal que añade información no se puede hacer adaptable al contenido, por lo que, si el mensaje de ayuda es demasiado largo, se añadirá una elipsis y un botón para mostrar el mensaje completo solo si el usuario lo desea. La elipsis permite mostrar la primera parte del mensaje, dejando entrever de que trata, pero evitando que el componente ocupe más de lo necesario. Al realizar la elipsis, en caso de que se haga, se añaden puntos suspensivos donde se corta el mensaje y seguidamente se añade un botón de "Leer más…" para mostrar el mensaje completo. Para evitar que el componente desaparezca mientras se muestra el mensaje completo, se mantiene el componente y solo se cerrará manualmente en caso de que se esté mostrando el mensaje completo o cuando se tiene el cursor encima.

6.2 Implementación de las conexiones

Para comunicar las dos aplicaciones se ha optado por la creación de un *end-point* junto al modelo implementado. La tecnología para la creación de este *end-point* es Flask, un *framework* de Python que permite crear aplicaciones web rápidamente y con un número mínimo de líneas. Al constar de un solo *end-point* se optó por este *framework* por su simplicidad y rapidez para crearlo.

El modelo de inferencia se ha modificado para que cuando se infieran las probabilidades guardase la lista en un archivo JSON (apéndice A.3) generado para ello. Para ello se añadió al archivo la librería *json*[14] de Python y cuando ya tiene las predicciones y las va a añadir a la gráfica, se añaden también al JSON. En caso de que ya hubiese una detección previa, actualiza los valores del JSON con las nuevas predicciones. Con cada fotograma analizado las emociones cambian y el archivo nunca está vacío. De esta manera, cuando se realiza una petición al *end-point*, no hay lugar a que devuelva un objeto vacío ni a que de error cuando llegan los datos al destino. La aplicación web (apéndice A.3) se aloja en el puerto 5000 del servidor local. El *end-point* se ha definido como un método GET y con la dirección '*/emotions*'. Este solo se encarga de abrir el archivo JSON y transmitir el contenido a la aplicación que lo solicita.

La aplicación, por otra parte, manda una petición (apéndice A.4) cada 2s para obtener la información de las emociones del usuario. Cuando la recibe recoge los datos de las emociones que nos interesan, en este caso enfado, tristeza y neutro. Las guarda en un *store* (apéndice A.5) para que todos los componentes que lo requieran tengan acceso a ello y que el pop-up tenga acceso a las últimas emociones detectadas y se abra o no en función de estas.

[14] Librería de Python que permite convertir datos entre los formatos JSON y las estructuras de datos de Python

6.3 Ayudas y explicaciones

La aplicación ya tiene las emociones y las ha almacenado, ahora tiene que determinar que se considera que el usuario requiere de ayuda. Para ello se hace un uso de la detección por webcam de manera empírica. Con este uso se puede determinar que límites poner y varias condiciones diferentes para mostrar la ayuda. Por otra parte, el mensaje de ayuda todavía no se ha diseñado, pero para ello debemos de tener varias cosas en cuenta, como ya veremos.

6.3.1 Detección de la ayuda

El método escogido para determinar el comportamiento del modelo, como ya se ha especificado antes, es el empírico. Realizando un estudio y pruebas con diferentes personas, se han determinado varias condiciones para mostrar al usuario la ayuda. Una de ellas se trata de cuando el usuario muestra una clara expresión de enfado, la cual se determina en el modelo con que la predicción de enfado esté por encima del 50%, es decir, el modelo determina que con la mayor probabilidad el usuario está enfadado. En esta situación el resto de las predicciones no son lo suficientemente altas como para considerarlas.

Otra de las situaciones detectadas es cuando el usuario siente un poco de frustración, no un enfado marcado. En ese caso la expresión neutra está en un 40-50%, la de tristeza está en un 20-25% y la de enfado en un 30-40%. En ese caso se determinó que, si neutro está por encima de 30%, tristeza por encima de 20% y enfado por encima de 30%, la aplicación ofreciese ayuda al usuario. Se determinó 30 para la expresión neutra dada la variabilidad del modelo con el resto de las expresiones. Con esta condición se arriesga un poco con los límites, ya que puede ofrecer falsos positivos, pero al tener un mensaje de ayuda no invasivo, se puede ignorar y en cualquier momento si se determina crítica la situación, subir los límites para evitar falsos positivos.

Otra de las condiciones que se han especificado es que el usuario transite entre las ventanas o pestañas de la aplicación sin realizar ninguna interacción que determine una acción clara, es decir, que se navegue por la aplicación buscando algo, pero no lo esté encontrando. Para esto, se ha añadido al *store* de la aplicación una variable que aumenta con cada entrada a las diferentes pestañas. Esta variable se reinicia a 0 cuando se ha ofrecido la ayuda al usuario y cuando el usuario realiza una acción clara, como buscar a otro usuario o entrar a un chat. Por último, para evitar la aparición constante del mensaje de ayuda, el mensaje solo se volverá a mostrar 2min después de la última apertura. Y para evitar molestar al usuario experto con el mensaje constantemente, el mensaje ofrece la opción de no volver a mostrase.

