Universidad San Jorge

Escuela de Arquitectura y Tecnología

Grado en Ingeniería Informática

Proyecto Final

Sistema de control de acceso para empresas mediante el uso de reconocimiento facial

Autor del proyecto: Jon Imaz Dravasa

Director del proyecto: Rafael del Hoyo Alonso

Zaragoza, 26 de junio de 2023

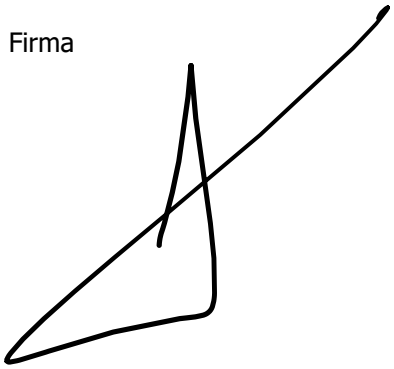
universidad SANJORGE
GRUPO SANVALERO

Este trabajo constituye parte de mi candidatura para la obtención del título de Graduado en Ingeniería Informática por la Universidad San Jorge y no ha sido entregado previamente (o simultáneamente) para la obtención de cualquier otro título.

Este documento es el resultado de mi propio trabajo, excepto donde de otra manera esté indicado y referido.

Doy mi consentimiento para que se archive este trabajo en la biblioteca universitaria de Universidad San Jorge, donde se puede facilitar su consulta.

Firma



Fecha

26 de junio del 2023

Dedicatoria y Agradecimiento

Con la finalización de este proyecto de fin de grado, quiero tomar un momento para reconocer y agradecer a las personas que han sido fundamentales en este proceso. A lo largo de este proyecto, he aprendido mucho y ha sido una experiencia valiosa para muchos aspectos de mi vida.

En primer lugar, deseo extender mi más sincero agradecimiento a mi mentor, Rafael del Hoyo, cuya sabiduría y dedicación han sido clave para mí. Gracias por estar siempre pendiente de mis avances y por estar siempre dispuesto a sacar tiempo para guiarme en todo lo que haya necesitado. Simplemente, me he sentido muy querido por ti y he disfrutado mucho de este tiempo juntos.

A mi familia, quiero decirles que son la razón de mis logros. Aunque la distancia nos haya separado físicamente en esta temporada, la resonancia de su apoyo y sacrificios me ha acompañado en cada paso. Este logro es tan mío como suyo, y es un testimonio de las bases que han establecido en mi vida.

También quiero dar las gracias a mis amigos, especialmente a Marta Simón y Germán Sánchez. Vosotros habéis estado a mi lado durante todo este proceso, y vuestro apoyo y ánimo han sido increíbles. Los buenos amigos hacen que los desafíos sean más fáciles de enfrentar.

Por último, una dedicatoria especial para mi novia, Hannah Pollet, mi compañera y confidente. En los momentos más difíciles, cuando la presión parecía insuperable, encontré consuelo y fuerza en tu apoyo y amor incondicional. Tu capacidad de comprenderme y tranquilizarme han sido un pilar en el que me he podido apoyar cuando lo he necesitado. No hay palabras que puedan expresar lo agradecido que estoy por tener a alguien tan maravillosa a mi lado.

En resumen, gracias a todos los que han contribuido de alguna manera a este proyecto. Vuestro apoyo ha sido invaluable, y estoy agradecido por cada uno de vosotros.

Índice de contenido

Índice de Figuras

Índice de tablas

Resumen

Este proyecto de grado propone diseñar e implementar un sistema de control de acceso avanzado para empresas mediante el uso de algoritmos de reconocimiento facial y *Deep Learning* en un dispositivo *hardware* embebido. El sistema ha sido desarrollado para superar las limitaciones de los métodos de seguridad convencionales basados en claves físicas o criptográficas, y en su lugar, ofrecer una solución que mejore sustancialmente la seguridad y eficiencia del proceso de autenticación.

El proyecto explora el uso de tecnologías biométricas de reconocimiento facial, centrándose en la aplicación de técnicas de *Deep Learning* para garantizar una alta precisión y robustez en la identificación facial, incluso en situaciones desafiantes donde la selección del algoritmo de reconocimiento facial y el *hardware* integrado se ha realizado después de un análisis exhaustivo de la tecnología actual y teniendo en cuenta tanto la robustez como los requisitos de *hardware*.

El sistema que ha sido desarrollado incluye la implementación de un módulo de gestión y control de usuarios, que permite la identificación de múltiples personas, la administración de los derechos de acceso y almacenar datos de forma segura en una base de datos. Además, con objetivo de asegurar su eficacia y confiabilidad, el sistema se ha sometido a un riguroso proceso de prueba y validación.

En resumen, este proyecto proporciona una solución innovadora y eficaz, mediante el uso de inteligencia artificial, para mejorar la seguridad de acceso a las instalaciones corporativas. Con este proyecto se demuestra el gran potencial de la tecnología de reconocimiento facial y *Deep Learning* en el campo de la seguridad.

Abstract

This degree project proposes to design and implement an advanced access control system for enterprises using facial recognition algorithms and Deep Learning in an embedded hardware device. The system has been developed to overcome the limitations of conventional security methods based on physical or cryptographic keys, and instead, offer a solution that substantially improves the security and efficiency of the authentication process.

