



TRABAJO FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN COMPUTACIÓN

ComunicELA: Software de Asistencia para la Comunicación en Pacientes con Esclerosis Lateral Amiotrófica

Estudiante: Pablo Páramo Telle
Dirección: José Joaquim de Moura Ramos
Luís María Hervella Nieto
Marcos Ortega Hortas

A Coruña, septiembre de 2024.

Este proyecto ha sido realizado con el fin de facilitar la comunicación de todas las personas que padecen [ELA](#) y dedicado a todas ellas por su esfuerzo y afán de superación.

Agradecimientos

Agradecer a la Cátedra NTT DATA en Diversidad y Tecnología el contacto con AGAELA y la propuesta del problema real que hemos intentado solucionar. También extender el agradecimiento a todas las personas que han participado en la recopilación de datos de manera voluntaria.

Resumen

La Esclerosis Lateral Amiotrófica (ELA) es una enfermedad neurodegenerativa irreversible que resulta en la pérdida gradual de neuronas, deteriorando progresivamente las funciones motrices, así como la capacidad de hablar, tragar o respirar. Las personas con ELA que carecen de capacidad de comunicarse experimentan una mejora significativa en su calidad de vida cuando utilizan sistemas de comunicación aumentativa y alternativa (CAA). Sin embargo, muchos de estos sistemas no están específicamente adaptados a sus necesidades únicas, lo que limita su efectividad.

Este proyecto, desarrollado en colaboración con la Asociación Galega de Afectados pola Esclerose Lateral Amiotrófica (AGAELA) y la Cátedra NTT DATA en Diversidad y Tecnología, propone un sistema basado en inteligencia artificial que mejora significativamente la calidad de comunicación de las personas con esta enfermedad. Gracias a la colaboración con AGAELA, se ha adaptado el diseño y las funcionalidades del sistema a las necesidades reales de los usuarios, asegurando su aplicabilidad y relevancia. La solución propuesta es una aplicación con una interfaz intuitiva, accesible, adaptable y sencilla, que facilita la comunicación mediante tableros personalizables controlados por movimientos oculares y parpadeo. Las frases formadas pueden ser reproducidas a través de síntesis de voz, mejorando la interacción y la autonomía de los pacientes.

El sistema fue validado tanto con un grupo control como con pacientes diagnosticados con ELA en diversas etapas de la enfermedad, cubriendo un amplio espectro de variaciones en sus habilidades de movilidad y comunicación. Este riguroso proceso de validación incluyó pruebas de usabilidad y efectividad, asegurando que el sistema funcione de manera efectiva bajo una variedad de condiciones y necesidades específicas.

Este enfoque integrador y colaborativo no solo mejora la comunicación sino que también amplía la interacción social y la calidad de vida de los afectados por la ELA. Diseñado para adaptarse a futuras innovaciones en tecnologías asistenciales, el sistema facilita la integración con otros dispositivos y plataformas, ofreciendo soluciones más personalizadas. Así, el proyecto no solo atiende las necesidades actuales sino que también evoluciona y se adapta a los avances tecnológicos y médicos, contribuyendo continuamente a la mejora de la autonomía y el bienestar de los usuarios.

Además, al estar disponible como código libre, el sistema invita a desarrolladores y usuarios de todo el mundo a contribuir, promoviendo un entorno abierto y colaborativo que enriquece continuamente su desarrollo y aplicación.

Abstract

Amyotrophic Lateral Sclerosis (ALS) is an irreversible neurodegenerative disease that results in the gradual loss of neurons, progressively impairing motor functions, as well as the ability to speak, swallow, or breathe. People with ALS who lack the ability to communicate experience a significant improvement in their quality of life when they use augmentative and alternative communication (AAC) systems. However, many of these systems are not specifically tailored to their unique needs, which limits their effectiveness.

This project, developed in collaboration with the Galician Association of People Affected by Amyotrophic Lateral Sclerosis (AGAELA) and the NTT DATA Chair in Diversity and Technology, proposes a system based on artificial intelligence that significantly improves the quality of communication for people with this disease. Thanks to the collaboration with AGAELA, the design and functionalities of the system have been adapted to the real needs of the users, ensuring its applicability and relevance. The proposed solution is an application with an intuitive, accessible, adaptable, and simple interface, which facilitates communication through customizable boards controlled by eye movements and blinking. The phrases formed can be reproduced through voice synthesis, improving interaction and patient autonomy.

The system was validated with both a control group and patients diagnosed with ALS at various stages of the disease, covering a wide spectrum of variations in their mobility and communication abilities. This rigorous validation process included usability and effectiveness tests, ensuring that the system functions effectively under a variety of conditions and specific needs.

This integrative and collaborative approach not only improves communication but also expands social interaction and the quality of life of those affected by ALS. Designed to adapt to future innovations in assistive technologies, the system facilitates integration with other devices and platforms, offering more personalized solutions. Thus, the project not only meets current needs but also evolves and adapts to technological and medical advances, continuously contributing to the improvement of autonomy and well-being of users.

Additionally, being available as open source, the system invites developers and users from around the world to contribute, promoting an open and collaborative environment that continually enriches its development and application.

Palabras clave principales:

- Esclerosis Lateral Amiotrófica
- Seguimiento ocular
- Aprendizaje automático
- Comunicación
- Interacción usuario-computadora
- Interfaces adaptativas
- Tecnologías asistivas

Key words:

- Amyotrophic Lateral Sclerosis
- Eye tracking
- Machine learning
- Communication
- Human-computer interaction
- Adaptive interfaces
- Assistive technologies

Índice general

1	Introducción	1
1.1	Contextualización	1
1.2	Motivación	2
1.3	Objetivos	3
1.4	Estructura	3
2	Esclerosis Lateral Amiotrófica	6
2.1	Revisión histórica: Evolución del conocimiento sobre la ELA	6
2.2	Diagnóstico: Procedimientos y Criterios	8
2.3	Opciones de Tratamiento: Enfoques actuales para la gestión de la ELA	8
3	Estado del arte	11
3.1	Tobii	11
3.2	Tallk	11
3.3	Comunicador dinámico de AsTerics Grid	12
3.4	Asistente de Voz AAC	12
3.5	Trabajo propuesto	13
4	Metodología	15
4.1	Metodología Scrum	15
4.2	Fases principales	16
4.3	Aplicación en el proyecto	16
4.3.1	Roles	16
4.3.2	Sprints	17
5	Planificación	19
5.1	Recursos	19
5.1.1	Recursos humanos	19

ÍNDICE GENERAL

5.1.2	Recursos materiales	20
5.2	Tareas	21
5.3	Seguimiento y costes	22
6	Análisis de requisitos	23
6.1	Actores	23
6.2	Análisis de los requisitos:	23
6.2.1	Requisitos funcionales:	24
6.2.2	Requisitos no funcionales:	25
6.3	Casos de uso	25
7	Diseño	30
7.1	Diseño de la aplicación	30
7.2	Diseños en Kivy	31
8	Implementación	33
8.1	Implementación del lector ocular	33
8.1.1	Criterios y restricciones de la recopilación de datos	33
8.1.2	Análisis de características	34
8.1.3	Base de datos inicial	38
8.1.4	Base de datos secundaria	39
8.1.5	Pre-procesado de datos	39
8.1.6	Métodos y métricas de error	40
8.1.7	Aproximación inicial	41
8.1.8	Mejoras del método	44
8.1.9	Primera aproximación: Perceptrón multicapa	49
8.1.10	Segunda aproximación: Redes neuronales convolucionales	53
8.1.11	Tercera aproximación: Redes neuronales híbridas	57
8.1.12	Entrenamiento final	59
8.2	Implementación de la aplicación	60
8.2.1	Base de la aplicación	60
8.2.2	Tutorial de la aplicación	61
8.2.3	Configuraciones de la pantalla principal	61
8.2.4	Calibración del parpadeo	65
8.2.5	Reentrenamiento del modelo	67
8.2.6	Tableros comunicativos	69
8.2.7	Pantallas de desarrollador	71
8.2.8	Recursos	73

ÍNDICE GENERAL

9 Pruebas de usabilidad del software	74
9.1 Participantes y materiales	74
9.2 Desarrollo de las pruebas	74
9.2.1 Preparación de los participantes	75
9.2.2 Pruebas	75
9.2.3 Recolección de datos	76
9.3 Evaluación y análisis	76
10 Conclusiones y Trabajos futuros	78
10.1 Conclusiones	78
10.2 Trabajo futuro	79
A Fundamentos tecnológicos	81
A.1 Visual Studio Code	81
A.2 Google Colab	81
A.3 GitHub	81
A.4 Python	82
A.5 Mediapipe Solutions	82
A.6 OpenCV	82
A.7 Pillow	82
A.8 Optuna	82
A.9 PyTorch	83
A.10 CUDA	83
A.11 cuDNN	83
A.12 Scikit-learn	83
A.13 Kivy	83
A.14 Canva	84
A.15 ARASAAC	84
A.16 Gemini API	84
A.17 ISpVoice	84
A.18 Balsamiq Wireframes	84
A.19 Creately	85
A.20 Overleaf	85
B Seguimiento del proyecto	86
B.1 Planificación inicial	86
B.2 Costes	87
B.2.1 Recursos materiales	87

ÍNDICE GENERAL

B.2.2 Recursos humanos	88
C Casos de uso del software	91
C.1 Casos de uso de la pre-configuración	91
C.2 Casos de uso del usuario común	95
C.3 Casos del usuario desarrollador	104
D Diseño de la aplicación	109
D.1 Bocetos de las pantallas de la aplicación	109
D.1.1 Bocetos de las pantallas de los usuarios comunes	109
D.1.2 Bocetos de las pantallas de desarrollador	114
D.2 Pantallas finales de la aplicación	117
D.2.1 Pantallas destinadas a los usuarios comunes	117
D.2.2 Pantallas destinadas a los desarrolladores	122
E Modelos de la segunda y tercera aproximación	125
E.1 Modelos de la segunda aproximación	125
E.1.1 Modelo 1	125
E.1.2 Modelo 2	126
E.1.3 Modelo 3	126
E.1.4 Modelo 4	127
E.1.5 Modelo 5	127
E.1.6 Modelo 6	128
E.1.7 Modelo 7	128
E.1.8 Modelo 8	128
E.2 Modelos de la tercera aproximación	129
E.2.1 Modelo híbrido 1	129
E.2.2 Modelo híbrido 2	130
E.2.3 Modelo híbrido 3	130
E.2.4 Modelo híbrido 4	130
E.2.5 Modelo híbrido 5	131
E.2.6 Modelo híbrido 6	131
E.2.7 Modelo híbrido 7	131
E.2.8 Modelo híbrido 8	132
E.2.9 Modelo híbrido 9	132
E.2.10 Modelo híbrido 10	133
Lista de acrónimos	134

ÍNDICE GENERAL

Glosario	135
Bibliografía	136

Índice de figuras

3.1	Uno de los modelos de lectores oculares de Tobii.	12
3.2	Captura de la aplicación web de AsTerics Grid.	13
6.1	Casos de uso de la pre-configuración.	27
6.2	Casos de uso del usuario común.	28
6.3	Casos de uso del usuario desarrollador.	29
7.1	Ejemplo de un botón creado con Kivy.	31
7.2	Ejemplo de un PopUp creado con Kivy.	31
7.3	Casilla con pictograma.	32
7.4	Casilla con texto.	32
8.1	(a) Distancias desde los puntos de referencia hasta la pupila. (b) Representación de los puntos de referencia detectados por Mediapipe.	34
8.2	Gráfico de las distancias de los ojos mirando a los extremos de la pantalla. . .	36
8.3	EAR durante un parpadeo.	36
8.4	Puntos 3D formando el modelo de la cara genérico en perspectiva lateral. . .	38
8.5	Puntos seleccionados de Mediapipe.	38
8.6	Suavizado de las distancias del indice 0 con sigma 5.	40
8.7	Gráfica del entrenamiento del modelo 3 para la beta del software.	45
8.8	Mapa de calor de la distancia euclidiana.	45
8.9	Función de ponderación de una dimensión.	47
8.10	Función de ponderación en el espacio de coordenadas.	48
8.11	Rangos de los valores de la BD originales y normalizados.	52
8.12	Resultados del método de recorte automático.	54
8.13	Gráfica del entrenamiento del modelo finalmente utilizado en el software. . .	60
8.14	Primer mensaje del tutorial.	61
8.15	Botón de cambio de idioma.	62

ÍNDICE DE FIGURAS

8.16 Menú flotante de configuración.	62
8.17 Selector de cámaras disponibles.	63
8.18 Imágenes representativas mostradas en la calibración del parpadeo.	65
8.19 Usuario colocado en una posición incorrecta.	66
8.20 Usuario colocado en la posición correcta.	66
8.21 Gráfico de un reentrenamiento comparado con el error general de la BD.	68
8.22 Hoja "TAB. INICIAL" de "tablero_es_ES.xlsx".	70
8.23 Representación visual del puntero: (a) en su estado inicial y (b) a un paso de bloquear una casilla.	71
 B.1 Diagrama de Gantt de la planificación inicial de "ComunicELA".	87
 D.1 Boceto de la pantalla principal.	110
D.2 Boceto del menú de configuración.	110
D.3 Boceto de la pantalla de calibración.	111
D.4 Boceto de la pantalla de reentrenar.	112
D.5 Boceto de la obtención de datos.	112
D.6 Boceto de la pantalla de los tableros comunicativos.	113
D.7 Boceto de los tableros comunicativos con pictogramas.	113
D.8 Boceto de los tableros comunicativos sin pictogramas.	114
D.9 Boceto de la pantalla principal con las opciones de desarrollador activadas.	115
D.10 Boceto de la pantalla de test.	115
D.11 Boceto de la pantalla de de recopilar.	116
D.12 Boceto de la pantalla de pruebas.	116
D.13 Pantalla principal.	117
D.14 Menú flotante de configuración.	118
D.15 Pantalla de calibración.	118
D.16 Pantalla de reentrenar.	119
D.17 Proceso de obtención de datos.	120
D.18 Pantalla de los tableros comunicativos.	120
D.19 Tableros comunicativos con pictogramas.	121
D.20 Tableros comunicativos sin pictogramas.	121
D.21 Pantalla principal con las opciones de desarrollador activadas.	122
D.22 Pantalla de test.	123
D.23 Pantallade de recopilar.	123
D.24 Pantalla de pruebas.	124

Índice de tablas

3.1	Comparación de las diferentes alternativas.	13
8.1	Correspondencia entre los puntos de Mediapipe y los P_i de la fórmula.	37
8.2	Modelos de RNA entrenados en la aproximación inicial.	42
8.3	Hiperparámetros utilizados en la búsqueda en cuadrícula para el modelo SVM. .	43
8.4	Mejores parámetros de la búsqueda en cuadrícula.	43
8.5	Resultados con los parámetros resultantes de la búsqueda.	43
8.6	Relaciones entre las esquinas y los límites bajos y altos en los ejes X e Y. . . .	48
8.7	Modelos de RNA de la aproximación inicial entrenados con la BD final. . . .	49
8.8	Modelos de topología similar al Modelo 3 entrenados con la BD final.	50
8.9	Resultados del Modelo 9 con el pre-procesado A.	51
8.10	Resultados del Modelo 9 con el pre-procesado B.	51
8.11	Resultados del Modelo 9 con el pre-procesado C.	52
8.12	Configuraciones del recorte automático utilizadas.	54
8.13	Modelos CNN evaluados en las primeras observaciones.	55
8.14	Modelos CNN evaluados en las segundas observaciones.	55
8.15	Modelos CNN evaluados en las terceras observaciones.	56
8.16	Modelos ResNet evaluados con las imágenes de la configuración 1.	56
8.17	Modelos híbridos evaluados en las primeras observaciones.	58
8.18	Modelos híbridos evaluados en las segundas observaciones.	58
9.1	Comparación de las diferentes pruebas realizadas.	77
B.1	Duración estimada.	86
B.2	Duración real.	86
B.3	Salarios de los recursos humanos.	89
B.4	Horas de trabajo de los recursos humanos del proyecto.	89
B.5	Costes de los recursos humanos del proyecto.	89

ÍNDICE DE TABLAS

B.6 Costes totales finales del proyecto.	90
--	----

Capítulo 1

Introducción

En este capítulo se presenta la motivación que dio origen al proyecto, los objetivos que se persiguen y la estructura general del documento. Además, se proporciona un contexto inicial que justifica la relevancia de la investigación y el desarrollo propuesto.

1.1 Contextualización

La Esclerosis Lateral Amiotrófica ([ELA](#)) es una enfermedad neurodegenerativa grave que afecta el sistema nervioso central, resultando en la degeneración progresiva de las neuronas motoras, lo que conduce a la debilidad muscular y, eventualmente, a la parálisis. A pesar de décadas de investigación, no existe una cura para la ELA; los tratamientos actuales solo pueden ralentizar el progreso de la enfermedad y ayudar a controlar los síntomas.

El impacto de la ELA sobre la autonomía de las personas afectadas es devastador, ya que progresivamente pierden la capacidad de moverse, respirar y comunicarse de manera efectiva. Las complicaciones más graves incluyen problemas respiratorios, dificultades para tragar y la pérdida de la capacidad para hablar, lo que genera una enorme carga emocional y psicológica tanto para los pacientes como para sus familias [1]. Además, la rápida progresión de la enfermedad y la limitada expectativa de vida agravan estos desafíos, generando una urgente necesidad de soluciones que ayuden a los pacientes a mantener su dignidad y calidad de vida durante las etapas avanzadas de la enfermedad.

En Europa, la [ELA](#) afecta entre 2 y 3 personas por cada 100.000 habitantes, con un riesgo estimado de 1 en 400 de desarrollar la enfermedad en algún momento de la vida. Generalmente, la ELA se presenta entre los 50 y los 75 años, afectando de manera desproporcionada a la población adulta [2]. A medida que envejece la población, es probable que estos números aumenten, incrementando la demanda de servicios de salud especializados y tecnologías asistenciales.

Existen dos formas principales de ELA, que se diferencian según los síntomas iniciales [3]:

- **ELA Bulbar:** Afecta principalmente las neuronas motoras localizadas en el tronco encefálico, lo que provoca dificultades para hablar y/o tragar en las primeras etapas. Este tipo representa aproximadamente el 25% de los casos de ELA.
- **ELA Medular o Espinal:** Se caracteriza por la debilidad muscular y la pérdida de fuerza en las extremidades, y abarca entre el 65% y el 70% de los casos.

La [ELA](#) también se clasifica según la presencia de antecedentes familiares [3]:

- **ELA familiar (ELAF):** Se diagnostica cuando hay al menos dos familiares de primer o segundo grado con la enfermedad, lo que ocurre en alrededor del 10% de los casos.
- **ELA esporádica (ELAS):** Es la forma más común, representando cerca del 90% de los casos, y aparece sin antecedentes familiares conocidos ni causas claras identificadas.

Dado el grave impacto de la ELA en la vida de los pacientes y la creciente prevalencia de la enfermedad, se hace imperativo desarrollar soluciones tecnológicas que ofrezcan un apoyo eficaz, especialmente en áreas como la comunicación, que es vital para mantener la autonomía y la calidad de vida. Para más detalles sobre la [ELA](#), véase el Capítulo 2.

1.2 Motivación

La capacidad de comunicarse es uno de los aspectos más fundamentales de la interacción humana, y su pérdida supone un impacto devastador en la calidad de vida de los pacientes con [ELA](#). A medida que la enfermedad avanza, los pacientes experimentan un deterioro gradual en sus habilidades motoras, lo que les dificulta realizar tareas básicas, incluidas aquellas relacionadas con la comunicación oral. Esta situación no solo afecta su autonomía, sino que también incrementa la sensación de aislamiento y dependencia.

Dado que las capacidades cognitivas y oculares de los pacientes con ELA generalmente permanecen intactas, surge una oportunidad valiosa para aprovechar estas funciones preservadas y proporcionarles una vía alternativa de comunicación. Este proyecto responde a esa necesidad crítica, orientándose a desarrollar un software que facilita la interacción mediante movimientos oculares. De esta manera, incluso aquellos pacientes en las etapas más avanzadas de la enfermedad pueden mantener una forma de comunicación eficaz y digna.

El sistema propuesto consiste en tableros de comunicación personalizables que permiten a los usuarios seleccionar palabras o frases con los ojos, las cuales son reproducidas mediante síntesis de voz. Este enfoque no solo mejora la capacidad de expresarse, sino que también promueve la autonomía y fortalece los vínculos con su entorno familiar y social, factores fundamentales para el bienestar emocional y psicológico.

Además, el desarrollo de esta herramienta está en línea con los avances recientes en el campo de las tecnologías asistivas, que buscan ofrecer soluciones accesibles y adaptadas a las necesidades particulares de cada paciente. En un contexto donde los sistemas de salud y asistencia se ven cada vez más presionados, soluciones tecnológicas como esta no solo alivian las barreras de comunicación, sino que también contribuyen a mejorar la calidad de vida y reducir el aislamiento de las personas con ELA [4][5].

1.3 Objetivos

El objetivo principal de este proyecto es desarrollar “ComunicELA”, un software eficiente y de fácil manejo para pacientes con ELA. Se busca facilitar una comunicación efectiva y mejorar la calidad de vida de los pacientes, permitiendo una interacción más fluida con su entorno. Los objetivos secundarios incluyen:

- **Evaluación del panorama actual:** Investigar las herramientas de comunicación existentes para identificar oportunidades de innovación.
- **Identificación de necesidades comunicativas:** Comprender los desafíos que enfrentan los pacientes con ELA, en colaboración con la [Asociación Galega de Afectados pola Esclerosa Lateral Amiotrófica \(AGAELA\)](#) y la Cátedra NTT DATA en Diversidad y Tecnología, asegurando que “ComunicELA” sea una solución práctica y adecuada.
- **Desarrollo de una interfaz intuitiva:** Crear una interfaz que sea accesible y fácil de usar para los pacientes con ELA, teniendo en cuenta sus limitaciones específicas.
- **Integración de tecnologías avanzadas:** Incorporar seguimiento ocular y adaptabilidad para diferentes dispositivos de asistencia.

1.4 Estructura

El documento está organizado en los siguientes capítulos, cada uno de los cuales aborda un aspecto fundamental del desarrollo y análisis del proyecto. A continuación, se ofrece un resumen de su contenido:

- **Capítulo 2 - Esclerosis Lateral Amiotrófica:** Se ofrece una visión en profundidad sobre la ELA, detallando sus síntomas, los métodos de diagnóstico y los tratamientos disponibles. Este capítulo también proporciona un contexto médico y científico que justifica la relevancia del proyecto, resaltando la importancia de desarrollar tecnologías que mejoren la calidad de vida de los pacientes.

- **Capítulo 3 - Estado del arte:** En este capítulo se recopilan y analizan las herramientas y soluciones tecnológicas existentes que tienen propósitos similares al proyecto. Se realiza una revisión crítica de las tecnologías actuales de comunicación asistiva para personas con ELA, identificando tanto sus ventajas como sus limitaciones, lo que sirve de base para proponer las mejoras en el proyecto.
- **Capítulo 4 - Metodología:** Este capítulo describe la metodología de desarrollo empleada, detallando las fases del proceso y los enfoques adoptados para asegurar una progresión eficiente y estructurada del proyecto. En particular, se explica cómo se ha aplicado la metodología ágil para garantizar la flexibilidad y adaptación del proyecto a medida que avanza su desarrollo.
- **Capítulo 5 - Planificación:** En este capítulo se expone el plan de trabajo realizado, detallando las distintas etapas del desarrollo, los plazos estimados y la distribución de los recursos. Además, se presentan estimaciones de costes, tanto materiales como humanos, proporcionando una visión completa de la gestión del proyecto desde su concepción hasta su finalización.
- **Capítulo 6 - Análisis de requisitos:** En este capítulo se describen los actores que interactúan con la aplicación, incluyendo tanto a los usuarios finales como a los desarrolladores. Se especifican los requisitos funcionales y no funcionales que el sistema debe cumplir para satisfacer las necesidades identificadas. Además, se incluyen los casos de uso que ilustran las interacciones clave entre los actores y el sistema.
- **Capítulo 7 - Diseño:** Este capítulo está dedicado a detallar el diseño de la aplicación. Se explican las decisiones de diseño tomadas en función de la experiencia de usuario, la accesibilidad y la interfaz gráfica, así como la arquitectura interna del sistema. El diseño busca asegurar que la aplicación sea intuitiva y adaptable a las necesidades de los usuarios.
- **Capítulo 8 - Implementación:** Se describe el proceso de desarrollo del software y del lector ocular, detallando cómo se han implementado las diferentes funcionalidades del sistema. Este capítulo incluye tanto la descripción del código como las técnicas empleadas para integrar los distintos módulos y asegurar el correcto funcionamiento del sistema.
- **Capítulo 9 - Pruebas de usabilidad del software:** En este capítulo se detallan las pruebas llevadas a cabo para validar el software, incluyendo pruebas funcionales y de usabilidad. Se describe la participación de usuarios con diferentes grados de ELA en las etapas de prueba, lo que permitió refinar y ajustar el sistema para garantizar que cumple con sus objetivos.

- **Capítulo 10 - Conclusiones y Trabajos Futuros:** Finalmente, se presentan las conclusiones del proyecto, valorando los resultados obtenidos y el impacto del sistema desarrollado. Además, se exponen los objetivos a futuro y posibles mejoras que podrían realizarse en el sistema, planteando líneas de trabajo adicionales para el crecimiento y evolución de la solución propuesta.

Capítulo 2

Esclerosis Lateral Amiotrófica

La Esclerosis Lateral Amiotrófica (ELA) es una enfermedad neurodegenerativa grave que afecta a las células nerviosas encargadas de controlar los movimientos musculares voluntarios, conocidas como motoneuronas. Estas células, que se extienden desde el cerebro hasta la médula espinal y de ahí a los músculos de todo el cuerpo, se deterioran con el tiempo. Como resultado, los músculos que se debilitan, se paralizan y finalmente se atrofian.

2.1 Revisión histórica: Evolución del conocimiento sobre la ELA

La Esclerosis Lateral Amiotrófica fue descrita por primera vez en el año 1830 por Charles Bell, quien documentó el caso de una paciente que presentaba afectación bulbar inicial, la cual progresivamente se extendió por su cuerpo [1]. Posteriormente, en 1865, el médico francés Jean-Martin Charcot (1825 - 1893) realizó una importante contribución al estudio de esta enfermedad al publicar el informe de una mujer con síntomas de debilidad muscular progresiva. En 1869, Charcot describió de manera precisa las principales características de la enfermedad, dándole el nombre con el que es conocida actualmente: “Esclerosis Lateral Amiotrófica” [6]. Las características principales que definen a la enfermedad son las siguientes:

- **“Esclerosis lateral”:** Hace referencia a la pérdida de fibras nerviosas, acompañada de un proceso de esclerosis o cicatrización glial en la parte lateral de la médula espinal.
- **“Amiotrófica”:** Indica la atrofia muscular causada por la inactividad crónica, ya que los músculos no reciben las señales nerviosas necesarias para su funcionamiento.

A lo largo de los años, ha habido avances significativos en el estudio y tratamiento de la ELA. De acuerdo con el Ministerio de Sanidad Español, algunos de los hitos más importantes en la investigación de esta enfermedad incluyen [1]:

- **Criterios diagnósticos de ELA:** Conocidos como los Criterios de El Escorial, estos

han facilitado la identificación y diagnóstico de la enfermedad, proporcionando un marco estándar para los profesionales médicos.

- **Creación de registros específicos de ELA:** La creación de registros ha permitido una recopilación más sistemática de datos, facilitando la investigación y el seguimiento de la evolución de la enfermedad en diferentes poblaciones.
- **Descubrimiento del "riluzol":** Este fármaco es actualmente el único con eficacia demostrada para prolongar la esperanza de vida de los pacientes con ELA, siendo un paso importante en el tratamiento farmacológico.
- **Manejo de problemas respiratorios:** Las complicaciones respiratorias son una de las principales causas de mortalidad en pacientes con ELA, y el avance en tecnologías de ventilación asistida ha sido crucial para mejorar la calidad de vida de estos pacientes.
- **Formación de equipos multidisciplinares:** La atención integral mediante equipos formados por neurólogos, fisioterapeutas, psicólogos y otros especialistas ha mejorado la atención que reciben los pacientes, abordando de manera holística sus necesidades en las diferentes etapas de la enfermedad.

Aunque se han logrado avances, la investigación sigue avanzando en busca de nuevos enfoques para el tratamiento y diagnóstico de la ELA [7]. Los científicos continúan investigando biomarcadores que permitan un seguimiento más preciso de la evolución de la enfermedad. Las líneas de investigación actuales se centran en:

- **Desarrollo de nuevos medicamentos:** El objetivo es encontrar tratamientos más efectivos que no solo prolonguen la vida de los pacientes, sino que también mejoren la calidad de vida al aliviar los síntomas más severos.
- **Terapias con células madre:** Estas investigaciones buscan regenerar las neuronas motoras dañadas, proporcionando esperanza para la reparación del tejido afectado por la enfermedad.
- **Terapia génica:** Enfocada en la modificación de los genes implicados en la progresión de la ELA, este campo busca frenar o detener completamente el avance de la enfermedad.
- **Factores neurotróficos:** Estos son moléculas que promueven la supervivencia y función de las neuronas, ofreciendo un enfoque alternativo para proteger las células nerviosas de la degeneración.

2.2 Diagnóstico: Procedimientos y Criterios

El diagnóstico de la ELA se basa principalmente en la evaluación clínica, complementada con pruebas que ayudan a descartar otras patologías que puedan imitar los síntomas. En 2020, con la introducción de los nuevos Criterios de Gold Coast, se ha simplificado el proceso, permitiendo un diagnóstico más temprano, lo que es fundamental para comenzar un tratamiento adecuado lo antes posible.

El primer paso en el diagnóstico es una anamnesis completa, seguida de una exploración física detallada. Las pruebas complementarias, como la electromiografía, juegan un papel crucial al detectar signos de afectación en la segunda motoneurona, incluso en zonas donde los síntomas son menos evidentes. Además, estudios neuroradiológicos y análisis de marcadores sanguíneos son fundamentales para descartar otras enfermedades que podrían presentar síntomas similares.

En los últimos años, los análisis genéticos se han convertido en una herramienta clave para diagnosticar ELA, particularmente en los casos de origen familiar. Sin embargo, un resultado negativo en estos análisis no excluye completamente la posibilidad de la ELA de origen esporádico [8].

Los síntomas iniciales varían, pero comúnmente incluyen la caída frecuente de objetos, tropiezos, fatiga inusual en los brazos y piernas, y dificultades en la movilidad y coordinación, especialmente en las manos, lo que afecta las actividades diarias. A medida que la enfermedad avanza, los pacientes desarrollan problemas más graves, como dificultad para tragar y respirar, lo que en última instancia puede requerir asistencia ventilatoria mecánica.

A pesar de la progresión de la enfermedad, es importante destacar que las capacidades cognitivas y los sentidos de los pacientes suelen permanecer intactos, así como las funciones sexuales y el control de esfínteres. Los síntomas de dolor no son comunes, aunque los calambres musculares y la limitación de movimiento pueden causar molestias, las cuales pueden aliviarse con medicación y ejercicios físicos adecuados. En algunos casos, los pacientes experimentan inestabilidad emocional, caracterizada por episodios de llanto o risa incontrolada, sin que ello implique la presencia de trastornos psiquiátricos subyacentes [9].

2.3 Opciones de Tratamiento: Enfoques actuales para la gestión de la ELA

La ELA es una enfermedad neurodegenerativa de curso progresivo para la cual, hasta el momento, no se ha descubierto una cura definitiva. Sin embargo, existen tratamientos farmacológicos y terapias de apoyo que han demostrado ser efectivos para ralentizar el avance de la enfermedad y mejorar la calidad de vida de los pacientes.

El manejo de los pacientes con **ELA** requiere de un enfoque multidisciplinario. Un equipo compuesto por neurólogos, fisioterapeutas, nutricionistas, terapeutas ocupacionales y otros especialistas trabaja en conjunto para diseñar un plan de tratamiento individualizado, adaptado a las necesidades y circunstancias particulares de cada paciente. Entre los tratamientos y terapias disponibles, se incluyen los siguientes [10][11]:

- **Riluzol:** Este medicamento, aprobado en 1995, es la única terapia modificadora del curso de la enfermedad autorizada por la Agencia Española de Medicamentos y Productos Sanitarios (AEMPS) para el tratamiento de pacientes con ELA en España. Riluzol actúa reduciendo la liberación de glutamato, un neurotransmisor excitatorio, y ha demostrado retrasar el progreso de la enfermedad, extendiendo la esperanza de vida de los pacientes en varios meses.
- **Terapia física:** Los ejercicios físicos de bajo impacto y la fisioterapia son fundamentales para mantener la movilidad, mejorar la fuerza muscular y prevenir las contracciones. La fisioterapia contribuye significativamente a mejorar el bienestar general de los pacientes, aliviando la rigidez muscular y ayudando a mantener la independencia funcional durante más tiempo.
- **Terapia del lenguaje:** A medida que la capacidad de hablar se deteriora, los pacientes pueden beneficiarse del trabajo con un terapeuta del lenguaje. Este tipo de terapia ofrece técnicas y estrategias para prolongar la claridad del habla, y en etapas avanzadas, se introduce el uso de dispositivos de asistencia como los sintetizadores de voz, que permiten a los pacientes comunicarse mediante la tecnología.
- **Apoyo nutricional:** La pérdida de peso es un problema frecuente en los pacientes con ELA debido a las dificultades para masticar y tragar. Un especialista en nutrición asegura que el paciente reciba una ingesta calórica adecuada mediante dietas personalizadas, y en etapas avanzadas, puede ser necesario el uso de dispositivos de succión para evitar la asfixia. En muchos casos, se requiere la colocación de una sonda de alimentación gástrica para garantizar una nutrición segura y efectiva.
- **Apoyo para respirar:** A medida que la enfermedad progresa, los pacientes experimentan dificultades respiratorias. La ventilación no invasiva (VNI), mediante el uso de una máscara, ayuda a mejorar la oxigenación, aliviando los síntomas y mejorando la calidad de vida. En etapas más avanzadas, los pacientes pueden requerir ventilación mecánica o traqueostomía para asistir la respiración de manera más constante. Técnicas como el *Breath stacking* también son utilizadas para mejorar la capacidad de tos y despejar las vías respiratorias.