6.3.2 Ayudas ofrecidas al usuario

Una vez se ha integrado el modelo de inferencia de emociones a la aplicación y se ha modificado la aplicación de la manera correspondiente, se procede a la construcción del mensaje de ayuda. Este debe de ser claro y conciso, de manera que el usuario no-experto que haga lo lea sepa que se le está diciendo y hacia donde se le está dirigiendo. Siguiendo otro trabajo de investigación de la UPV [19] en el que se propone un modelo conceptual para caracterizar las explicaciones. Adaptando el trabajo a nuestro caso podemos determinar, en base a las características de las explicaciones y las situaciones que se especifican, nuestros mensajes.

Para las características de la explicación hay que determinar el tipo de acción a explicar y el perfil del usuario. En nuestro caso, el tipo de acción a explicar se basa en las acciones del usuario y qué infiere el sistema que está buscando. Por otra parte, para determinar el perfil del usuario se va a establecer uno fijo, que es el de las personas que más van a hacer uso de los mensajes de ayuda y las explicaciones. Los usuarios expertos ignorarán el mensaje en caso de que se les muestre o lo cerrarán sin leerlo dado que no necesitan una explicación. Una vez tenemos estas variables establecidas, toca determinar el nivel de atención y el nivel de criticidad de la acción, ambas determinadas por un número del 1 al 3. El nivel de atención se va a establecer en un 3 (alto), puesto que se asume que el usuario no-experto está dedicando todas sus capacidades y atención en qué está haciendo y en qué está pasando. Además, el nivel de criticidad de la acción se establece en un 2 (intermedia) ya que no se espera la explicación de ninguna acción crítica, pero sí se va a explicar la acción que desea realizar el usuario.

También se especifica el tipo de explicación a ofrecer, si se repite o no, de qué manera se proporciona y si es *feedback* o *feedforward*. En nuestro caso se explica qué tiene que hacer el usuario para realizar la acción que ha determinado el sistema que se desea hacer, por lo tanto, el tipo de explicación será *Cómo*. El mensaje o la explicación se le ofrecerá al usuario una vez el modelo de inferencia de emociones haya detectado una expresión de frustración o enfado y la aplicación haya recibido esa información y se den las condiciones apropiadas para que se muestre el mensaje. Teniendo en cuenta el contexto, el mensaje o explicación se mostrará de manera sincrónica, cuando el usuario lo necesite. Además, para evitar frustración adicional o si se ha olvidado el mensaje o ha pasado tiempo entre usos y no se recuerda cómo realizar la acción, el mensaje es repetitivo. Se le mostrará al usuario tantas veces como lo requiera. Por último, se trata de un mensaje *feedforward* ya que explica las acciones que ha de seguir el usuario y no qué realiza o está realizando el sistema.

Con todas estas características ya establecidas, se puede proceder con la explicación. El título que mostrará el componente de ayuda ha de ser amigable y cercano, pero determinante y conciso, ya que se trata de un título. En nuestro caso se ha optado por: “Ey, parece que estás un poco perdido”. Este mensaje es algo informal y cercano, ya que se trata de una red social y se intenta ser amigable con el usuario, pero se asegura de que si necesita ayuda porque no encuentra lo que busca, se leerá la explicación. Esto evita que el usuario experto preste demasiada atención desde un principio al estímulo visual de la aparición del componente. Como cuerpo del mensaje se ha de dar una explicación clara de las acciones a seguir, pero no muy extensa porque se perdería la atención del usuario. Un ejemplo de explicación podría ser: “Creo que estás buscando [qué se ha inferido que busca el usuario]. Para llegar a ello debes ir a [pasos a seguir por el usuario separados por >].”. Este mensaje ayuda al usuario dándole los pasos a seguir en el orden correspondiente, pero asumiendo que el usuario va a encontrar los elementos que se le marcan. Esto deja aún un margen de error en la explicación, dado que se asume que los que van a hacer uso de la ayuda son usuarios no-expertos. Por lo tanto, se puede optar por otra opción que dé una descripción más detallada de las acciones a seguir del usuario incluyendo las indicaciones correspondientes para que el usuario encuentre los elementos necesarios, pero sin excederse con la longitud del mensaje para evitar que el usuario pierda la atención. Esto nos enfrenta a otro problema, que es la longitud del cuerpo del mensaje. Para que el componente no sea demasiado grande y no moleste al usuario experto, se ha optado por la inclusión de una elipsis del mensaje y que si se quiere leer entero se le dé clic al botón de “Leer más”, para que el componente muestre el mensaje completo. También, para que el usuario no pierda el hilo de la explicación, se mantiene abierto el componente hasta que se cierre entre pestañas solo si se le ha dado al botón de “Leer más”.