The project explores the use of biometric facial recognition technologies, focusing on the application of Deep Learning techniques to ensure high accuracy and robustness in facial identification, even in challenging situations where the selection of the facial recognition algorithm and embedded hardware has been made after a thorough analysis of the current technology and considering both robustness and hardware requirements.

The system that has been developed includes the implementation of a user management and control module, which allows the identification of multiple people, the administration of access rights and the secure storage of data in a database. In addition, to ensure its effectiveness and reliability, the system has undergone a rigorous testing and validation process.

In summary, this project provides an innovative and effective solution, using artificial intelligence, to improve the security of access to corporate facilities. This project demonstrates the great potential of facial recognition technology and Deep Learning in the field of security.

1. Introducción

1.1. Descripción del problema

La seguridad hoy en día es una de las mayores preocupaciones para los seres humanos, este es el motivo por el que existe el control de acceso. Es un elemento fundamental en la seguridad de cualquier instalación, más aún si se trata de empresas o corporaciones donde una brecha de seguridad puede poner en peligro tanto la información como a los empleados de esta. Por esta razón, los sistemas de control de acceso son vitales para prevenir este tipo de eventos, convirtiéndose en una parte vital de la seguridad, ya que estos sistemas presentan varios problemas e inconvenientes que pueden afectar tanto a la seguridad como a la comodidad de los usuarios.

Los sistemas de control de acceso también permiten a las empresas cumplir con las normativas de seguridad establecidas, así como las políticas internas de seguridad para garantizar la seguridad de los empleados (Alarcón et al., 2017). Además, la implementación de estos sistemas puede contribuir a mejorar la eficiencia operativa de la empresa, permitiendo una mejor gestión de los recursos humanos y materiales.

La mayoría de las empresas ya cuentan con algún sistema de control de acceso implementado, aunque la mayoría de los casos se traten de llaves físicas, tarjetas de identificación o códigos numéricos. Sistemas que pueden ser vulnerables a la pérdida de llaves o clonación de tarjetas, además, estos sistemas no tienen ninguna forma de evitar el acceso no autorizado, siempre y cuando estos posean las llaves o tarjetas de identificación.

Otro inconveniente de estos sistemas se presenta cuando un empleado deja la empresa, el cambio de cerradura o reprogramación de los sistemas de acceso es necesario para garantizar que ninguna persona no autorizada pueda entrar en las instalaciones. Este proceso puede ser costoso y engorroso, especialmente si la compañía en cuestión cuenta con un alto volumen de personal.

Hablando de seguridad, los humanos siempre han sido considerados la mayor vulnerabilidad informática ya que el informe de investigación de brechas de datos de Verizon en 2019 reveló que el 34% de las violaciones de datos se debieron a factores humanos (Selvam, 2020). Por esto podemos afirmar que los humanos representan uno de los eslabones más débiles en la cadena de ciberseguridad. En este contexto, la implementación de sistemas de control de acceso sólidos y confiables se vuelve crucial. Al utilizar tecnologías como el reconocimiento facial, se puede minimizar la dependencia de métodos de autenticación basados en la interacción humana, que a menudo son susceptibles a errores humanos, pérdida o robo. Un sistema de control de acceso mediante reconocimiento facial puede automatizar el proceso de autenticación, y al hacerlo, reduce las vulnerabilidades asociadas con los factores humanos, lo que resulta en un entorno más seguro y protegido.

1.2. Antecedentes y justificación del proyecto

El crecimiento de la tecnología biométrica les ha permitido a los sistemas de control de acceso basados en reconocimiento facial convertirse en una solución viable para la identificación de personas en multitud de aplicaciones, desde desbloquear dispositivos móviles hasta controlar el acceso a edificios. Entre las distintas modalidades biométricas, el reconocimiento facial destaca por su carácter no intrusivo y potencial de alto rendimiento.

A pesar de los beneficios inherentes del reconocimiento facial, su aplicación en los sistemas de control de acceso sigue siendo reducida, en gran parte debido a las limitaciones de la tecnología

existente. Entre estas limitaciones se incluyen la alta demanda de poder de cálculo, que resulta especialmente problemática para sistemas embebidos con recursos limitados; la sensibilidad a las variaciones en las condiciones de iluminación, que puede afectar a la precisión del reconocimiento; la privacidad y seguridad de los datos, dado que el almacenamiento y procesamiento de datos biométricos pueden ser sensibles; y los desafíos en la escalabilidad. En este contexto, es imperativo trabajar en superar estas limitaciones para aprovechar de manera efectiva el potencial del reconocimiento facial en sistemas de control de acceso, especialmente en plataformas con capacidades de procesamiento más limitadas, como los sistemas embebidos.

Sin embargo, el reconocimiento facial basado en *Deep Learning* se ha mostrado muy prometedor para superar algunas de estas limitaciones. El *Deep Learning* es una subcategoría de inteligencia artificial que utiliza redes neuronales de varias capas para modelar y comprender datos complejos. En el contexto del reconocimiento facial, el *Deep Learning* se puede usar para extraer y aprender características faciales que puedan utilizarse posteriormente para una identificación sólida incluso en situaciones difíciles.