- **Apoyo psicológico:** El diagnóstico de ELA tiene un fuerte impacto emocional, no solo para los pacientes, sino también para sus familias. El apoyo psicológico juega un papel crucial para ayudar a las personas a enfrentar el estrés, la ansiedad y la depresión asociados con la progresión de la enfermedad. Además, proporciona herramientas para que tanto el paciente como sus familiares puedan adaptarse emocionalmente a los cambios que impone la enfermedad.

En resumen, aunque no existe una cura definitiva para la ELA, estos tratamientos y terapias permiten gestionar los síntomas y mejorar la calidad de vida de los pacientes. Un enfoque integral y multidisciplinario es clave para asegurar que las diferentes necesidades de los pacientes sean abordadas de manera efectiva.

Capítulo 3

Estado del arte

En este capítulo se recopilan las herramientas y tecnologías actuales que tienen un propósito similar al de este proyecto. Estas herramientas están enfocadas en mejorar la calidad de vida de personas con discapacidades motoras o de comunicación, ofreciendo soluciones innovadoras basadas en el control mediante seguimiento ocular o interfaces accesibles. A continuación, se detallan algunas de las principales herramientas disponibles en el mercado.

3.1 Tobii

“**Tobii**” es una empresa pionera en el desarrollo de tecnología de seguimiento ocular, que permite a los usuarios controlar dispositivos mediante el movimiento de sus ojos [12]. El sistema funciona emitiendo luz infrarroja hacia los ojos del usuario, lo que permite capturar imágenes precisas de la posición ocular. A partir de estas imágenes, se calcula la dirección de la mirada, habilitando la interacción con pantallas y otras interfaces digitales. Uno de los dispositivos desarrollados por Tobii se muestra en la Figura 3.1. Aunque la tecnología de Tobii es muy avanzada, uno de sus principales inconvenientes es que requiere la compra de hardware específico, lo que puede limitar su accesibilidad para algunos usuarios.

3.2 Tallk

“**Tallk**” es una aplicación desarrollada por Samsung, compatible únicamente con algunos de sus dispositivos, que permite el control del dispositivo mediante la mirada [13]. Su tecnología se basa en el uso de la cámara del dispositivo para detectar la dirección ocular del usuario, aplicando algoritmos que interpretan estos movimientos para facilitar la interacción. Tallk proporciona un conjunto de frases predefinidas, así como teclados virtuales, lo que permite a los usuarios con problemas de movilidad comunicarse de manera eficiente. A pesar de sus ventajas, el hecho de que solo esté disponible para dispositivos Samsung limita su accesibili-



Figura 3.1: Uno de los modelos de lectores oculares de Tobii [12].

dad a una gama más amplia de usuarios. Además, según algunas asociaciones, la distribución de esta herramienta ha sido limitada.

3.3 Comunicador dinámico de AsTerics Grid

El “**Comunicador dinámico de AsTerics Grid**” es una plataforma multiplataforma diseñada para facilitar la comunicación de personas con dificultades motoras o del habla [14]. Esta herramienta permite la creación y personalización de tableros de comunicación, utilizando pictogramas, imágenes y texto. Una de sus características más destacadas es la posibilidad de personalizar completamente los tableros según las necesidades del usuario, lo que le otorga una gran flexibilidad y adaptabilidad. En la Figura 3.2, se muestra un ejemplo de uno de sus tableros de comunicación personalizables. Este comunicador está valorado positivamente por las asociaciones de pacientes, debido a su capacidad de edición y la posibilidad de incorporar tecnologías de texto a voz (*text-to-speech*), lo que aumenta su potencial como herramienta asistencial.

3.4 Asistente de Voz AAC

El “**Asistente de Voz AAC**” es una aplicación móvil diseñada para ayudar a personas con dificultades en el habla y la comunicación [15]. Esta aplicación ofrece una serie de tableros con palabras y frases predefinidas, lo que permite a los usuarios comunicarse de manera rápida y eficiente sin necesidad de utilizar un teclado virtual convencional. Aunque no dispone de las capacidades de personalización de otras herramientas como AsTerics Grid, su simplicidad y accesibilidad en dispositivos móviles la convierten en una opción popular para usuarios que buscan una solución rápida y sencilla para problemas de comunicación.

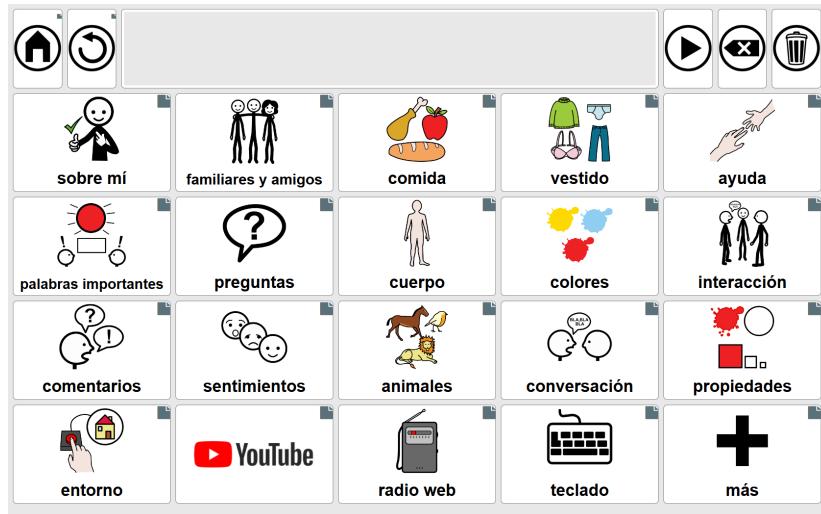


Figura 3.2: Captura de la aplicación web de AsTerics Grid [14].

3.5 Trabajo propuesto

En esta sección se detalla la propuesta de este proyecto, cuyo objetivo principal es desarrollar una solución de comunicación asistida dirigida a personas con Esclerosis Lateral Amiotrófica (ELA) u otras condiciones que afecten la movilidad y la capacidad de comunicación. A través de un análisis de las alternativas existentes, se identificaron varias características clave que se tendrán en cuenta para mejorar las opciones actuales y ofrecer una solución más accesible y eficaz.

La Tabla 3.1 presenta una comparación de las herramientas existentes descritas previamente. Como se puede observar, las soluciones que incorporan control ocular, como Tobii y Tallk, requieren de dispositivos especializados, lo que puede limitar su accesibilidad debido a los costos asociados. A su vez, todas las herramientas analizadas permiten la personalización de los tableros de comunicación, lo que es esencial para adaptarse a las necesidades específicas de cada usuario. No obstante, solo algunas de estas herramientas, como AsTerics Grid, ofrecen la posibilidad de usar pictogramas, facilitando una comunicación más visual y rápida.

	Tobii [12]	Tallk [13]	AsTerics Grid [14]	Asistente de Voz AAC [15]
Control ocular	✓	✓	✗	✗
Necesidad de un dispositivo dedicado	✓	✓	✗	✗
Tableros personalizables	✓	✓	✓	✓
Posibilidad de uso con pictogramas	✓	✗	✓	✗

Tabla 3.1: Comparación de las diferentes alternativas.

A partir de esta comparación, el proyecto propone una solución que combina las mejores

CAPÍTULO 3. ESTADO DEL ARTE

características de las herramientas existentes, a la vez que elimina algunas de las barreras actuales. El objetivo es desarrollar un software que permita el control mediante seguimiento ocular sin la necesidad de un dispositivo especializado dedicado, reduciendo así los costos y mejorando la accesibilidad. El sistema también ofrecerá tableros de comunicación altamente personalizables que podrán ser configurados según las preferencias y necesidades específicas de cada usuario, al igual que permitirá el uso de pictogramas para facilitar una comunicación más visual y rápida.

Este software se diferenciará por su accesibilidad, ya que estará diseñado para funcionar con hardware estándar, como cámaras web integradas en portátiles, eliminando la necesidad de adquirir costosos equipos adicionales. Además, se centrará en la simplicidad de uso y la capacidad de adaptación, lo que lo hará adecuado para una amplia gama de usuarios, desde aquellos en las primeras etapas de la enfermedad hasta quienes se encuentran en fases más avanzadas.

Por lo tanto, el trabajo propuesto busca ofrecer una herramienta inclusiva, adaptable y económica, que permita a los pacientes con ELA y otras condiciones similares mantener una comunicación eficaz y autónoma, mejorando así su calidad de vida.

Capítulo 4

Metodología

Las metodologías de ingeniería de software son fundamentales para garantizar que los proyectos se desarrollen de manera adecuada, proporcionando un marco que permite la planificación, el seguimiento y la entrega eficiente de resultados. En este sentido, el uso de metodologías ágiles se ha convertido en una práctica común en proyectos que requieren adaptabilidad y respuestas rápidas a cambios o nuevas necesidades. Para el desarrollo de “ComunicELA”, se ha elegido Scrum como metodología principal, debido a su enfoque iterativo y adaptable, lo cual es ideal para proyectos con incertidumbres o requisitos cambiantes, como es el caso de los sistemas asistenciales personalizados.

4.1 Metodología Scrum

Scrum, ampliamente reconocido en el mundo del desarrollo de software, divide el trabajo en ciclos denominados *Sprints* que generalmente duran de dos a cuatro semanas. Esta estructura permite enfrentar de manera eficaz la incertidumbre inherente a proyectos innovadores como este. Los pasos básicos de cada Sprint son:

- **Planificación del Sprint:** Se decide por parte del equipo cuáles de las tareas del *Product Backlog* (lista de requisitos y características del software) son más prioritarias y deben ser abordadas en el Sprint actual.
- **Desarrollo:** Durante este periodo, se trabaja en equipo de forma colaborativa en las tareas elegidas, apuntando a cumplir los objetivos establecidos en la planificación.
- **Revisión del Sprint:** Al final de cada Sprint, se presenta el progreso a los *stakeholders*. Esta etapa es crucial ya que es donde se recibe retroalimentación que permite adaptar y refinar “ComunicELA” tomando como base las necesidades reales de los usuarios.
- **Retrospectiva del Sprint:** Esta fase está destinada a la autoevaluación del equipo, identificando fortalezas, debilidades y oportunidades de mejora en el proceso.

4.2 Fases principales

El desarrollo del proyecto se estructura en varias fases clave, cada una de ellas diseñada para garantizar que el software final sea eficaz y relevante para los pacientes con ELA. A continuación, se detallan las fases principales del desarrollo:

- **Revisión bibliográfica y contextualización:** Antes de comenzar con el desarrollo del software, es crucial tener un conocimiento de la ELA, los desafíos comunicativos que enfrentan los pacientes y las soluciones tecnológicas existentes. Esta fase permite identificar las carencias en el campo y orienta el inicio del desarrollo del software.
- **Interacción con AGAELA:** Se establecen contactos regulares con la Asociación Galega de Afectados pola Esclerose Lateral Amiotrófica para comprender directamente las necesidades de los pacientes.
- **Diseño de la interfaz:** Es importante centrarse en la facilidad de uso y en la adaptabilidad para diseñar la interfaz teniendo en cuenta las limitaciones físicas y las necesidades específicas de los pacientes con ELA.
- **Desarrollo técnico:** Esta fase implica la programación e integración de las funcionalidades clave, incluyendo los tableros comunicativos y el lector ocular.
- **Pruebas piloto:** Una vez que la versión beta del software está lista, se prueba con un grupo usuarios, entre ellos pacientes con ELA, para obtener retroalimentación directa e identificar áreas de mejora.
- **Optimización:** Teniendo en cuenta el feedback recibido, se realizan ajustes y mejoras en el software, asegurando satisfacer las necesidades de los usuarios finales.
- **Elaboración de la memoria:** Como se describe anteriormente, se documenta todo el proceso, las decisiones tomadas, los resultados obtenidos y las conclusiones.

4.3 Aplicación en el proyecto

4.3.1 Roles

En el contexto de la metodología Scrum [16], se distinguen varios roles fundamentales que, en conjunto, forman el “Scrum Team”. Estos roles son claves para asegurar la buena organización y ejecución del proyecto, promoviendo la colaboración y la eficiencia. Los roles principales son los siguientes:

- **Scrum Máster:** Facilita el proceso Scrum y ayuda al equipo a seguir prácticas ágiles. En este proyecto, el rol de Scrum Máster se asigna a los tutores, quienes mantienen reuniones semanales con el alumno para guiar el avance del proyecto.
- **Desarrolladores y analistas:** Los analistas son los encargados de realizar los estudios necesarios para el proyecto. Por otra parte, los desarrolladores son las personas que se comprometen a crear cualquier aspecto de un incremento utilizable en cada Sprint. En “ComunicELA”, los analistas se encargan de desarrollar el análisis histórico y la evaluación de las herramientas actuales. El equipo de desarrollo se encarga de la programación, diseño e integración de tecnologías como el seguimiento ocular. Estos roles lo desempeña el alumno.
- **Product Owner:** Es la persona responsable de maximizar el valor del producto y del trabajo del equipo de desarrollo. En el contexto de “ComunicELA”, se asegura de que las funcionalidades desarrolladas respondan a las necesidades de la aplicación. Este rol lo desempeña [AGAELA](#), asegurando que el software sea adecuado para las necesidades de los pacientes con ELA.
- **Stakeholders:** Incluyen a todas las personas y organizaciones que tienen un interés en el proyecto. En este caso, los stakeholders principales son los pacientes con ELA y [AGAELA](#). Su retroalimentación es crucial para asegurar que el producto final sea útil y relevante.

4.3.2 Sprints

El desarrollo del proyecto se organiza en 6 *Sprints*, cada uno con una duración aproximada de dos a 4 semanas. Los sprints permiten dividir el trabajo en ciclos más pequeños, con objetivos específicos a corto plazo, lo que facilita la entrega continua de mejoras y la adaptación a cambios o nuevos requisitos. A continuación, se describe el enfoque general para cada sprint:

- **Sprint 1: Análisis histórico y actual.** Se estudian las limitaciones que sufren los enfermos con ELA y se estudian las diferentes herramientas que existen actualmente para favorecer la comunicación de estas personas. También se establecen los requisitos que debe cumplir el software.
- **Sprint 2: Diseño de la aplicación.** Se realiza un diseño inicial enfocando en la facilidad de uso, permitiendo así una interacción sencilla teniendo en cuenta las limitaciones que puedan tener los pacientes con ELA.
- **Sprint 3: Implementación de la aplicación.** Se implementa el software con los tableros de comunicación, basado en el diseño propuesto en el Sprint anterior.

- **Sprint 4: Implementación del seguimiento ocular.** Se realizan diferentes aproximaciones con métodos de aprendizaje automático supervisado y se valora cuantitativamente cual es la solución más efectiva para el seguimiento ocular.
- **Sprint 5: Optimización y pruebas.** Una vez desarrollada la versión beta de la aplicación, se organiza una reunión con AGAELA para revisar y ajustar las sugerencias proporcionen, con el objetivo de aumentar la calidad del software y la satisfacción de los usuarios finales. Una vez se implementan estos cambios, se realizan las pruebas piloto con usuarios cercanos al alumno, esto permite identificar y recoger mejoras para una mayor optimización antes de organizar un encuentro con los miembros de AGAELA, donde se repiten las pruebas con pacientes afectados por la ELA y la asociación valida la aplicación.
- **Sprint 6: Conclusión.** El encuentro realizado al final del Sprint anterior proporciona el feedback necesario para modificar el software con los detalles finales. Tras la implementación de estos cambios, se concluye el desarrollo de este proyecto, aportando unas nuevas pruebas con usuarios cercanos al desarrollador.

Capítulo 5

Planificación

EN este capítulo se describe la planificación detallada para llevar a cabo el desarrollo del proyecto, asegurando que las actividades se realicen de manera estructurada y eficiente. Además, se incluye una estimación de los recursos y costes necesarios, considerando tanto los aspectos técnicos como los humanos. La planificación adecuada es esencial para cumplir con los plazos y objetivos establecidos, minimizando riesgos y gestionando adecuadamente los recursos disponibles.

5.1 Recursos

Para lograr un desarrollo eficiente y garantizar una planificación óptima, los recursos necesarios se clasifican en dos categorías principales: recursos humanos y recursos materiales. Ambos son fundamentales para asegurar que el proyecto se lleve a cabo con éxito, cumpliendo con los plazos y la calidad requerida.

5.1.1 Recursos humanos

- **Scrum Máster:** Encargado de supervisar y coordinar las fases, asegurando que se cumplan los plazos y objetivos establecidos.
- **Analista de Business Intelligence:** Responsable de la realización de los diferentes análisis necesarios y la evaluación de los aspectos relevantes para el proyecto.
- **Diseñador UX/UI:** Responsable de diseñar la experiencia de usuario (UX) y la interfaz de usuario (UI) asegurando que la aplicación sea intuitiva, fácil de usar y que cumpla con los requisitos para los usuarios finales.
- **Programador Python y AI:** Se ocupa de la programación y desarrollo de la aplicación así como del seguimiento ocular, adaptándose a los diseños que otorgue el diseñador y los requisitos establecidos previamente.

Como ya se detalló en la Sección 4.3.1, los tutores ejercen el rol de Scrum Máster, mientras que el alumno es el encargado de los análisis, el diseño y la programación.

5.1.2 Recursos materiales

- **Ordenador portátil:**
 - **CPU:** Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz, 2592 Mhz, 6 Core(s), 12 Logical Processor(s).
 - **Disco duro:** SSD Samsung, M.2, 1TB.
 - **Tarjeta gráfica:** NVIDIA GeForce RTX 2060.
 - **Memoria RAM:** 16GB 2933Mhz.
 - **SO:** Windows 11 Home 64 bits.
- **Software:** A continuación se explican los diferentes recursos software que se utilizan en el desarrollo, cabe destacar que ya han sido comentados en el Capítulo A.
 - **IDEs de desarrollo:** Se utilizan Visual Studio Code y Google Colab para el desarrollo del código fuente y ejecución del mismo.
 - **Bakend de la aplicación:** Se emplea Python, un lenguaje de alto nivel popular en la industria.
 - **Frontend de la aplicación:** Kivy, una biblioteca de Python que se utiliza para crear interfaces de usuario interactivas y dinámicas, proporcionando una experiencia de usuario agradable y eficiente. También se emplea Canva para la creación de los recursos gráficos que sean convenientes para una interfaz atractiva.
 - **Control de versiones:** Se utiliza GitHub para alojar el repositorio del proyecto, aprovechando el uso de Git para gestionar de manera eficiente los cambios en el código fuente del proyecto.
 - **Implementación de tableros comunicativos:** Para la creación y personalización de los tableros, se utiliza como plataforma de referencia ARASAAC, que ofrece una amplia colección de pictogramas diseñados para facilitar la comunicación de personas con dificultades del habla. Adicionalmente, la Gemini API es empleada para asegurar que las frases construidas por los usuarios a través de los tableros sean coherentes desde un punto de vista gramatical. Finalmente, la reproducción de las frases se realiza mediante ISpVoice. ISpVoice permite transformar el texto en voz, utilizando las voces preinstaladas en el sistema operativo, lo que elimina la necesidad de conexión a internet para reproducir las frases, garantizando así que los usuarios puedan escuchar sus mensajes de forma inmediata y clara.

- **Implementación del lector ocular:** Se emplean las herramientas Mediapipe Solutions, OpenCV, Pillow, Optuna, PyTorch, CUDA, cuDNN y Scikit-learn en su conjunto.

Toda la fundamentación tecnológica detallada, que incluye las herramientas, lenguajes de programación, bibliotecas y plataformas utilizadas durante el desarrollo del proyecto, así como las razones detrás de cada decisión técnica adoptada, puede consultarse en el ([Apéndice A](#)). En dicho apéndice se profundiza en cada una de las tecnologías empleadas, justificando su elección y explicando su papel en la implementación de la solución propuesta.

5.2 Tareas

Las tareas que se deben llevar a cabo se planifican de manera eficiente en los Sprints nombrados en la Sección [4.3.2](#), garantizando siempre que se cumpla el objetivo específico de cada Sprint.

- **Sprint 1: Análisis histórico y actual**
 - Análisis histórico sobre el [ELA](#).
 - Estudio sobre las herramientas actuales.
 - Análisis sobre las limitaciones de los usuarios y las adaptaciones de diseño necesarias.
 - Especificación de requisitos.
- **Sprint 2: Diseño de la aplicación**
 - Bocetos de las vistas de la aplicación.
 - Diseños con Kivy de los bocetos.
- **Sprint 3: Implementación de la aplicación**
 - Implementación de las funcionalidades de desarrollador.
 - Implementación de las funcionalidades de los usuarios.
- **Sprint 4: Implementación del seguimiento ocular**
 - Análisis de características importantes y recopilación de datos.
 - Aproximación inicial: Establecer camino a seguir.
 - Aproximación 1: Perceptrones multicapa.
 - Aproximación 2: Redes neuronales convolucionales.

- Aproximación 3: Redes neuronales híbridas.

- **Sprint 5: Optimización y pruebas**

- Reunión con AGAELA.
- Desarrollo de las mejoras propuestas por AGAELA.
- Pruebas piloto con grupo de control.
- Implementación de mejoras en base al feedback recibido.
- Pruebas piloto con miembros de AGAELA y validación del software.
- Implementación de mejoras en base al feedback recibido.

- **Sprint 6: Conclusión**

- Pruebas finales con un grupo de control.
- Redacción de la memoria.

5.3 Seguimiento y costes

Durante el desarrollo del proyecto, se llevaron a cabo reuniones semanales de seguimiento entre los tutores y el equipo de desarrollo para evaluar el progreso y proporcionar orientación en cada fase del proyecto. Estas reuniones permitieron ajustar la planificación cuando fue necesario y asegurar que el desarrollo se mantuviera alineado con los objetivos establecidos.

Adicionalmente, se realizaron reuniones periódicas con distintos especialistas de la AGAELA, quienes aportaron valiosa retroalimentación sobre la funcionalidad y adaptabilidad del sistema a las necesidades reales de los pacientes. Como parte del proceso de validación, se realizaron pruebas *in situ* con pacientes diagnosticados con ELA en diversas fases de la enfermedad. Estas pruebas permitieron evaluar la efectividad del sistema en un entorno real, lo que proporcionó información crucial para realizar ajustes y mejorar la experiencia del usuario.

En el Apéndice B se incluye el diagrama de Gantt del proyecto, el cual ofrece una visión clara de la organización de las tareas y las fases del desarrollo de “ComunicELA”. Además, se presenta un análisis detallado del tiempo invertido y los costes esperados, comparados con los valores reales finales. Esto incluye el desglose del tiempo total de duración del proyecto y los gastos asociados a los recursos empleados, tanto humanos como materiales.

Capítulo 6

Análisis de requisitos

Un análisis de requisitos claramente especificado es fundamental para el desarrollo exitoso de cualquier proyecto de software. En este capítulo se detallan los requisitos que este debe cumplir, los cuales se han planificado con la colaboración de AGAELA, a través de diversas reuniones, con el objetivo de desarrollar una aplicación que satisfaga las necesidades de los usuarios finales.

6.1 Actores

En el contexto de esta aplicación, los actores son las entidades o usuarios que interactúan con el sistema. Dado que la aplicación está diseñada con menús de desarrollo principalmente ocultos al usuario final, se clasifican los actores en dos categorías principales:

- **Usuario Común:** Representa a los usuarios finales que utilizan la aplicación para el propósito para el cual ha sido desarrollada, es decir, facilitar la comunicación. Estos usuarios no tienen acceso a las funciones avanzadas o de desarrollo, interactuando únicamente con las opciones habilitadas en la interfaz de usuario principal.
- **Usuario Desarrollador:** Son los usuarios involucrados en el desarrollo y mantenimiento del software. Tienen acceso a las opciones de desarrollador y menús avanzados que permiten realizar modificaciones en la estructura del código, la configuración del sistema, y probar nuevas funcionalidades.

6.2 Análisis de los requisitos:

En esta sección se detallan los requisitos funcionales y no funcionales de la aplicación. Estos requisitos se dividen en dos grandes categorías: aquellos asociados a los actores que

interactúan con el sistema y los relacionados con la pre-configuración del software. Esta distinción permite una comprensión más clara de las necesidades tanto de los usuarios como de los desarrolladores.

6.2.1 Requisitos funcionales:

Requisitos de pre-configuración:

- **Tutorial del software:** Al iniciar la aplicación, se debe presentar un tutorial con la opción de no volver a mostrarlo.
- **Selección de idioma:** La aplicación debe permitir elegir el idioma deseado.
- **Selección de cámara:** Es necesario ofrecer la opción de seleccionar entre las cámaras disponibles.
- **Selección de voz:** La personalización de la voz para la síntesis de voz es esencial para la comodidad de los usuarios finales.
- **Activar / desactivar el conjugador automático de frases:** Se debe proporcionar la opción de conjugar automáticamente las frases escritas antes de reproducirlas en voz alta.

Requisitos de usuario común:

- **Calibrar el parpadeo:** La aplicación debe permitir calibrar el gesto de parpadeo para que el usuario pueda marcar una pulsación con los ojos.
- **Reentrenar el modelo:** Es necesario permitir el reentrenado y optimización del modelo con los datos del usuario para mejorar la precisión. También se debe ofrecer la opción de reestablecer el modelo original descartando los datos registrados en los reentrenamientos.
- **Tableros comunicativos con o sin pictogramas y personalizables:** Los tableros deben permitir el uso con o sin pictogramas, al igual que personalizar el número de casillas, palabras y pictogramas.
- **Escritura y borrado de palabras en los tableros:** Cuando una casilla es seleccionada, debe escribirse en el apartado de texto la palabra correspondiente. Por otra parte, también se debe permitir borrar la última palabra escrita o la frase completa.
- **Reproducción de frases con síntesis de voz:** Las frases deben poder ser reproducidas en voz alta utilizando un método de síntesis de voz.

- **Alarma en el tablero:** Los tableros deben contar con un botón de alarma que active un modo avisador.
- **Control ocular de los tableros:** Se debe proporcionar un sistema de control ocular, sin la necesidad de un dispositivo dedicado.
- **Modo descanso:** Mientras se utiliza el control ocular, se debe ofrecer un modo descanso de fácil acceso para que los usuarios puedan bloquear el sistema.

Requisitos de desarrollador:

- **Pantalla de test:** Se debe incluir un apartado donde el desarrollador pueda observar el funcionamiento del modelo del lector ocular y modificar el postprocesado aplicado.
- **Recopilar datos:** Es necesario ofrecer una opción para recopilar datos de voluntarios y crear las bases de datos necesarias para entrenar el modelo encargado del seguimiento ocular.
- **Pruebas:** Se deben proporcionar unas pruebas automatizadas para evaluar el software.

6.2.2 Requisitos no funcionales:

- **Requisitos de interfaz:** Se siguen las indicaciones de AGAELA para producir una interfaz intuitiva, sencilla y accesible, adaptando los diseños según sus sugerencias.
- **Requisitos hardware:** La aplicación y el lector ocular deben ser ejecutados eficientemente en CPU, desarrollando así un software más accesible.
- **Requisitos de usabilidad:** El software debe contar con la documentación y guías necesarias para su uso.
- **Tolerancia a fallos:** El software debe estar preparado para afrontar diferentes situaciones de error, proporcionando mensajes de error con el feedback necesario.

6.3 Casos de uso

En esta sección se presentan los diferentes casos de uso identificados para la aplicación en desarrollo. Estos casos de uso describen las interacciones entre los usuarios y el sistema, detallando las funcionalidades principales que la aplicación debe ofrecer. La información aquí proporcionada sirve como una introducción a los escenarios de uso más relevantes. Una descripción ampliada y detallada de cada caso de uso, incluyendo diagramas y explicaciones exhaustivas, se puede encontrar en el Apéndice C.

Casos de uso de la pre-configuración:

A continuación, se presentan los casos de uso para la pre-configuración de la aplicación, donde los usuarios pueden establecer diferentes parámetros según sus preferencias. Estos se ven representados en la Figura 6.1 y son comentados a continuación:

- **Selección de idioma:** Al pulsar sobre el ícono del idioma, este cambia alternando entre los disponibles.
- **Tutorial del software:** Se muestra un tutorial al inicio de la aplicación, donde se abren diferentes ventanas guiando el uso de la aplicación. En la última, en caso de seleccionar “No volver a mostrar”, no se despliega al abrir de nuevo la aplicación.
- **Configuración:** Menú donde el usuario configura el software y puede volver a mostrar el tutorial.
 - **Selección de cámara:** Se muestran las cámaras disponibles y la opción de actualizar la lista, permitiendo al usuario escoger la cámara a utilizar para el control ocular.
 - **Selección de voz:** Se muestran las voces disponibles y la opción de actualizar la lista, permitiendo al usuario seleccionar la que se adapte a sus gustos.
 - **Activar / desactivar el conjugador automático:** Cambia el estado del conjugador automático de las frases.

Casos de uso del usuario común:

Los casos de uso del usuario común representan las interacciones que tiene el usuario final con la aplicación. Se ven representados en la Figura 6.2 y son comentados a continuación:

- **Calibrar el parpadeo:** La calibración del parpadeo ajusta al perfil del usuario el gesto utilizado para captar pulsaciones con los ojos.
- **Reentrenar el modelo:** Se presentan las instrucciones sobre el reentrenamiento del modelo.
 - **Reentreno:** El usuario debe seguir un punto en movimiento por la pantalla y posteriormente, el sistema se reentrena y optimiza con las métricas del usuario para ajustarse al mismo.
 - **Descartar reentrenamientos:** Restablece el sistema devolviendo el modelo del lector ocular a sus características por defecto, permitiendo así recuperar la precisión en caso de pérdida debido a un sobreajuste.

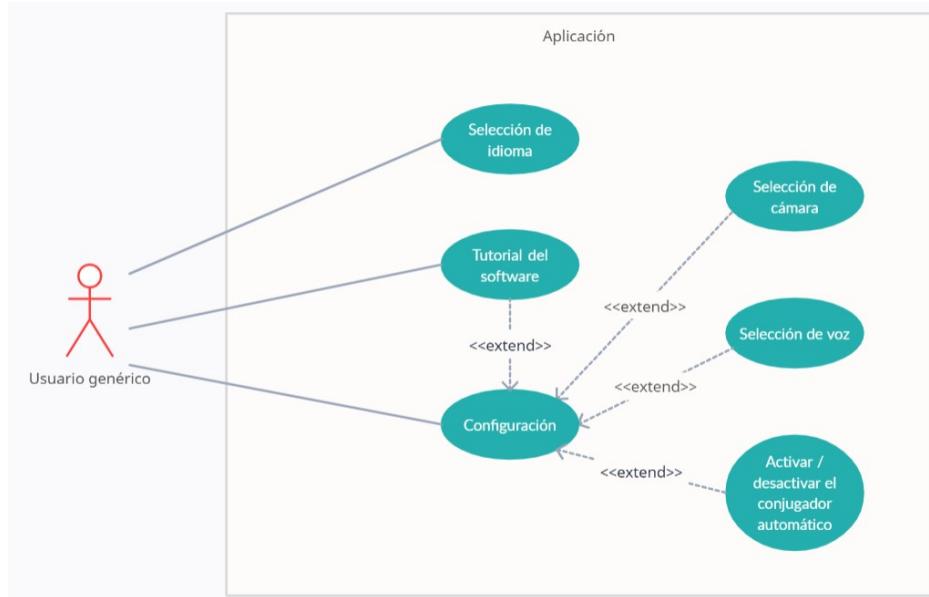


Figura 6.1: Casos de uso de la pre-configuración.

- **Tableros comunicativos con o sin pictogramas:** Se presentan las instrucciones para interactuar con los tableros, donde el usuario elige si desea utilizar los tableros con pictogramas o solamente con texto.
 - **Escritura de palabra:** El usuario escoge la casilla que quiere seleccionar, una vez confirmada la selección, se añade la palabra correspondiente a la frase que se está formando.
 - **Borrado único:** Borra la última palabra de la frase.
 - **Borrado total:** La frase se borra por completo, permitiendo crear una nueva.
 - **Reproducción de frases con síntesis de voz:** Se reproduce en alto la frase utilizando la voz seleccionada por el usuario en la pre- configuración.
 - **Alarma:** Suena una alarma con el fin de avisar al personal de asistencia que esté pendiente del usuario.
 - **Control ocular:** El usuario controla el puntero de los tableros con movimientos oculares. Al mantenerlo sobre una casilla, esta se bloquea pudiendo confirmar la selección mediante el parpadeo.
 - **Modo descanso:** Durante el uso del control ocular, cerrar los ojos durante tres segundos activa el modo descanso de la aplicación, bloqueando la pantalla. Para desactivar este modo, basta con mantener los ojos cerrados otros tres segundos.