Ahora que el sistema está al completo, el modelo es capaz de inferir la emoción del usuario a partir de sus expresiones faciales y almacenar los datos. Cuando la aplicación solicita la información, a través del *end-point*, se le envían los datos. A partir de las variables especificadas, infiere qué está buscando el usuario. Si determina que el usuario lo requiere o necesita, el sistema proporciona el mensaje y la ayuda correspondientes para guiarlo hacia lo que busca. Usando el mensaje adecuado se mejora la experiencia del usuario no-experto, se le ayuda a adaptarse al entorno y se le facilita el acceso a las tecnologías.

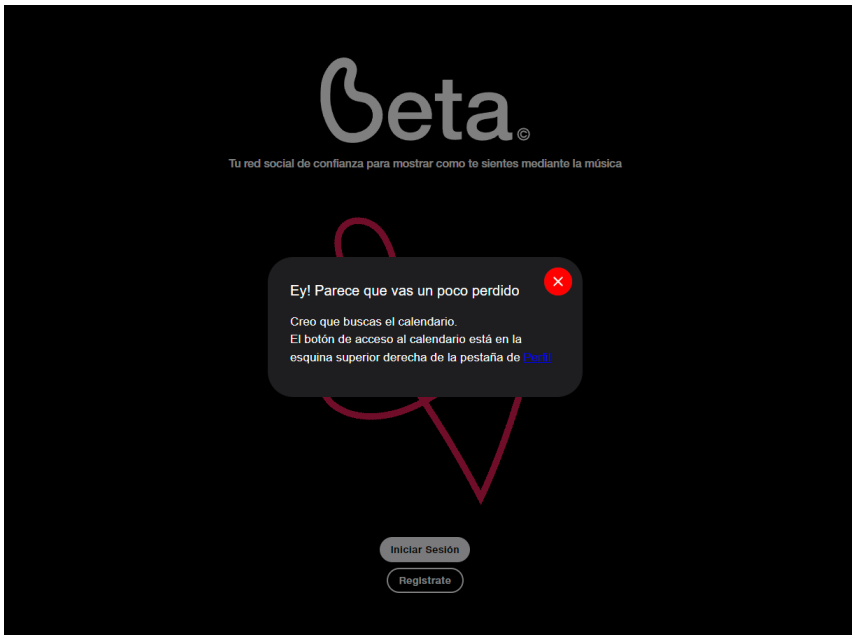

Fig. 8 Ejemplo del modal en la aplicación de prueba

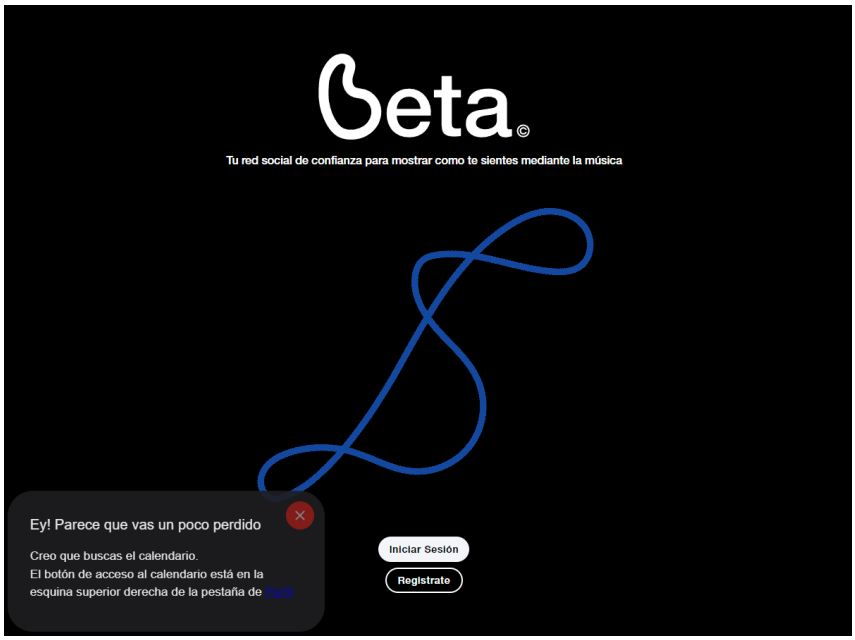

Fig. 9 Ejemplo del pop-up en la aplicación de prueba

7. Conclusiones

Llegando al final de la memoria, en este último capítulo se van a presentar y explicar las conclusiones extraídas respecto a los objetivos inicialmente marcados en la sección 1.2 de esta memoria.

En primer lugar, en el capítulo 5: *Tecnologías y puesta a punto*, se ha descrito y explicado la elección y el desarrollo del modelo de inferencia de emociones en base a expresiones faciales y cómo funciona. Además, se ha descrito la aplicación escogida para realizar las pruebas y las tecnologías usadas para ambas. También se ha descrito que conexiones se han planteado entre ambos componentes, así como que elementos se han diseñado para mostrar la ayuda.