El uso de técnicas de *Deep Learning* en el sistema aumentará la precisión y robustez del reconocimiento facial (Schroff et al., 2015). Esto es especialmente importante en situaciones en las que la cara está ligeramente oculta o cuando se usan máscaras, ya que estas cosas dificultan la identificación correcta de las personas. Por otro lado, la implementación del sistema en dispositivos embebidos con recursos limitados tiene como objetivo garantizar una solución rentable y sencilla para las empresas. El sistema debe ser fácil de instalar y operar y tiene como objetivo evitar ninguna inversión en costosos equipos de mantenimiento y evitar las comunicaciones con otros sistemas, por ejemplo, en la nube, en caso de algún ataque a alguno de los sistemas de la compañía, el sistema de control de acceso podría verse afectado.

En definitiva, el control de acceso basado en reconocimiento facial permite a las empresas aumentar su seguridad al tener más control sobre quién entra y sale de sus instalaciones, además de brindar una solución eficiente, precisa y rentable para la identificación de las personas en las empresas (Bravo et al., 2018).

El proyecto que se presenta es novedoso por varias razones clave. En primer lugar, aprovecha las capacidades del *Deep Learning*, que es una de las tecnologías más avanzadas en el campo de la inteligencia artificial, para mejorar la precisión y confiabilidad del reconocimiento facial. Esto es crítico para minimizar los falsos negativos y falsos positivos, lo cual es esencial para la seguridad. Además, el sistema está diseñado para ser robusto frente a diversas condiciones, como variaciones en la iluminación y ocultamiento parcial de la cara, lo que tradicionalmente ha sido un desafío en el reconocimiento facial. También, la capacidad de ser embebido en *hardware*, como sistemas de bajo consumo, lo hace altamente adaptable y escalable. Esto no solo permite una implementación más fácil y rentable, sino que también abre la puerta para ser utilizado en una amplia gama de aplicaciones y entornos.

En términos más generales, este proyecto tiene el potencial de contribuir a un cambio de paradigma en los sistemas de control de acceso, reemplazando los sistemas obsoletos tales como las llaves físicas, tarjetas magnéticas y códigos numéricos con una solución biométrica avanzada. Este cambio en las empresas conllevaría a un aumento en la seguridad de sus instalaciones, al mismo tiempo que estarían mejorando la comodidad de sus usuarios, facilitando la gestión del control de acceso.

1.3. Objetivos del proyecto

El objetivo principal de este proyecto es ofrecer una solución al sistema de control de acceso basado en reconocimiento facial utilizando algoritmos de *Deep Learning*, optimizado para su implementación en un sistema embebido con poder de cálculo limitado. Por ello, para que el sistema de reconocimiento facial sea factible, se han planteado los siguientes objetivos:

- Analizar el estado del arte y los algoritmos para la identificación facial:
  Se realizará una revisión exhaustiva de la literatura existente y de los trabajos previos en el campo del reconocimiento facial, con un enfoque en las técnicas de *Deep Learning*. Dicha revisión permitirá obtener una visión completa y actualizada del campo, identificar las mejores prácticas y detectar posibles oportunidades de mejora.
- Escoger el sistema embebido a utilizar:
  Antes de la implementación, se seleccionará el sistema embebido más adecuado. Se tomarán en cuenta factores como la capacidad de cálculo, la memoria, el consumo energético, el costo y otros aspectos relevantes.
- Evaluar la mejor solución basándose en robustez y requisitos de *hardware*:
  Tras analizar el estado del arte, se seleccionan y evalúan diferentes algoritmos y técnicas de reconocimiento facial, teniendo en cuenta su robustez (es decir, capacidad para manejar variaciones de luz, postura, ocultación, entre otros factores) y sus requisitos de *hardware*.
- Desarrollar una aplicación para un sistema embebido de reconocimiento facial:
  Este objetivo implica la implementación real del algoritmo elegido en el sistema embebido elegido, lo que requerirá ajustar y optimizar el algoritmo para que funcione con recursos limitados que dispongamos.
- Permitir el reconocimiento de varias personas diferentes:
  El sistema desarrollado será capaz de identificar a múltiples usuarios diferentes, lo que implica la capacidad de manejar y distinguir entre múltiples conjuntos de características faciales.
- Desarrollar un sistema de gestión y control de los distintos usuarios:
  Además del desarrollo de un sistema de reconocimiento facial, también se desarrollará un sistema de gestión y control de usuarios que permitirá agregar y eliminar usuarios, asignar y revocar derechos de acceso y monitorear los registros de acceso, que serán guardados posteriormente en un fichero.
- Almacenar de forma segura los datos en una base de datos:
  También, se implementará una base de datos para almacenar de forma segura los datos de los usuarios y los registros de acceso. Esta base de datos debe diseñarse e implementarse de manera que garantice la integridad y seguridad de los datos, en cumplimiento con la normativa vigente en materia de protección de datos.
- Implementar el sistema en un sistema embebido de recursos limitados:
  Finalmente, el sistema en su totalidad será implementado en el sistema embebido seleccionado. Hay que considerar esto, ya que este sistema dispondrá de recursos muy limitados y la eficiencia será un punto clave del desarrollo.

En resumen, este proyecto busca no solo desarrollar un sistema de reconocimiento facial basado en *Deep Learning*, sino también integrar este sistema en una solución completa de control de acceso que sea segura, eficiente y fácil de usar para cualquiera.