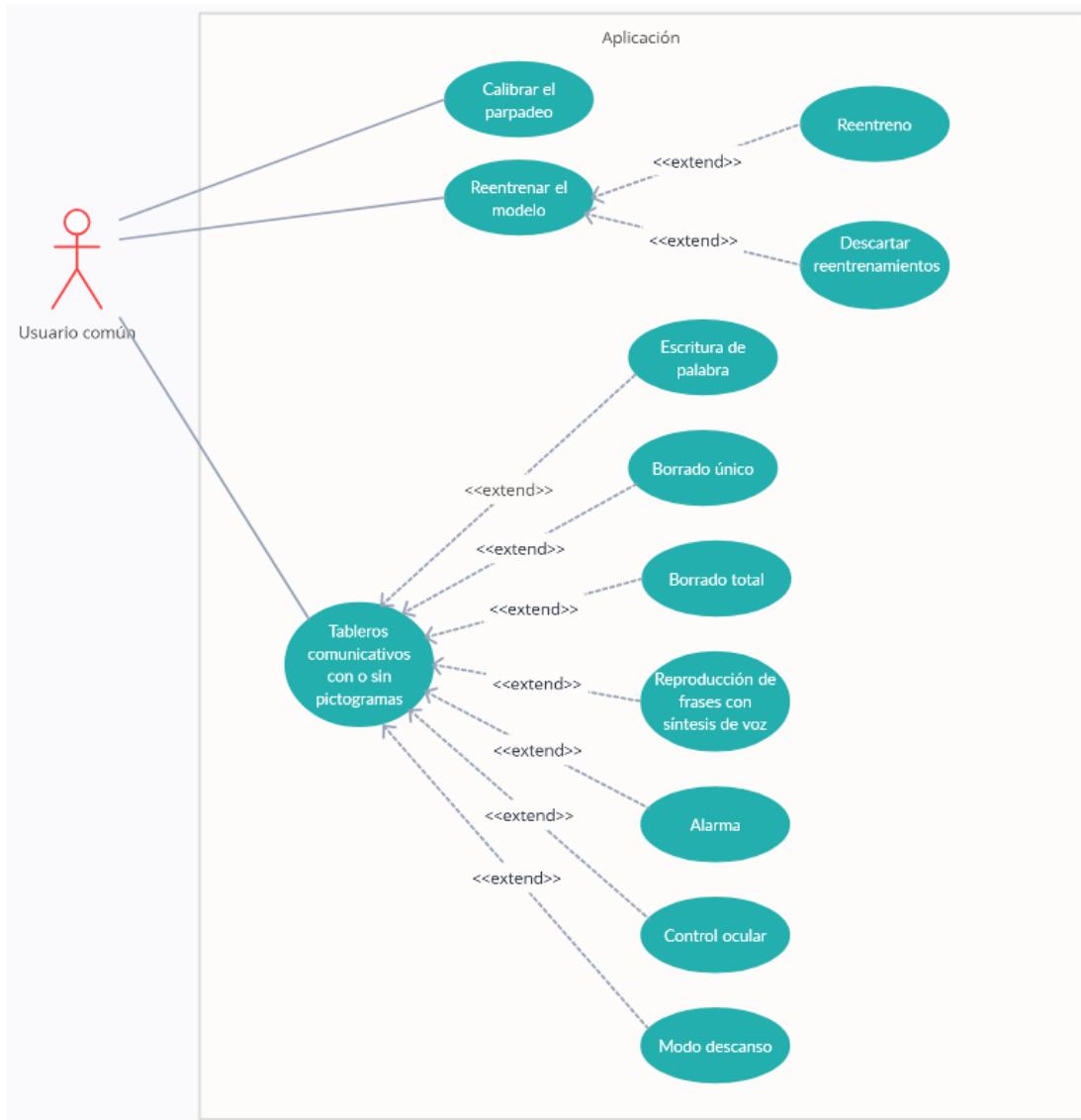


Figura 6.2: Casos de uso del usuario común.

Casos de uso del usuario desarrollador:

A continuación, se describen los casos de uso específicos para los desarrolladores. Estos se ilustran en la Figura 6.3 y son comentados a continuación:

- **Activar / desactivar opciones de desarrollador:** Permite cambiar el estado de las opciones de desarrollador desde la pantalla principal, pulsando la tecla “D”.
- **Modo Test:** El desarrollador puede ver el comportamiento del modelo encargado del seguimiento ocular.

- **Ajustar postprocesado:** El desarrollador personaliza el postprocesado que se aplica a la salida del modelo del lector ocular para ajustarlo convenientemente.
- **Recopilar datos:** El desarrollador recopila datos de voluntarios, los cuales deben seguir un punto en movimiento por la pantalla, registrando los datos necesarios para crear las BD necesarias para el desarrollo del modelo del lector ocular.
- **Pruebas:** Se muestra la prueba a realizar en cada momento. Las pruebas se detallan en la Sección 9.
- **Comenzar prueba:** Se inicia el cronómetro y los diferentes registros para guardar los resultados de la prueba actual.
- **Saltar prueba:** Se salta a la siguiente prueba sin realizar la actual.
- **Prueba anterior:** Se regresa a la prueba anterior.

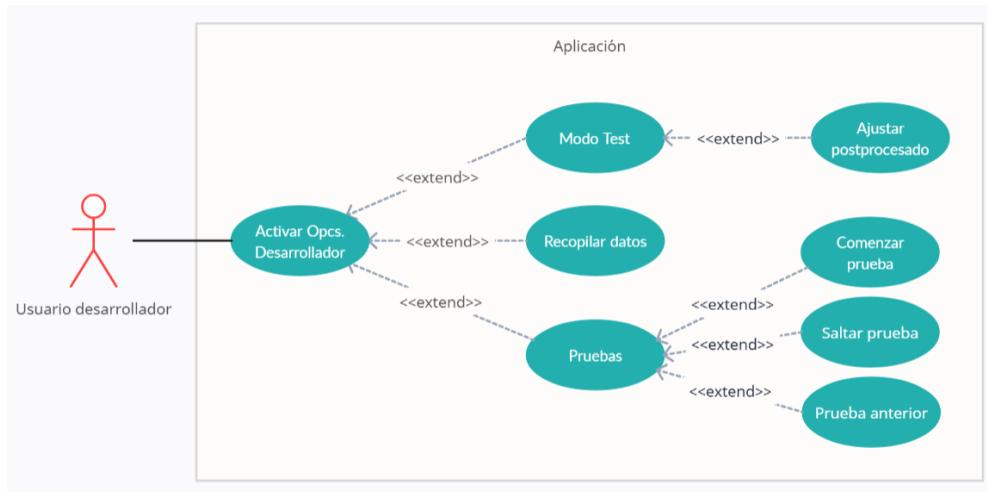


Figura 6.3: Casos de uso del usuario desarrollador.

Capítulo 7

Diseño

ESTE capítulo está dedicado a detallar el diseño de la aplicación, con especial énfasis en la creación de una interfaz accesible, intuitiva y amigable para los usuarios. Se describen los principios de diseño adoptados, así como las decisiones técnicas y estéticas que guían el desarrollo de la interfaz y la experiencia de usuario. Además, se incluyen las consideraciones sobre la accesibilidad para garantizar que la aplicación sea funcional para personas con diferentes grados de movilidad y necesidades específicas.

7.1 Diseño de la aplicación

El diseño de la interfaz es un aspecto crucial para tener en cuenta cuando se desarrolla una aplicación para personas afectadas por la [ELA](#) u otras patologías. Gracias a la colaboración con [AGAELA](#), se ha podido orientar el desarrollo de estos diseños de manera adecuada, adaptándolos a las necesidades de los usuarios finales para crear una aplicación accesible, sencilla y amigable. Esta asociación ha proporcionado consejos y guías para realizar un diseño responsable, asegurando los cuidados necesarios para que los usuarios finales no tengan problemas al utilizar la aplicación. Para ello, se han seguido las recomendaciones sobre colores, fuentes, tamaños de botones y contrastes.

Los bocetos de los diseños de las pantallas de la aplicación se encuentran en el Apéndice D. Por otra parte, las vistas finales de las pantallas, que muestran cómo se implementaron estos diseños en la aplicación, también se presentan en el mismo apéndice. Estas vistas permiten apreciar cómo los bocetos iniciales se transformaron en interfaces funcionales y accesibles.

7.2 Diseños en Kivy

La aplicación ha sido desarrollada en Python utilizando la herramienta Kivy, que ofrece una gran personalización en la creación de interfaces de usuario. El diseño de los componentes visuales se ha desarrollado en un archivo .kv, lo que permite separar la presentación visual del resto de la lógica del código, manteniendo una estructura organizada y modular.

Los botones de la interfaz han sido diseñados con esquinas redondeadas para mejorar su apariencia estética, brindándoles un aspecto moderno. Siguiendo las indicaciones de AGAE-LA, se ha optado por letras negras sobre un fondo claro para asegurar la accesibilidad y una fácil lectura. En la Figura 7.1 se puede ver un ejemplo de botón creado con estas características.



Figura 7.1: Ejemplo de un botón creado con Kivy.

El diseño de los **Spinner** sigue un patrón estético coherente con el resto de la aplicación, manteniendo la simplicidad y funcionalidad. Los **PopUp** se han desarrollado con la misma lógica visual, ofreciendo una interfaz intuitiva para las ventanas emergentes. Un ejemplo de PopUp se puede observar en la Figura 7.2.

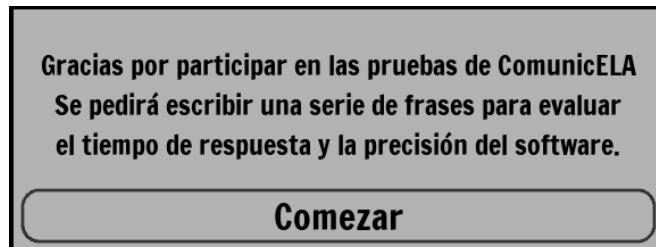


Figura 7.2: Ejemplo de un PopUp creado con Kivy.

Las casillas de los tableros comunicativos han sido diseñadas en dos variantes: con pictogramas y sin ellos. Estas variantes permiten adaptarse a las preferencias y necesidades del usuario. En la Figura 7.3 se puede observar el diseño de las casillas con pictogramas, mientras que en la Figura 7.4 se muestra el diseño para las casillas de texto. Adicionalmente, el color de las casillas varía según su función: aquellas que abren un tablero tienen un tono diferente de las casillas que escriben una palabra.

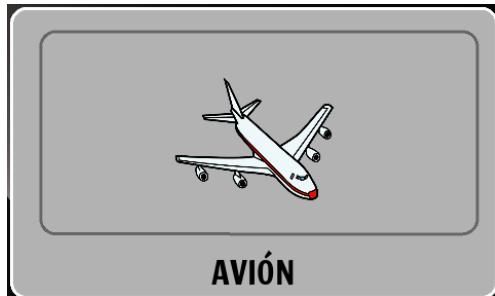


Figura 7.3: Casilla con pictograma.

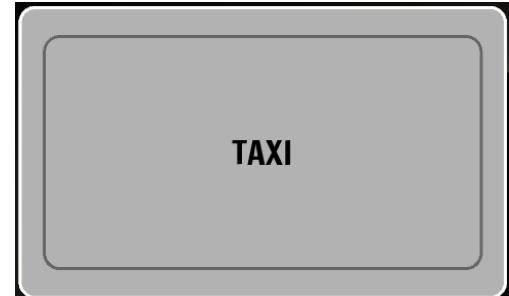


Figura 7.4: Casilla con texto.

Para la comunicación de mensajes informativos al usuario, se han implementado notificaciones de tipo [Toast](#). Estos mensajes se superponen de manera temporal al contenido de la aplicación, apareciendo en la parte inferior central de la pantalla durante tres segundos. Utilizan una animación de aparición y desvanecimiento suave para no interferir con la interacción del usuario.

Capítulo 8

Implementación

ESTE capítulo está dedicado al desarrollo de la aplicación y del lector ocular, detallando los procesos y metodologías empleados para la implementación de ambos componentes.

8.1 Implementación del lector ocular

Para el desarrollo del lector ocular, se propone entrenar un modelo de aprendizaje automático supervisado. Dado que no existe una base de datos ([BD](#)) predefinida que cumpla con los requisitos específicos de este proyecto, se opta por diseñar una base de datos propia. Esta decisión permite recopilar y organizar los datos necesarios de forma eficiente, asegurando que el modelo se entrene con información precisa y relevante para abordar el problema específico en cuestión.

8.1.1 Criterios y restricciones de la recopilación de datos

La creación de la base de datos para el lector ocular sigue una serie de criterios y restricciones esenciales para garantizar la calidad y relevancia de los datos recopilados. Los principales criterios y restricciones incluyen:

- **Consentimiento y privacidad:** Todos los usuarios deben ser personas cercanas al desarrollador y deben dar su consentimiento, asegurándose de que los datos recopilados no serán publicados y se mantendrá su privacidad.
- **Sin gafas:** Los usuarios no deben llevar gafas durante la recopilación de datos. Las pruebas iniciales mostraron que el uso de gafas disminuye el rendimiento del sistema, por lo que en esta versión del software no se utilizarán.
- **Iluminación adecuada:** El rostro del usuario debe estar bien iluminado durante la recopilación para asegurar una captura de datos precisa.

- **Distancia, posición y calibrado:** Se debe respetar la distancia requerida entre el rostro y la pantalla, colocar la cámara a la altura adecuada y calibrar el parpadeo antes de la recopilación. El proceso de calibrado se detalla más adelante en la Sección 8.2.4.
- **Precisión en la mirada:** Los usuarios deben ser responsables y no desviar la mirada del punto a seguir, evitando así la recopilación de datos erróneos.

8.1.2 Análisis de características

El desarrollo de una base de datos (BD) adecuada para nuestro sistema de lector ocular implica una selección cuidadosa de características que son cruciales para resolver el problema específico de seguimiento ocular. Tras un análisis exhaustivo de los factores que pueden influir en la precisión del sistema, se han seleccionado las siguientes características para ser incluidas en la base de datos:

Distancias desde puntos fijos hasta las pupilas

Se plantea que, al rodear un ojo con 16 puntos fijos y conectarlos con la pupila, las longitudes de estas conexiones (medidas en píxeles) reflejan la orientación del ojo en relación con la cabeza, siempre y cuando la cabeza esté lo más recta y centrada posible. Esta idea se ilustra en la Figura 8.1a. Entre los puntos detectados por *Mediapipe Face Mesh* se incluyen los 16 puntos alrededor de cada ojo y las pupilas. Esto permite medir las longitudes de las conexiones para cada ojo de manera individual, como se muestra en la Figura 8.1b. La calidad del reconocimiento de puntos de Mediapipe ha sido previamente evaluada y se ha optado por su uso debido al bajo nivel de ruido en los puntos localizados y al bajo coste computacional requerido para su ejecución.

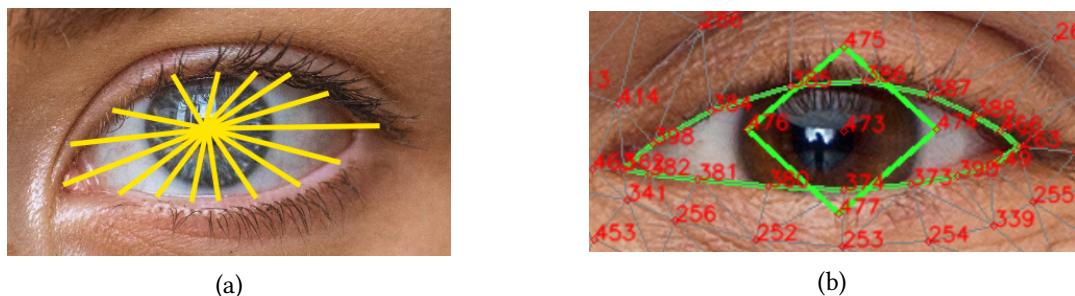


Figura 8.1: (a) Distancias desde los puntos de referencia hasta la pupila. (b) Representación de los puntos de referencia detectados por Mediapipe.

Considerando que el diámetro del globo ocular humano varía entre 2,2 cm y 2,5 cm [17], es previsible que estas distancias sean similares en todos los usuarios, siempre que se mantenga una distancia estándar entre el rostro y la cámara durante las mediciones. Esta distancia, que

oscila entre 50 cm y 60 cm, garantiza una correcta detección de los puntos y evita que el usuario tenga que acercarse demasiado a la pantalla, proporcionando así una experiencia cómoda.

Para el cálculo de las características relevantes como la proporción de aspecto de los ojos (EAR) y otras medidas relacionadas con el seguimiento ocular, se utilizan puntos de referencia específicos obtenidos del modelo de Mediapipe. Estos puntos son cruciales para obtener medidas precisas de las características oculares. A continuación se detallan los puntos de referencia usados para cada ojo:

- **Pupila izquierda:** Punto 473, que indica la posición central de la pupila izquierda.
- **Contorno ojo izquierdo:** Puntos 362, 398, 384, 385, 386, 387, 388, 466, 263, 249, 390, 373, 374, 380, 381, 382, que forman el contorno del ojo izquierdo y son esenciales para determinar la apertura del párpado y otros movimientos.
- **Pupila derecha:** Punto 468, que indica la posición central de la pupila derecha.
- **Contorno ojo derecho:** Puntos 33, 246, 161, 160, 159, 158, 157, 173, 133, 155, 154, 153, 145, 144, 163, 7, que forman el contorno del ojo derecho y son utilizados para cálculos similares a los del ojo izquierdo.

Se calculan las distancias euclidianas entre la pupila y los puntos del contorno del ojo, obteniendo un vector de 16 distancias en píxeles por ojo. La Figura 8.2 muestra un gráfico del promedio de estas distancias de ambos ojos mientras se mira hacia los extremos horizontales de la pantalla a una altura media. El gráfico revela que las distancias son indicativas de la posición del ojo, permitiendo observar las diferencias en los valores cuando se mira hacia la derecha o la izquierda.

EAR

El [Eye Aspect Ratio](#) es la relación de aspecto del ojo, la cual es constante mientras el ojo está abierto, pero tiende a cero cuando el ojo está cerrado. En la figura 8.3, extraída del artículo de Soukupova y Čech [18], se muestra un gráfico donde el [EAR](#) disminuye debido a un parpadeo. La fórmula utilizada para su cálculo, también proveniente del artículo, es la siguiente:

$$\text{EAR} = \frac{\|p2 - p6\| + \|p3 - p5\|}{2 \cdot \|p1 - p4\|} \quad (8.1)$$

Para obtener esta medida, se deben definir los puntos de referencia de *Mediapipe Face Mesh* que corresponden a los diferentes P_i necesarios para el cálculo. Esta correspondencia se muestra en la Tabla 8.1.

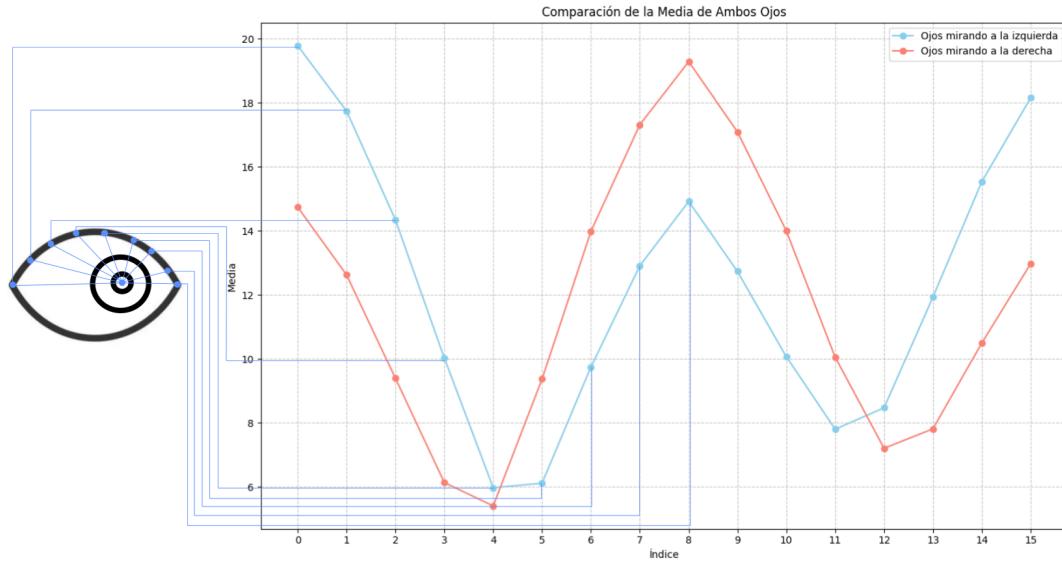


Figura 8.2: Gráfico de las distancias de los ojos mirando a los extremos de la pantalla.

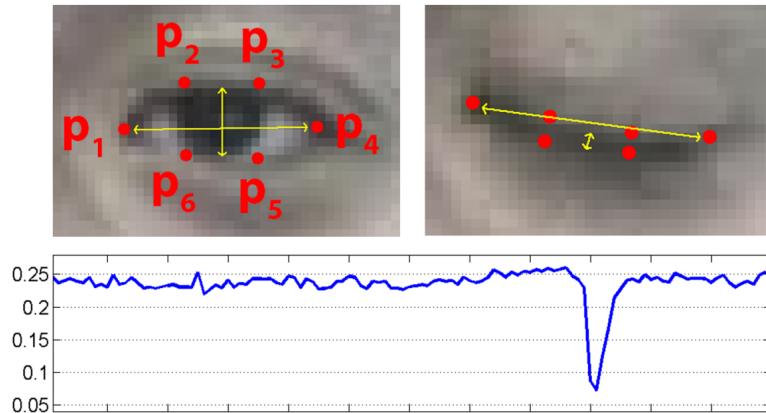


Figura 8.3: EAR durante un parpadeo.

Esta medida, junto con el umbral determinado durante la calibración que indica cuándo el usuario tiene el ojo cerrado (como se detalla en la Sección 8.2.4), son elementos cruciales para el correcto preprocesado de los datos. Este proceso se explicará en detalle más adelante en la Sección 8.1.5.

Orientación de la cabeza

La orientación de la cabeza influye en la interpretación de las distancias capturadas porque las distancias relativas varían si la cabeza está centrada o girada, incluso si la vista permanece fija en un punto. Por lo tanto, es esencial capturar la orientación de la cabeza para una correcta interpretación de los datos.

Ojo Izquierdo		Ojo Derecho	
Punto de Mediapipe	P_i	Punto de Mediapipe	P_i
362	P_1	133	P_1
385	P_2	158	P_2
387	P_3	160	P_3
263	P_4	33	P_4
373	P_5	144	P_5
380	P_6	153	P_6

Tabla 8.1: Correspondencia entre los puntos de Mediapipe y los P_i de la fórmula.

Siguiendo el método propuesto por Aflalo *et al.* [19], se seleccionan puntos específicos de Mediapipe que corresponden a un rostro genérico en 3D. Estos puntos se mapean a un plano tridimensional para obtener los grados de inclinación de la cabeza en los tres ejes principales. Los puntos utilizados para este mapeo y su correspondencia en el rostro genérico se ilustran en las Figuras 8.4 y 8.5.

- Punto de la nariz: 4
- Punto del mentón: 152
- Esquinas de los ojos (izquierdo y derecho): 263, 33
- Esquinas de la boca (izquierda y derecha): 287, 57

Utilizando la función `solvePnP` de OpenCV y los puntos definidos, se calculan los ángulos de Euler para determinar la orientación de la cabeza en los ejes X, Y y Z. Para el eje Y, debido a inconsistencias en las pruebas, se calcula y normaliza la pendiente de la línea que une las esquinas de los ojos al rango [0, 1]. Con la cabeza en posición recta frente a la cámara, esta pendiente es 0, que se normaliza a 0.5. El vector completo de la orientación de la cabeza es representado como [0.5, 0.5, 0.5], indicando una vista frontal y recta. Valores extremos corresponden a un giro máximo de 45° para cada eje.

Posición de la cabeza en el marco

La posición de la cabeza en el marco es importante, ya que la perspectiva varía según el punto de vista. Para determinar la posición de la cabeza, se utiliza el punto de referencia 8 de

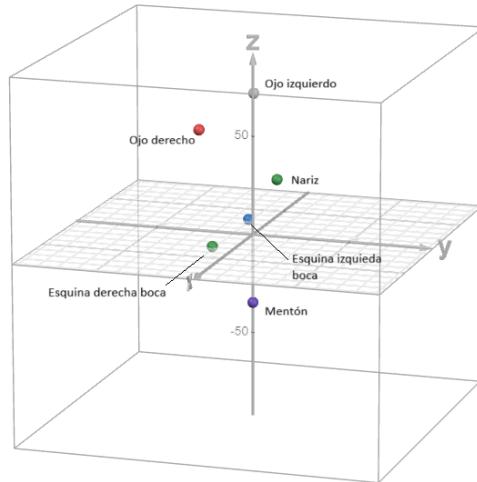
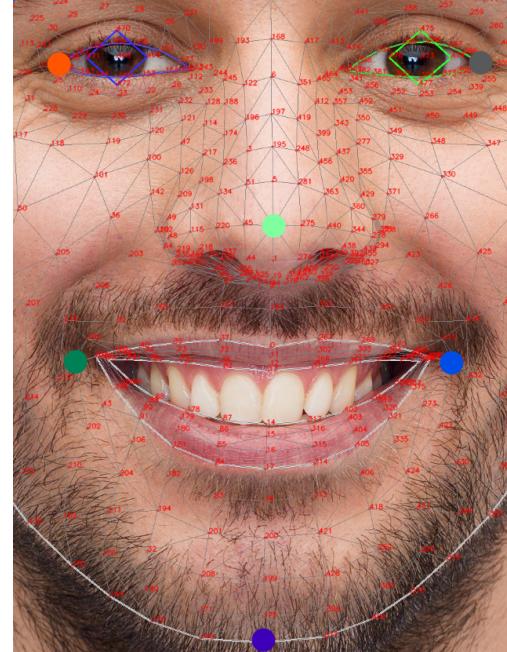


Figura 8.4: Puntos 3D formando el modelo de la cara genérico en perspectiva lateral.



Por otra parte, el archivo `outputs.txt` contiene en cada línea dos valores en punto flotante en el rango [0, 1]. El primer valor representa la coordenada en el eje horizontal, donde 0 es el borde izquierdo de la pantalla, y el segundo valor representa la coordenada en el eje vertical, donde 0 es la parte inferior de la pantalla. Para la creación de esta BD, se contó con el apoyo de 23 voluntarios, quienes siguieron el método de recopilación de datos descrito en la Sección 8.2.7, logrando reunir un total de 63.080 registros.

8.1.4 Base de datos secundaria

Gracias al enfoque incremental adoptado durante el desarrollo del proyecto, surgieron nuevas ideas que requerían el uso de imágenes para su implementación. Estas necesidades no estaban contempladas en la BD inicial. Por ello, se decidió crear una segunda base de datos con la misma estructura que la anterior, pero incluyendo los *frames* capturados, con el índice de la línea correspondiente como nombre del archivo. De esta manera, se podrían emplear nuevas arquitecturas que trabajen con imágenes, como las redes neuronales convolucionales.

La creación de esta BD ha seguido el mismo enfoque que la primera. Sin embargo, se ha contado con la colaboración de diferentes voluntarios, ya que en caso de que las imágenes no sean necesarias, ambas bases de datos podrían llegar a fusionarse para formar una única BD más grande y completa, evitando duplicidades.

En esta ocasión, se contó con la participación de 15 nuevos voluntarios, quienes siguieron el método de recopilación, logrando almacenar correctamente un total de 33.352 registros junto con sus *frames* correspondientes.

8.1.5 Pre-procesado de datos

El pre-procesado de los datos es un paso crucial en proyectos de aprendizaje automático, ya que mejora la calidad y eficiencia del modelo final. Para esto, se eliminan los valores atípicos que puedan distorsionar el resultado y se normalizan los datos, asegurando que todos tengan la misma importancia para la red neuronal.

Para evitar el ruido, se suavizan los índices [0 – 36] utilizando un filtro Gaussiano [20] de una dimensión a lo largo de la línea temporal, sin afectar a los valores del EAR (índices [37, 38]). Es importante no suavizar el último registro de un usuario con el primero del siguiente, ya que el proceso de recopilación no siempre termina en la misma coordenada en la que empieza y se estarían mezclando datos que no tienen relación. Por lo tanto, es necesario suavizar los registros de cada voluntario individualmente. Para seleccionar el valor de σ , se probaron varios valores. Finalmente, se optó por una $\sigma = 5$, ya que ofrecía un suavizado adecuado de las distancias sin reducir su varianza natural. En la Figura 8.6, se muestra el resultado del suavizado en los primeros 300 registros del índice 0.

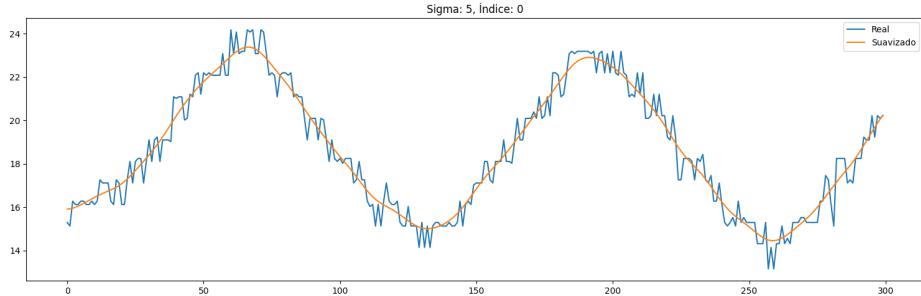


Figura 8.6: Suavizado de las distancias del índice 0 con sigma 5.

Como se mencionó anteriormente, los dos últimos índices no se suavizan. El último índice representa el umbral definido en la calibración, a partir del cual se considera que el usuario tiene los ojos cerrados, por lo que este valor no varía. El penúltimo índice corresponde al valor actual del EAR, que experimenta caídas bruscas de corta duración debido a los parpadeos involuntarios. Suavizar estos valores podría elevar el valor del EAR durante esas caídas, al combinarse con los datos de ojos abiertos en los registros próximos. Esto interferiría con la correcta eliminación de los registros donde los ojos están cerrados, un paso necesario para evitar entrenar con datos erróneos. Se eliminan aquellos registros cuyo valor en el índice 37 sea menor que el del índice 38.

A continuación, se procede a la normalización de los datos. Los tres índices que representan la orientación de la cabeza ([32 , 33 , 34]), los dos que indican su posición en el frame ([35 , 36]) y los del EAR ([37 , 38]) ya están ajustados al rango establecido en la propia base de datos. Por lo tanto, solo se normalizan los índices que corresponden a las distancias de los ojos, utilizando la técnica Min-Max de forma individual para cada ojo. De esta manera, se asegura que se evalúe la relación entre las distancias y no sus valores absolutos, permitiendo una mayor flexibilidad en la distancia del rostro a la cámara y en el tamaño de los ojos, aunque como se comentó en la Sección 8.1.2, este tamaño casi no varía.

Este proceso de normalización garantiza que todas las características se encuentren en la misma escala, dándoles la misma importancia durante los entrenamientos de los modelos, ya que todas las entradas y salidas se encuentran en el mismo rango de 0 a 1.

8.1.6 Métodos y métricas de error

Para estimar el error y minimizar la influencia del azar en el estudio, se emplea el método de *k-fold cross-validation leave-one-out*. Este enfoque implica realizar tantos experimentos (*folds*) como voluntarios tenga la BD, utilizando N-1 voluntarios para entrenar y dejando al restante para la prueba. Además, se realizan 5 ejecuciones por cada *fold* para asegurar la robustez y consistencia de los resultados.

Dado que el problema de regresión a resolver se basa en minimizar la distancia del punto predicho al punto real de la pantalla donde se está mirando, se han utilizado diferentes métricas para la evaluación el error:

- **Mean Square Error (EMS):** El error cuadrático medio, mide la diferencia promedio al cuadrado entre las coordenadas predichas y las reales. Con esta medida los errores más grandes tienen un impacto mayor en el valor final.
- **Distancia euclíadiana:** Proporciona una medida visualmente intuitiva del error, expresando la distancia entre el punto predicho y el punto real en una escala de 0 a 1, donde 1 representa el tamaño completo de la pantalla.
- **Desviación típica de los errores:** Proporcionan información sobre la dispersión de los errores, dando una visión mas detallada de la consistencia y robustez del modelo.

El proceso de entrenamiento se enfoca en minimizar el EMS. Esta métrica penaliza más severamente los errores más grandes, ya que al tener gradientes mayores, se realizan ajustes más significativos en los pesos durante la retropropagación, mejorando así la precisión del modelo. Además, se utiliza una técnica de ajuste del *learning rate* para mejorar la eficiencia del entrenamiento. En esta técnica, el *learning rate* se reduce a la mitad si el error de validación no mejora durante 20 épocas. Esto permite que el modelo se ajuste de manera más precisa a los datos conforme avanza el entrenamiento. Para evitar el sobreajuste, se implementa un mecanismo de *early stopping*, el cual detiene el entrenamiento si el error de validación no mejora después de 30 épocas. Este mecanismo garantiza que el modelo no se ajuste demasiado a los datos de entrenamiento, preservando su capacidad de generalización.

8.1.7 Aproximación inicial

Descripción

La aproximación inicial tiene como objetivo desarrollar una versión beta del sistema de seguimiento ocular, que sirva como base para el desarrollo posterior del proyecto. Para ello, se utiliza la BD inicial, la cual contiene 63.080 filas de datos completos con todas sus características, que son preprocesadas como se ha explicado previamente. Los datos incluyen 39 entradas y 2 salidas, todas normalizadas en el rango entre 0 y 1. Dado que la base de datos está compuesta por información de 23 voluntarios, se realiza un experimento de validación cruzada de 23 pliegues (*23-fold cross-validation leave-one-out*) para evaluar el rendimiento de los modelos.

En esta fase, se experimenta con diversas RNA y SVM, evaluando cuál ofrece el mejor rendimiento. La solución óptima se utilizará como base para desarrollar aproximaciones más avanzadas.

Resultados RNA

Se han entrenado varias **RNA** con diferentes topologías para realizar un estudio inicial y determinar qué características son esenciales para la red neuronal. Basado en las pruebas iniciales, se ha concluido que un tamaño de lote de 6.000 y un *learning rate* de 0.0002 son las configuraciones óptimas. Estas configuraciones fueron seleccionadas porque, durante los entrenamientos, los modelos mostraron una disminución estable y continua del error, en contraste con otras configuraciones que presentaban fluctuaciones y picos significativos.

Además, el número máximo de ciclos de entrenamiento se ha restringido a 250, ya que se ha comprobado que a partir de este número los modelos tienden al sobreajuste. No obstante, la técnica de *early-stopping* detiene el entrenamiento si el error de validación no mejora durante las épocas establecidas.

En todos los modelos utilizados, cada capa densa está seguida de una capa de activación ReLU, lo que introduce no linealidad y permite aprender relaciones complejas. La excepción es la capa de salida, que utiliza una función de activación sigmoide para reducir el rango de la salida entre 0 y 1, ya que las coordenadas de la pantalla están normalizadas a este rango.

Para esta aproximación inicial, se seleccionaron diferentes topologías de manera que sus dimensiones aumentaran con respecto a las anteriores, con el objetivo de obtener una visión preliminar y determinar el camino a seguir. Se muestran los resultados obtenidos para cada una de ellas en la Tabla 8.2.

Modelo	Topología	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
1	[50]	0.022237	0.010132	0.179768	0.047463
2	[90]	0.023406	0.010505	0.184501	0.047065
3	[50 100]	0.020754	0.011287	0.171945	0.044621
4	[300 400 500 400]	0.020916	0.011712	0.172251	0.045879
5	[100 300 400 500 100]	0.024812	0.012513	0.187073	0.050906

Tabla 8.2: Modelos de RNA entrenados en la aproximación inicial.

El modelo con la topología [50, 100] ha demostrado ser el más eficiente en términos de la media de la distancia euclíadiana y la media del **EMS**, mostrando además las menores desviaciones en los errores. Estos resultados sugieren que una topología más compleja no necesariamente mejora el rendimiento. Un enfoque más equilibrado, como el del Modelo 3, puede ofrecer mejores resultados en cuanto a precisión y estabilidad.