Por otra parte, en los capítulos 4: *Diseño de la solución* y 6: *Desarrollo*, se ha explicado y desarrollado el proceso de creación de la herramienta. Se ha descrito y explicado los pasos a seguir para integrar el modelo a cualquier aplicación de manera clara y concisa, al igual que se ha descrito las modificaciones que han sufrido ambas partes para poder acoplarse entre ellas y funcionar correctamente. De la misma manera se han descrito las opciones y las decisiones tomadas para conseguir el mejor diseño para el mensaje y su contenedor y se han mostrado ejemplos. Y, además, se ha descrito cómo modificar una aplicación cliente permitiendo a esta hacer un uso correcto de los datos obtenidos de la inferencia de emociones en base a expresiones faciales.

Y finalmente, se ha modificado la aplicación y añadido los componentes necesarios para que, una vez obtenidos los datos provenientes del modelo de inferencia de emociones en base a expresiones faciales, se compruebe si el usuario necesita o requiere la ayuda y en caso afirmativo, se le muestre el mejor mensaje de ayuda posible de la mejor forma posible.

En conclusión, a lo largo de este proyecto se ha desarrollado un modelo de inferencia de emociones en base a expresiones faciales, se ha adaptado para su integración en otras aplicaciones y que estas aplicaciones hagan uso de la información que proporciona. En la aplicación de prueba se han realizado las modificaciones correspondientes para la lectura correcta de los datos provenientes del modelo de inferencia y se han añadido los componentes para mostrar la ayuda, al igual que se han determinado los límites para considerar mostrarla o no. Además, se ha explicado cada paso dado hasta conseguir el producto final y el porqué de estas decisiones. Finalmente, se ha experimentado con una herramienta de este tipo, abriendo la puerta a su explotación e investigación, y a que se puedan desarrollar herramientas de este tipo para todos los sistemas y aplicaciones y ampliar su accesibilidad y facilitar su uso.

7.1 Trabajos futuros

Con las ideas aquí planteadas se plantean varios frentes que no son excluyentes los unos de los otros. En primero lugar se podría añadir otro modelo que detecte el habla y las expresiones y sonidos no verbales que se producen en situaciones de frustración o enfado. De esta manera, juntando los datos obtenidos mediante la inferencia de la expresión facial y los datos obtenidos mediante la detección auditiva, la detección de la necesidad de ayuda sería más certera. Dado que a mayor cantidad de información que verifique los datos obtenidos, mejorarían las predicciones de las situaciones y el uso de la herramienta para mejorar la accesibilidad. Este modelo se había propuesto en un principio como un añadido al modelo de inferencia de emociones en base a expresiones faciales, pero finalmente no se ha podido llevar a cabo.

Por otra parte, usando el mismo modelo de detección auditiva, se podría medir la respiración del usuario. Como se explicó en el punto 3: Tecnologías y puesta a punto, las emociones se pueden determinar mediante señales física, como hemos realizado en este trabajo, o mediante señales fisiológicas. La respiración entra dentro de estas últimas y valorando su variación a lo largo del tiempo se pueden inferir las emociones que siente un usuario. La mayor dificultad a la hora de añadir este modelo sería encontrar datos para su entrenamiento. Estos tendrían que recogerse realizando un estudio, ya que todos los conjuntos de datos de este tipo son privados, dada la sensibilidad de estos. Superando este inconveniente, la adición de este modelo, al igual que el anterior, mejorarían mucho la efectividad de la herramienta y su precisión a la hora de discernir cuando necesita ayuda el usuario.

Otra adición, en este caso al modelo que hace uso de la cámara, sería la detección de la edad del usuario. Haciendo uso de esta información, se podría modificar dinámicamente el mensaje a mostrar para que fuese más o menos descriptivo y explicativo. Además, esta información también podría modificar la aplicación en tiempo de ejecución, aumentando el tamaño de las letras o cambiando los colores para los usuarios más mayores o con discapacidades visuales, facilitando su uso y navegabilidad.

Hasta el momento solo se ha hablado de esta herramienta en relación con su uso como herramienta de ayuda al usuario. Pero esta misma herramienta, incluyendo las adiciones anteriormente comentadas, también permite muchos otros usos no comentados hasta ahora. Por ejemplo, en el campo del marketing, conocer las emociones y el estado de ánimo de un cliente es vital para poder guiarles e indicarles hacia lo que se desea vender. Con una herramienta como esta, se facilita la primera parte, permitiendo, por ejemplo, cambiar qué producto se ofrece en función de la edad y de la emoción predominante, o que, si se han realizado varias búsquedas relacionadas, se te ofrezcan otros productos también relacionados para disminuir las interacciones que requieren de cambio entre el ratón y el teclado, así como facilitar el proceso de búsqueda y compra.