1.4. Metodología del trabajo

La metodología Scrum ha sido elegida para desarrollar este proyecto. Scrum es una estructura de trabajo ágil que permite desarrollar e implementar proyectos en fases, enfocado en obtener

el máximo valor en el menor tiempo posible. Aunque el método Scrum es ampliamente utilizado y efectivo, ha sido difícil de implementar por completo debido a la carga de trabajo. A pesar de estos desafíos, se ha tratado de utilizar los elementos más importantes de Scrum que fueran aplicables y valiosos en la gestión del proyecto.

Herramientas:

Para la organización y gestión de tareas, se ha utilizado Trello [1] como herramienta de administración de proyectos, aunque funcionó principalmente como backlog, tal y como se puede observar en la figura, y no fue la opción más adecuada para la administración del proyecto. Para la comunicación y coordinación se empleó Microsoft Teams [2] y correos electrónicos. Adicionalmente, se hizo uso de ChatGPT [3], como una herramienta utilizada para mejorar la calidad del lenguaje y la redacción en diversas secciones del proyecto. Esta última herramienta también ha servido como una fuente de orientación y asistencia en el desarrollo del prototipo, proporcionando sugerencias que contribuyeron a la efectividad del proceso de desarrollo.

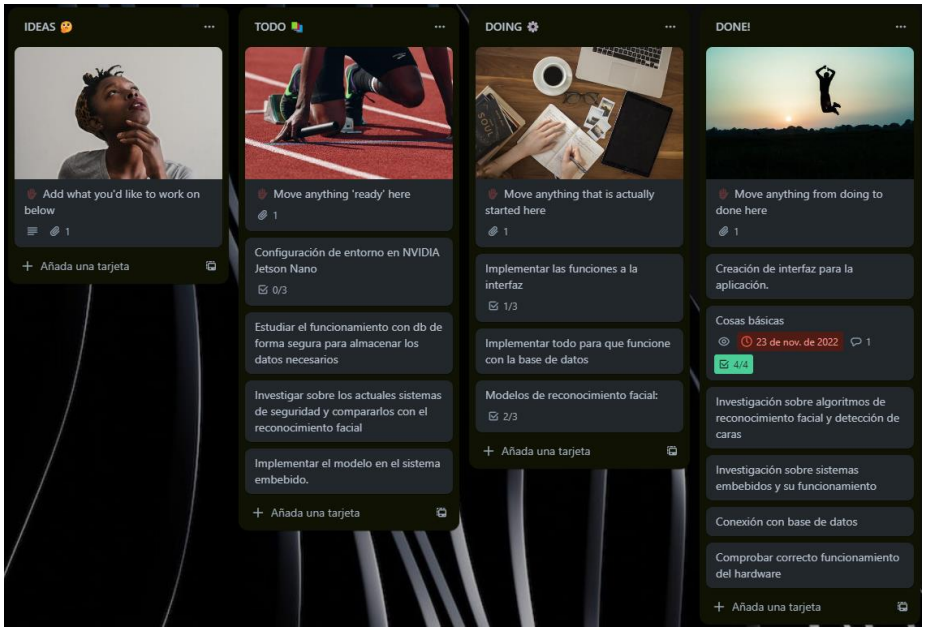


Figura 1: Gestión de tareas mediante Trello

FUENTE: ELABORACIÓN PROPIA

Reuniones:

Se realizaron reuniones regulares para discutir el progreso del proyecto y recibir retroalimentación. En este caso, el tutor actuó como cliente dentro de la metodología Scrum. A continuación, se muestra un resumen de las reuniones:

10/11/2022: Se acordó realizar una reunión cada dos semanas los miércoles a las 19:30. También se estableció un calendario con objetivos propuestos y se decidió seleccionar el *hardware* necesario para el proyecto.

1 https://trello.com/es

2 https://www.microsoft.com/es-es/microsoft-teams/log-in

3 https://openai.com/chatgpt

08/02/2023: Se discutió la selección de *hardware* y la elección de un modelo de reconocimiento facial. Se sugirió utilizar caras generadas mediante inteligencia artificial como conjuntos de datos. Se acordó también visitar el Instituto Tecnológico de Aragón (ITA) para examinar el *hardware*.

03/03/2023: Se revisaron los avances del prototipo y se identificaron áreas de mejora. La reunión se centró en el desarrollo técnico del proyecto.

29/03/2023: Se revisó el índice de la memoria y se acordó enfocarse en la implementación de otros modelos de *Deep Learning* para hacer una comparación de ellos.

09/05/2023: Se presentaron los avances en la implementación del modelo FaceNet. Se enfatizó la necesidad de escribir la memoria para cumplir con los plazos.

24/05/2023: Se discutió el progreso en la escritura de la memoria, la organización del contenido, y la correcta elaboración de la bibliografía. Se sugirió añadir también la dificultad que hay para la obtención del *hardware*.

31/05/2023: Se propusieron cambios al análisis de mercado y se introdujo el concepto de niveles de madurez tecnológica (TRL). Se hizo énfasis en mejorar la sección de segmentación de mercado y competidores, y se sugirió incluir un apartado sobre los costes.