Resultados SVM

Aunque anteriormente se ha mencionado el uso de técnicas como *early-stopping* o el ajuste del *learning rate*, estos conceptos no son aplicables a los modelos de **SVM** debido a su naturale-

za distinta. Los **SVM** no se entrenan mediante un proceso iterativo como las redes neuronales, sino que buscan una solución óptima a partir de los datos.

Los **SVM** no pueden manejar dos salidas directamente, lo cual es necesario para resolver este problema específico. Por ello, se ha utilizado un método de Scikit-learn que entrena dos modelos independientes, uno para cada salida. El rendimiento de los **SVM** depende en gran medida de los parámetros seleccionados. Para encontrar la mejor combinación de parámetros, se ha llevado a cabo una búsqueda en cuadrícula (*grid search*), combinando todos los parámetros mostrados en la Tabla 8.3 y evaluando cada modelo mediante *23-fold cross-validation leave-one-out*.

Hiperparámetro	Valores
C	0.01, 0.1, 1, 10, 100
ϵ	0.0001, 0.001, 0.01, 0.1
Kernel	linear, rbf, poly
Grado Kernel	1, 2
γ_{kernel}	scale, auto, 0.1, 1

Tabla 8.3: Hiperparámetros utilizados en la búsqueda en cuadrícula para el modelo SVM.

Es importante destacar que el grado del kernel solo es aplicable cuando se utiliza el kernel polinómico, y que el parámetro γ_{kernel} solo tiene efecto para los kernels *rbf* y polinómico. Por lo tanto, el espacio de búsqueda se reduce a un total de 160 combinaciones. Los resultados obtenidos mediante la búsqueda en cuadrícula muestran que los mejores parámetros para este problema son los presentados en la tabla 8.4, obteniendo los resultados que se pueden observar en la Tabla 8.5.

C	ϵ	Kernel	Grado Kernel	γ_{kernel}
1	0.1	poly	1	1

Tabla 8.4: Mejores parámetros de la búsqueda en cuadrícula.

μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
0.029901	0.014035	0.210091	0.048591

Tabla 8.5: Resultados con los parámetros resultantes de la búsqueda.

Conclusiones

El análisis de los resultados obtenidos en esta aproximación inicial revela que los modelos basados en [RNA](#) superan consistentemente a los [SVM](#), mostrando tasas de error medias inferiores. En particular, se ha identificado que la topología del Modelo 3, con una estructura de [50 100], ofrece un equilibrio óptimo entre complejidad y rendimiento. Aunque las topologías más complejas pueden parecer prometedoras, no siempre garantizan mejores resultados. Por lo tanto, un enfoque equilibrado como el del Modelo 3 resulta más adecuado para el desarrollo del sistema de seguimiento ocular. En base a estos hallazgos, las siguientes aproximaciones seguirán esta dirección, aprovechando la simplicidad y eficacia de las redes neuronales con topologías moderadamente complejas.

Entrenamiento del modelo para la versión beta

Para el desarrollo de una versión beta del software, se entrenó una [RNA](#) utilizando la topología del Modelo 3, detallada anteriormente en la Tabla 8.2, ya que ha sido la que ha aportado mejores resultados.

Para este entrenamiento se seleccionaron aleatoriamente dos voluntarios para formar el conjunto de validación. No se utilizó un conjunto de prueba con el objetivo de emplear la mayor cantidad de datos posible en el entrenamiento del modelo optimizando la generalización del mismo, sin restringir la validación a un solo usuario. Se puede ver la evolución de su entrenamiento en la Figura 8.7.

Durante el proceso de entrenamiento, se guardó una lista con los modelos generados en cada época. Al finalizar, se seleccionó el modelo correspondiente a la época 246, ya que aunque en las últimas épocas las mejoras fueron mínimas debido al bajo *learning rate*, el rendimiento en el conjunto de validación continuó mejorando sin indicios de sobreentrenamiento. Este modelo se utiliza como base para la versión beta del software, mientras se desarrollan las siguientes aproximaciones.

8.1.8 Mejoras del método

Tras la aproximación inicial, se ha observado que los modelos de [RNA](#) tienden a concentrarse en el centro de la pantalla, mostrando una baja precisión en las esquinas, como se puede ver en la Figura 8.8, donde se muestra un mapa de calor donde se representa el error mediante la distancia euclídea media en cada punto del espacio de coordenadas. Para solucionar esto, se ha desarrollado un método que extiende los valores hacia las esquinas, facilitando así el alcance de estas áreas de la pantalla.

Primeramente, se consideró utilizar la Campana de Gauss de manera inversa. Esto daría más peso a los valores cercanos a los extremos del rango [0, 1], tanto para el eje vertical como

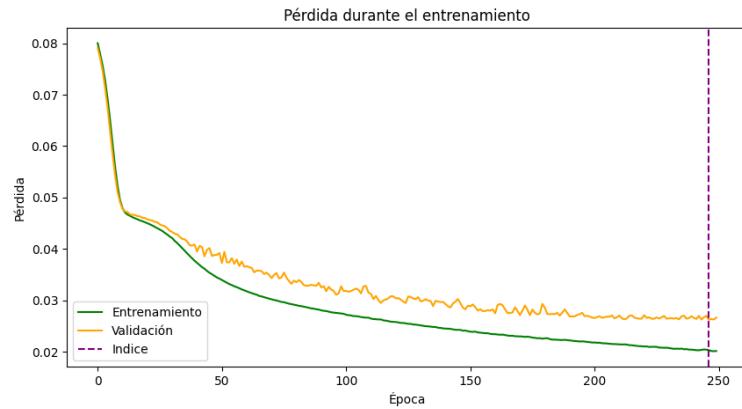


Figura 8.7: Gráfica del entrenamiento del modelo 3 para la beta del software.

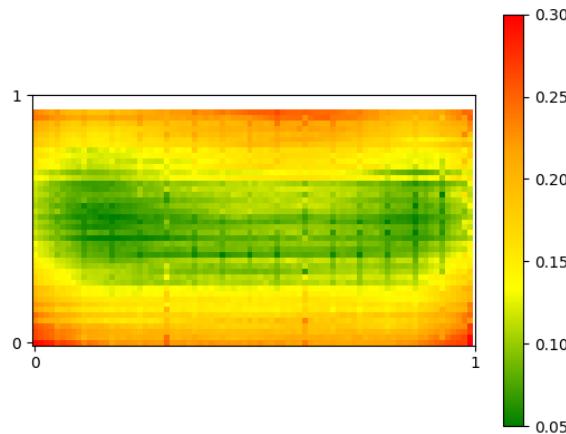


Figura 8.8: Mapa de calor de la distancia euclidiana.

para el horizontal, haciendo que estos valores sean más significativos cuanto más cerca estén del extremo. Sin embargo, este método presentaba una falta de control para personalizar cada esquina individualmente. Por lo tanto, se decidió desarrollar un método que permite ajustar la ponderación de manera más precisa y controlada.

Para facilitar la explicación y comprensión, primero se abordará en una sola dimensión y luego en su totalidad. Se explicará la función considerando únicamente el eje vertical y posteriormente se integrarán los conceptos. Partiendo de la necesidad de ponderar un valor lineal entre 0 y 1, que representa los valores desde la parte inferior hasta la superior de la pantalla, el objetivo es estirar los valores en los extremos mientras se mantiene la zona central sin modificar, aunque permitiendo el ajuste de esta mediante un punto modificador central. Se

busca una función polinómica de cuarto grado ajustable mediante los siguientes cinco puntos preestablecidos:

- **Límite Bajo (LB):** Desde menos infinito hasta este punto, la función debe devolver 0. Esto establece el límite con un valor ligeramente superior al original, desplazando a 0 todo lo que esté por debajo de este valor.
- **Comienzo Zona Afectada (CZA):** Desde el punto anterior, la función debe aumentar su valor hasta llegar a este punto. Aquí comienza la zona no afectada, que corresponde al centro de la pantalla, por lo que los valores deben ser lo más parecidos posible a una función lineal con pendiente 1. Este valor se calcula automáticamente, siendo la mitad entre el límite bajo y el centro.
- **Centro (C):** Establece el centro, permitiendo desplazar la zona no afectada, ajustando la zona central de la función.
- **Fin Zona Afectada (FZA):** Donde termina la zona no afectada, a partir de este punto, la función debe aumentar su valor hasta el límite alto. También es calculado automáticamente, siendo la mitad entre el centro y el límite superior.
- **Límite Alto (LA):** A partir de este punto, todos los valores deben devolver 1, estableciendo así el límite superior por debajo del original y desplazando al extremo todo lo que se encuentre por encima de este.

Para personalizar la función, se deben modificar los límites extremos y el punto central, ya que los límites de la zona no afectada se calculan automáticamente. Luego, se crea la función polinómica, estableciendo los puntos con los valores necesarios. El valor se pasa a través de esta función y se recorta al rango [0, 1]. La implementación de la función se muestra a continuación y posteriormente, se añade un gráfico en la Figura 8.9 para observar su funcionamiento.

```
1  def ponderar1d(x, LimiteBajo, LimiteAlto, Centro):
2      # Calcular los límites de la zona no afectada
3      ComienzoZonaNoAfectada = LimiteBajo + (Centro - LimiteBajo)
4          / 2
5      FinZonaNoAfectada = LimiteAlto - (LimiteAlto - Centro) / 2
6
7      # Calcular los puntos para la función polinómica
8      puntos = np.array([LimiteBajo, ComienzoZonaNoAfectada,
9      Centro, FinZonaNoAfectada, LimiteAlto])
10     valores = np.array([0, ComienzoZonaNoAfectada, 0.5,
11     FinZonaNoAfectada, 1])
```

```
10      # Crear la función polinómica
11      polinomio = np.poly1d(np.polyfit(puntos, valores, 4))
12
13      # Calcular el valor ponderado
14      return np.clip(polinomio(x), 0, 1)
```

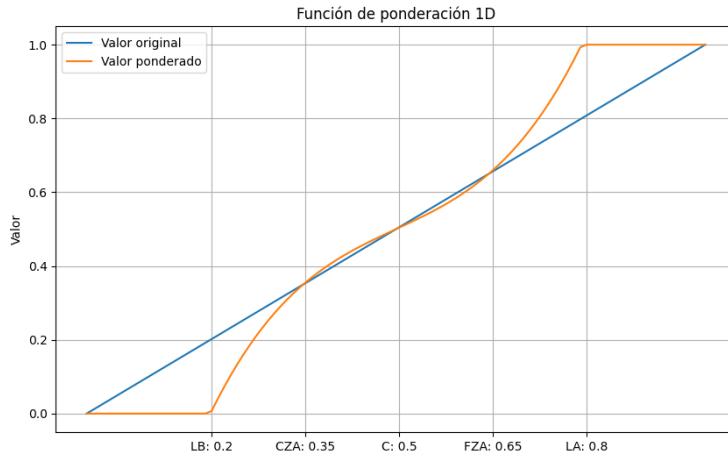


Figura 8.9: Función de ponderación de una dimensión.

Una vez comprendido el concepto básico en el que se basa la ponderación realizada, se puede profundizar en cómo funciona este proceso para las dos dimensiones, tanto horizontal como verticalmente y para las cuatro esquinas a la vez.

Primero, se definen cinco puntos (X, Y): cuatro en las esquinas y uno en el centro. Los puntos de las esquinas establecen los límites de la función, a la vez que el punto central se utiliza para ajustar la zona céntrica.

Una vez definidos estos puntos, se pueden realizar dos ponderaciones unidimensionales para cada esquina. En la ponderación horizontal, se emplean los valores X de los puntos y en la vertical se utilizan los valores Y. Los valores del punto central C utilizados para la función de ponderación son los del punto central definido, mientras que la relación de los valores límite utilizados para ponderar cada esquina se detalla a continuación en la Tabla 8.6:

Esquinas	Ponderación X		Ponderación Y	
	Límite B.	Límite A.	Límite B.	Límite A.
Inf. Izq.	Pt. Inf. Izq.	Pt. Inf. Der.	Pt. Inf. Izq.	Pt. Sup. Izq.
Inf. Der.	Pt. Inf. Izq.	Pt. Inf. Der.	Pt. Inf. Der.	Pt. Sup. Der.
Sup. Izq.	Pt. Sup. Izq.	Pt. Sup. Der.	Pt. Inf. Izq.	Pt. Sup. Izq.
Sup. Der.	Pt. Sup. Izq.	Pt. Sup. Der.	Pt. Inf. Der.	Pt. Sup. Der.

Tabla 8.6: Relaciones entre las esquinas y los límites bajos y altos en los ejes X e Y.

La ponderación se realiza para los dos ejes de las cuatro esquinas a la vez, después, se realiza la media ponderada entre ellas dándole más importancia a la que se encuentra más cerca. Por lo tanto, se calcula la distancia euclídea desde el punto a ponderar hasta los puntos establecidos de las esquinas. Posteriormente, se calculan los pesos de ponderación, de manera que sean inversamente proporcionales a las distancias, es decir, cuanto más cerca se encuentra el punto de una esquina, mayor es el peso de esa esquina. Después se normalizan los pesos mediante la técnica Min-Max, la cual los ajusta para que estén en el rango $[0, 1]$ y se calcula la ponderación final combinando las ponderaciones de cada esquina según estos pesos normalizados. El resultado es una función de ponderación que cumple con el objetivo preestablecido y proporciona una experiencia fluida en toda la pantalla.

En la Figura 8.10 se muestra un ejemplo del funcionamiento de esta función, para el que se han representado 10,000 puntos distribuidos por el espacio de coordenadas de la pantalla y se han ponderado utilizando esta función. Cada flecha indica el punto inicial y el punto resultante, con colores que varían en función de la distancia desplazada para cada punto. Para una mejor visualización del gráfico, se ha modificado la función eliminando el límite de los valores resultantes al rango $[0, 1]$ y se ha establecido una reducción del 10% de la pantalla para todas las esquinas.

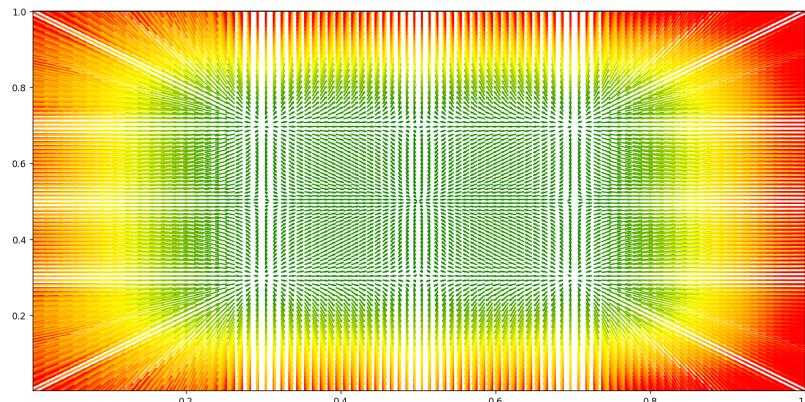


Figura 8.10: Función de ponderación en el espacio de coordenadas.

8.1.9 Primera aproximación: Perceptrón multicapa

Descripción

Cuando se desarrolla esta aproximación, la BD final explicada en la Sección 8.1.4 ya se ha creado, por lo que es la utilizada en todas las aproximaciones a partir de este punto para asegurar una correcta comparativa entre los modelos analizados. Esta base de datos consta de 33.352 registros de datos y fotografías, aunque para esta aproximación solamente se hace uso de los datos numéricos. Dado que contiene los registros de 15 voluntarios, se desarrolla un experimento de validación cruzada de 15 pliegues, *15-fold cross-validation leave-one-out*.

El objetivo de esta aproximación es encontrar el modelo de RNA que cumpla con las necesidades requeridas en términos de precisión y probar diferentes métodos de preprocesamiento de los datos para verificar si se obtienen mejoras al modificar la forma en que estos se preprocesan.

Resultados

Debido al cambio de la BD para la realización de los experimentos, no se pueden comparar los resultados de nuevos modelos con los obtenidos en la anterior aproximación, por lo que se han vuelto a evaluar los modelos probados en la aproximación inicial pero con la BD final, manteniendo los valores del *learning rate* y de las épocas máximas. Los resultados se pueden observar en la Tabla 8.7.

Modelo	Topología	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
1	[50]	0.020503	0.008308	0.177835	0.177835
2	[90]	0.019898	0.008445	0.174896	0.038224
3	[50 100]	0.018698	0.008145	0.168518	0.038051
4	[300 400 500 400]	0.020138	0.007714	0.173231	0.034264
5	[100 300 400 500 100]	0.027064	0.011137	0.201583	0.045816

Tabla 8.7: Modelos de RNA de la aproximación inicial entrenados con la BD final.

Tras analizar los resultados, se observa que los valores de error son menores en comparación con la aproximación inicial. Esto sugiere que, aunque la base de datos inicial cuenta con más registros, su mayor diversidad hace que el problema sea más complejo de resolver.

Los resultados obtenidos confirman las conclusiones de la aproximación inicial, mostrando que una topología más equilibrada, como la del Modelo 3, produce mejores resultados. Por lo tanto, se ha decidido realizar un nuevo estudio utilizando modelos con una topología similar a la del Modelo 3, los cuales incluyen una capa de activación ReLU tras cada capa densa y una capa de activación sigmoide en la salida.

Modelo	Topología	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
6	[70 100]	0.016571	0.007993	0.157550	0.041097
7	[100 200]	0.016892	0.00817	0.158907	0.041273
8	[100 100 100]	0.016479	0.008264	0.156414	0.040202
9	[80 100 80]	0.016311	0.007269	0.156911	0.036545
10	[150]	0.017785	0.009189	0.163745	0.043026

Tabla 8.8: Modelos de topología similar al Modelo 3 entrenados con la BD final.

Los resultados obtenidos en este nuevo estudio, visibles en la Tabla 8.8, demuestran que el Modelo 9 con la topología [80 100 80] es el más eficiente y consistente, ya que ha obtenido los mejores resultados en base a la media de la distancia euclíadiana y la media del EMS. Además, su consistencia se confirma con las desviaciones de los errores más bajas obtenidas hasta el momento.

Tras identificar el modelo con el mejor rendimiento entre los evaluados, se procede a realizar un estudio variando el pre-procesado de los datos. El objetivo es probar diferentes formas de pre-procesar los datos para determinar si estos cambios pueden mejorar los resultados obtenidos. Para ello, se implementan y evalúan tres pre-procesados diferentes utilizando el Modelo 9, que ha demostrado ser el más eficiente y consistente en los experimentos previos.

Pre-procesado A

Al calcular la media de las distancias de los ojos antes de aplicar la normalización Min-Max, se puede reducir la dimensionalidad del problema, evitando información extra y aspirando a mejorar los resultados. Esto se basa en el hecho de que ambos ojos siempre apuntan al mismo punto de la pantalla simultáneamente.

Originalmente, hay 32 índices que representan las distancias de los ojos. Estos se reducen a 16 al calcular la media de las distancias correspondientes de ambos ojos. Es decir, para cada par de índices que representan la misma característica en ambos ojos, se calcula la media y se utiliza este valor combinado, lo que da como resultado un total de 23 entradas.

En la Tabla 8.9 se pueden ver los resultados que se han obtenido tras la evaluación del Modelo 9 con este conjunto de datos. Estos revelan un aumento del error, lo que demuestra que tener la información de cada ojo por separado aporta más información que la media de los dos, siendo mejor tener una mayor dimensionalidad en la entrada de la RNA, por lo que se rechaza el pre-procesado presentado.

μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
0.018484	0.007496	0.169487	0.032330

Tabla 8.9: Resultados del Modelo 9 con el pre-procesado A.

Pre-procesado B

Dado que la reducción de la dimensionalidad mediante la media de los dos ojos no ha dado buenos resultados, se mantienen los primeros 31 índices como en el pre-procesado inicial. En este caso, se evalúa si dar más importancia a la orientación y posición de la cabeza mejora los resultados.

Al reducir el rango de los valores de orientación y posición, estos muestran mayores variaciones con los mismos movimientos. Aunque la intención es mantener la cabeza estable en el centro de la pantalla, se ha considerado este enfoque debido a los movimientos involuntarios que ocurren incluso al intentar mantener la cabeza quieta.

Para ello, se hace uso de la normalización Min-Max, estableciendo el valor límite inferior como 0.3 y el superior como 0.7 para los índices que representan los valores de la orientación y la posición. Esta normalización se realiza de la siguiente manera:

$$\text{Valor Normalizado} = \frac{\text{Valor} - 0.3}{0.7 - 0.3} \quad (8.2)$$

Dado que puede existir el caso en el que los valores sobrepasen los límites establecidos, lo que resultaría en un valor normalizado negativo o superior a 1, se ajustan todos los valores que superen estos límites a 0 o a 1, respectivamente. En la Figura 8.11, se representa el rango que cubren los datos con esta normalización en comparación con el rango que cubren los valores originales.

Tras la evaluación del Modelo 9 utilizando este pre-procesado, se obtienen los resultados que se pueden apreciar en la Tabla 8.10 donde se puede observar un aumento del error con respecto a los resultados obtenidos con el pre-procesado inicial, por lo que también se descarta este método.

μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
0.017335	0.008026	0.161310	0.037795

Tabla 8.10: Resultados del Modelo 9 con el pre-procesado B.

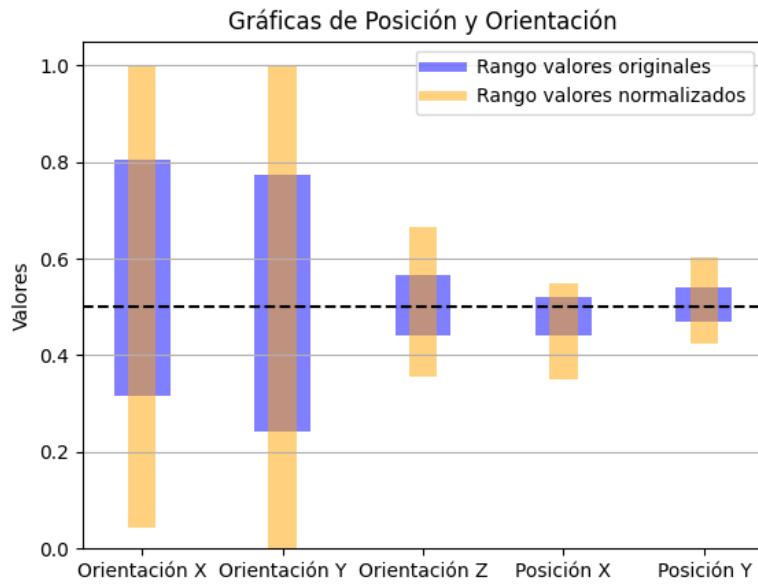


Figura 8.11: Rangos de los valores de la BD originales y normalizados.

Pre-procesado C

Inicialmente, los valores del [EAR](#), tanto el actual como el umbral, se agregaron a la [BD](#) para eliminar los registros con el ojo cerrado y evitar introducir datos sucios en los entrenamientos. Posteriormente, en todos los estudios se incluyeron estos datos en los entrenamientos, sin embargo, debido a la falta de comprensión de la relación entre estos valores y la posición de la mirada del usuario en la pantalla, se decidió basar este pre-procesado en el original, excluyendo los índices del [EAR](#), lo que resulta en un total de 37 entradas.

Los resultados obtenidos con este pre-procesado se pueden ver en la Tabla 8.11, estos demuestran que los datos sobre el [EAR](#) sí aportan información valiosa, ya que se observa un aumento del error al excluir estos valores. Debido a esto, se concluye que la información del [EAR](#) es beneficiosa para mejorar la precisión del modelo, por lo que este método también se descarta.

μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
0.020622	0.007701	0.175699	0.031552

Tabla 8.11: Resultados del Modelo 9 con el pre-procesado C.

Conclusiones

Tras los diferentes estudios realizados en esta aproximación, se concluye que el Modelo 9, con la topología [80 100 80], es el que mejor aprende la relación entre los datos, ofreciendo las tasas de error más bajas y desviaciones equivalentes. Esto indica un aprendizaje robusto y consistente por parte del modelo. Por otro lado, aunque se exploraron diversas alternativas de pre-procesado de los datos, se llegó a la conclusión de que el pre-procesado inicial es el que mejor representa la información, ya que produjo los mejores resultados.

8.1.10 Segunda aproximación: Redes neuronales convolucionales

Descripción

El objetivo de esta aproximación es emplear el aprendizaje profundo y las redes neuronales convolucionales, dado que están especialmente diseñadas para el reconocimiento de imágenes. Estas redes suelen ofrecer excelentes resultados en tareas de reconocimiento visual, lo que puede ser muy útil para identificar la orientación de los ojos y predecir la ubicación de la mirada en la pantalla.

En cuanto a las imágenes utilizadas, se ha desarrollado un método para recortar automáticamente la zona de interés de los frames, simplificando el análisis a la zona requerida en lugar de usar la imagen completa. Esto evita introducir información irrelevante del frame completo en el modelo, lo que añadiría ruido a la medición.

Utilizando los puntos de las pupilas obtenidos con *Mediapipe*, se calcula un rectángulo que enmarca los ojos, añadiendo un margen tanto vertical como horizontalmente. Luego, se ajusta este rectángulo para que coincida con la relación de aspecto deseada, añadiendo píxeles a los lados o arriba y abajo según sea necesario. Una vez ajustado el rectángulo a la relación de aspecto necesaria, se recorta esta área del frame original. Finalmente, el rectángulo recortado se ajusta a las dimensiones deseadas, manteniendo la relación de aspecto.

Para ello, este método acepta como argumentos de entrada el ratio deseado, el margen horizontal que se desea añadir desde los puntos de las pupilas y los márgenes superior e inferior, para así tener un método flexible para poder realizar diferentes pruebas con diferentes características y deducir cuál es el cuadro más representativo en el rostro de una persona para la detección de la mirada en él. Este proceso asegura que todos los frames tengan el mismo tamaño, respetando la misma relación de aspecto, centrados en los ojos y sin modificar las proporciones de la imagen.

Para esta aproximación, se utilizan las configuraciones descritas en la Tabla 8.12. La primera configuración abarca únicamente el rectángulo de los ojos, la segunda incluye las cejas y la tercera abarca tanto las cejas como parte de la nariz. En la Figura 8.12 se puede apreciar un ejemplo del funcionamiento de este método con cada una de estas configuraciones, en el

orden mencionado.

Configuración	Tamaño imagen	Margen horizontal	Margen superior	Margen inferior
1	200x50	15px	15px	15px
2	200x70	15px	35px	15px
3	210x120	20px	35px	55px

Tabla 8.12: Configuraciones del recorte automático utilizadas.



Figura 8.12: Resultados del método de recorte automático.

Todas las fotografías de la [BD](#) fueron procesadas con las tres configuraciones y transformadas a escala de grises. Ya que el color de la imagen no influye en la posición de los ojos, esto debería aumentar el rendimiento reduciendo la dimensionalidad. Con las imágenes procesadas se crearon tres archivos `.pt`, cada uno para un tamaño de imagen, logrando una mejor organización y acceso más rápido a los datos durante el entrenamiento.

Los modelos utilizados para esta aproximación se detallan en el apéndice [E](#). A lo largo del desarrollo de esta aproximación, los modelos se identificarán con el número correspondiente. Por otra parte, al igual que en la aproximación anterior, se realiza un experimento de validación cruzada de 15 pliegues, *15-fold cross-validation leave-one-out*.

Resultados

Para el primer estudio, se comenzó utilizando las imágenes de mayor tamaño, aplicando la configuración 3 mencionada anteriormente, de modo que se partía del punto con mayor cantidad de información. Se evaluaron cuatro modelos con un *learning rate* de 0.002 y un tamaño de lote de 1000. Los resultados de esta evaluación se presentan en la Tabla [8.13](#).

Modelo	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
E.1.1	0.080904	0.007083	0.377027	0.019934
E.1.2	0.109332	0.041784	0.418969	0.064975
E.1.3	0.080557	0.005956	0.376285	0.017284
E.1.4	0.080494	0.007062	0.376062	0.019972

Tabla 8.13: Modelos CNN evaluados en las primeras observaciones.

Como se puede visualizar en la Tabla (8.13), los resultados de los modelos no han alcanzado un óptimo global, quedándose la mayoría en óptimos locales y raramente bajando de un EMS de 0.08. Tras varias pruebas, se descubrió que los modelos con una capa de activación sigmoide en la última capa no lograban converger adecuadamente y que la inicialización de pesos mejoraba la convergencia.

Estos dos ajustes, junto con la reducción del tamaño del lote a 100, permitieron unos entrenamientos más efectivos durante diferentes pruebas, evitando el estancamiento en óptimos locales. Por ello se realizó un nuevo estudio con cuatro nuevos modelos, manteniendo características de los anteriores pero añadiendo estas mejoras.

Modelo	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
E.1.5	0.033889	0.022414	0.215833	0.085742
E.1.6	0.06257	0.009087	0.323710	0.026975
E.1.7	0.030716	0.022826	0.202843	0.085179
E.1.8	0.030300	0.022239	0.202727	0.086479

Tabla 8.14: Modelos CNN evaluados en las segundas observaciones.

Analizando los resultados del nuevo estudio, visibles en la Tabla 8.14, se puede apreciar cómo los modelos E.1.7 y E.1.8 han ofrecido mejores resultados, siendo el modelo E.1.8 el que aporta una tasa de error medio más baja. En base a los resultados obtenidos, se realiza un nuevo estudio con estos dos modelos, utilizando las imágenes recortadas con las otras dos configuraciones restantes, manteniendo las características que han permitido una convergencia adecuada. Cabe destacar que para utilizar tamaños de imágenes diferentes, es necesario adaptar los modelos, modificando las dimensiones de la capa densa posterior a la de aplanamiento. Los resultados se representan en la Tabla 8.15.

Modelo	Configuración	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
E.1.7	1	0.021238	0.012093	0.169621	0.057109
E.1.8	1	0.028247	0.020473	0.193669	0.082398
E.1.7	2	0.034485	0.023500	0.213987	0.093562
E.1.8	2	0.027377	0.018009	0.191518	0.078174

Tabla 8.15: Modelos CNN evaluados en las tercera observaciones.

Aunque ningún modelo de los presentados en la Tabla 8.15 ha superado los resultados de la primera aproximación, se destaca la baja tasa de error del modelo E.1.7 con la configuración 1 en cuanto al EMS medio y la distancia euclídea media, en comparación con el resto de modelos y configuraciones, lo que se tendrá en cuenta para la siguiente aproximación.

Ya que los mejores resultados se han obtenido con la configuración 1 para el recorte automático de las imágenes, esta ha sido la escogida para un nuevo estudio, en el que se evalúan dos modelos preentrenados: ResNet-18 y ResNet-34. Estos ya han sido entrenados en grandes conjuntos de datos de reconocimiento de imágenes, por lo que deberían presentar un buen rendimiento para la tarea que se plantea resolver.

Primero hay que adaptarlos para el problema en cuestión. Esto se realiza modificando la primera capa convolucional para aceptar imágenes en escala de grises y ajustando la capa totalmente conectada final para que se adapte a las salidas necesarias para este problema, en este caso dos, las coordenadas de la mirada en la pantalla.

Los resultados del *15-fold cross-validation one-leave-out* son expuestos en la Tabla 8.16. Cabe destacar que para los entrenamientos de estos dos modelos se ha establecido un tamaño de lote de 100 y una tasa de aprendizaje de 0.0005. Estos parámetros han sido escogidos en base a diferentes pruebas realizadas hasta conseguir una evolución del entrenamiento suave y constante.

Modelo	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
Resnet-18	0.022426	0.017142	0.171793	0.074130
Resnet-34	0.020974	0.014823	0.169894	0.066299

Tabla 8.16: Modelos ResNet evaluados con las imágenes de la configuración 1.

Analizando los resultados de la Tabla 8.16, se puede percibir que el modelo de ResNet más profundo evaluado, ResNet-34, ha conseguido los mejores resultados, ofreciendo unas tasas de error más bajas que ResNet-18 y muy semejantes a las del modelo E.1.7.

Conclusiones

Después de analizar todos los resultados obtenidos de los diferentes estudios realizados con modelos basados en [CNN](#), se destaca la capacidad del modelo [E.1.7](#) y del ResNet-34 para resolver la tarea planteada, ya que ofrecen tasas de error medias bastante reducidas. Sin embargo, a pesar de su rendimiento aceptable, estos modelos no logran igualar los resultados obtenidos en la primera aproximación. Por lo tanto, el modelo resultante de la aproximación 1 se mantiene, por el momento, como el candidato más adecuado para ser el modelo final.

8.1.11 Tercera aproximación: Redes neuronales híbridas

Descripción

Esta aproximación surge de la inquietud por explorar los posibles resultados al combinar los mejores modelos de cada aproximación. El objetivo de esta combinación es observar si se puede aprovechar las fortalezas de ambos enfoques y mejorar el rendimiento general del sistema. Para ello, se emplean redes neuronales híbridas para optimizar los resultados, combinando diferentes arquitecturas de redes neuronales. Este enfoque permite que los modelos aprendan características complementarias, lo que generalmente se traduce en mejores resultados [21]. Los modelos utilizados en esta aproximación se detallan en el Apéndice [E](#), y a lo largo del desarrollo se identifican con su respectivo número.

La entrada de estos modelos combina las características de la primera aproximación con las imágenes utilizadas en la segunda. Internamente, los modelos están formados por las arquitecturas que ofrecieron los mejores resultados en las aproximaciones anteriores. Los modelos ResNet no se utilizaron debido a motivos de eficiencia, ya que no ofrecían una mejora significativa en comparación con otros modelos de [CNN](#), pero sí incrementaban notablemente los tiempos de entrenamiento.

Es importante destacar que los mejores modelos de las aproximaciones previas fueron el Modelo 9 de la primera y el modelo [E.1.7](#) de la segunda. Por lo tanto, las topologías de las redes híbridas siguen estructuras similares, adaptando la topología o eliminando la capa de salida para conectarlas entre ellas.

Resultados

Para realizar este estudio, tras las pruebas iniciales, se determinó que el *learning rate* más adecuado es 0.0001, con un número máximo de 250 épocas. Se parte de los cinco primeros modelos, evaluando cuáles probar posteriormente en función de los resultados de las primeras observaciones.