Otra aplicación posible de esta herramienta sería en el campo de la medicina. La adaptación de la herramienta para que dentro de las consultas se pueda analizar la respuesta del paciente ante la información que recibe, cambia cómo se atiende a un paciente y mejora la asistencia del médico a este mismo. Detectando como cambia la expresión del paciente o como cambian sus respuestas fisiológicas, el médico puede optar por otra manera de explicarse o preguntar qué es lo que no se entiende. Otra aplicación dentro del mundo de la medicina sería en

los campos de psiquiatría, junto a la psicología. Permitiría un análisis exhaustivo de las respuestas físicas y fisiológicas de un paciente, pudiendo diagnosticar trastornos y enfermedades mentales con mayor facilidad.

Para finalizar, teniendo en cuenta el uso que se le podría dar a esta herramienta, se podría cambiar su diseño para que su uso se pueda acoplar a tantas aplicaciones y sistemas como sea posible. Esto se podría conseguir convirtiendo el modelo, o conjunto de modelos, en un servicio externo o un SDK, permitiendo que su uso se globalice y adapte según las necesidades de cada aplicación.

Como se ha comentado a lo largo de este trabajo, esta herramienta pretende abrir la puerta a la explotación de este tipo de herramientas y con ello abrir un gran abanico de oportunidades en muchos sectores diferentes, desde la medicina, hasta el marketing.

Bibliografía

[1] European Commission. Joint Research Centre., *AI watch: defining Artificial Intelligence : towards an operational definition and taxonomy of artificial intelligence*. LU: Publications Office, 2020. Accedido: 7 de junio de 2024. [En línea]. Disponible en: https://data.europa.eu/doi/10.2760/382730

[2] «¿Qué es la inteligencia artificial? - Explicación de la inteligencia artificial (IA) - AWS», Amazon Web Services, Inc. Accedido: 8 de junio de 2024. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/artificial-intelligence/

[3] «¿Funcionalidad de la IA? Características, tipos y usos.» Accedido: 12 de junio de 2024. [En línea]. Disponible en: https://esdima.com/la-inteligencia-artificial-definicion-caracteristicas-tipos-usos/

[4] «Virtual assistant», *Wikipedia*. 17 de mayo de 2024. Accedido: 11 de junio de 2024. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Virtual_assistant&oldid=1224250705

[5] S. Feuerriegel, J. Hartmann, C. Janiesch, y P. Zschech, «Generative AI», *Bus. Inf. Syst. Eng.*, vol. 66, n.º 1, pp. 111-126, feb. 2024, doi: 10.1007/s12599-023-00834-7.

[6] Shadows, «Generative AI: Midjourney and DALL-E May Soon face Copyright Infrigement lawsuits», 3DVF. Accedido: 11 de junio de 2024. [En línea]. Disponible en: https://3dvf.com/en/generative-ai-midjourney-and-dall-e-facing-copyright-issues/

[7] C. Mauran, «“Adobe does not train Firefly Gen AI models on customer content”: Company responds to backlash», Mashable. Accedido: 12 de junio de 2024. [En línea]. Disponible en: https://mashable.com/article/adobe-does-not-train-firefly-gen-ai-customer-content-policy-backlash

[8] «Cómo evitar que Meta use tus fotos y datos de Instagram para entrenar su inteligencia artificial», ELMUNDO. Accedido: 12 de junio de 2024. [En línea]. Disponible en: https://www.elmundo.es/como/2024/05/29/665745dde9cf4a0e708b4595.html

[9] Reuters, «Meta detiene su proyecto para entrenar a la IA con publicaciones de Facebook e Instagram en Europa», El País. Accedido: 18 de junio de 2024. [En línea]. Disponible en: https://elpais.com/tecnologia/2024-06-14/meta-detiene-su-proyecto-para-entrenar-a-la-ia-con-publicaciones-de-facebook-e-instagram-en-europa.html

[10] RTVE.es/AGENCIAS, «La Unión Europea aprueba definitivamente la ley de inteligencia artificial», RTVE.es. Accedido: 12 de junio de 2024. [En línea]. Disponible en: https://www.rtve.es/noticias/20240521/union-europea-aprueba-definitivamente-ley-inteligencia-artificial/16112399.shtml

[11] «¿Qué es la Estrategia de Inteligencia Artificial 2024?» Accedido: 12 de junio de 2024. [En línea]. Disponible en: https://www.lamoncloa.gob.es/serviciosdeprensa/notasprensa/transformacion-digital-y-funcion-publica/Paginas/2024/ia-inteligencia-artificial-estrategia-espana.aspx

[12] luciaclemares, «¿Qué beneficios tiene la Inteligencia Artificial en la medicina?», Telefónica. Accedido: 12 de junio de 2024. [En línea]. Disponible en: https://www.telefonica.com/es/sala-comunicacion/blog/que-beneficios-tiene-la-inteligencia-artificial-en-la-medicina/

[13] C. SER, «Un hospital de València realiza con éxito la primera cirugía asistida con un robot en España», cadena SER. Accedido: 13 de junio de 2024. [En línea]. Disponible en: https://cadenaser.com/comunitat-valenciana/2024/02/18/un-hospital-de-valencia-realiza-con-exito-la-primera-cirugia-asistida-con-un-robot-en-espana-radio-valencia/

[14] R. D. Moreno Padilla, «La llegada de la inteligencia artificial a la educación», *Rev. Investig. En Tecnol. Inf.*, vol. 7, n.º 14, pp. 260-270, dic. 2019, doi: 10.36825/RITI.07.14.022.