05/06/2023: Se propusieron ideas para continuar con el desarrollo de la memoria, estableciendo como objetivo terminar todas las secciones exceptuando la implementación para la siguiente semana. Se resaltó la importancia de incluir más referencias bibliográficas, y que estas fueran de calidad.

14/06/2023: Se revisaron algunos avances sobre la memoria y se propuso seguir avanzando con la escritura de esta para así poder acabar de redactar el apartado de la implementación del sistema.

19/06/2023: En esta reunión se revisaron los avances de todo el documento y se propusieron varios cambios como añadir alguna captura de pantallas o gráfico para que la lectura del documento sea más visible y fácil de seguir.

21/06/2023: Se discutió sobre la estructura general del proyecto y se recomendó varios cambios para mejorar la coherencia y flujo del documento. Además, se abordaron ciertos aspectos relacionados con el formato del documento, enfatizando la importancia de la presentación y la claridad. Por último, acordamos la inclusión de ChatGPT en la sección de herramientas.

22/06/2023: Se discutieron sobre diferentes formas de hacer la conclusión y de analizar los resultados del reconocimiento facial.

23/06/2023: Se mencionó una forma de mejorar el análisis y resultados, además, se indicaron varios cambios relacionados con la estructura del apartado de diseño e implementación del sistema.

Problemas encontrados:

Durante el desarrollo del proyecto, se encontraron varios desafíos, como la carga de trabajo académica que condicionaba y dificultaba la implementación completa de la metodología Scrum.

También mencionar, que la obtención del *hardware* fue todo un reto por falta de stock, por lo que se estuvieron barajando distintas opciones al respecto en caso de no poder obtenerlo a tiempo.

1.4.1. Fases de la Metodología

Planificación inicial y preparación

En esta etapa inicial, se han definido los objetivos y el alcance del proyecto. Además, se han especificado requisitos técnicos que fueran compatibles con NVIDIA Jetson Nano [4] y la selección del lenguaje de programación Python [5]. TensorFlow [6] también fue escogido como entorno de desarrollo debido a su amplio soporte y capacidades para el reconocimiento facial. Todo esto se ha definido de acuerdo con la representación de la siguiente figura:

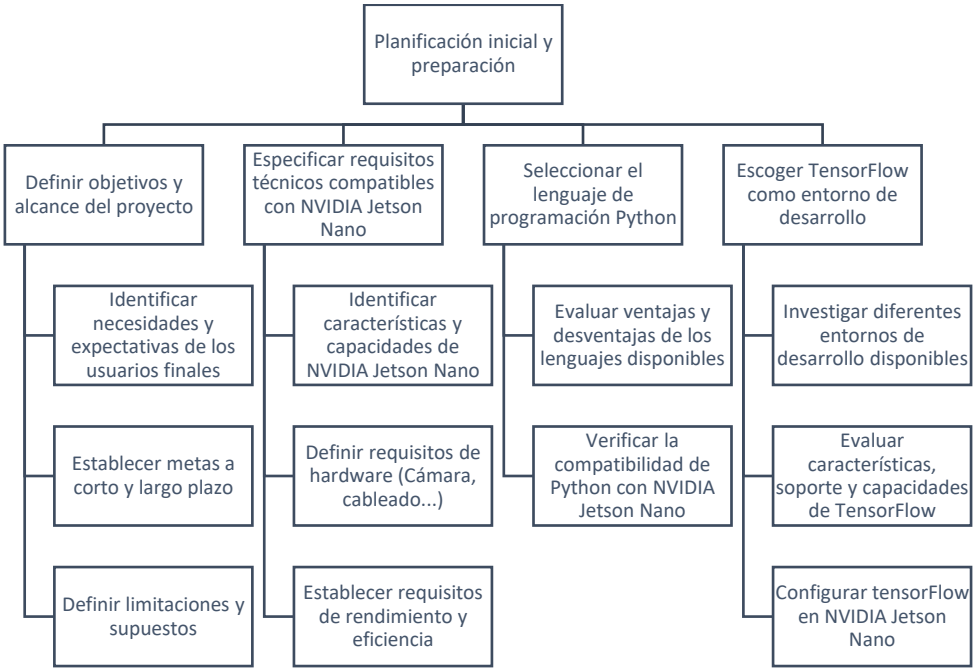


Figura 2: Planificación inicial y preparación

FUENTE: ELABORACIÓN PROPIA

Sprints de desarrollo

El trabajo se dividió en varios sprints, que eran intervalos cortos y fijos de entre 2 semanas a 1 mes. Cada sprint funciona en un conjunto específico de características seleccionadas del backlog del producto. Al final de cada sprint, el proyecto se vería elevado, es decir, cada vez el proyecto iría teniendo mayores capacidades.

- Sprint 1: Durante este sprint inicial, se realizó la investigación preliminar sobre sistemas de control de acceso y tecnologías de reconocimiento facial, identificando requisitos clave.
- Sprint 2: Se continuó con la investigación, obteniendo una comprensión más clara de los objetivos y el alcance del proyecto, y seleccionando tecnologías clave.