Modelo	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
E.2.1	0.023476	0.016806	0.179547	0.071025
E.2.2	0.023327	0.016459	0.179298	0.070489
E.2.3	0.022146	0.013927	0.176091	0.060301
E.2.4	0.020626	0.012950	0.170318	0.056190
E.2.5	0.018364	0.010418	0.161811	0.050218

Tabla 8.17: Modelos híbridos evaluados en las primeras observaciones.

Tras analizar los resultados obtenidos en las primeras observaciones, presentados en la Tabla 8.17, no se ha observado una mejora significativa en comparación con los modelos de la primera aproximación. Sin embargo, se puede notar que los modelos con una estructura relativamente simple, pero no excesivamente simplificada, tienden a ofrecer mejores resultados. Un ejemplo de ello es el modelo E.2.5, ya que las estructuras demasiado pequeñas o demasiado complejas no consiguen una tasa de error media baja. Estas observaciones orientan hacia la realización de un nuevo estudio con modelos similares a aquellos que han mostrado un mejor rendimiento.

Modelo	μ_{MSE}	σ_{MSE}	$\mu_{Dist.Euc.}$	$\sigma_{Dist.Euc.}$
E.2.6	0.020408	0.013430	0.168435	0.059243
E.2.7	0.019643	0.011911	0.166694	0.054712
E.2.8	0.018010	0.010876	0.159067	0.051334
E.2.9	0.018769	0.010475	0.163221	0.052096
E.2.10	0.030235	0.014588	0.206019	0.057817

Tabla 8.18: Modelos híbridos evaluados en las segundas observaciones.

Las segundas observaciones realizadas muestran una mejora en los resultados con respecto a las primeras, como se aprecia en la Tabla 8.18. El modelo que ha presentado la tasa de error más baja ha sido el Modelo E.2.8, destacando sobre los demás. Esto confirma que, para este problema, una estructura que no sea excesivamente compleja ofrece mejores resultados. Sin embargo, ninguno de los modelos ha superado los resultados obtenidos en la primera aproximación, por lo que no serán considerados para el modelo final que utilizará el software.

Conclusiones

Los modelos híbridos evaluados en esta aproximación han mostrado que una combinación de características y arquitecturas no siempre garantiza una mejora significativa. Tras completar la tercera aproximación, se puede confirmar que los modelos de [RNA](#) son los que ofrecen los mejores resultados, presentando tasas de error medio más bajas y desviaciones menores en comparación con el resto de modelos probados en las diferentes aproximaciones.

8.1.12 Entrenamiento final

Después de realizar las diferentes aproximaciones, se ha decidido utilizar el modelo de [RNA](#) con la topología [80 100 80] y el pre-procesado inicial de los datos, ya que esta configuración ha proporcionado los mejores resultados. Este modelo no utiliza imágenes, lo que ofrece la posibilidad de unir las dos [BD](#).

Al combinar las dos bases de datos, se pueden compensar las diferencias observadas en la primera aproximación (Tabla 8.7), donde se obtenían diferentes resultados con el mismo modelo para las diferentes bases de datos. Entrenar los modelos con ambas bases de datos permite fusionar estas diferencias, proporcionando una mayor generalización de los datos al contar con un conjunto más amplio y diverso de personas, aumentando el rendimiento final del modelo.

La unión de las dos [BD](#) resulta en un total de 96,432 registros provenientes de 38 voluntarios diferentes. Esta cifra representa una excelente base para el entrenamiento, ya que contar con un volumen tan amplio de datos de diversas personas mejora la capacidad de generalización del modelo, lo que se traduce en un rendimiento más robusto y fiable.

Para realizar el entrenamiento, al igual que se hizo para el modelo entrenado tras la aproximación inicial, se separan los datos de dos voluntarios para formar el conjunto de validación. Esto tiene como objetivo obtener un conjunto más genérico, no basado en un solo usuario, sobre el cual evaluar la parada del entrenamiento. No se emplea un conjunto de prueba, con el fin de maximizar la cantidad de datos disponibles para el entrenamiento, buscando así una mejor generalización del modelo final. El entrenamiento se lleva a cabo utilizando los parámetros establecidos en la primera aproximación: un *learning rate* de 0.0002 y un límite de 250 épocas.

La gráfica del entrenamiento del modelo final se presenta en la Figura 8.13. En ella, se puede observar que aproximadamente a partir de la época 200, el modelo comienza a mostrar signos de sobreajuste. Sin embargo, gracias a la técnica de *early stopping*, el entrenamiento se detiene alrededor de la época 230. Posteriormente, se selecciona el modelo correspondiente a la época 205, antes de que inicie el sobreajuste, para obtener un modelo final sin los efectos del sobreentrenamiento en las últimas épocas. Este modelo es el que se utiliza en el software

final.

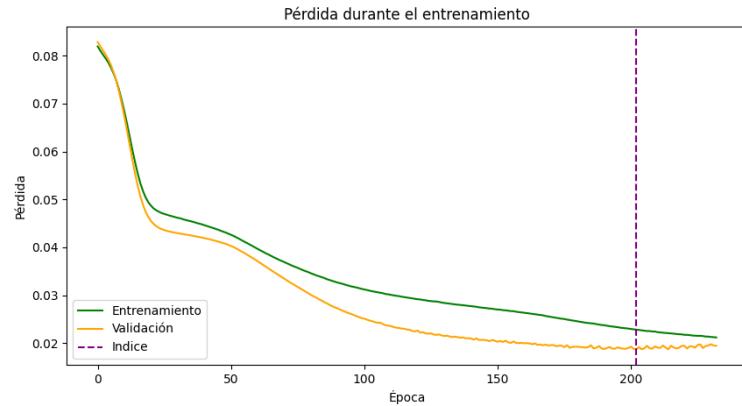


Figura 8.13: Gráfica del entrenamiento del modelo finalmente utilizado en el software.

8.2 Implementación de la aplicación

El desarrollo de la aplicación “ComunicELA” se ha llevado a cabo utilizando modernas técnicas de ingeniería de software para asegurar que el producto final sea robusto, eficiente y fácil de mantener. Un aspecto clave de este proceso ha sido la elección de un patrón de diseño estructural adecuado.

8.2.1 Base de la aplicación

En base al patrón de diseño utilizado, se ha escogido el [Model View Presenter \(MVP\)](#) ya que permite crear un código fácil de mantener gracias a una clara separación de responsabilidades [22].

- **Modelo:** Define los datos y la lógica de negocio que se operan en la interfaz de usuario. Gestiona el comportamiento de la aplicación y responde a las solicitudes de información y a las instrucciones para cambiar su estado.
- **Vista:** Es una interfaz pasiva que muestra los datos y envía comandos del usuario al presentador para actuar sobre ellos. Responsable de renderizar los datos, la vista delega la manipulación de las acciones al presentador.
- **Presentador:** Actúa como el intermediario entre el modelo y la vista. Manipula tanto el modelo como la vista, formateando los datos del modelo para mostrarlos en la vista y respondiendo a las acciones del usuario para cambiar los datos como sea necesario.

8.2.2 Tutorial de la aplicación

Para facilitar el primer contacto de los usuarios con la aplicación “ComunicELA”, se ha implementado un tutorial guiado que mejora significativamente la comprensión del funcionamiento del software y de las opciones disponibles. Este tutorial utiliza PopUp explicativos que se muestran en secuencia, siguiendo el orden de las acciones que el usuario debe realizar.

Cada PopUp se posiciona junto al elemento de la interfaz que describe, como los botones y menús, proporcionando una orientación directa y relevante. Esta estrategia ayuda a los usuarios a familiarizarse rápidamente con la aplicación, permitiéndoles aprender de manera intuitiva cómo navegar y utilizar las diferentes funcionalidades desde el inicio.

Como se ilustra en la Figura 8.14, el último mensaje del tutorial ofrece la opción de no volver a mostrarlo. Al seleccionar “Continuar” y confirmar esta elección, la preferencia del usuario se guarda en el archivo config.json. En futuros accesos, el sistema recordará la configuración elegida y mostrará o no el tutorial según la preferencia del usuario.



Figura 8.14: Primer mensaje del tutorial.

8.2.3 Configuraciones de la pantalla principal

Idioma

Desde la pantalla principal, los usuarios pueden cambiar el idioma de la interfaz mediante un ícono ubicado en la esquina superior derecha, como se muestra en la figura 8.15. Este ícono representa la bandera del idioma actual con el nombre del idioma escrito debajo. Al seleccionar este ícono, el idioma de la aplicación cambia inmediatamente. La selección del nuevo idioma se almacena automáticamente en el archivo de configuración, asegurando que el idioma elegido se mantenga para futuras sesiones.

Para facilitar la implementación de múltiples idiomas y permitir futuras extensiones de idiomas, se ha establecido una estructura organizada en la carpeta strings. Dentro de esta carpeta, se almacenan archivos en formato .json, los cuales contienen todas las cadenas de texto utilizadas en la aplicación.



Figura 8.15: Botón de cambio de idioma.

Menú flotante de configuración

Al abrir la configuración desde la pantalla principal, aparece un menú flotante que oscurece el fondo y muestra varias opciones de personalización. Este menú, que se puede ver en la Figura 8.16, permite seleccionar la cámara y la voz a utilizar, cambiar el estado del conjugador automático de frases y volver a mostrar el tutorial.

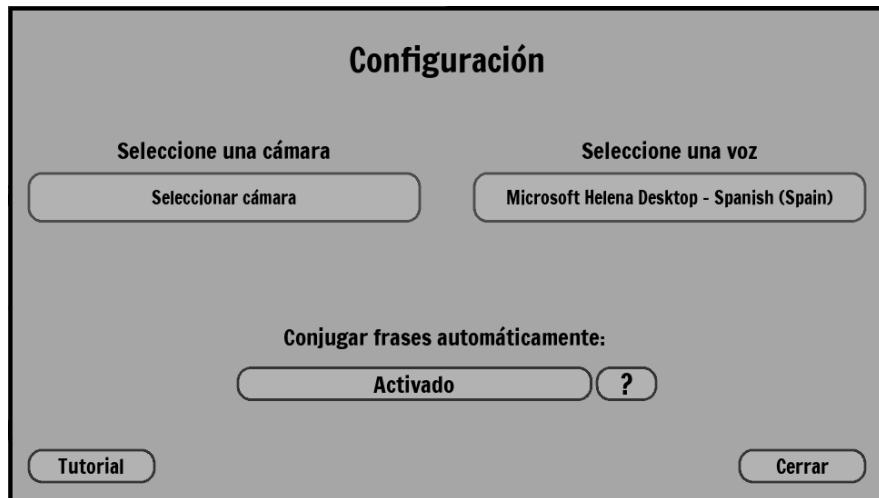


Figura 8.16: Menú flotante de configuración.

Cámara

En el contexto del control de la cámara en nuestro software, se ha aplicado el patrón Adapter [23] mediante la creación de la clase `Camara`. Esta clase adapta la interfaz proporcionada por OpenCV, que es compleja y versátil, a una más simple y adecuada para las necesidades específicas de nuestra aplicación. La clase `Camara` opera la cámara en un hilo separado para evitar interferencias con la interfaz gráfica del usuario. El selector de cámaras, como se ilustra en la Figura 8.17, permite al usuario no solo ver las cámaras disponibles sino también actualizar la lista en caso de nuevas conexiones mientras la aplicación está activa. Al seleccionar una cámara, la aplicación inicia esta cámara automáticamente y guarda su índice en el archivo

`config.json`. Esto asegura que en futuros inicios de la aplicación, la misma cámara pueda ser activada automáticamente sin requerir configuración adicional por parte del usuario.

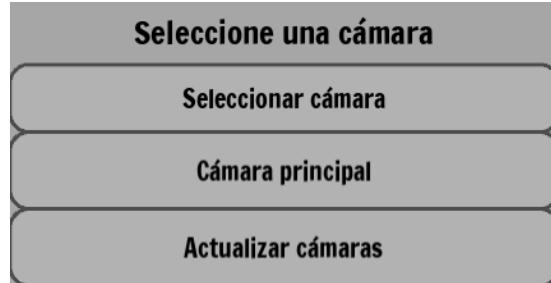


Figura 8.17: Selector de cámaras disponibles.

Voz

Para la síntesis de voz se utiliza ISPVoice [24], ya que permite reproducir texto mediante Text-to-Speech. Esta interfaz de Windows emplea las voces del Narrador de Windows, las cuales se pueden instalar desde la configuración del sistema operativo. Estas voces se presentan en el selector de voz, el cual tiene el mismo aspecto que el mencionado anteriormente para las cámaras, permitiendo al usuario elegir la preferida. Según AGAELA, es un factor muy relevante, ya que los usuarios no se sienten cómodos si utilizan una voz que no concuerda con su género. Una vez seleccionada la voz, esta se guarda en el archivo de `config.json` para mantenerla en futuras sesiones.

Conjugar frases automáticamente

Aunque los tableros comunicativos aún no se han descrito en detalle, se planea que contengan palabras en infinitivo. Sin embargo, al construir frases y reproducirlas en voz alta, estas pueden sonar incompletas, como por ejemplo, “Yo querer comer”, lo que podría resultar incómodo para el usuario.

Por lo tanto, se ha implementado un conjugador automático de frases, utilizando el modelo de lenguaje Google Gemini, específicamente el modelo Flash 1.5, que es el más rápido disponible y ofrece un uso gratuito dentro de ciertos límites [25].

Para utilizar esta función, el usuario debe seguir la guía de instalación proporcionada con el software para agregar una variable de entorno llamada `GOOGLE_API_KEY` con el valor de la clave API obtenida de Google Gemini. Una vez configurado y después de reiniciar la aplicación para detectar la nueva variable de entorno, el usuario puede activar o desactivar la opción de conjugación automática. Si está activada, las frases se conjugan automáticamente antes de ser convertidas a voz.

La conjugación se realiza enviando a Google Gemini un *prompt* específico que se ha diseñado para recibir una frase en infinitivo y el idioma correspondiente, y devolver la frase conjugada. Este es el *prompt* utilizado:

Recibes una frase con palabras en infinitivo y el idioma en el que está escrita (Español o Gallego). Tu tarea es conjugar la frase para que las palabras estén en la forma correcta y coherente entre sí, siendo coherente también con el idioma.

No se permiten signos de puntuación, solo palabras y el símbolo de interro-gación (esto solamente en caso de que se añada en el input).

Devuelve SOLAMENTE la frase conjugada en la forma más previsible posible que crees que significa lo que quiere decir.

Si la frase tiene solamente una palabra, puedes agregar un determinante si es necesario.

Ejemplo 1 (es): Entrada: TÚ COMER CARNE?

Respuesta: ¿Tú comes carne?

Ejemplo 2 (es): Entrada: YO QUERER COMER CARNE

Respuesta: Yo quiero comer carne

Ejemplo 3 (gal): Entrada: EU NECESITAR VER DOUTOR

Respuesta: Eu necesito ver ao doutor

Ejemplo 4 (gal): Entrada: EU QUERER TI MOITO

Respuesta: Eu quérote moito

Frase: *"Frase que el usuario ha escrito en infinitivos"*

Idioma: *"Idioma de la aplicación en el momento actual"*

A continuación se presentan algunos resultados de las pruebas realizadas en los dos idiomas disponibles, demostrando la eficacia del método:

Pruebas en castellano:

- “*Yo comer mucho carne poco lechuga*”: **“Yo como mucha carne y poca lechuga”**
- “*Yo no necesitar ayuda yo sentir bien*”: **“Yo no necesito ayuda, me siento bien”**
- “*Nosotros-as querer viajar autobús parque*”: **“Nosotros queremos viajar en autobús al parque”**
- “*Tú necesitar ayuda para beber?*”: **“¿Tú necesitas ayuda para beber?”**

Pruebas en gallego:

- “*Ti querer reloxo moito diñeiro*”: “**Ti queres un reloxo moi caro**”
- “*Necesitar anteollos para ver televisión*”: “**Necesito anteollos para ver la televisión**”
- “*Ti poder ir ver doutor para axuda eu?*”: “**¿Ti podes ir ver ao doutor para axudarme?**”
- “*El-a pechar porta tenda chave viaxar tren pobo*”: “**Ela pechou a porta da tenda coa chave e viaxou de tren para o pobo**”

Los resultados indican una buena adaptación del modelo a frases en español, mientras que en gallego se nota una calidad ligeramente menor, lo cual era esperado dado que Gemini no está completamente adaptado a este idioma.

8.2.4 Calibración del parpadeo

El parpadeo es el método mediante el cual el usuario puede confirmar las selecciones. Por lo tanto, es fundamental detectar con precisión el momento en que el usuario realiza esta acción, evitando que los parpadeos involuntarios sean interpretados como pulsaciones.

Se explicó en detalle cómo se obtiene esta medida en la Sección 8.1.2. Dado que el EAR no suele llegar a cero con los ojos cerrados, sino que se queda un poco por encima, es difícil asignar un valor fijo para detectar una pulsación. Por ello, es necesario calibrar el sistema específicamente para cada usuario. La calibración del parpadeo está dividida en tres fases, ilustradas en la Figura 8.18.

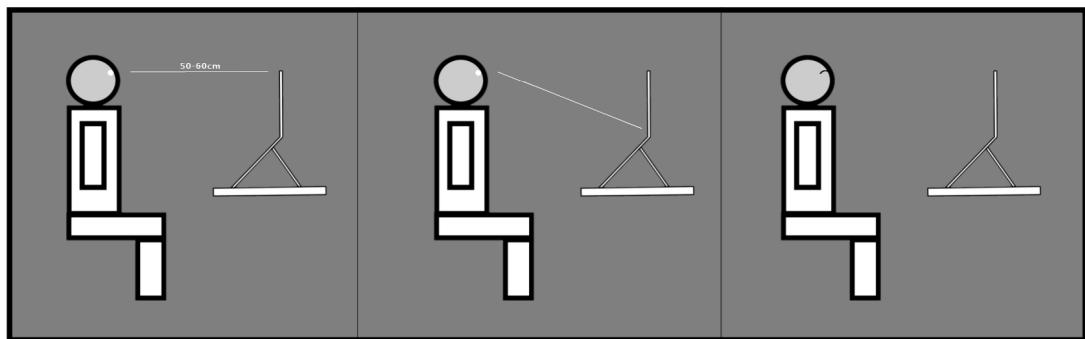


Figura 8.18: Imágenes representativas mostradas en la calibración del parpadeo.

Fase 1: Colocar al usuario y el ordenador en la posición correcta

El primer paso en la calibración del parpadeo consiste en asegurar que el usuario esté adecuadamente posicionado frente al ordenador. Es importante que el usuario se sitúe a una

distancia de entre 50 y 60 cm de la pantalla. Además, la pantalla debe estar colocada perpendicularmente a la superficie de la mesa para minimizar distorsiones en la captura de la imagen. Es crucial que la cámara web esté a la altura de los ojos del usuario para capturar de manera eficaz el movimiento de los párpados. Esta configuración inicial es vital para el correcto funcionamiento del sistema de detección de parpadeos, como se ilustra en la figura de la izquierda en la Figura 8.18.

Para asegurar que tanto la cámara como el usuario estén correctamente posicionados, la pantalla de calibración muestra un frame en el lado derecho con la silueta difuminada de una persona superpuesta. Además, utilizando *Mediapipe Face Mesh*, se identifica y se visualiza el punto de referencia número 8, ubicado en el centro entre las cejas del usuario. Cuando este punto se alinea correctamente dentro del círculo central de la pantalla, se ilumina en verde, indicando una posición óptima. Si el usuario no está correctamente alineado, aparece una flecha dinámica que varía en tamaño según la distancia del punto de las cejas al centro del círculo, ofreciendo una guía visual intuitiva para ajustar la posición. Esta funcionalidad está visualmente representada en las Figuras 8.19 y 8.20, donde se muestra la posición incorrecta y correcta respectivamente.



Figura 8.19: Usuario colocado en una posición incorrecta.

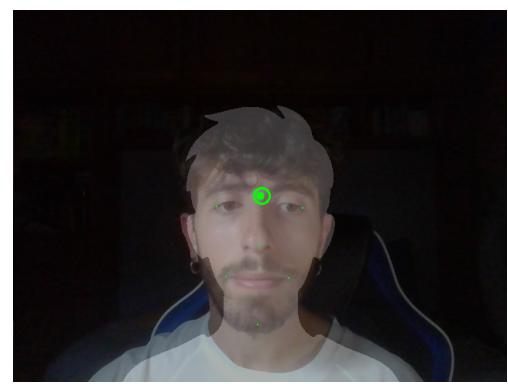


Figura 8.20: Usuario colocado en la posición correcta.

Fase 2: Tomar la medida del EAR mientras el usuario mira a la parte inferior de la pantalla

Una vez que el usuario está en la posición correcta, se muestra un punto amarillo en el centro del borde inferior de la pantalla. El usuario debe mirar ese punto mientras se pulsa “continuar”. En este momento, se toma la medida del **EAR** de cada ojo y se calcula la media de ambas, obteniendo así la **Medida_Inferior**.

Fase 3: Tomar la medida del EAR mientras el usuario mantiene los ojos cerrados

En el último paso de la calibración, se instruye al usuario a cerrar los ojos. Mientras los ojos permanecen cerrados, el operador debe pulsar “continuar”. En este momento, se captura la medida del [EAR](#) de ambos ojos cerrados, y se calcula el promedio de estas medidas para obtener el valor de *Medida_Cerrados*. Con esto, se concluye el proceso de calibración.

Una vez finalizada la calibración, se establece el umbral de detección para considerar que el usuario tiene los ojos cerrados. Este umbral se calcula usando una media ponderada de las dos medidas relevantes, como se muestra en la siguiente ecuación:

$$\text{Umbral EAR} = (\text{Medida_Inferior} \times 0.4 + \text{Medida_Cerrados} \times 0.6) \quad (8.3)$$

En la Ecuación 8.3, *Medida_Inferior* se refiere al valor promedio de EAR cuando los ojos están abiertos, ajustado para ser un poco más bajo, actuando como un nivel de seguridad para evitar falsos positivos. Por otro lado, *Medida_Cerrados* es el promedio de EAR cuando los ojos están completamente cerrados.

Para detectar pulsaciones, no solo se identifican los valores de EAR que caen por debajo del umbral. Dado que un parpadeo puede capturar varios frames con los ojos cerrados, se emplea un contador para evitar múltiples detecciones en un corto intervalo. Si el EAR se mantiene por debajo del umbral durante al menos 4 frames consecutivos, se considera una pulsación intencionada. Cuando el EAR supera el umbral (indicando que los ojos se han abierto), el contador se reinicia a cero. Este método asegura que los parpadeos involuntarios no sean contados como selecciones y que solo se registre una pulsación, aunque los ojos permanezcan cerrados por un período prolongado.

8.2.5 Reentrenamiento del modelo

Dado que el modelo del lector ocular puede no ajustarse con la misma precisión a todos los usuarios, se ofrece la opción de reentrenarlo utilizando datos específicos del usuario actual.

Recopilación de datos

El proceso comienza con la recopilación de datos del usuario. Desde el menú principal, el usuario accede a la pantalla de “Reentrenar”, donde se le instruye a seguir un punto en movimiento en la pantalla. En esta se aprecia una imagen dinámica, generada con Canvas, que proporciona una vista previa del proceso. Al seleccionar el botón de “Reentrenar”, la aplicación muestra solamente el punto a seguir y comienza una cuenta regresiva de 5 segundos antes de registrar los datos, como se detalla en la Sección 8.2.7.

Procesamiento y entrenamiento

A diferencia del procedimiento estándar de recopilación de datos, aquí los datos no se almacenan en la base de datos (BD). En su lugar, se procesan siguiendo el mismo método descrito en la Sección 8.1.5 y se utilizan inmediatamente para el reentrenamiento. El modelo se entrena durante 40 épocas con un *learning rate* de 0.00001 para adaptarse a las características del usuario sin incurrir en sobreajuste. La evolución del entrenamiento se visualiza en tiempo real y se ilustra en la Figura 8.21, donde se destaca cómo el modelo ajusta los datos del usuario manteniendo la generalidad.

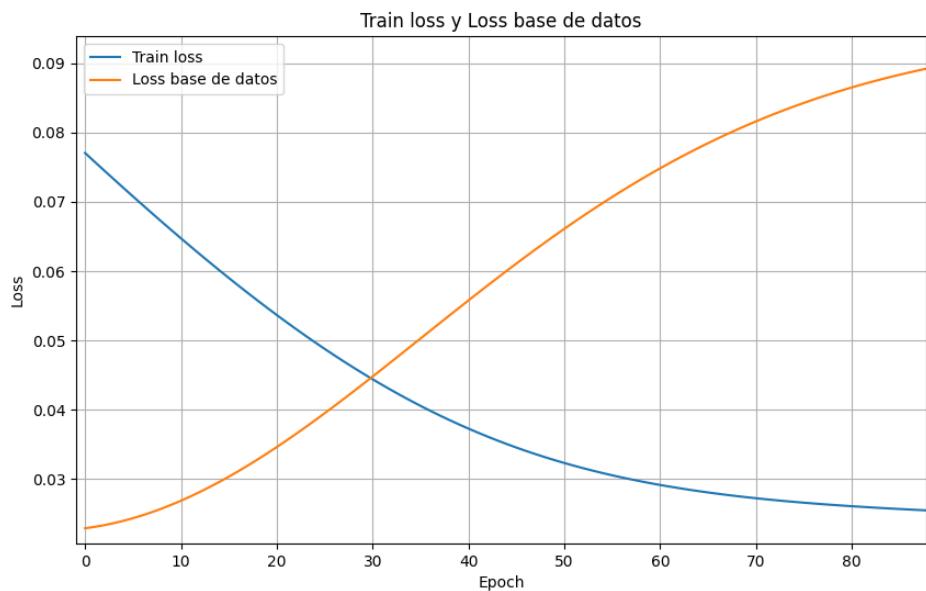


Figura 8.21: Gráfico de un reentrenamiento comparado con el error general de la BD.

Optimización de hiperparámetros

Posteriormente, se realiza una búsqueda de hiperparámetros con Optuna [26], involucrando 100 iteraciones en tres hilos para explorar los valores óptimos de ajuste postprocesado. Este proceso busca la combinación que minimice el error en los datos del usuario, optimizando el rendimiento de la red. Durante esta fase, el progreso también se informa en tiempo real.

Gestión de datos y reajustes

Los datos del usuario se almacenan temporalmente y se incluyen en futuros reentrenamientos para reducir el riesgo de sobreajuste. Si se detecta una reducción en la calidad del

modelo, indicando un posible sobreajuste, el usuario tiene la opción de restablecer los ajustes del modelo a sus valores predeterminados y eliminar todos los registros de reentrenamiento.

8.2.6 Tableros comunicativos

Para la implementación de los tableros comunicativos, se ha utilizado el patrón *Composite*, que permite manejar conjuntos de elementos, tanto simples como compuestos, de manera uniforme [23]. En este contexto, los tableros están formados por casillas, que pueden incluir o no pictogramas, pero todas se gestionan de la misma manera.

Acceso y configuración de los tableros

Desde la pantalla de “Tableros”, accesible desde el menú principal, el usuario puede acceder a las instrucciones de uso. Aquí, puede elegir si desea utilizar los tableros con o sin pictogramas. Tras seleccionar la opción deseada, se despliega el tablero comunicativo correspondiente.

Personalización de los tableros

Los tableros se diseñan para ser altamente personalizables, permitiendo a los usuarios modificar el contenido de las casillas, ya sea cambiando las palabras, los pictogramas o el número de casillas por tablero. Esta flexibilidad facilita la adaptación del sistema a las habilidades y necesidades específicas de cada usuario en relación con el lector ocular.

Estructura y almacenamiento de los tableros

Los datos de los tableros se almacenan en la carpeta `tableros`, que contiene un archivo `.xlsx` para cada idioma soportado. Dentro de cada archivo, los diferentes tableros están organizados en hojas. La carpeta `pictogramas` almacena todas las imágenes utilizadas para los pictogramas, las cuales son comunes para todos los idiomas.

Cuando se abren los tableros comunicativos, el sistema carga automáticamente el primer tablero, correspondiente a la primera hoja del archivo. En los tableros incluidos por defecto en el software, esta hoja se denomina “TAB. INICIAL”, como se muestra en la Figura 8.22.

Cada casilla tiene dos columnas reservadas: la primera para el nombre de la fotografía correspondiente de la carpeta de pictogramas y la segunda para el nombre de la palabra a mostrar. De esta manera, si un usuario quiere personalizar una casilla, solo debe añadir la fotografía con el nombre deseado a la carpeta de pictogramas y escribir este nombre en la primera columna reservada para esa casilla.

Cuando la palabra de la casilla comienza por “TAB. ”, esta adapta su diseño estableciendo un tono más oscuro, ya que es considerada por el software como un enlace a otro tablero, es

CAPÍTULO 8. IMPLEMENTACIÓN

	A	B	C	D	E	F	G	H
1	tab-rapido.png	TAB. RÁPIDO	tab-verbos.png	TAB. VERBOS	tab-comida.png	TAB. COMIDA	tab-objetos.png	TAB. OBJETOS
2	tab-personas.png	TAB. PERSONAS	tab-lugares.png	TAB. LUGARES	tab-transporte.png	TAB. TRANSPORTE	tab-casa.png	TAB. CASA
3	tab-animales.png	TAB. ANIMALES	tab-cuerpo.png	TAB. CUERPO	tab-conceptos.png	TAB. CONCEPTOS	int.png	?

Figura 8.22: Hoja "TAB. INICIAL" de "tablero_es_ES.xlsx".

decir, a otra hoja del archivo. Por ejemplo, al seleccionar "TAB. RÁPIDO", se abre el tablero correspondiente a la hoja "TAB. RÁPIDO".

Las instrucciones de edición de los tableros se proporcionan con el software de manera clara, permitiendo a los usuarios un control completo sobre los tableros que desean utilizar. En caso de un error de formato, se avisa al usuario de que los tableros no están en el formato adecuado para la aplicación. En el caso de que el error se encuentre en el nombre de la foto y este nombre no se encuentre en el directorio de los pictogramas, se muestra una cruz negra en lugar de la imagen indicando el error.

Dentro de la pantalla de los tableros, al pulsar sobre una palabra esta se añade a la frase que se está formando, permitiendo borrar la última palabra o borrar la frase por completo.

También disponen de un botón de alarma que emite un sonido avisador. Esta función fue solicitada por [AGAELA](#) ya que, en ocasiones, los usuarios desean avisar a su cuidador o responsable. Esta es una manera sencilla de hacerlo, permitiendo que el encargado del usuario acuda rápidamente.

Al pulsar sobre el espacio de texto donde se forma la frase, si el conjugador automático está activado y hay conexión a internet, la frase se modifica reflejando el cambio y posteriormente se reproduce en voz alta con la voz seleccionada en la configuración, mediante síntesis de voz. En caso de que el conjugador no esté activado o no haya conexión a internet, la frase simplemente se reproduce en su forma original. Para comprobar la conexión a internet, se intenta establecer una conexión con el servidor DNS de Google (8.8.8.8) en el puerto 53, el cual está asignado para el protocolo de Domain Name Server por la [Internet Assigned Numbers Authority \(IANA\)](#) [27].

Para controlar los tableros, se ofrece la posibilidad de utilizar la entrada estándar del ordenador, lo que permite la adaptabilidad a otras tecnologías de asistencia, como mouses adaptados. El manejo de los tableros con la entrada estándar es sencillo, basta con hacer una pulsación sobre una casilla o un botón para que se accione.

Por otro lado, cuando el control ocular está activado, aparece un puntero en la pantalla, como se muestra en las Figuras 8.23a y 8.23b. Este puntero en forma de cruz tiene un círculo en su interior que al mantenerlo sobre una casilla, comienza a aumentar de tamaño, pero



Figura 8.23: Representación visual del puntero: (a) en su estado inicial y (b) a un paso de bloquear una casilla.

solamente mientras se está sobre la misma. En caso de cambiar de casilla reinicia el ciclo desde su tamaño más pequeño. Una vez que el círculo alcanza el tamaño de la cruz, la casilla o botón sobre el que se encuentra se bloquea, oscureciendo su tono. La casilla permanece bloqueada hasta que se bloquee otra casilla, permitiendo al usuario confirmar la selección de la casilla bloqueada con un parpadeo, sin importar si el puntero se mueve de la casilla deseada.

Cabe destacar que el puntero no marca específicamente la posición predicha por el modelo del lector ocular en cada instante. En su lugar, se aplica el postprocesado descrito en la Sección 8.1.8 con una reducción del 5% para cada esquina y el valor central sin modificar, junto con una cadena de dos suavizados. Primero, se calcula la mediana de los últimos 30 valores registrados, lo que reduce el ruido en las predicciones del modelo sin retrasar mucho el movimiento. Luego, se realiza la media de las últimas 5 medianas, proporcionando un movimiento suave del puntero por la pantalla.

Por otro lado, AGAELA también sugirió la implementación de un modo descanso, ya que puede haber momentos en los que los usuarios se sientan cansados de utilizar el lector ocular y quieran detenerlo. Este modo se ha implementado de manera sencilla para evitar la frustración que puede surgir al intentar alcanzar un punto específico de la pantalla, al que el usuario no da movido el cursor para activar el botón de modo descanso, como ocurre aparentemente con otras tecnologías. Para activar el modo descanso, el usuario simplemente debe mantener los ojos cerrados durante 3 segundos. El sistema se bloquea, atenuando la pantalla y mostrando un mensaje que indica que está bloqueado y para desbloquearlo se debe realizar la misma acción: cerrar los ojos durante 3 segundos.

8.2.7 Pantallas de desarrollador

Test

Esta pantalla tiene como finalidad la observación del rendimiento de los diferentes modelos de lectores oculares desarrollados, permitiendo manipular su postprocesado de una manera sencilla. Como se mencionó en la Sección 8.1.8, el postprocesado principal depende de 5 puntos

visibles en la pantalla de prueba: uno en cada esquina y uno en el centro. Estos puntos son fácilmente manejables desde esta pantalla, ya que se pueden arrastrar con el ratón a la posición deseada, cambiando así el valor del postprocesado. Cuanto más se aleje un punto de su esquina correspondiente, más se estira la ponderación hacia esa dirección. De esta manera, se permite un control sencillo y visual del ajuste del postprocesado de la red, permitiendo ver los cambios en tiempo real, lo que ayuda al desarrollador a implementar mejoras en el software.