[15] P. Cardon, C. Fleischmann, J. Aritz, M. Logemann, y J. Heidewald, «The Challenges and Opportunities of AI-Assisted Writing: Developing AI Literacy for the AI Age», *Bus. Prof. Commun. Q.*, vol. 86, n.º 3, pp. 257-295, sep. 2023, doi: 10.1177/23294906231176517.

[16] S. K. Khare, V. Blanes-Vidal, E. S. Nadimi, y U. R. Acharya, «Emotion recognition and artificial intelligence: A systematic review (2014–2023) and research recommendations», *Inf. Fusion*, vol. 102, p. 102019, feb. 2024, doi: 10.1016/j.inffus.2023.102019.

[17] «Challenges in Representation Learning: Facial Expression Recognition Challenge». Accedido: 15 de junio de 2024. [En línea]. Disponible en: https://kaggle.com/competitions/challenges-in-representation-learning-facial-expression-recognition-challenge

[18] «Cómo Reconocer Emociones con OpenCV y TensorFlow», DataSmarts Español. Accedido: 18 de junio de 2024. [En línea]. Disponible en: https://datasmarts.net/es/como-reconocer-emociones-con-tensorflow/

[19] A. Mestre, M. Gil, M. Albert, J. I. Panach Navarrete, y V. Pelechano, «Caracterización dinámica de explicaciones en sistemas autónomos con participación humana», sep. 2022, Accedido: 24 de junio de 2024. [En línea]. Disponible en: https://biblioteca.sistedes.es/entities/art%C3%ADculo/a2531d45-bdfa-47d6-b2c1-ac1a60480d87

Apéndices

A Archivos del código fuente

A.1 Archivo YAML para la creación del entorno

```yaml
name: tf-emotion-recognizer
channels:
  - defaults
  - conda-forge
dependencies:
  - python=3.8
  - opencv
  - imutils
  - tensorflow-gpu
```

A.2 Código fuente del *Modal*

```
export const ModalComponent = () => {

  const [ openModal, setOpenModal ] = useState<boolean>(false);

  const emotions = useAppSelector(getEmotions);
  const perfilEntries = useAppSelector(getEntries);

  useEffect(() => {
    if ((emotions[1] > 20 && emotions[2] > 20 && emotions[0] >= 30
      || emotions[1] > 50)
      && perfilEntries > 2
    ) {
      handleOpenModal();
    }
  }, [emotions]);

  const handleOpenModal = () => {
    setOpenModal(true);
  }

  const handleClose = () => {
    setOpenModal(false);
  };

  return (
    <Modal
      open={openModal}
      aria-labelledby='modal-modal-title'
      aria-describedby='modal-modal-description'
      sx={{ display: 'flex', alignItems: 'center', justifyContent:
'center' }}
    >
```

```
32.       <Box
33.         sx={{
34.           backgroundColor: '#1E1E20',
35.           p: 4,
36.           height: 200,
37.           width: 450,
38.           borderRadius: 9,
39.           color: 'white',
40.           position: 'relative'
41.         }}
42.       >
43.         <IconButton
44.           sx={{
45.             backgroundColor: 'red',
46.             color: 'white',
47.             '&:hover': {
48.               backgroundColor: 'lightgrey',
49.               color: 'black',
50.               borderColor: 'black',
51.             },
52.             position: 'absolute',
53.             top: '15px',
54.             right: '15px'
55.           }}
56.           onClick={handleClose}
57.         >
58.           <CloseRounded />
59.         </IconButton>
60.         <Typography
61.           className='modal-modal-title'
62.           variant='h6'
63.           component='h2'
64.         >
65.           Ey! Parece que vas un poco perdido
66.         </Typography>
67.
68.         <Typography
69.           className='modal-modal-description'
70.           sx={{ mt: 2, fontSize: 17 }}
71.         >
72.           Creo que buscas el calendario. <br />
73.           El botón de acceso al calendario está en la esquina
    superior derecha de la pestaña de <Link to='/perfil' style={{ color:
    'blue' }}>Perfil</Link>
74.         </Typography>
75.       </Box>
76.     </Modal>
77.   );
78. };
```