4 https://www.nvidia.com/es-es/autonomous-machines/embedded-systems/jetson-nano/education-projects/

5 https://www.python.org/

6 https://www.tensorflow.org/

- Sprint 3: Se centró en la conceptualización y diseño inicial, esbozando la arquitectura y seleccionando herramientas como Python y TensorFlow.
- Sprint 4: Se inició el desarrollo de *software* e implementación de la base de datos, además de la selección y adquisición de componentes de *hardware*.
- Sprint 5: Se continuó con el desarrollo de *software*, implementando reconocimiento facial y desarrollando la interfaz de usuario.
- Sprint 6: Enfocado en la integración de *hardware* y *software*, con pruebas iniciales para identificar problemas y área de mejora.
- Sprint 7: Realización de ajustes basados en pruebas iniciales y preparación de la documentación en la memoria.
- Sprint 8: Segunda ronda de pruebas y validación de mejoras. Continuación de la documentación.
- Sprint 9: Desarrollo y pruebas de características adicionales, y revisión de la documentación del proyecto.
- Sprint 10: Pruebas finales y optimización del sistema. Preparación para la fase final de entrega.
- Sprint 11: Preparación de la documentación final del proyecto.
- Sprint 12: Revisión final y ajustes basados en sus comentarios.
- Sprint 13: Ajustes finales, revisión exhaustiva y preparación para la entrega final.

Una vez definidos los sprints del proyecto, es el momento de representarlos en un diagrama de Gantt para ver los intervalos de tiempo de forma más visual.

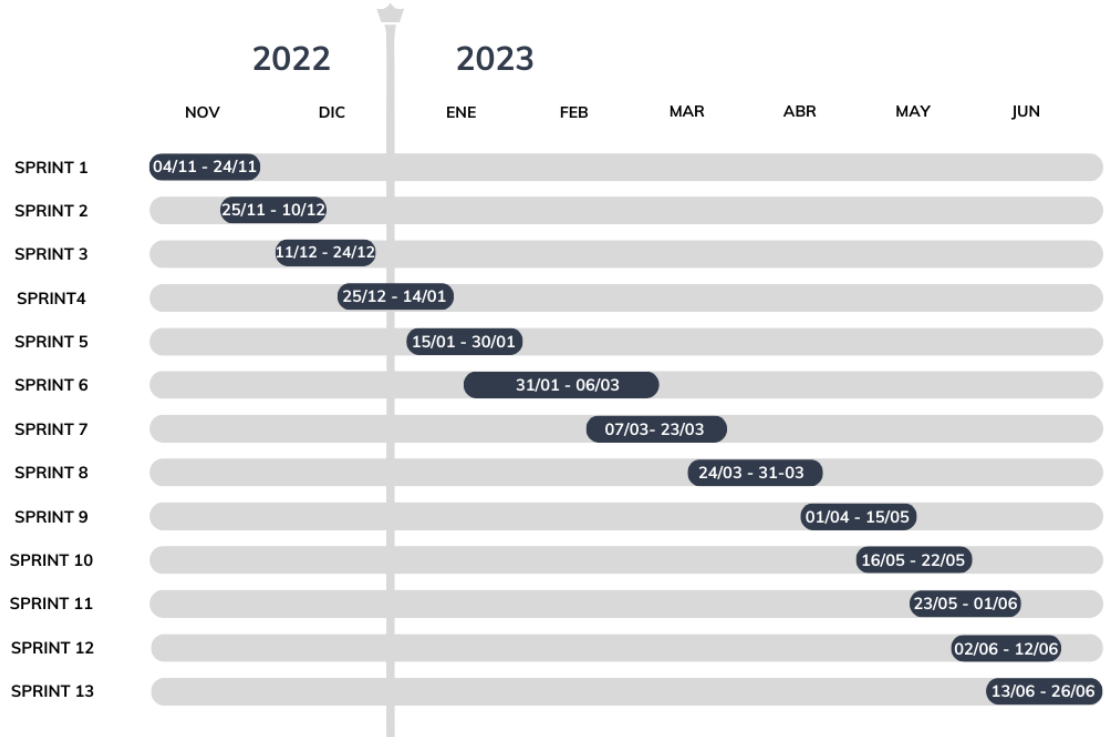


Figura 3: Diagrama de Gantt de los sprints

FUENTE: ELABORACIÓN PROPIA

Revisiones y retroalimentación

Al finalizar cada sprint, se realiza una evaluación del trabajo realizado. Esto permite recibir comentarios y ajustar el progreso del proyecto según sea necesario. Si bien las sesiones de

revisión suelen ser menos formales, estas fueron cruciales para garantizar que el proyecto estuviera en el camino correcto.

Pruebas y validación

En esta etapa se realizaron pruebas para verificar la efectividad y precisión de los algoritmos de reconocimiento facial implementados. Esto incluye probar la integración con el *hardware* NVIDIA Jetson Nano y verificar la compatibilidad de las librerías utilizadas.

Documentación y entrega

Finalmente, se registraron los resultados, problemas y las soluciones implementadas. La documentación es crucial para garantizar que el conocimiento obtenido del proyecto esté disponible para trabajos y posibles desarrollos futuros. La entrega final incluyó tanto el código fuente como la documentación detallada del proyecto.

1.4.2. Tecnologías Aplicadas

Se eligió Python como lenguaje de programación porque se usa ampliamente en la comunidad de la inteligencia artificial y por su amplia gama de bibliotecas. La principal preocupación fue que todas las librerías fueran compatibles en el *hardware* NVIDIA Jetson Nano.