Recopilar

La pantalla de recopilación de datos se ha utilizado para crear las bases de datos necesarias para entrenar los diferentes modelos desarrollados en este proyecto. En esta pantalla, la cámara del usuario se encuentra en la parte inferior derecha, para evitar movimientos imprevistos que descoloquen al usuario de la posición determinada antes de cada recopilación y también se encuentra debajo de esta el botón para iniciar la recopilación. Cuando este botón es pulsado, comienza una cuenta regresiva de 5 segundos.

Al llegar a cero, un punto rojo empieza a moverse a una velocidad constante de izquierda a derecha. Cada vez que alcanza los bordes de la pantalla, aumenta su posición en el eje vertical y cambia de dirección horizontal. Una vez que llega al borde superior, el punto continúa su movimiento hacia abajo hasta alcanzar nuevamente el borde inferior. La velocidad a la que el puntero se mueve es constante, mientras que los saltos verticales se producen de manera aleatoria dentro de un rango de valores, contribuyendo así a una BD lo más completa posible sin repetir para todos los usuarios las mismas coordenadas.

Durante este proceso, se recopilan todas las características mencionadas en la Sección 8.1.4, incluyendo los frames, aunque solamente desde el momento en que se detecta que pueden ser útiles para el desarrollo del lector ocular. Al finalizar la recopilación, se muestra un PopUp de agradecimiento y se ofrece la opción de guardar o no los datos. Esto es importante, ya que en ocasiones los usuarios pueden perder la concentración y desviar la mirada, o pueden surgir situaciones en las que es preferible descartar los datos para evitar introducir información incorrecta o sucia en las bases de datos.

Pruebas

Se han implementado pruebas automatizadas para la validación del software, en las cuales el propio software es capaz de medir y registrar diversas métricas durante la realización de diferentes tareas a través del lector ocular. Para su desarrollo, se ha decidido evaluar el software utilizando los propios tableros, midiendo el tiempo que los usuarios tardan en escribir una palabra o una frase, con o sin pictogramas, además de la precisión obtenida y los errores cometidos. Las pruebas se presentan como *PopUps* sobre los tableros, donde cada uno

explica la prueba a realizar y tiene un botón para comenzarla, saltarla o regresar a la anterior, asegurando así la fluidez a la hora de la realización de las mismas.

Están diseñadas para ser automáticas, requiriendo solo iniciar la prueba, completarla y luego proceder a la siguiente, mecanizando todo el proceso de registro de datos para una evaluación posterior. Las diferentes pruebas realizadas se detallan en el Capítulo D.24 donde se comenta qué registros se realizan y también se analizan los resultados obtenidos. Todos los registros se almacenan en un archivo .xlsx para su posterior análisis y evaluación.

8.2.8 Recursos

Para la realización de este software se han utilizado diferentes recursos de uso recogidos de diversas fuentes asegurando su libertad de uso y otros han sido creados por el alumno, se citan a continuación junto con las respectivas referencias:

- Fondo de pantalla: Foto de [Andrew Kliatskyi](#) en [Unsplash](#)
- Fuentes: [FrancoisOne-Regular](#) y [Orbitron-Regular](#), de [Google Fonts](#).
- Sonido de click: [Camera shutter click](#) (recortado)
- Sonido de bloqueo: [Door Lock](#) en [Freesound](#)
- Sonido de campana: [Campana](#) en [Freesound](#)
- Autor pictogramas: Sergio Palao. Origen: [ARASAAC](#). Licencia: CC (BY-NC-SA). Propiedad: Gobierno de Aragón (España)
- Silueta del frame e icono de la aplicación: Generados con DALL·E 3.
- Por otra parte, el resto de recursos, ya sean las imágenes de la calibración, iconos de idiomas, o la animación de la vista previa del reentrenamiento, han sido producidos por el alumno.

Capítulo 9

Pruebas de usabilidad del software

El objetivo de las pruebas es evaluar la usabilidad del software y verificar la precisión con la que detecta y sigue la posición de la mirada en la pantalla. Además, se busca identificar posibles áreas de mejora para garantizar una experiencia de usuario óptima y precisa.

9.1 Participantes y materiales

Tras el desarrollo de la versión beta del software, se realizaron pruebas con un grupo de control y dos miembros de la asociación AGAELA, quienes son pacientes diagnosticados con ELA en diferentes etapas de la enfermedad, presentando variaciones en sus habilidades de movilidad y comunicación. Durante estas pruebas, se utilizó la versión beta del software, la cual implementaba el modelo del lector ocular entrenado tras la primera aproximación, detallado en la Sección 8.1.7. Posteriormente, tras implementar las mejoras identificadas, se realizaron nuevas pruebas con tres usuarios del grupo de control. En estas pruebas se utilizó la versión final del software, que incluía el modelo del lector ocular entrenado tras las distintas aproximaciones, tal como se detalla en la Sección 8.1.12.

9.2 Desarrollo de las pruebas

Las pruebas realizadas tenían como objetivo evaluar tanto la usabilidad como la efectividad del software ComunicELA en diferentes escenarios y con distintos tipos de usuarios. Se dividieron en varias fases que abarcaban desde la familiarización con el sistema hasta la evaluación de su rendimiento en tareas específicas de comunicación. Estas pruebas se realizaron con usuarios de control y miembros de la asociación AGAELA, quienes presentaban diferentes niveles de movilidad debido a las distintas etapas de la enfermedad.

9.2.1 Preparación de los participantes

Para comenzar, se debe obtener el consentimiento formal de los participantes. A continuación, se les explican las instrucciones que deben seguir y el propósito de las pruebas. Antes de iniciar las evaluaciones, se permite a cada usuario familiarizarse con el control ocular en los tableros comunicativos, brindándoles la oportunidad de practicar y sentirse cómodos con el sistema. Esta fase previa es crucial para asegurar que los participantes comprendan plenamente el funcionamiento del software y puedan interactuar con él de manera fluida durante las pruebas de usabilidad del software.

9.2.2 Pruebas

Las pruebas se dividen en diferentes niveles de dificultad, permitiendo evaluar progresivamente la usabilidad del software a medida que los usuarios escriben oraciones de complejidad creciente:

- **Nivel básico:** Se evalúa la capacidad de selección de respuestas rápidas y básicas, seleccionando palabras individualmente. Para este nivel no se emplean pictogramas, se utiliza un tablero que solo dispone de las palabras en formato de texto. Tampoco se requiere que los usuarios cambien de tablero localizando el necesario, ya que todas las palabras necesarias se muestran desde el principio para que los usuarios solo tengan que elegir la que se solicita. Las palabras solicitadas son: “Sí”, “No”, “Bien” y “Mal”.
- **Nivel intermedio:** Se evalúa la capacidad de elaboración de oraciones o frases sencillas, utilizando tableros que disponen de pictogramas para facilitar la rápida localización de las casillas. Para esta prueba, es necesario que los usuarios naveguen entre los diferentes tableros disponibles. Las palabras y frases que se utilizan son: “Querer beber”, “Pierna derecha” y “Abrir ventana”.
- **Nivel experto:** Se evalúa la capacidad de los usuarios para formar oraciones complejas y estructuradas, utilizando tableros avanzados que incluyen tanto palabras como pictogramas. En este nivel, se espera que los usuarios naveguen entre múltiples tableros y seleccionen las palabras necesarias para construir frases completas y coherentes. Este nivel muestra una perspectiva realista de cómo los usuarios interactúan con el software en situaciones cotidianas y más exigentes. Las oraciones solicitadas son: “¿Yo poder dormir ahora?”, “Yo necesitar ver doctor”, “Él/ella antes viajar mucho” y “¿Nosotros/as poder ir parque?”.

9.2.3 Recolección de datos

Para evaluar la efectividad y usabilidad del software ComunicELA, se recopilan y analizan datos tanto cuantitativos como cualitativos:

Datos cuantitativos

- **Tiempos de respuesta:** Se mide el tiempo que tarda el usuario en escribir cada palabra y el tiempo total desde que comienza la prueba hasta que se reproduce la frase, proporcionando una perspectiva completa del uso del software, desde que el usuario piensa en la frase hasta que esta se reproduce auditivamente.
- **Precisión en la selección:** Se registran los errores cometidos por los usuarios durante las pruebas, el número de veces que el usuario selecciona una opción incorrecta y la borra para corregirla antes de completar la oración. La precisión final se calcula como el porcentaje de selecciones correctas sobre el total de selecciones realizadas. Además, se mide la desviación entre el punto central de la casilla a seleccionar y los puntos a donde se detecta que está mirando el usuario. Para ello se registran todas las posiciones donde se encuentra el puntero desde que la casilla se empieza a bloquear hasta que esta es seleccionada mediante el parpadeo.

Datos cualitativos

A través de un cuestionario se mide la satisfacción de los usuarios miembros de AGAELA, evaluando su bienestar con el software y recopilando sugerencias para posibles mejoras. Con ello, se pueden identificar áreas de mejora y asegurar que el software ComunicELA ofrezca una experiencia de usuario óptima basada en el feedback recibido.

9.3 Evaluación y análisis

En las pruebas iniciales y finales, los usuarios tuvieron el tiempo necesario para familiarizarse con el sistema antes de comenzar las evaluaciones formales, lo que les permitió interactuar con el software de manera efectiva.

En contraste, los usuarios de la asociación AGAELA presentaban distintos niveles de movilidad debido a las diferentes etapas de la enfermedad. El tiempo limitado disponible, junto con las restricciones de movilidad, impidió obtener resultados satisfactorios en el nivel avanzado para este grupo. Además, el corto periodo de familiarización con el control ocular generó resultados menos óptimos en comparación con los otros usuarios, tanto en términos de tiempo como de precisión en las selecciones, como se muestra en la Tabla 9.1.

	Iniciales	AGAELA	Finales
Tiempo por palabra	6.03s	23.16s	4.84s
Tiempo medio oraciones nivel básico	15.76s	34.95s	10.15s
Tiempo medio oraciones nivel intermedio	37.18s	88.99s	30.85s
Tiempo medio oraciones nivel experto	97.65s	-	68.49s
Porcentaje selecciones erróneas	20.27%	45%	5.83%
Desviación media del centro de la casilla	9.38%	12.71%	10.13%

Tabla 9.1: Comparación de las diferentes pruebas realizadas. **Iniciales** se refiere al grupo de control con la aplicación en fase beta; **AGAELA**, a las pruebas realizadas con la asociación utilizando la aplicación en fase beta; **Finales**, al grupo de control con la aplicación en su versión final.

Por otra parte, los resultados de las pruebas iniciales y finales muestran una clara evolución del software desde su versión beta hasta la versión final. El tiempo medio por palabra disminuyó de 6.03 segundos a 4.84 segundos, lo que se refleja también en los tiempos de los diferentes niveles de dificultad. Además, el porcentaje de selecciones erróneas se redujo notablemente, pasando del 20.27% al 5.83%.

En conclusión, los resultados demuestran una reducción significativa en los tiempos promedio para completar las oraciones en todos los niveles, así como una notable disminución en el porcentaje de selecciones erróneas, lo que indica una mayor eficiencia en la interacción con el sistema. Estos resultados subrayan la importancia del tiempo de familiarización con el software, ya que se observó que, a mayor tiempo de uso, mayor es la precisión de los usuarios. Esto también refuerza los hallazgos de las pruebas realizadas con **AGAELA**, donde un menor tiempo de familiarización impactó en los resultados obtenidos.

Capítulo 10

Conclusiones y Trabajos futuros

En este capítulo se presentan las conclusiones obtenidas a lo largo del desarrollo de este proyecto, así como las posibles líneas de trabajo futuro que podrían complementar o ampliar los resultados alcanzados. A través de un análisis exhaustivo de las fases del proyecto, se evaluarán los logros obtenidos y las limitaciones encontradas, con el fin de proporcionar una visión clara del impacto y la viabilidad de la solución implementada.

10.1 Conclusiones

El proyecto ha logrado un desarrollo efectivo, culminando en la creación del software “ComunicELA”, una herramienta robusta y funcional que mejora considerablemente la calidad de vida de los pacientes con ELA. Desde el inicio, todos los objetivos planteados se han cumplido satisfactoriamente, proporcionando una alternativa comunicativa que permite a los usuarios formar frases mediante el seguimiento ocular y reproducirlas usando síntesis de voz.

La investigación inicial sobre herramientas de comunicación para pacientes con ELA reveló oportunidades significativas para la innovación. Este análisis fundamentó el desarrollo de una solución de código abierto que responde a las necesidades específicas de estos pacientes sin requerir dispositivos dedicados.

La colaboración con la [Asociación Galega de Afectados pola Esclerose Lateral Amiotrófica \(AGAELA\)](#) ha sido crucial, proporcionando orientación precisa mediante consejos y sugerencias que han garantizado que el software esté alineado con las necesidades reales de los usuarios. Además, la interfaz del software se diseñó meticulosamente con características accesibles, como botones de gran tamaño y adecuados contrastes visuales, lo que facilita su uso a un amplio rango de pacientes, independientemente de la severidad de su condición.

El lector ocular implementado es fundamental en el funcionamiento del software, permitiendo a los usuarios interactuar con los tableros comunicativos mediante el seguimiento de la mirada. Esta tecnología, crucial para pacientes en etapas avanzadas de la enfermedad

con movilidad limitada, ha demostrado ser eficaz y precisa en pruebas realizadas en entornos reales.

El proyecto no solo ha generado una herramienta tecnológica avanzada, sino que también ha creado una solución impactante para la vida de muchas personas. Esto no habría sido posible sin el apoyo de AGAELA, de la Cátedra NTT DATA en Diversidad y Tecnología, y el esfuerzo de los voluntarios involucrados en la recopilación y validación de la herramienta.

Para facilitar el acceso a las mejoras introducidas por “ComunicELA”, ofrecemos este software como una solución de código abierto, disponible para su descarga sin costo alguno a través del siguiente enlace¹. Estamos comprometidos con la mejora continua y la adaptación del software para satisfacer las necesidades de los usuarios, y valoramos enormemente cualquier comentario que nos permita perfeccionar aún más esta herramienta.

10.2 Trabajo futuro

Aunque el software ha alcanzado los objetivos iniciales, existen varias áreas en las que se puede continuar trabajando para mejorar y expandir sus capacidades:

- **Mejora de la precisión del lector ocular:** A pesar de los buenos resultados obtenidos, siempre hay margen para mejorar la precisión. Explorar nuevos métodos y tecnologías podría optimizar aún más el rendimiento del lector ocular.
- **Desarrollo de una versión móvil:** La creación de una versión para dispositivos móviles y tabletas aumentaría significativamente la accesibilidad del software, permitiendo que los usuarios puedan utilizar “ComunicELA” en una mayor variedad de contextos.
- **Ampliación del soporte lingüístico:** Incluir soporte para más idiomas permitirá que el software sea accesible a una mayor diversidad de usuarios a nivel global, expandiendo su impacto.
- **Editor de tableros:** Desarrollar un editor intuitivo para gestionar y personalizar los tableros de comunicación directamente desde el menú de la aplicación, evitando la necesidad de editar manualmente los archivos.

¹ Descargue “ComunicELA” desde: <https://github.com/TsolidarioFG/2024-AGAELA-comunicacion>

Apéndices

Apéndice A

Fundamentos tecnológicos

En este apéndice se describen las herramientas y tecnologías empleadas para el desarrollo de este proyecto. Estas herramientas cubren una amplia gama de necesidades, desde la edición de código y el control de versiones hasta el procesamiento de imágenes y el desarrollo de modelos de aprendizaje automático, cada una con un rol esencial en la construcción de la solución propuesta.

A.1 Visual Studio Code

“**Visual Studio Code**” es un editor de código fuente desarrollado por Microsoft para sistemas operativos como Windows, macOS y Linux [28]. Este editor cuenta con soporte para depuración, integración con Git [29], y una gran cantidad de extensiones que facilitan el desarrollo en múltiples lenguajes de programación. Es el entorno de desarrollo integrado (IDE) principal utilizado en este proyecto debido a su versatilidad y capacidad de personalización.

A.2 Google Colab

“**Google Colab**” es una plataforma basada en la nube que permite escribir y ejecutar código en Python desde el navegador, con acceso a recursos de procesamiento en GPU [30]. Para este proyecto, se utiliza Google Colab para realizar el entrenamiento de los modelos de aprendizaje automático, aprovechando su capacidad para manejar cargas computacionales pesadas sin la necesidad de hardware local avanzado.

A.3 GitHub

“**GitHub**” es una plataforma que permite el desarrollo colaborativo y el control de versiones mediante Git [29]. Se utiliza en este proyecto para alojar el repositorio del código y

mantener un historial de versiones, facilitando tanto la colaboración como el seguimiento de cambios a lo largo del ciclo de desarrollo [31].

A.4 Python

“Python” es un lenguaje de programación de alto nivel conocido por su simplicidad y flexibilidad, lo que lo convierte en una opción ideal para proyectos que requieren programación orientada a objetos y manejo eficiente de datos [32]. En este proyecto, Python se utiliza como el lenguaje principal debido a su robusto ecosistema de bibliotecas que soportan tareas como el procesamiento de imágenes, el aprendizaje automático y el desarrollo de interfaces de usuario.

A.5 Mediapipe Solutions

“Mediapipe Solutions” es una biblioteca de código abierto que ofrece una serie de soluciones para la visión por computadora mediante aprendizaje automático [33]. En este proyecto, se utiliza “MediaPipe Face Mesh”, que permite el seguimiento en tiempo real de 468 puntos de referencia faciales, lo cual es clave para el desarrollo del lector ocular, que permite la interacción a través del seguimiento de los ojos.

A.6 OpenCV

“OpenCV” es una biblioteca de visión por computadora ampliamente utilizada en la industria para el procesamiento de imágenes y videos [34]. En este proyecto, OpenCV es fundamental para manejar tareas como la detección y el procesamiento de las imágenes obtenidas del lector ocular, facilitando la implementación de algoritmos de visión por computadora.

A.7 Pillow

“Pillow” (PIL) es una biblioteca de Python utilizada para la manipulación y edición de imágenes [35]. Se emplea en este proyecto para transformar y optimizar las imágenes utilizadas en los tableros de comunicación, facilitando su personalización y visualización adecuada.

A.8 Optuna

“Optuna” es un marco de optimización automática de hiperparámetros [26]. En este proyecto, Optuna se utiliza para optimizar los hiperparámetros de los modelos de aprendizaje

automático, lo que mejora la precisión y el rendimiento de los modelos entrenados para el lector ocular.

A.9 PyTorch

“**PyTorch**” es una biblioteca de código abierto especializada en el desarrollo y entrenamiento de modelos de aprendizaje profundo [36]. En este proyecto, PyTorch se utiliza para la creación de redes neuronales profundas que permiten la detección y análisis de los movimientos oculares.

A.10 CUDA

“**CUDA**” es una plataforma de computación paralela desarrollada por NVIDIA que permite el uso de GPUs para acelerar aplicaciones computacionales intensivas [37]. En este proyecto, CUDA se utiliza para acelerar el entrenamiento de los modelos de aprendizaje profundo en PyTorch, permitiendo entrenamientos más rápidos y eficientes.

A.11 cuDNN

“**cuDNN**” es una biblioteca desarrollada por NVIDIA que proporciona primitivas optimizadas para redes neuronales profundas, mejorando la velocidad y eficiencia de los entrenamientos en GPU [38]. Se utiliza junto con CUDA para aprovechar al máximo las capacidades de los modelos en PyTorch.

A.12 Scikit-learn

“**Scikit-learn**” es una biblioteca de Python muy popular para la implementación de algoritmos de aprendizaje automático y análisis de datos [39]. En este proyecto, Scikit-learn se emplea para tareas de preprocesamiento de datos y para la evaluación de algunos modelos de clasificación.

A.13 Kivy

“**Kivy**” es un marco de desarrollo de aplicaciones de código abierto para Python, que permite la creación de aplicaciones multiplataforma con interfaces de usuario táctiles [40]. En este proyecto, Kivy se utiliza para desarrollar la interfaz de usuario que los pacientes utilizarán para interactuar con el sistema a través de los tableros de comunicación personalizables.

A.14 Canva

“**Canva**” es una herramienta de diseño gráfico en línea que permite crear contenido visual de manera sencilla [41]. En este proyecto, se utiliza Canva para crear y diseñar los elementos visuales y gráficos que forman parte de la interfaz y los tableros de comunicación de la aplicación.

A.15 ARASAAC

“**ARASAAC**” es un portal que ofrece recursos gráficos y pictogramas para facilitar la comunicación en personas con dificultades [42]. Los pictogramas de ARASAAC se utilizan en este proyecto para crear tableros de comunicación accesibles y visuales, ayudando a los usuarios a localizar y expresar palabras de manera más intuitiva [43].

A.16 Gemini API

“**Gemini API**” es una API avanzada de procesamiento de lenguaje natural que permite corregir y mejorar la coherencia gramatical de frases [25]. En este proyecto, se utiliza para optimizar las frases generadas por los usuarios a través de los tableros de comunicación, asegurando que las conjugaciones verbales y la concordancia de género y número sean correctas antes de que las frases sean reproducidas por el sistema de síntesis de voz.

A.17 ISpVoice

“**ISpVoice**” es una interfaz de síntesis de voz desarrollada por Microsoft [24]. En este proyecto, se utiliza para reproducir las frases que los usuarios escriben en los tableros de comunicación, aprovechando las voces instaladas en el sistema operativo, sin necesidad de conexión a Internet.

A.18 Balsamiq Wireframes

“**Balsamiq Wireframes**” es una herramienta de diseño que permite crear wireframes de manera rápida y sencilla [44]. Se utiliza en este proyecto para realizar los prototipos de las pantallas de la aplicación, facilitando el diseño y la organización de la interfaz de usuario.

A.19 **Createley**

“Createley” es una herramienta en línea para la creación de diagramas, como diagramas UML [45]. En este proyecto, se emplea para la creación de los diagramas de casos de uso, proporcionando una representación visual clara de los procesos y la interacción entre los usuarios y el sistema.

A.20 **Overleaf**

“Overleaf” es un editor de LaTeX en línea que permite la creación y edición colaborativa de documentos científicos [46]. En este proyecto, Overleaf se utiliza para la redacción y organización de la memoria del proyecto, aprovechando su capacidad para manejar referencias bibliográficas y ecuaciones complejas con facilidad [47].

Apéndice B

Seguimiento del proyecto

B.1 Planificación inicial

En la Figura B.1, se presenta el Diagrama de Gantt que muestra la planificación inicial diseñada para el desarrollo del proyecto “ComunicELA”. Este diagrama permite visualizar de manera clara las fases del proyecto, los hitos clave, y las dependencias entre las tareas. Además, facilita el seguimiento del progreso y asegura que los plazos establecidos se cumplan de manera eficiente.

Tras haber creado una planificación inicial y asignado las tareas correspondientes a cada recurso, se puede estimar la duración del desarrollo del software, la cual se detalla en la Tabla B.1. Sin embargo, durante el desarrollo surgieron ciertas complicaciones, principalmente relacionadas con el tiempo adicional necesario para recompilar el dataset y realizar el entrenamiento de los modelos de aprendizaje automático, lo que provocó un pequeño retraso que afectó la planificación original. La duración real del proyecto, incluyendo el impacto de estos retrasos, se presenta en la Tabla B.2.

Duración estimada	
<i>Fecha inicio</i>	01-03-2024
<i>Fecha fin</i>	24-07-2024
<i>Días duración</i>	206 días
<i>Trabajo</i>	845 horas

Tabla B.1: Duración estimada.

Duración real	
<i>Fecha inicio</i>	01-03-2024
<i>Fecha fin</i>	06-08-2024
<i>Días duración</i>	225 días
<i>Trabajo</i>	919 horas

Tabla B.2: Duración real.

APÉNDICE B. SEGUIMIENTO DEL PROYECTO

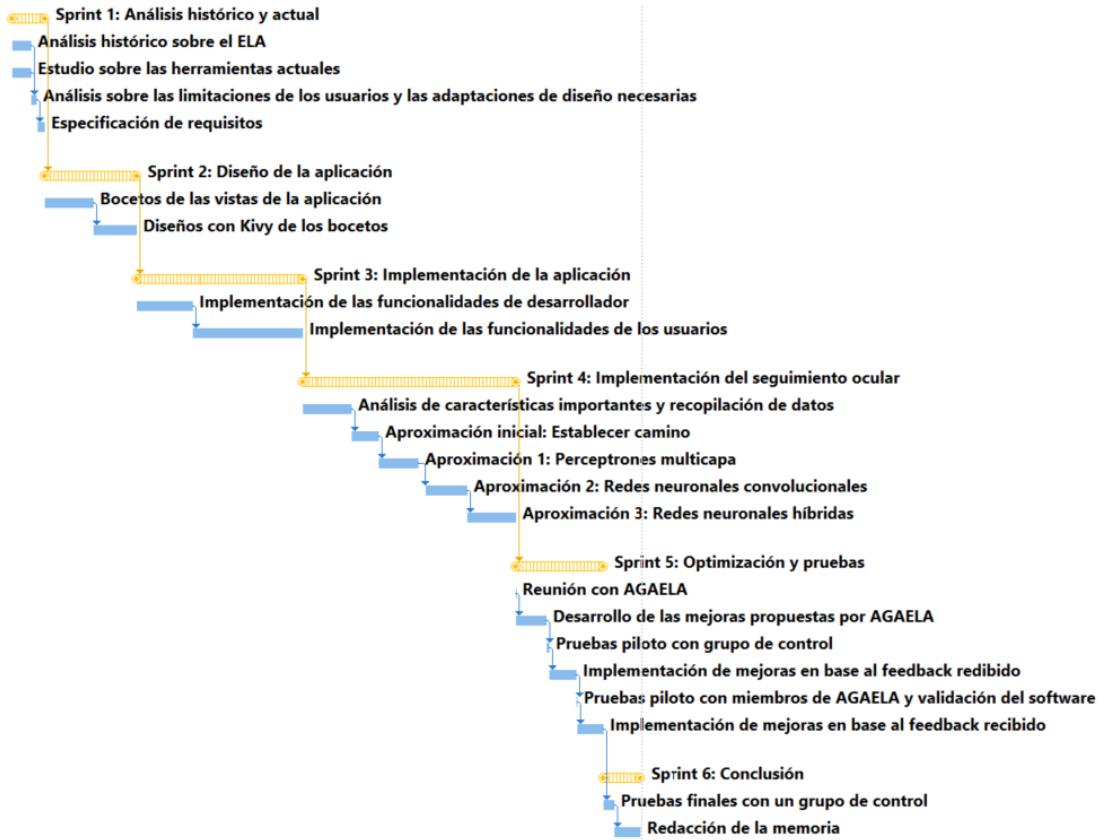


Figura B.1: Diagrama de Gantt de la planificación inicial de "ComunicELA".

B.2 Costes

Los costes de este proyecto se estiman considerando tanto los recursos materiales como los recursos humanos.

B.2.1 Recursos materiales

El único costo previsto durante la planificación del proyecto es el de un ordenador portátil, que tiene un valor de **1.200,00€** en el momento de su compra. Debido a las limitaciones de hardware al realizar las diferentes aproximaciones de los modelos de aprendizaje profundo, se decidió adquirir el plan (*Colab Pro+* de *Google Colab*). Este proporciona acceso a GPUs más potentes que agilizan el proceso, pero incrementan los costos del proyecto en **51,12€**. Es importante destacar que, para el desarrollo de este proyecto, se utilizó exclusivamente **software libre**, lo que permitió reducir significativamente los costos de licencias y herramientas de desarrollo, favoreciendo un enfoque accesible y colaborativo.

B.2.2 Recursos humanos

Para estimar los costos de los recursos humanos involucrados en el proyecto, se han utilizado las medias salariales anuales obtenidas de Prosperity Digital [48], correspondientes a los roles desempeñados en este desarrollo. La conversión de estos salarios a un coste por hora se ha realizado tomando como referencia la jornada laboral efectiva máxima anual, que en 2024 es de 1787 horas anuales [49]. A continuación, se detalla esta información en la Tabla B.3.

APÉNDICE B. SEGUIMIENTO DEL PROYECTO

Rol	Salario medio anual	Salario por hora
<i>Scrum Master</i>	47.500,00 €	26,58 €
<i>Analista de Business Intelligence</i>	40.500,00 €	22,66 €
<i>Diseñador UX/UI</i>	44.000,00 €	24,62 €
<i>Programador Python y AI</i>	43.500,00 €	24,34 €

Tabla B.3: Salarios de los recursos humanos.

Dado que la duración estimada del proyecto se vio alterada, aumentando su duración, los costes previstos para los recursos humanos también se incrementaron. En la Tabla B.4 se muestra la desviación que ha experimentado cada recurso en términos de horas de trabajo, y en la Tabla B.5 se detallan los costos adicionales que esta desviación ha supuesto.

Recurso	Horas previstas	Horas finales
<i>Scrum Master</i>	18	18
<i>Analista de Business Intelligence</i>	93	94
<i>Diseñador UX/UI</i>	61	71
<i>Programador Python y AI</i>	673	736

Tabla B.4: Horas de trabajo de los recursos humanos del proyecto.

Recurso	Coste previsto	Coste final
<i>Scrum Master</i>	478,44 €	478,44 €
<i>Analista de Business Intelligence</i>	2.107,38 €	2.130,04 €
<i>Diseñador UX/UI</i>	1.501,82 €	1.748,02 €
<i>Programador Python y AI</i>	16.380,82 €	17.914,24 €

Tabla B.5: Costes de los recursos humanos del proyecto.

APÉNDICE B. SEGUIMIENTO DEL PROYECTO

Para concluir este apéndice, se presenta la Tabla B.6, en la que se detalla el desglose de los costes finales del proyecto. Esta tabla proporciona una visión clara de los recursos utilizados y el impacto que tuvieron en el presupuesto total, reflejando tanto los costes previstos como los finales.

Recursos	Coste previsto	Coste final
Recursos materiales	1.200,00 €	1.251,12 €
Recursos humanos	20.468,46 €	22 270,74 €
TOTAL	21.668,46 €	23.521,86 €

Tabla B.6: Costes totales finales del proyecto.

Apéndice C

Casos de uso del software

En este apéndice se describen los principales casos de uso del software, que representan las interacciones clave entre los usuarios y el sistema. Estos casos de uso ayudan a entender cómo los diferentes actores, tanto usuarios comunes como desarrolladores, interactúan con las diversas funcionalidades de la aplicación. La correcta definición de los casos de uso permite asegurar que el software cumpla con los requisitos funcionales y no funcionales identificados durante la fase de análisis.

C.1 Casos de uso de la pre- configuración

Caso de uso	Selección de idioma
Actor	Usuario genérico
Precondiciones	Aplicación en la pantalla principal.
Postcondiciones	La aplicación cambia de idioma.
Flujo principal	El usuario toca sobre el ícono del idioma alternando entre los disponibles.
Excepciones	-

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Tutorial del software
Actor	Usuario genérico
Precondiciones	Abrir el software sin haber marcado anteriormente la casilla “No volver a mostrar” o haber tocado el botón del tutorial desde la configuración.
Postcondiciones	Se guarda el valor de la casilla “No volver a mostrar” para el próximo inicio.
Flujo principal	El usuario lee y cierra las ventanas del tutorial, en la última marca si desea volver a mostrarlo.
Excepciones	-

Caso de uso	Configuración
Actor	Usuario genérico
Precondiciones	Aplicación en la pantalla principal.
Postcondiciones	El menú de configuración está visible.
Flujo principal	El usuario toca el botón de configuración y el menú flotante se abre.
Excepciones	-

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Selección de cámara
Actor	Usuario genérico
Precondiciones	Aplicación en el menú de configuración.
Postcondiciones	La cámara seleccionada se activa o se actualiza la lista.
Flujo principal	El usuario elige la cámara deseada entre las disponibles o actualiza la lista.
Excepciones	Ninguna cámara conectada: No aparecerá ninguna opción en la lista de cámaras disponibles.

Caso de uso	Selección de voz
Actor	Usuario genérico
Precondiciones	Aplicación en el menú de configuración.
Postcondiciones	Se escoge la voz a utilizar o se actualiza la lista.
Flujo principal	El usuario escoge la voz deseada o actualiza la lista.
Excepciones	-

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Activar / desactivar el conjugador automático
Actor	Usuario genérico
Precondiciones	Aplicación en el menú de configuración, API configurada según la guía de uso y conexión a internet disponible.
Postcondiciones	Se activa o desactiva el conjugador automático.
Flujo principal	El usuario activa o desactiva el conjugador.
Excepciones	No tener la API de Gemini configurada o no tener conexión a internet: el botón aparecerá como “No disponible”.

C.2 Casos de uso del usuario común

Caso de uso	Calibrar el parpadeo
Actor	Usuario común
Precondiciones	Aplicación en la pantalla principal.
Postcondiciones	La detección del parpadeo se calibra a los características del usuario.
Flujo principal	Se divide en tres fases: Se coloca el ordenador en la posición indicada, después el usuario mira a un punto marcado y posteriormente el usuario cierra los ojos.
Excepciones	Ninguna cámara seleccionada: Se avisa al usuario mediante un mensaje informativo.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Reentrenar el modelo
Actor	Usuario común
Precondiciones	Aplicación en la pantalla principal.
Postcondiciones	-
Flujo principal	El usuario ve las instrucciones y decide qué hacer entre las opciones disponibles.
Excepciones	Cámara no seleccionada o parpadeo no calibrado: Se avisa al usuario mediante un mensaje informativo.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Reentreno
Actor	Usuario común
Precondiciones	Aplicación en la pantalla de reentrenar.
Postcondiciones	El modelo se reentrena y optimiza con los datos del usuario.
Flujo principal	El usuario mira al punto marcado en la pantalla hasta que este termina de moverse, después el sistema se reentrena y optimiza mostrando el proceso.
Excepciones	Muy pocos datos recopilados: en el caso de que el sistema no capture la suficiente información como para reentrenarse, este no se reentrenará avisando al usuario mediante un mensaje informativo.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Descartar reentrenamientos
Actor	Usuario común
Precondiciones	Aplicación en la pantalla de reentrenar.
Postcondiciones	El modelo se reestablece a las configuraciones de serie y los valores de la optimización tambien se reestablecen a los de serie.
Flujo principal	El modelo y los valores de la optimización se reestablecen a los de serie.
Excepciones	Modelo sin reentrenar: Se avisa al usuario mediante un mensaje informativo.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Tableros comunicativos con o sin pictogramas
Actor	Usuario común
Precondiciones	Aplicación en la pantalla principal
Postcondiciones	Se ven las instrucciones de los tableros y las opciones para su uso con pictogramas o sin ellos.
Flujo principal	El usuario ve las instrucciones de uso y decide si utilizar los tableros con pictogramas o sin ellos.
Excepciones	-

Caso de uso	Escritura de palabra
Actor	Usuario común
Precondiciones	Tableros comunicativos abiertos.
Postcondiciones	La palabra de la casilla seleccionada se añade a la frase.
Flujo principal	El usuario selecciona una casilla y la palabra de esta se añade a la frase.
Excepciones	-

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Borrado único
Actor	Usuario común
Precondiciones	Tableros comunicativos abiertos.
Postcondiciones	La última palabra de la frase se borra.
Flujo principal	El usuario selecciona la casilla de borrar y la última palabra de la frase se elimina.
Excepciones	Ninguna palabra escrita: No sucede nada.