A.3 Código fuente del *Pop-up*

```
export const PopupComponent = () => {

  const [ openPopup, setOpenPopup ] = useState<boolean>(false);

  const emotions = useAppSelector(getEmotions);
  const perfilEntries = useAppSelector(getEntries);

  useEffect(() => {
    if ((emotions[1] > 20 && emotions[2] > 20 && emotions[0] >= 30
      || emotions[1] > 50)
      && perfilEntries > 2
    ) {
      handleOpenPopup();
    }
  }, [emotions]);

  const handleOpenPopup = () => {
    setOpenPopup(true);
  }

  const handleClose = () => {
    setOpenPopup(false);
  };

  return (
    <Popper open={openPopup}
      aria-labelledby='modal-modal-title'
      aria-describedby='modal-modal-description'
      sx={{
        position: 'fixed',
        bottom: '30px',  // Ajustar según sea necesario
        ml: '20px',    // Ajustar según sea necesario
        display: 'flex',
        alignItems: 'end',
        justifyContent: 'end',
      }}
    >
      <Box
        sx={{
          opacity: '90%',
          backgroundColor: '#1E1E20',
          p: 4,
          height: 200,
          width: 450,
          borderRadius: 9,
          color: 'white',
          position: 'relative'
        }}
      >
      <IconButton
        sx={{
          opacity: '50%',
          backgroundColor: 'red',
          color: 'white',
          '&:hover': {
            backgroundColor: 'lightgrey',
            color: 'black',
            borderColor: 'black',
          },
```

```
60.           position: 'absolute',
61.           top: '15px',
62.           right: '15px'
63.         }}
64.         onClick={handleClose}
65.       >
66.         <CloseRounded />
67.       </IconButton>
68.       <Typography
69.         className='modal-modal-title'
70.         variant='h6'
71.         component='h2'
72.       >
73.         Ey! Parece que vas un poco perdido
74.       </Typography>
75.
76.       <Typography
77.         className='modal-modal-description'
78.         sx={{ mt: 2, fontSize: 17 }}
79.       >
80.         Creo que buscas el calendario. <br />
81.         El botón de acceso al calendario está en la esquina
    superior derecha de la pestaña de <Link to='/perfil' style={{
    color: 'blue' }}>Perfil</Link>
82.       </Typography>
83.       </Box>
84.     </Popper>
85.   );
```

A.4 Archivo JSON donde se guardan las predicciones del modelo

```
1. {
2.  "detection":
3.       [107, 21, 195, 195],
4.  "emotions": {
5.       "enfadado": 0.04089689254760742,
6.       "asustado": 0.062379758805036545,
7.       "feliz": 0.004785735160112381,
8.       "triste": 0.48671674728393555,
9.       "sorprendido": 0.0031562712974846363,
10.      "neutro": 0.4020645320415497
11.  }
12.}
```

A.5 Código fuente del *end-point*

```python
from flask import Flask, jsonify
from flask_cors import CORS
import json

app = Flask(__name__)
CORS(app)

@app.route('/emotions', methods=['GET'])
def emotions_route():
  with open('../detected_emotions.json') as f:
    detected_emotions = json.load(f)
  return jsonify(detected_emotions)

if __name__ == '__main__':
  print("Starting Flask server...")
  app.run(host='0.0.0.0', port=5000, debug=True)
```

A.6 Código fuente del *fetch* para las emociones

```javascript
import { useEffect } from "react";
import { setAnger, setNeutral, setSadness, useAppDispatch } from
"../../store";
const url = "http://localhost:5000/emotions";

export const useFetch = () => {

  const dispatch = useAppDispatch();

  useEffect(() => {
    const interval = setInterval(() => getFetch(), 2000);

    return () => {
      clearInterval(interval);
    }
  }, [])

  const getFetch = async() => {
    try{
      const resp = await fetch(url,
          { method: 'GET',
            cache: 'default'
        })

      const detection = await resp.json();

      dispatch(setNeutral(detection.emotions.neutro));
      dispatch(setAnger(detection.emotions.enfadado));
      dispatch(setSadness(detection.emotions.triste));

    } catch (e: any) {
      throw new Error(e.message)
    }
  }
}
```

A.7 Código fuente del *store* para las emociones

```
1. import { createSlice } from '@reduxjs/toolkit';
2. import { RootState } from '../store';
3.
4. export const emotionsSlice = createSlice({
5.     name: 'emotions',
6.     initialState: {
7.       neutral: 0,
8.       anger: 0,
9.       sadness: 0
10.    },
11.    reducers: {
12.        setNeutral: (state, action ) => {
13.            state.neutral = action.payload * 100;
14.        },
15.        setAnger: (state, action ) => {
16.          state.anger = action.payload * 100;
17.        },
18.        setSadness: (state, action ) => {
19.          state.sadness = action.payload * 100;
20.        },
21.    }
22. });
23.
24. // Action creators are generated for each case reducer function
25. export const { setNeutral, setAnger, setSadness } =
    emotionsSlice.actions;
26.
27. export const getEmotions = (state: RootState) =>
    [state.emotions.neutral, state.emotions.anger,
    state.emotions.sadness];
28.
29. export default emotionsSlice.reducer; }));
```

B Objetivos de Desarrollo Sostenible (ODS)