Sin embargo, es importante mencionar que Python puede no ser el lenguaje de programación más eficiente para el desarrollo de sistemas embebidos en un entorno de producción. Dado que Python es un lenguaje interpretado, puede ser más lento en comparación con lenguajes compilados como C o C++.

TensorFlow fue elegido entorno de desarrollo. Es una de los entornos más populares y potentes para desarrollar, entrenar y desplegar modelos de aprendizaje, y cuenta con un amplio soporte y gran comunidad.

2. Marco teórico

2.1. Sistemas biométricos de identificación

Los sistemas de identificación biométrica se utilizan cada vez más en muchas aplicaciones debido a su capacidad para proporcionar una autenticación segura y fiable. La biometría se refiere al uso de las características físicas o de comportamiento únicas de un individuo para verificar su identidad. Las características físicas incluyen huellas dactilares, iris, forma de la mano y rasgos faciales, mientras que los rasgos de comportamiento pueden incluir firma, voz o patrones de pulsación de teclas (Quisilema, 2017).

Un sistema biométrico típico consta de cinco componentes principales: El sensor, el módulo de extracción de características, el módulo de comparación, el módulo de decisión y la base de datos (Arslan, 2017), tal y como se puede observar en la siguiente figura.

Figura 4: Componentes de un sistema biométrico

FUENTE: ELABORACIÓN PROPIA

La principal ventaja de un sistema biométrico es su capacidad para proporcionar una autenticación segura e intransferible (Yang et al., 2018). A diferencia de las contraseñas o los PIN, que se pueden compartir, perder o robar fácilmente, las características biométricas son únicas para cada individuo y no se pueden copiar fácilmente. Además, debido a su naturaleza inherente, los sistemas biométricos ofrecen un alto grado de comodidad ya que los usuarios no necesitan recordar contraseñas ni llevar consigo dispositivos físicos.

La autenticación biométrica se basa generalmente en dos categorías: Basada en rasgos físicos y basada en rasgos de comportamiento (Machine and *Deep Learning* in Biometric Authentication: A Review, 2023).

2.1.1. Autenticación Basada en Rasgos Físicos

Esta categoría de autenticación biométrica implica el uso de características físicas del cuerpo humano. Los rasgos físicos son generalmente inmutables y únicos para cada individuo. Algunos ejemplos de características físicas pueden ser:

- **Huellas dactilares:** El patrón de crestas y valles en la superficie de la punta de los dedos es único para cada persona, y estas son utilizadas para identificar a las personas.
- **Reconocimiento facial:** Analiza las características faciales, como la distancia entre los ojos y la forma de la nariz.
- **Reconocimiento de retina:** Se basa en patrones únicos en el iris o la retina del ojo. El iris es la parte coloreada del ojo, mientras que la retina es una capa posterior del ojo.
- **Geométrica de la mano:** Analiza y mide la forma y el tamaño de la mano de la persona.

2.1.2. Autenticación Basada en Rasgos de Comportamiento

Esta categoría se enfoca en las características del comportamiento de un individuo, que son a menudo adquiridas y pueden cambiar con el tiempo. Algunos ejemplos de características de comportamiento pueden ser:

- **Dinámica de la firma:** No solo analiza la imagen de la firma, sino que también la forma en que una persona la hace, como la velocidad y la presión aplicada durante el proceso.
- **Reconocimiento de voz:** Se basa en las características acústicas de la voz, como el tono y el acento.
- **Patrón de tecleo:** Analiza la forma en que alguien teclea en un teclado, como la velocidad y el ritmo de tecleo.

Tabla 1: Comparación de tipos de autenticaciones biométricas

Criterio	**Autenticación basada en rasgos físicos**	**Autenticación basada en rasgos de comportamiento**
Ejemplos	Huellas dactilares, reconocimiento facial, reconocimiento del iris.	Dinámica de la firma, reconocimiento de voz, patrón de tecleo.
Inmutabilidad	Inmutable a lo largo de la vida.	Puede cambiar con tiempo debido a factores como la edad.
Único	Casi siempre único para cada individuo.	Puede ser menos distintivo en comparación con los rasgos físicos.
Sensibilidad a las condiciones ambientales	Puede ser sensible a condiciones como suciedad, lesiones…	Generalmente menos sensible a condiciones ambientales.
Riesgo de suplantación	Menor, debido a la dificultad de replicar características físicas únicas.	Mayor, puede ser posible imitar el comportamiento humano.
Aplicaciones típicas	Control de acceso, seguridad fronteriza…	Autenticación en sistemas, verificación de firmas…

Es importante destacar que cada tipo de autenticación biométrica tiene sus propias ventajas y desventajas como se puede observar en la tabla anterior, y la elección de una u otra dependerá de la aplicación y los requisitos específicos (Albalawi, 2022). Como parte de este proyecto, estamos particularmente interesados en el reconocimiento facial como método de identificación biométrica. El reconocimiento facial tiene varias ventajas sobre otros métodos biométricos. En

primer lugar, no es intrusivo, ya que se puede realizar de forma remota y no requiere contacto físico. En segundo lugar, es universal porque todo el mundo tiene un rostro que puede utilizarse para la identificación. Sin embargo, también presenta algunos desafíos, como las diferencias en la iluminación, las diferencias en la postura y las expresiones faciales, y el envejecimiento. En las siguientes secciones, discutiremos cómo el uso de técnicas de aprendizaje profundo puede ayudar a superar estos desafíos y mejorar la efectividad del reconocimiento facial.