Caso de uso	Borrado total
Actor	Usuario común
Precondiciones	Tableros comunicativos abiertos.
Postcondiciones	La frase se borra.
Flujo principal	El usuario selecciona la casilla de borrar todo y la frase se elimina.
Excepciones	Ninguna palabra escrita: No sucede nada.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Reproducción de frases con síntesis de voz
Actor	Usuario común
Precondiciones	Tableros comunicativos abiertos.
Postcondiciones	La frase se conjuga si el conjugador está activo y se reproduce en alto.
Flujo principal	El usuario selecciona la casilla de reproducir, en el caso de que el conjugador esté desactivado, esta se reproduce, en caso contrario se conjuga y se reproduce en alto.
Excepciones	Volumen desactivado: No se escucha nada. Ninguna palabra escrita: Se avisa al usuario mediante un mensaje informativo.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Alarma
Actor	Usuario común
Precondiciones	Tableros comunicativos abiertos.
Postcondiciones	Suena una alarma.
Flujo principal	El usuario selecciona la casilla de alarma y esta se reproduce.
Excepciones	Volumen desactivado: No se escucha nada.

Caso de uso	Control ocular
Actor	Usuario común
Precondiciones	Tableros comunicativos abiertos y cámara seleccionada.
Postcondiciones	-
Flujo principal	El puntero sigue la dirección de la mirada del usuario, al mantenerlo sobre una casilla, cuando se el circulo que lo rodea llega a su tamaño máximo, la bloquea y una vez bloqueada, al parpadear es seleccionada.
Excepciones	Cara no detectada: Se avisa al usuario mediante un mensaje informativo.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Modo descanso
Actor	Usuario común
Precondiciones	Tableros comunicativos abiertos y cámara seleccionada.
Postcondiciones	Pantalla bloqueada o desbloqueada dependiendo del estado anterior.
Flujo principal	El usuario cierra los ojos durante 3 segundos para activar o desactivar el modo descanso, este bloquea el lector ocular y la pantalla.
Excepciones	Cara no detectada: Se avisa al usuario mediante un mensaje informativo.

C.3 Casos del usuario desarrollador

Caso de uso	Activar opciones desarrollador
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla principal.
Postcondiciones	Opciones de desarrollador activadas o desactivadas dependiendo del estado anterior.
Flujo principal	El desarrollador pulsa la tecla “D” y las opciones se activan o desactivan.
Excepciones	-

Caso de uso	Modo test
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla principal y las opciones de desarrollador activadas.
Postcondiciones	Se abre la pantalla de test
Flujo principal	El desarrollador entra en el modo test, donde puede ver el funcionamiento del modelo de seguimiento ocular.
Excepciones	Cámara no seleccionada: no se verá el modelo de seguimiento ocular en funcionamiento.

Caso de uso	Ajustar postprocesado
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla de test.
Postcondiciones	Los valores de los parámetros se modifican según lo que establezca el desarrollador.
Flujo principal	El desarrollador mueve los puntos visibles en la pantalla para modificar el postprocesado del modelo.
Excepciones	-

Caso de uso	Recopilar datos
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla principal y las opciones de desarrollador activadas.
Postcondiciones	Se muestra un mensaje de agradecimiento.
Flujo principal	El desarrollador inicia la recopilación y el voluntario sigue el punto con la mirada para así almacenar los datos que este ha proporcionado.
Excepciones	Errores del voluntario durante la recopilación: Se permite la posibilidad de descartar estos datos antes de guardarlos. Cámara no seleccionada o parpadeo no calibrado: Se avisa al usuario mediante un mensaje informativo.

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Pruebas
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla principal y las opciones de desarrollador activadas.
Postcondiciones	Se muestra la introducción a las pruebas.
Flujo principal	El desarrollador inicia las pruebas y se muestra la introducción a las mismas.
Excepciones	Cámara no seleccionada o parpadeo no calibrado: Se avisa al usuario mediante un mensaje informativo.

Caso de uso	Saltar prueba
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla de pruebas.
Postcondiciones	Se salta la prueba actual y se muestra la siguiente.
Flujo principal	El desarrollador salta la prueba actual, mostrando así la siguiente sin realizar la actual.
Excepciones	Última prueba: opción no disponible

APÉNDICE C. CASOS DE USO DEL SOFTWARE

Caso de uso	Prueba anterior
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla de pruebas.
Postcondiciones	Se muestra la prueba anterior a la actual.
Flujo principal	El desarrollador vuelve a la prueba anterior.
Excepciones	Primera prueba: opción no disponible

Caso de uso	Comenzar prueba
Actor	Usuario desarrollador
Precondiciones	Aplicación en la pantalla de pruebas.
Postcondiciones	Se muestra un agradecimiento y se guardan los resultados de la misma.
Flujo principal	El desarrollador inicia la prueba y el voluntario realiza lo indicado, guardando después los resultados.
Excepciones	-

Apéndice D

Diseño de la aplicación

En este apéndice se describen los aspectos clave del diseño de la aplicación, con especial atención en la interfaz de usuario y la experiencia del usuario final. Se incluyen los bocetos preliminares de las pantallas que componen la aplicación, explicando su funcionalidad y la disposición de los elementos. El diseño se ha realizado teniendo en cuenta la accesibilidad y facilidad de uso para los usuarios finales, lo que es esencial para un software destinado a personas con ELA.

D.1 Bocetos de las pantallas de la aplicación

A continuación, se presentan los bocetos de las pantallas que componen la interfaz de usuario de la aplicación. Estos bocetos fueron diseñados para asegurar una interacción intuitiva y accesible, cubriendo todas las funciones principales que los usuarios necesitan. El diseño preliminar se basa en las necesidades detectadas durante el análisis de requisitos, lo que asegura que la interfaz esté alineada con los objetivos del proyecto.

D.1.1 Bocetos de las pantallas de los usuarios comunes

En esta sección se presentan los bocetos de las pantallas diseñadas específicamente para los usuarios finales, quienes interactuarán con la aplicación en su uso cotidiano. El objetivo principal del diseño es proporcionar una interfaz accesible y fácil de usar, permitiendo que los usuarios naveguen por las distintas funcionalidades de manera intuitiva.

En la Figura D.1, se puede observar el boceto de la pantalla principal del software. Esta pantalla contiene las opciones clave de la aplicación, organizadas de manera clara para facilitar su acceso.

APÉNDICE D. DISEÑO DE LA APLICACIÓN

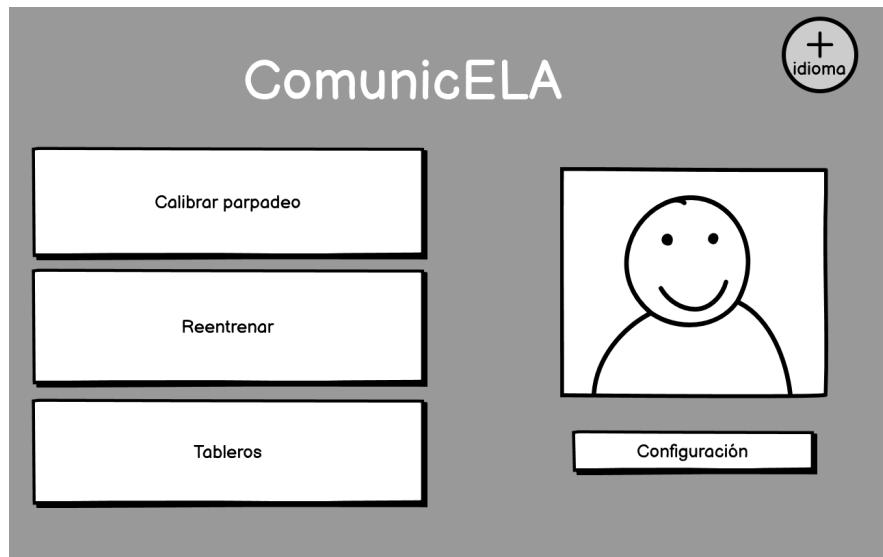


Figura D.1: Boceto de la pantalla principal.

A continuación, en la Figura D.2, se muestra el boceto del menú de configuración, donde los usuarios pueden personalizar diversos aspectos de la aplicación. Este menú ofrece opciones clave para adaptar la experiencia de uso según las preferencias individuales.

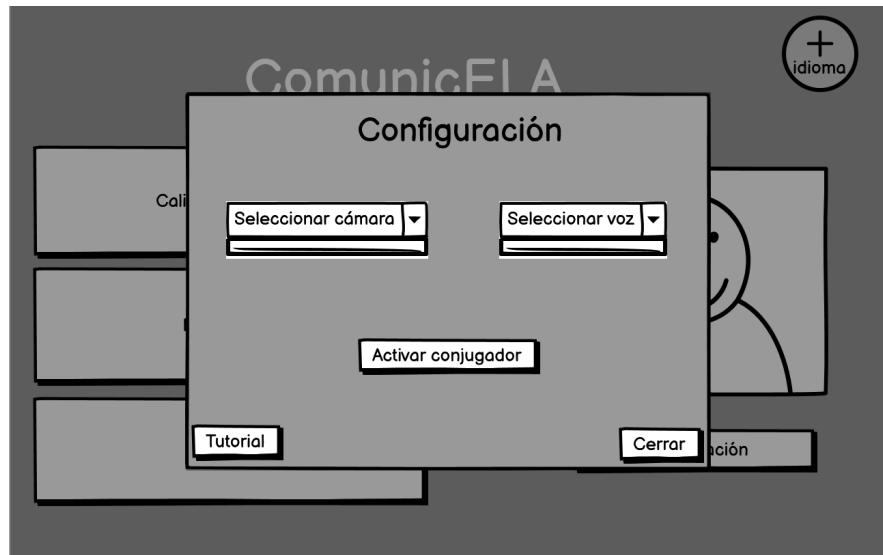


Figura D.2: Boceto del menú de configuración.

Una vez dentro del apartado “Calibrar parpadeo” (Figura D.3), se presentan tres fases, todas con un diseño similar. A la izquierda de la pantalla, se muestra una imagen representativa de cada fase, diseñada para que el proceso de calibración sea más intuitivo y fácil de seguir. Debajo de cada imagen se encuentran las instrucciones correspondientes, detallando los pasos

APÉNDICE D. DISEÑO DE LA APLICACIÓN

que el usuario debe seguir en cada fase del proceso. A la derecha de la pantalla, se muestra la imagen capturada por la cámara, lo que permite al usuario verificar en tiempo real el correcto funcionamiento de la calibración ocular. Este diseño ha sido optimizado para guiar al usuario de manera clara a través del proceso, asegurando que la calibración del parpadeo se realice de forma precisa y efectiva.

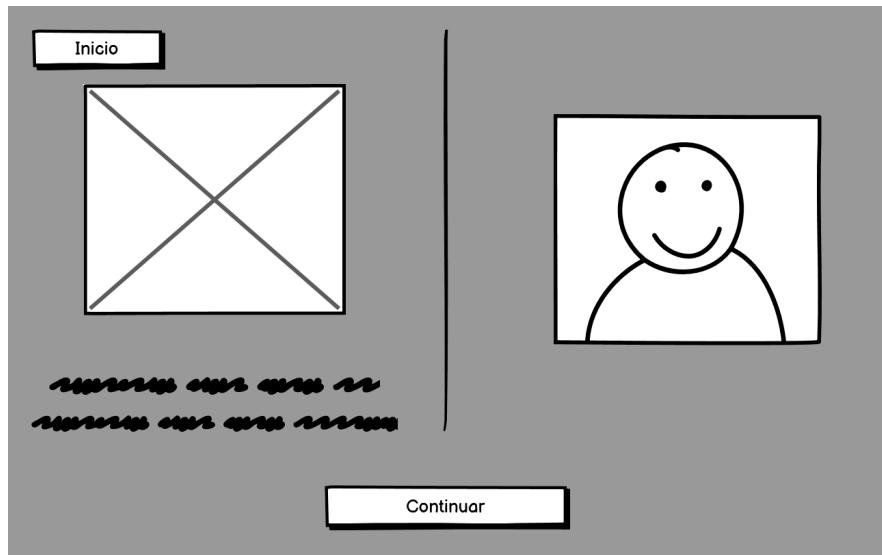


Figura D.3: Boceto de la pantalla de calibración.

Por otra parte, la pantalla de “Reentrenar” (Figura D.4) está diseñada para guiar al usuario a través del proceso de reentrenamiento. En el centro de la pantalla se encuentran las instrucciones detalladas sobre cómo proceder con el reentrenamiento, mientras que a la derecha se localiza un botón que permite iniciar el proceso. Además, hay una opción adicional para descartar los reentrenamientos realizados previamente, en caso de que sea necesario empezar desde el principio. Al pulsar el botón de reentrenar, se accede a una nueva pantalla (Figura D.5), donde el usuario verá un punto rojo que se moverá por la pantalla siguiendo una secuencia predefinida. Este movimiento permite que el sistema ajuste la calibración basada en los nuevos datos recogidos durante el reentrenamiento, mejorando la precisión del seguimiento ocular.

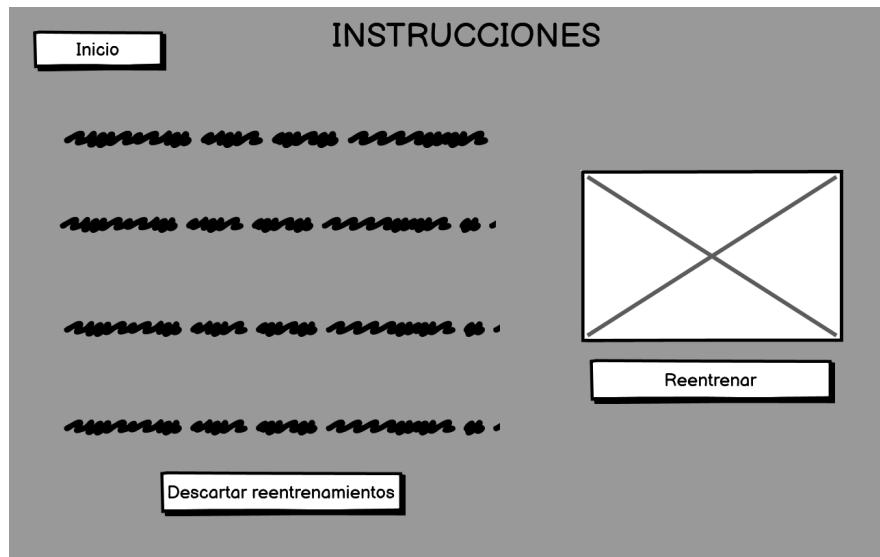


Figura D.4: Boceto de la pantalla de reentrenar.

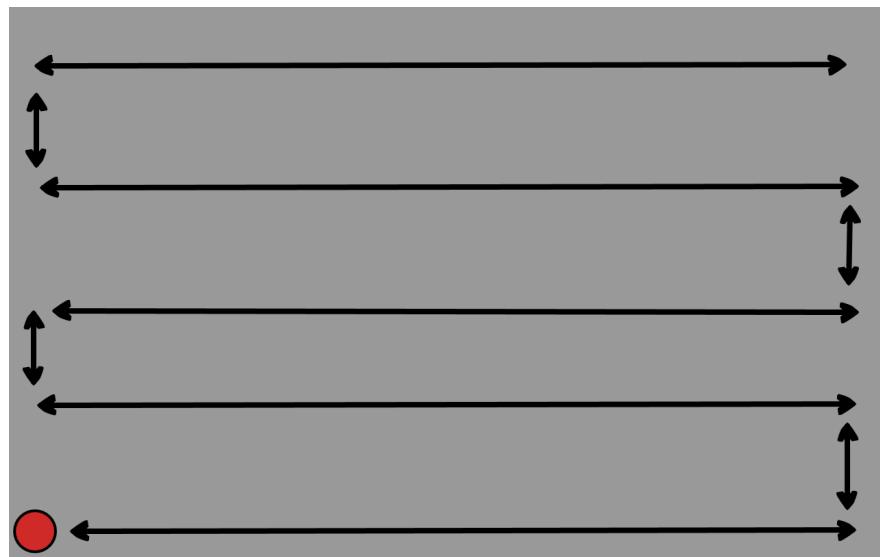


Figura D.5: Boceto de la obtención de datos.

La pantalla de los tableros (Figura D.6) presenta las instrucciones sobre cómo utilizar los tableros de comunicación, ofreciendo al usuario dos opciones principales: abrir los tableros con pictogramas o abrirlos con solo texto. Esta flexibilidad permite adaptar la interfaz a las preferencias o necesidades del usuario, facilitando la comunicación según el nivel de familiaridad con pictogramas o texto. En la Figura D.7, se muestra un ejemplo del tablero comunicativo con pictogramas, diseñado para aquellos usuarios que prefieren o necesitan comunicarse de forma visual. Por otro lado, en la Figura D.8, se presenta el mismo tablero, pero utilizando solo

APÉNDICE D. DISEÑO DE LA APLICACIÓN

texto, ofreciendo una opción más simple para los usuarios que prefieren una comunicación basada en palabras.

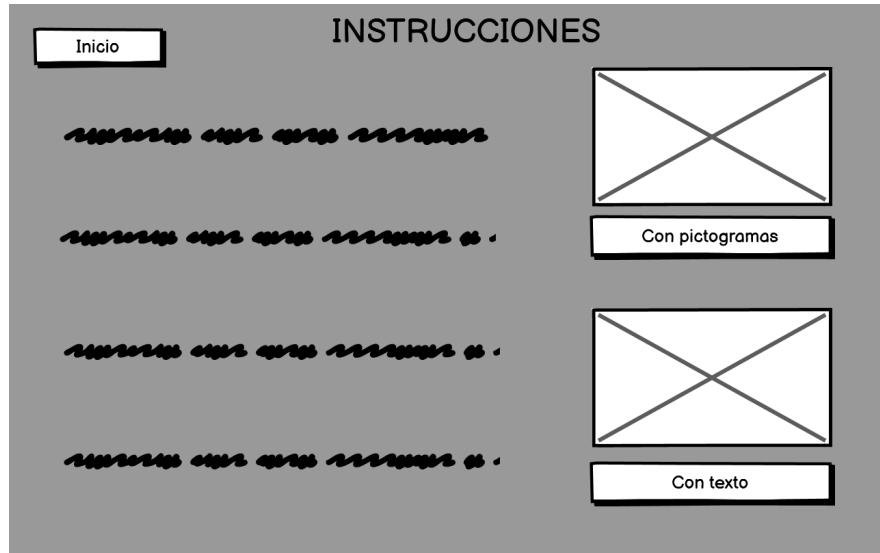


Figura D.6: Boceto de la pantalla de los tableros comunicativos.

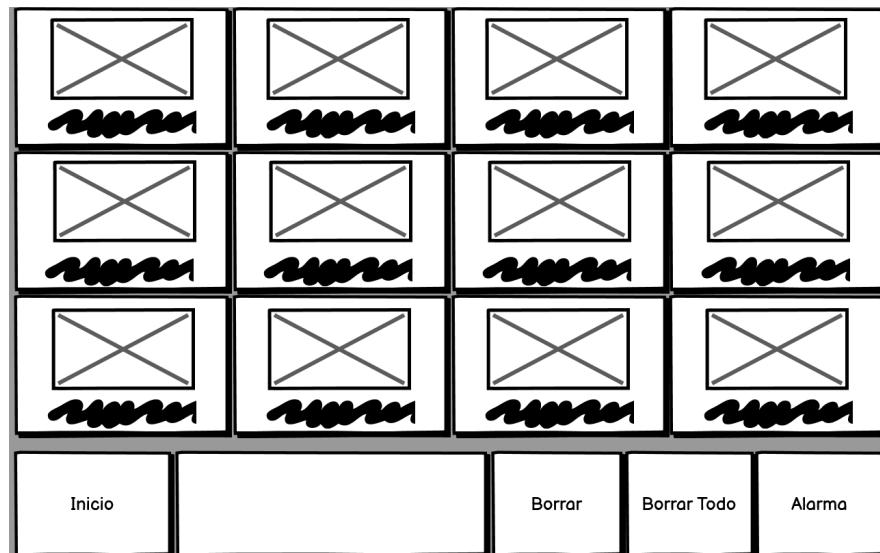


Figura D.7: Boceto de los tableros comunicativos con pictogramas.

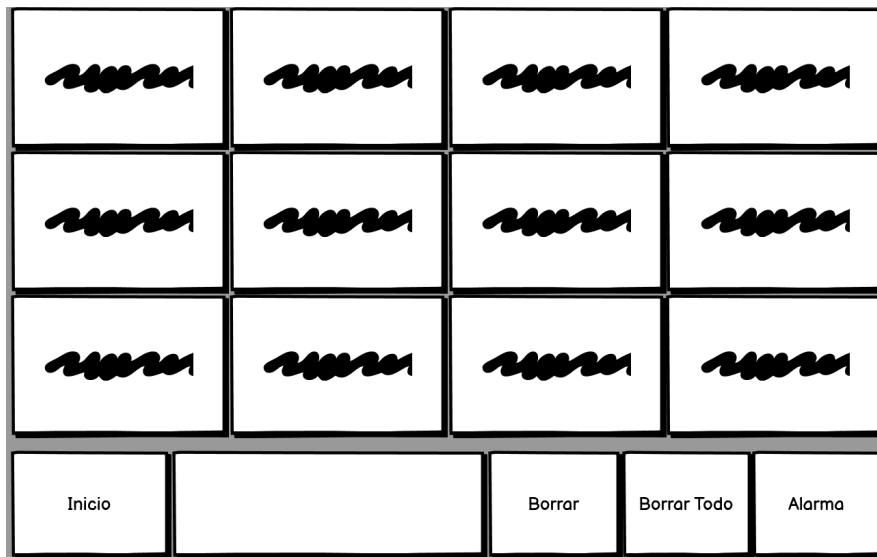


Figura D.8: Boceto de los tableros comunicativos sin pictogramas.

D.1.2 Bocetos de las pantallas de desarrollador

A continuación se presentan los bocetos de las pantallas diseñadas específicamente para el usuario desarrollador. Estas pantallas contienen opciones avanzadas que permiten acceder a funciones adicionales y realizar ajustes técnicos en la aplicación.

Al activar las opciones de desarrollador pulsando la tecla “D” en la pantalla principal, la estructura de la pantalla cambia, reorganizando los botones y añadiendo nuevas opciones de configuración. En la Figura D.9, se muestra cómo se reestructura la pantalla principal con las “opciones de desarrollador activadas”. Estas opciones adicionales permiten realizar tareas como el ajuste fino de parámetros, pruebas de nuevas funcionalidades y la modificación de configuraciones internas del sistema.

La pantalla de test se utiliza para personalizar el postprocesado de la red neuronal y verificar el funcionamiento del modelo del lector ocular. Esta pantalla permite al usuario desarrollador ajustar parámetros específicos del modelo y observar en tiempo real cómo se comporta el sistema en función de estos ajustes. En la Figura D.10 se puede observar la disposición de los elementos de esta pantalla, que facilita la visualización del modelo en acción y proporciona opciones de ajuste para mejorar la precisión y eficiencia del lector ocular.

Por otra parte, la pantalla de recopilar se puede observar en la Figura D.11. En esta pantalla, se muestra una imagen de referencia para que el usuario mantenga su posición en el punto donde se realizó la calibración, mientras espera que comience el proceso de recopilación de datos. Además, cuenta con un botón que permite iniciar dicho proceso.

El procedimiento de recopilación de datos sigue una estructura similar al del reentre-

APÉNDICE D. DISEÑO DE LA APLICACIÓN

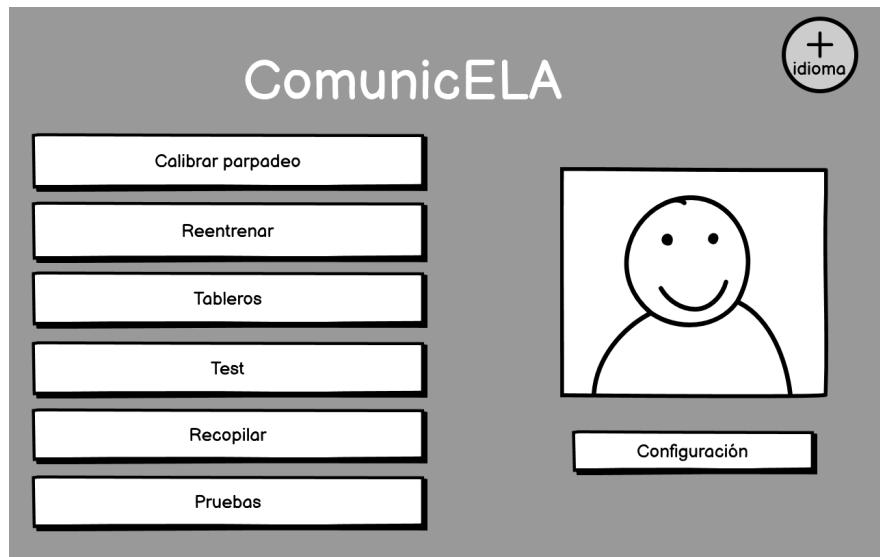


Figura D.9: Boceto de la pantalla principal con las opciones de desarrollador activadas.

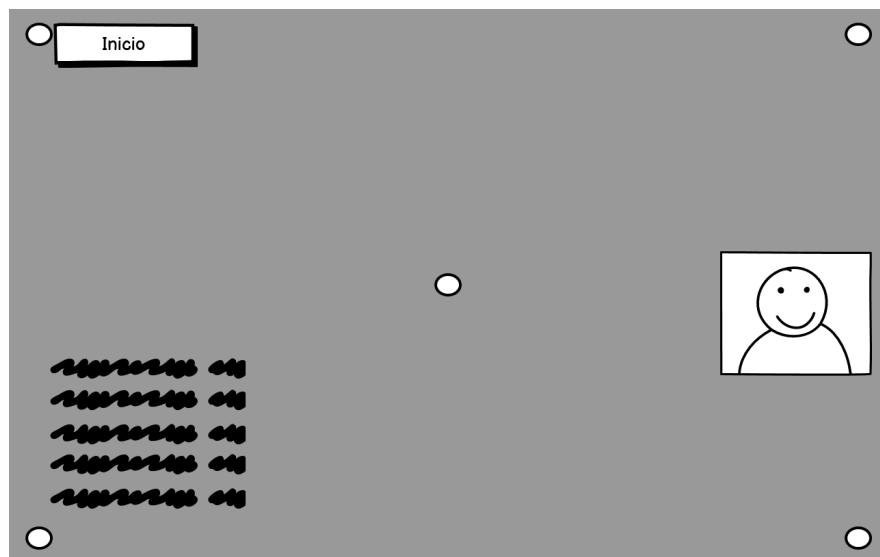


Figura D.10: Boceto de la pantalla de test.

namiento del modelo, por lo que la secuencia visual mostrada en la Figura D.5 también es representativa para este caso, con el punto rojo moviéndose por la pantalla en una secuencia predefinida que facilita la obtención de datos precisos para el lector ocular.

APÉNDICE D. DISEÑO DE LA APLICACIÓN



Figura D.11: Boceto de la pantalla de de recopilar.

Por último, durante las pruebas, se incluyen *PopUps explicativos* que guían al usuario en cada etapa del proceso. Estos PopUps presentan las opciones correspondientes para cada prueba, como se muestra en la Figura D.12.

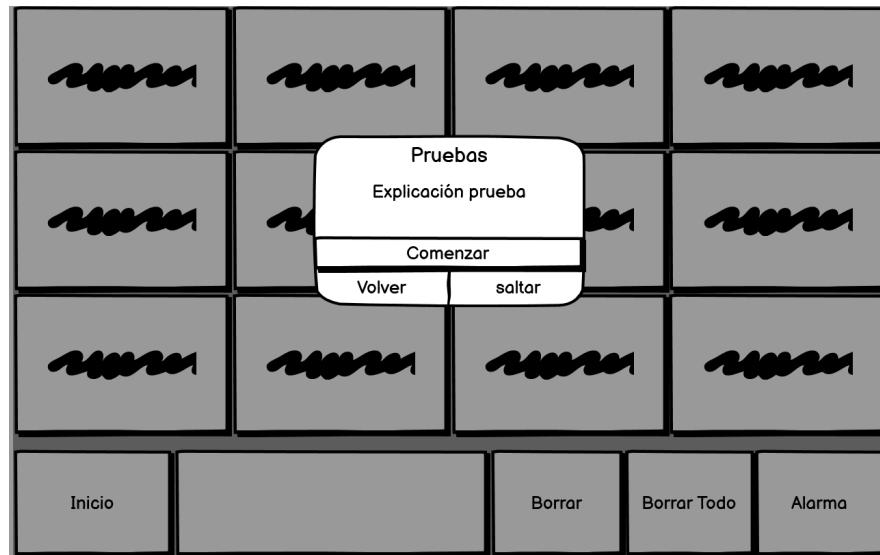


Figura D.12: Boceto de la pantalla de pruebas.

D.2 Pantallas finales de la aplicación

A continuación, se presentan las pantallas finales de la aplicación, que han sido diseñadas siguiendo de cerca los bocetos detallados en el apéndice anterior. El objetivo ha sido mantener la coherencia entre el diseño conceptual y la implementación final, garantizando que las funcionalidades y la estética propuestas inicialmente se reflejen en el producto final. Estas pantallas muestran la estructura definitiva de la interfaz, ajustada tras las pruebas y evaluaciones realizadas durante el proceso de desarrollo.

D.2.1 Pantallas destinadas a los usuarios comunes

En la pantalla *principal* (Figura D.13), se pueden observar las opciones previamente descritas, incluyendo los botones principales para la navegación y configuración. A la derecha de la pantalla, se encuentra una silueta que indica la posición donde debe colocarse el rostro del usuario, ayudando a alinear su posición correctamente para el control mediante el seguimiento ocular. Justo encima del botón de configuración se ubica esta silueta, mientras que el botón de configuración, al ser presionado, abrirá un menú flotante (Figura D.14) que permite realizar ajustes en la configuración del sistema.



Figura D.13: Pantalla principal.



Figura D.14: Menú flotante de configuración.

La pantalla de *calibrar* se puede observar en su versión final en la Figura D.15. Esta pantalla permite al usuario realizar el proceso de calibración de manera intuitiva, asegurando que el sistema esté correctamente ajustado para el control mediante seguimiento ocular.

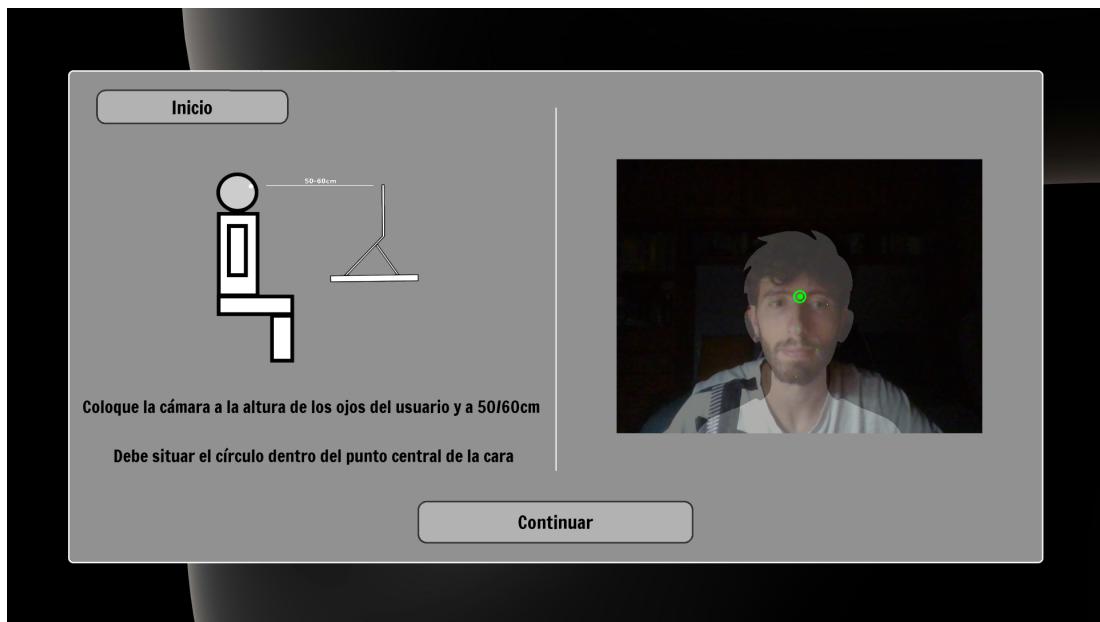


Figura D.15: Pantalla de calibración.

Por otra parte, en la pantalla de *reentrenar* (Figura D.16), se presenta una imagen en movi-

APÉNDICE D. DISEÑO DE LA APLICACIÓN

miento que ofrece una vista previa del proceso de reentrenamiento. Este proceso se detalla en la Figura D.17, donde se muestra una pelota roja moviéndose por la pantalla en una secuencia predefinida. Este movimiento es idéntico al presentado en el boceto original (Figura D.5), lo que permite al usuario seguir fácilmente la secuencia visual y completar el reentrenamiento de manera eficiente.