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**			X	
ODS 2. **Hambre cero.**			X	
ODS 3. **Salud y bienestar.**			X	
ODS 4. **Educación de calidad.**	X			
ODS 5. **Igualdad de género.**		X		
ODS 6. **Agua limpia y saneamiento.**			X	
ODS 7. **Energía asequible y no contaminante.**			X	
ODS 8. **Trabajo decente y crecimiento económico.**			X	
ODS 9. **Industria, innovación e infraestructuras.**		X		
ODS 10. **Reducción de las desigualdades**.	X			
ODS 11. **Ciudades y comunidades sostenibles**.			X	
ODS 12. **Producción y consumo responsables.**			X	
ODS 13. **Acción por el clima.**			X	
ODS 14. **Vida submarina.**			X	
ODS 15. **Vida de ecosistemas terrestres.**			X	
ODS 16. **Paz, justicia e instituciones sólidas.**			X	
ODS 17. **Alianzas para lograr objetivos.**			X	

Reflexión sobre la relación del TFG con los ODS y con los ODS más relacionados.

En este momento se asume que el lector tiene los suficientes conocimientos sobre el sistema desarrollado en este TFG, sus posibles aplicaciones y su uso y utilidad desarrolladas en esta memoria. El sistema propuesto tiene como objetivo acercar la tecnología a todo aquel que lo requiera o lo necesite independientemente de su situación o contexto, por lo tanto, aporta ventajas y herramientas a campos como la educación o la banca. Amplía la accesibilidad a cualquier sistema o aplicación y abre un nuevo camino en la investigación y el desarrollo de este tipo de herramientas. Por estos motivos, los ODS más relacionados con el trabajo son:

1. **Educación de calidad.** Como se ha comentado en varios puntos del trabajo, este sistema acerca la tecnología aquellos que menos conocimientos tiene sobre esta. Con la aplicación correcta de este sistema en el sistema educativo, permitiría una educación temprana en la tecnología y su uso evitando una gran frustración por parte de los más pequeños. Además, como se ha comentado en el párrafo anterior, el sistema desarrollado en este trabajo abre la puerta a que se investigue y se desarrolle más herramientas de este tipo o se adapte a diferentes ámbitos.

2. **Igualdad de género.** Aún, a día de hoy, existe una brecha entre mujeres y hombres en los campos de la ciencia y la tecnología (STEM o CTIM en español). Tanto en cantidad de personas trabajando como en el número de reconocimientos recibidos o "papers" publicados. La brecha es cada vez menor y se aplaca con más firmeza y recursos, pero sigue existiendo. Como se ha comentado en el punto anterior, con una correcta adaptación del sistema, se podría aplicar al sistema educativo. Con recursos como este, se abre el abanico de oportunidades a todos los jóvenes, incluyendo las mujeres. Con una exposición temprana a todas las opciones se brinda a las niñas y jóvenes una mayor libertad de elección. Al experimentar con las disciplinas de STEM pueden explorar y descubrir cuáles son sus verdaderas vocaciones e intereses. Este sistema no solo les permitiría adquirir una base sólida, sino que también les ayuda a ganar la confianza necesaria para tomar libremente la decisión que ellas consideren. Al experimentar con las diferentes disciplinas de STEM, las niñas y jóvenes pueden encontrar y desarrollar sus pasiones, así como fomentar su desarrollo personal y profesional a largo plazo.

3. **Industria, innovación e infraestructuras.** El sistema desarrollado en este trabajo es fácilmente adaptable a cualquier disciplina de la industria para optimizar interfaces, así como modificarlas para que sean más amigables ofreciendo la misma información. Al igual, que se puede usar para entrenar a los novatos y ayudar en casos de emergencia a seguir el curso de acción acertado. Por otra parte, a lo largo del trabajo se ha comentado en varias ocasiones que esta herramienta abre la puerta a que se investiguen y se desarrollen más herramientas de este tipo. Este tipo de innovaciones son necesarias para el acceso universal a la tecnología y hoy en día son aún más necesarias con la digitalización de tantos trámites y acciones que antes eran presenciales y personalizadas.

4. **Reducción de las desigualdades.** Como se ha comentado en el punto anterior, la digitalización de trámites y acciones que antes eran en oficinas y sucursales es cada

vez más común. Estas acciones tienen como objetivo reducir costes y tiempos de espera, pero marginan a un gran colectivo como es las personas de la tercera edad. Con el uso de esta herramienta se acerca a las personas de este colectivo a la tecnología y su uso y poco a poco les elimina el miedo o ansiedad que pueda generar enfrentarse por primera vez a estas situaciones. El desarrollo y adaptación de esta herramienta también mejora la accesibilidad para todas las personas en el ámbito laboral. Al mejorar la accesibilidad, se abren oportunidades para una mayor diversidad, permitiendo acceder a nuevos trabajos a personas que anteriormente habrían estado fuera de su alcance. Esto incluye a personas con diferentes habilidades y antecedentes, que se pueden beneficiar de un entorno laboral más inclusivo. Con estas herramientas y adaptaciones se fomenta la inclusión en el mercado laboral de todo tipo de personas, independientemente de su situación o contexto, asegurando que tengan las mismas oportunidades para prosperar en una carrera laboral.