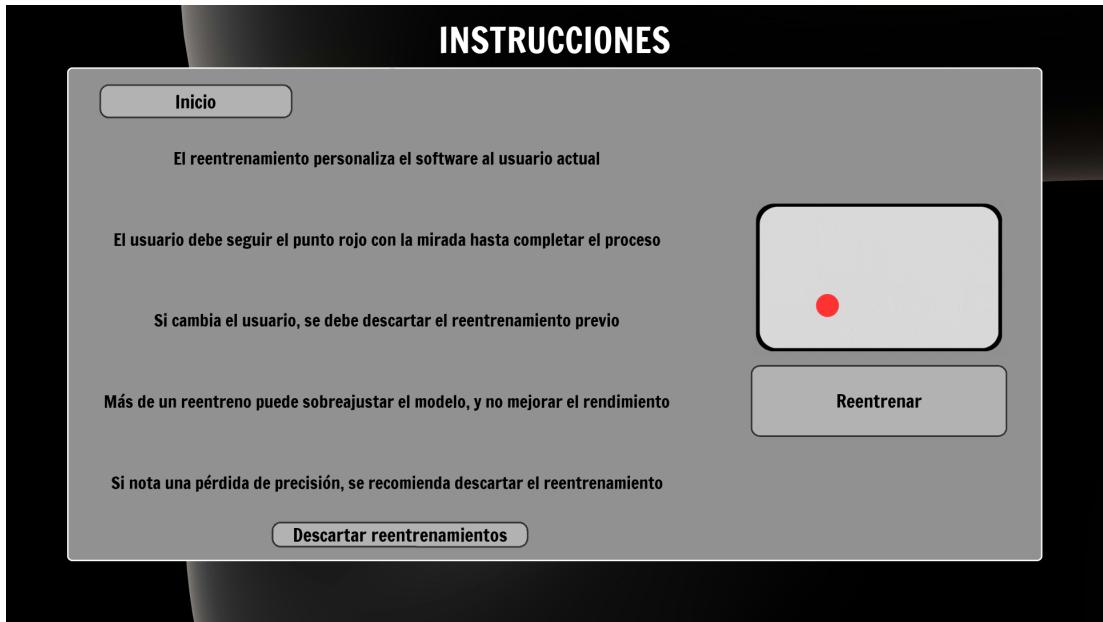


Figura D.16: Pantalla de reentrenar.

Por otra parte, la pantalla de los tableros (Figura D.18) se elige la opción de su uso con tableros (Figura D.19) o sin ellos (Figura D.20).

APÉNDICE D. DISEÑO DE LA APLICACIÓN

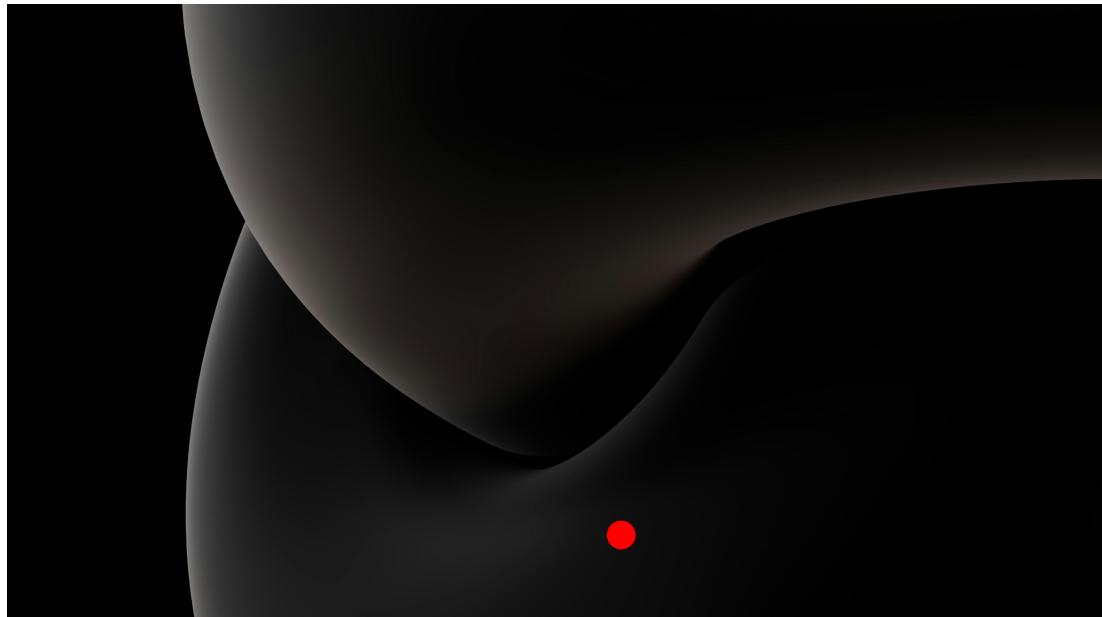


Figura D.17: Proceso de obtención de datos.

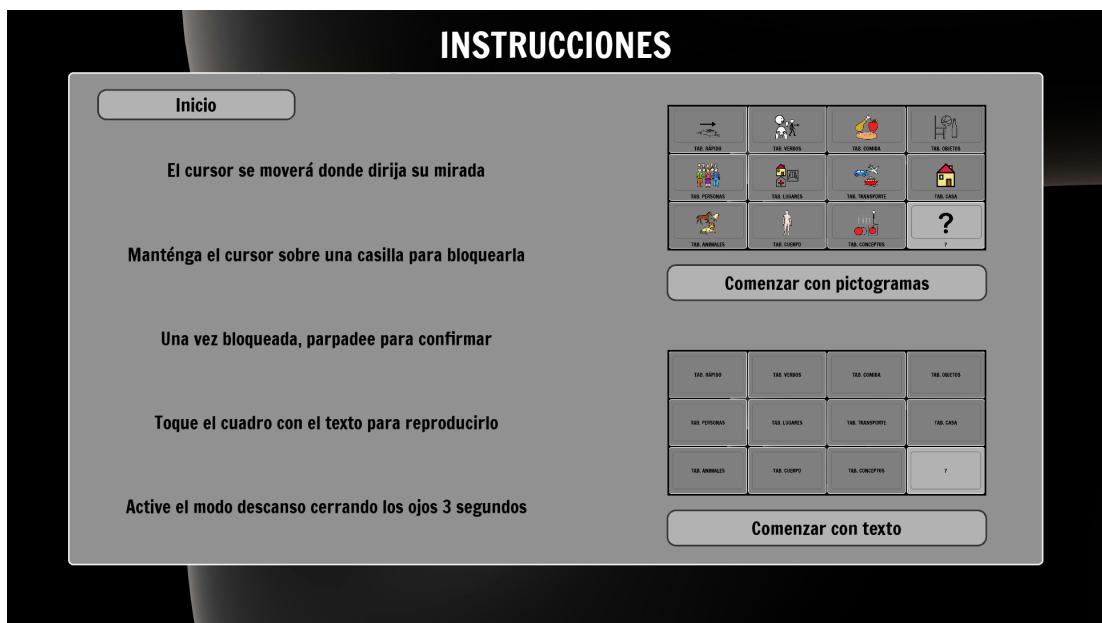


Figura D.18: Pantalla de los tableros comunicativos.

APÉNDICE D. DISEÑO DE LA APLICACIÓN

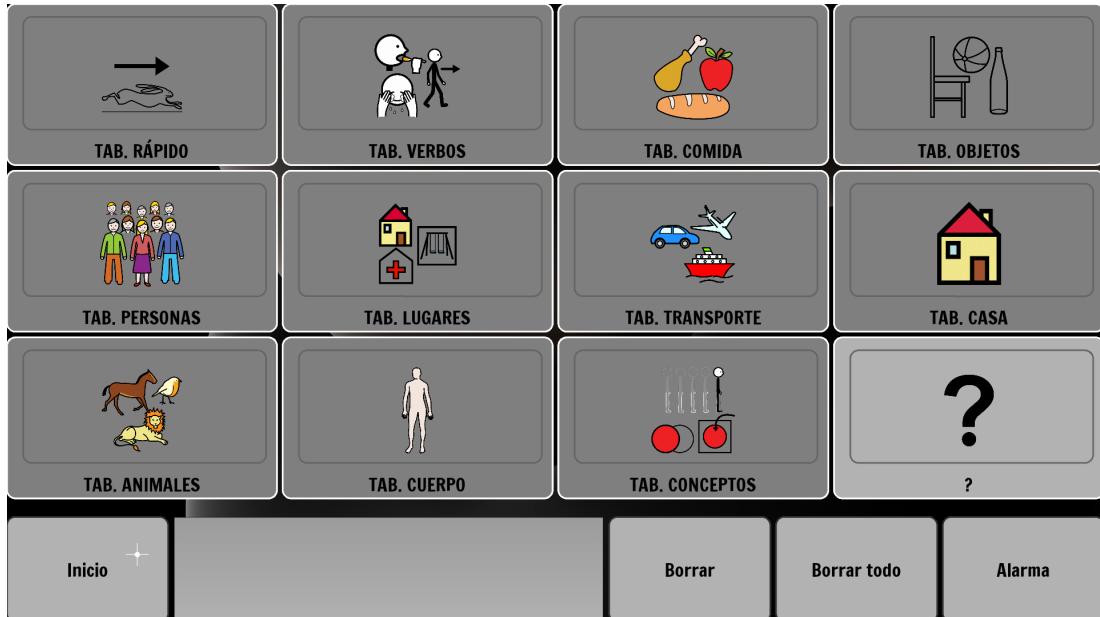


Figura D.19: Tableros comunicativos con pictogramas.

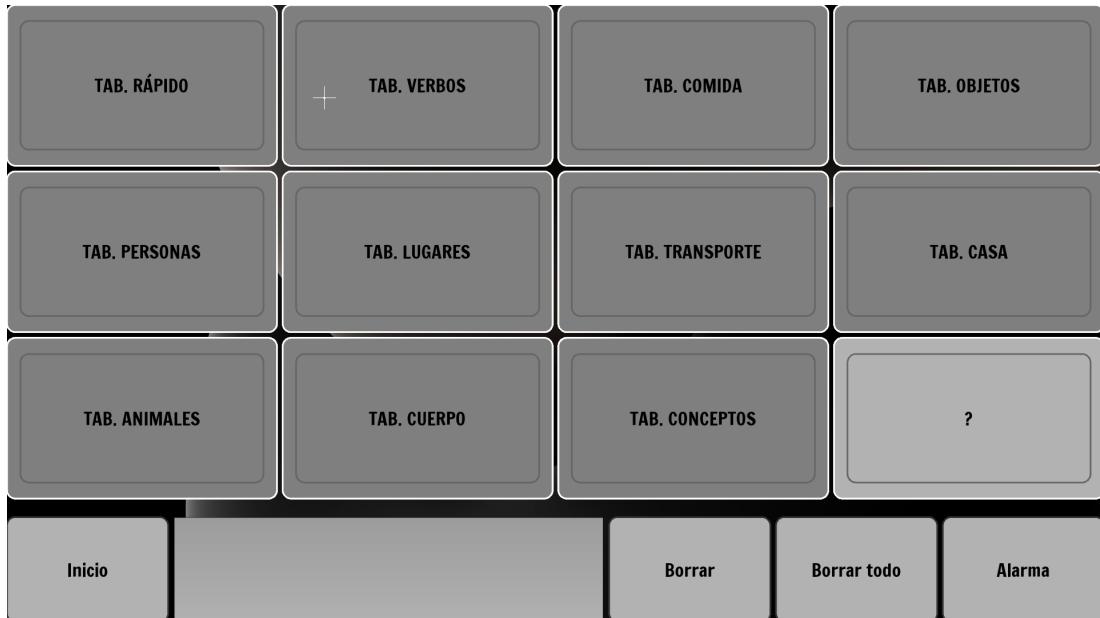


Figura D.20: Tableros comunicativos sin pictogramas.

D.2.2 Pantallas destinadas a los desarrolladores

En la Figura D.21, se presenta la pantalla *principal* redistribuida con las *opciones de desarrollador activadas*. Estas opciones proporcionan acceso a herramientas avanzadas para ajustar parámetros y probar funcionalidades específicas del sistema. Además, en esta pantalla aparece un mensaje explicativo que indica cómo desactivar las opciones de desarrollador, facilitando la transición entre los modos de usuario común y desarrollador.



Figura D.21: Pantalla principal con las opciones de desarrollador activadas.

En la pantalla de *test* (Figura D.22), se pueden observar los diferentes *puntos verdes arrastreables*, los cuales permiten al usuario desarrollador modificar el postprocesado del lector ocular. Estos puntos controlan parámetros específicos que influyen en el funcionamiento y precisión del sistema de seguimiento ocular, proporcionando una interfaz visual que facilita el ajuste de los valores en tiempo real. Al arrastrar y ajustar estos puntos, el desarrollador puede ver inmediatamente cómo los cambios afectan el rendimiento del modelo, lo que permite optimizar el postprocesado para mejorar la precisión y la experiencia del usuario final.

En la Figura D.23 se puede observar la pantalla de *recopilar*, cuyo diseño sigue el mismo patrón que el utilizado en el proceso de reentrenamiento. Al igual que en este último, la pantalla guía al usuario a través de la secuencia de movimientos visuales, utilizando una pelota roja que se desplaza por la pantalla. Este proceso de recolección de datos es fundamental para ajustar y optimizar el rendimiento del lector ocular.

APÉNDICE D. DISEÑO DE LA APLICACIÓN

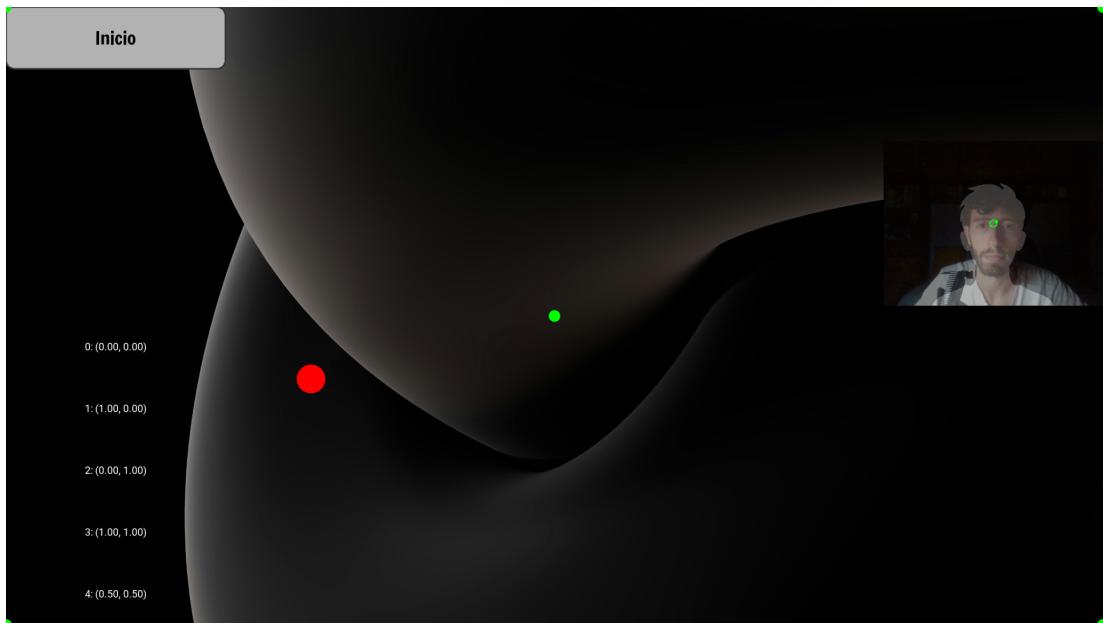


Figura D.22: Pantalla de test.

Dado que ambos procedimientos son similares, la Figura D.17, que muestra la secuencia de la pelota en el proceso de reentrenamiento, también es representativa para el caso de la recolección de datos. Esto asegura que la interfaz sea coherente y familiar para el usuario, facilitando el proceso de recopilación.

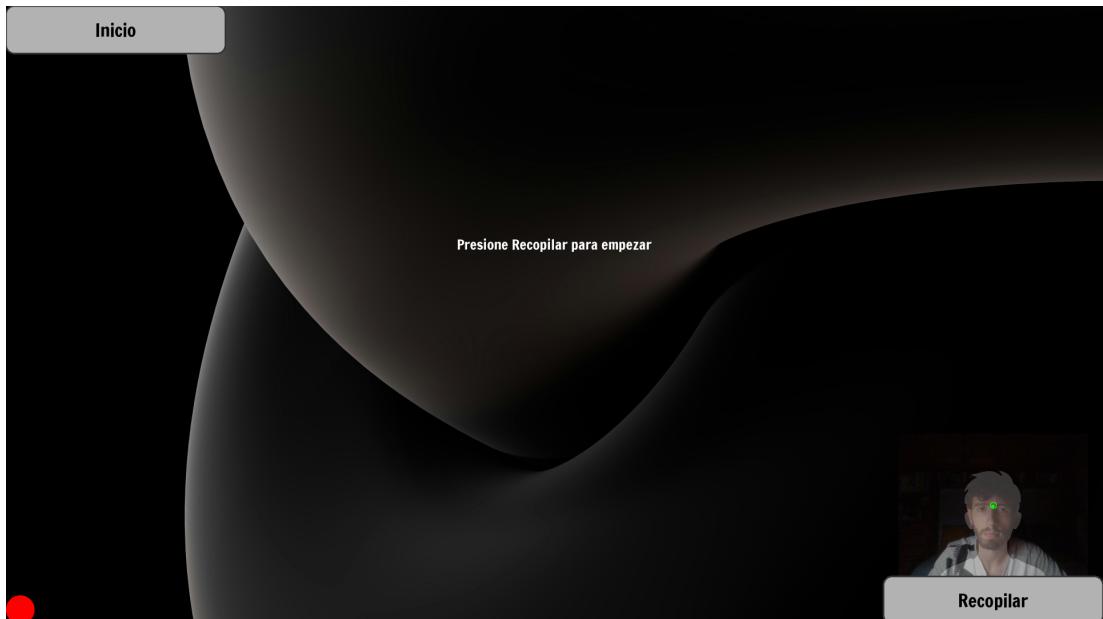


Figura D.23: Pantallade de recopilar.

APÉNDICE D. DISEÑO DE LA APLICACIÓN

Por último, en la Figura D.24 se puede observar la pantalla de *pruebas*. Tal como se describió previamente en los bocetos, esta pantalla ha sido diseñada con *PopUps* que aparecen sobre los tableros de comunicación. Los *PopUps* proporcionan instrucciones claras y permiten al usuario seguir el flujo de cada prueba de manera guiada, mientras mantienen visibles las opciones del tablero. Este diseño asegura que los usuarios puedan interactuar fácilmente con las pruebas sin perder de vista las herramientas de comunicación, mejorando tanto la usabilidad como la accesibilidad del sistema. Además, los *PopUps* ofrecen opciones para continuar con la prueba, saltarla o volver a una prueba anterior, proporcionando flexibilidad en el proceso de evaluación.



Figura D.24: Pantalla de pruebas.

Apéndice E

Modelos de la segunda y tercera aproximación

En este apéndice se detallan los modelos utilizados en las aproximaciones segunda y tercera del desarrollo del sistema. Cada uno de estos modelos ha sido diseñado con el objetivo de mejorar tanto la precisión como la eficiencia del lector ocular, optimizando su rendimiento a lo largo de las diferentes etapas del proyecto. A lo largo de las pruebas, se fueron implementando ajustes progresivos para refinar los modelos, asegurando un sistema más robusto y adaptado a las necesidades de los usuarios finales.

E.1 Modelos de la segunda aproximación

Esta sección está dedicada a la especificación de los modelos empleados en la segunda aproximación del desarrollo del sistema. Todos los modelos utilizados en esta fase se basan en redes neuronales convolucionales ([CNN](#)), debido a su capacidad para procesar eficientemente grandes cantidades de datos visuales, como las imágenes capturadas por el lector ocular.

E.1.1 Modelo 1

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolucional 1:** 4 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.
- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 80 neuronas.

- **Activación 2:** ReLU.
- **Capa Densa 2:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.1.2 Modelo 2

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolutinal 1:** 8 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.
- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 100 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 2:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.1.3 Modelo 3

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolutinal 1:** 8 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.
- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 150 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 2:** 100 neuronas.
- **Activación 3:** ReLU.
- **Capa Densa 3:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.1.4 Modelo 4

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolucional 1:** 8 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.
- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Capa Convolucional 2:** 16 filtros de tamaño 3×3 .
- **Activación 2:** ReLU.
- **Capa de Pooling 2:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 250 neuronas.
- **Activación 3:** ReLU.
- **Capa Densa 2:** 100 neuronas.
- **Activación 4:** ReLU.
- **Capa Densa 3:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.1.5 Modelo 5

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolucional 1:** 4 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.
- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 80 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 2:** 2 neuronas.

E.1.6 Modelo 6

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolucional 1:** 8 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.
- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 100 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 2:** 2 neuronas.

E.1.7 Modelo 7

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolucional 1:** 8 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.
- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 150 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 2:** 100 neuronas.
- **Activación 3:** ReLU.
- **Capa Densa 3:** 2 neuronas.

E.1.8 Modelo 8

- **Entrada:** Imágenes de tamaño 210×210 píxeles.
- **Capa Convolucional 1:** 8 filtros de tamaño 3×3 .
- **Activación 1:** ReLU.

- **Capa de Pooling 1:** Max pooling con tamaño de ventana 2×2 .
- **Capa Convolucional 2:** 16 filtros de tamaño 3×3 .
- **Activación 2:** ReLU.
- **Capa de Pooling 2:** Max pooling con tamaño de ventana 2×2 .
- **Aplanamiento:** Convierte la salida 2D en un vector 1D.
- **Capa Densa 1:** 2 neuronas.

E.2 Modelos de la tercera aproximación

Esta sección está dedicada a la especificación de las arquitecturas empleadas en la tercera aproximación del sistema, las cuales están basadas en arquitecturas híbridas. Estos modelos combinan las capacidades de las ([RNA](#)) y las ([CNN](#)) estudiadas en las aproximaciones anteriores.

El uso de arquitecturas híbridas permite aprovechar las fortalezas de ambas tecnologías: mientras que las [CNN](#) se encargan de procesar y extraer características clave de las imágenes, las [RNA](#) son responsables de realizar la clasificación o predicción basada en dichas características. Esta combinación optimiza el rendimiento global del sistema, mejorando tanto la precisión en el seguimiento ocular como la eficiencia en el procesamiento de los datos. A continuación, se detallan las especificaciones técnicas de los modelos híbridos utilizados, así como los resultados obtenidos durante las pruebas en esta tercera aproximación.

E.2.1 Modelo híbrido 1

- Entrada:
 - [RNA](#) [80 100] con 80 salidas.
 - [CNN E.1.7](#) con 80 salidas.
- **Capa Densa 1:** 80 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.2 Modelo híbrido 2

- Entrada:
 - RNA [80 100] con 80 salidas.
 - CNN E.1.7 con 80 salidas.
- **Capa Densa 1:** 200 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 100 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 3:** 100 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.3 Modelo híbrido 3

- Entrada:
 - RNA [80 100] con 80 salidas .
 - CNN E.1.7 con 50 salidas.
- **Capa Densa 1:** 80 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.4 Modelo híbrido 4

- Entrada:
 - RNA [80 100] con 25 salidas .
 - CNN E.1.7 con 15 salidas.
- **Capa Densa 1:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.5 Modelo híbrido 5

- Entrada:
 - RNA [80 100] con 25 salidas .
 - CNN E.1.7 con 15 salidas.
- **Capa Densa 1:** 50 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 30 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 3:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.6 Modelo híbrido 6

- Entrada:
 - RNA [80 100] con 25 salidas .
 - CNN E.1.7 con 15 salidas.
- **Capa Densa 1:** 100 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 50 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 3:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.7 Modelo híbrido 7

- Entrada:
 - RNA [80 100] con 25 salidas .
 - CNN E.1.7 con 15 salidas.
- **Capa Densa 1:** 30 neuronas.

- **Activación 1:** ReLU.
- **Capa Densa 2:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.8 Modelo híbrido 8

- Entrada:
 - [RNA](#) [80 100] con 25 salidas .
 - [CNN E.1.7](#) con 15 salidas.
- **Capa Densa 1:** 30 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 15 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 3:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.9 Modelo híbrido 9

- Entrada:
 - [RNA](#) [80 100] con 25 salidas .
 - [CNN E.1.7](#) con 15 salidas.
- **Capa Densa 1:** 30 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 20 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 3:** 10 neuronas.
- **Activación 3:** ReLU.
- **Capa Densa 4:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

E.2.10 Modelo híbrido 10

- Entrada:
 - RNA [80 100] con 25 salidas .
 - CNN E.1.7 con 15 salidas.
- **Capa Densa 1:** 30 neuronas.
- **Activación 1:** ReLU.
- **Capa Densa 2:** 20 neuronas.
- **Activación 2:** ReLU.
- **Capa Densa 3:** 10 neuronas.
- **Activación 3:** ReLU.
- **Capa Densa 4:** 5 neuronas.
- **Activación 4:** ReLU.
- **Capa Densa 5:** 2 neuronas.
- **Activación de Salida:** Sigmoide.

Lista de acrónimos

AGAELA Asociación Galega de Afectados pola Esclerose Lateral Amiotrófica. 3, 16–18, 23, 25, 30, 31, 63, 70, 71, 74, 76–78

BD Base de Datos. vii, viii, 29, 33, 34, 38–41, 49, 50, 52, 54, 59, 68, 72

CNN Red Neuronal Convolucional (Convolutional Neural Network). viii, 55–57, 125, 129–133

EAR Eye Aspect Ratio. 35, 39, 40, 52, 65–67

ELA Esclerosis Lateral Amiotrófica. 1–3, 6, 8, 9, 16, 21, 30, 74

EMS Mean Square Error. 41, 42, 50, 55, 56

IANA Internet Assigned Numbers Authority. 70

MVP Model View Presenter. 60

RNA Red de Neuronas Artificiales. 41, 42, 44, 49, 50, 59, 129–133

SVM Máquinas de Soporte Vectorial. 41–44

Glosario

Breath stacking Técnica utilizada para facilitar la tos mediante la acumulación de aire en los pulmones antes de exhalar.. [9](#)

PopUp Ventana emergente que muestra información o solicita una acción del usuario.. [31](#), [61](#), [72](#)

Spinner Widget desplegable con opciones para su selección que muestra el texto correspondiente al valor seleccionado actualmente.. [31](#)

Sprints Periodos de tiempo fijo, durante los cuales un equipo de Scrum trabaja para completar un conjunto específico de tareas y alcanzar un objetivo definido.. [15](#)

Toast Notificaciones que muestran información de manera temporal al usuario.. [32](#)

Bibliografía

- [1] M. de sanidad y política social, “Guía para la atención de la esclerosis lateral amiotrófica (ela) en España,” 2009. [En línea]. Disponible en: <https://www.sanidad.gob.es/profesionales/prestacionesSanitarias/publicaciones/docs/esclerosisLA.pdf>
- [2] O. Hardiman, L. H. Van Den Berg, and M. C. Kiernan, “Clinical diagnosis and management of amyotrophic lateral sclerosis,” *Nature reviews neurology*, vol. 7, no. 11, pp. 639–649, 2011.
- [3] Fundación Francisco Luzón, “Tipos de ELA.” [En línea]. Disponible en: <https://www.ffluzon.org/ela/tipos-de-ela>
- [4] Fundación AINDACE, “Esclerosis Lateral Amiotrófica (ELA).” [En línea]. Disponible en: <https://fundacionaindace.org/esclerosis-lateral-amiotrofica-ela/>
- [5] B. Poletti, F. Solca, L. Carelli, A. Diena, E. Colombo, S. Torre, A. Maranzano, L. Greco, F. Cozza, A. Lizio *et al.*, “Association of clinically evident eye movement abnormalities with motor and cognitive features in patients with motor neuron disorders,” *Neurology*, vol. 97, no. 18, pp. e1835–e1846, 2021.
- [6] AGAELA, “Qué es la ELA?” [En línea]. Disponible en: <https://agaela.es/que-es-la-ela>
- [7] National Institutes of Health, “Progresos en ELA.” [En línea]. Disponible en: <https://salud.nih.gov/recursos-de-salud/nih-noticias-de-salud/progresos-en-ela>
- [8] Clínica Universidad de Navarra, “Esclerosis Lateral Amiotrófica.” [En línea]. Disponible en: <https://www.cun.es/enfermedades-tratamientos/enfermedades/esclerosis-lateral-amiotrofica>
- [9] AGAELA, “Asociación Galega de Esclerosis Lateral Amiotrófica.” [En línea]. Disponible en: <https://agaela.es/>

BIBLIOGRAFÍA

- [10] Clínica Mayo, “Esclerosis Lateral Amiotrófica.” [En línea]. Disponible en: <https://www.mayoclinic.org/es/diseases-conditions/amyotrophic-lateral-sclerosis/diagnosis-treatment/drc-20354027>
- [11] Fundación Francisco Luzón, “Tratamiento de la ELA.” [En línea]. Disponible en: <https://www.fluzon.org/ela/tratamiento-de-la-ela>
- [12] Tobii, “Tobii.” [En línea]. Disponible en: <https://www.tobii.com/>
- [13] Samsung, “Tallk.” [En línea]. Disponible en: <https://www.samsung.com/es/tecnologiaconproposito/accesibilidad-bienestar/TALLK/>
- [14] “Asterics grid.” [En línea]. Disponible en: https://aulaabierta.arasaac.org/asterics-grid_inicio
- [15] ASoft.nl, “Asistente de voz aac.” [En línea]. Disponible en: <https://play.google.com/store/apps/details?id=nl.asoft.speechassistant&gl=DE>
- [16] K. Schwaber and J. Sutherland, “The scrum guide,” *Scrum Alliance*, vol. 21, no. 1, pp. 1–38, 2011.
- [17] C. oftalmológica Novovisión, “Partes del ojo humano – conceptos básicos.” [En línea]. Disponible en: <https://www.clinicasnovovision.com/blog/partes-del-ojo-humano/#:~:text=Directamente%20conectado%20al%20cerebro%2C%20el,entre%207%20y%208%20gramos>.
- [18] J. Cech and T. Soukupova, “Real-time eye blink detection using facial landmarks,” *Cent. Mach. Perception, Dep. Cybern. Fac. Electr. Eng. Czech Tech. Univ. Prague*, pp. 1–8, 2016.
- [19] A. Aflalo, “Eye gaze estimation using a webcam.” [En línea]. Disponible en: <https://medium.com/@amit.aflalo2/eye-gaze-estimation-using-a-webcam-in-100-lines-of-code-570d4683fe23>
- [20] OpenCV, “Filtro gaussiano.” [En línea]. Disponible en: https://docs.opencv.org/4.x/dc/dd3/tutorial_gaussian_median_blur_bilateral_filter.html#autotoc_md565
- [21] A. Requena, R. Quintanilla, J. Bolarín, A. Vázquez, A. Bastida, J. Zúñiga, and L. Tomás, “Nuevas tecnologías y contaminación de atmósferas, para pymes,” *Universidad de Murcia*. <https://www.um.es/LEQ/Atmosferas>, p. 3, 2020.
- [22] Y. Zhang and Y. Luo, “An architecture and implement model for model-view-presenter pattern,” in *2010 3rd international conference on computer science and information technology*, vol. 8. IEEE, 2010, pp. 532–536.

BIBLIOGRAFÍA

- [23] F. P. U. de Valladolid, “Patrones de diseño.” [En línea]. Disponible en: <https://www.infor.uva.es/~felix/datos/priii/tema7.pdf>
- [24] Microsoft, “Ispvoice.” [En línea]. Disponible en: [https://learn.microsoft.com/en-us/previous-versions/office/developer/speech-technologies/jj127823\(v=msdn.10\)](https://learn.microsoft.com/en-us/previous-versions/office/developer/speech-technologies/jj127823(v=msdn.10))
- [25] Google, “Gemini api.” [En línea]. Disponible en: <https://ai.google.dev/gemini-api>
- [26] “Optuna.” [En línea]. Disponible en: <https://optuna.org/>
- [27] I. A. N. Authority, “Service name and transport protocol port number registry.” [En línea]. Disponible en: <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml?search=53>
- [28] Microsoft, “Visual studio code.” [En línea]. Disponible en: <https://code.visualstudio.com/>
- [29] L. Torvalds, “Git.” [En línea]. Disponible en: <https://git-scm.com/>
- [30] Google, “Google colab.” [En línea]. Disponible en: <https://colab.research.google.com/>
- [31] Microsoft, “Github.” [En línea]. Disponible en: <https://github.com/>
- [32] P. S. Foundation, “Python.” [En línea]. Disponible en: <https://www.python.org/>
- [33] Google, “Mediapipe solutions.” [En línea]. Disponible en: <https://ai.google.dev/edge/mediapipe>
- [34] I. Corporation, “Opencv.” [En línea]. Disponible en: <https://opencv.org>
- [35] J. A. Clark, “Pillow.” [En línea]. Disponible en: <https://pypi.org/project/pillow/>
- [36] “Pytorch.” [En línea]. Disponible en: <https://pytorch.org/>
- [37] N. Corporation, “Cuda.” [En línea]. Disponible en: <https://developer.nvidia.com/cuda-toolkit>
- [38] NVIDIA, “cudnn - gpu accelerated deep learning,” <https://developer.nvidia.com/cudnn>, 2024.
- [39] “Scikit-learn.” [En línea]. Disponible en: <https://scikit-learn.org/stable/>
- [40] K. Organization, “Kivy.” [En línea]. Disponible en: <https://kivy.org/>
- [41] “Canva.” [En línea]. Disponible en: https://canva.com/es_es
- [42] “Arasaac.” [En línea]. Disponible en: <https://arasaac.org/>

BIBLIOGRAFÍA

- [43] S. Palao, “Pictogramas arasaac,” 2024, licencia: CC (BY-NC-SA). Propiedad: Gobierno de Aragón (España). [En línea]. Disponible en: <http://www.arasaac.org>
- [44] “Balsamiq.” [En línea]. Disponible en: <https://balsamiq.com/>
- [45] “Creately.” [En línea]. Disponible en: <https://creately.com/es/home/>
- [46] “Overleaf.” [En línea]. Disponible en: <https://es.overleaf.com>
- [47] “Latex.” [En línea]. Disponible en: <https://www.latex-project.org/>
- [48] P. Digital, “La guía salarial prosperity digital 2024,” cortesía de prosperitydigital.es, la agencia de reclutamiento especializada en el sector digital en España. [En línea]. Disponible en: <https://www.prosperitydigital.es/guia-salarial-espana-2024>
- [49] “Boletín oficial del estado,” 2024, núm. 108, Sec. III, Pág. 50744, Artículo 18.