2.2. Reconocimiento facial y Deep Learning

El reconocimiento facial es una técnica biométrica de identificación personal que interpreta y utiliza patrones físicos y rasgos faciales para autenticar la identidad de un individuo. Desde sus inicios, ha demostrado ser una de las formas de identificación biométrica más naturales, comunes y menos intrusivas, ya que no requiere contacto físico y se puede realizar de forma remota.

A pesar de los beneficios, el reconocimiento facial conlleva una serie de desafíos relacionados con las variaciones de iluminación en el entorno, escala y las variaciones de las poses de la cabeza en las imágenes utilizadas (Wu, 2023). Afortunadamente, en los últimos años, el desarrollo de técnicas de *Deep Learning* ha proporcionado nuevas y poderosas herramientas para abordar estos desafíos (Nzegha et al., 2020).

El *Deep Learning* es una subcategoría de Machine Learning basado en algoritmos inspirados en la estructura y función del cerebro conocidos como redes neuronales artificiales (Silvestrini & Lavagna, 2022). En lugar de organizar los datos para pasar por ecuaciones predefinidas, *Deep Learning* pone parámetros básicos en los datos y entrena el ordenador para que aprenda por sí mismo a través de capas de procesamiento de nodos de redes neuronales, de manera muy similar al aprendizaje del cerebro humano, pero dentro de un ordenador.

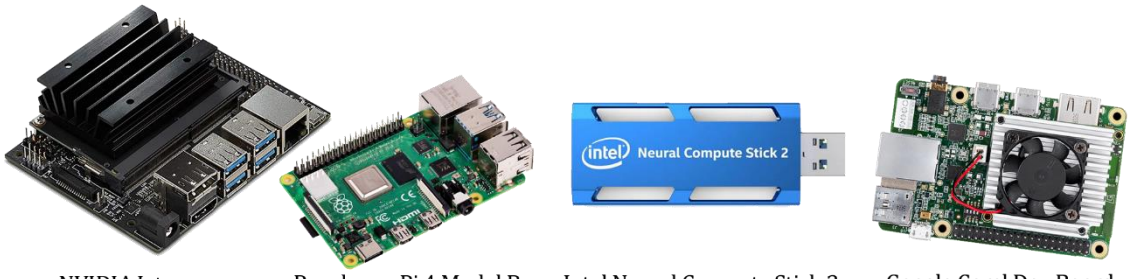
Existen múltiples algoritmos de *Deep Learning* específicos para llevar a cabo de forma excepcional la tarea de reconocimiento facial, entre los que destacan las redes siamesas y el modelo FaceNet. Ambos representan enfoques innovadores para el reconocimiento facial y han demostrado su eficacia en distintas aplicaciones y entornos. Sin embargo, en el [Capítulo 5] de este trabajo, se realizará un análisis más detallado de los algoritmos disponibles y se seleccionará el que mejor se adapte a las necesidades específicas.

2.3. Hardware embebido

El *hardware* embebido se refiere a sistemas informáticos especializados diseñados para realizar funciones especializadas, a menudo con requisitos informáticos y de energía específicos (Zhang, 2022). En este proyecto, el *hardware* embebido juega un papel importante, ya que proporciona la base sobre la cual se desarrolla y opera el sistema de reconocimiento facial basado en *Deep Learning*. Elegir el *hardware* correcto puede afectar significativamente el rendimiento y funcionamiento del sistema. Por lo tanto, es importante elegir un sistema integrado que pueda manejar las necesidades informáticas intensivas del modelo de *Deep Learning* sin sacrificar la eficiencia energética.

En el contexto de este proyecto, el *hardware* embebido es la plataforma en la que se desarrolla y ejecuta todo el sistema de reconocimiento facial basado en *Deep Learning*. Considerando el rendimiento computacional, la eficiencia energética y el costo, se ha seleccionado NVIDIA Jetson Nano para él. Para entender por qué, es importante realizar una comparación detallada con otros

sistemas de *hardware* embebidos populares que podrían adecuarse a: Raspberry Pi 4 Model B [7], Intel Neural Compute Stick 2 (NCS2) [8] y [9], los cuales pueden verse en la próxima figura.



NVIDIA Jetson nano | Raspberry Pi 4 Model B | Intel Neural Compute Stick 2 | Google Coral Dev Board

Figura 5: Sistemas hardware embebidos

FUENTE: Externa [10]

2.3.1. Rendimiento computacional

El rendimiento computacional es crucial para el procesamiento de cualquier programa informático, más aún si estamos ante un proyecto de inteligencia artificial, donde muchos parámetros tienen que ser calculados de forma eficiente para asegurar el correcto funcionamiento del programa. Es por esto, que se necesita un sistema embebido capaz de manejar cálculos intensivos en un tiempo razonable.

- NVIDIA Jetson Nano: Ofrece un rendimiento excepcionalmente alto para aplicaciones de IA, con hasta 472 GFLOPs para operaciones de punto flotante de precisión simple (FP32).
- Raspberry Pi 4 Model B: Este sistema cuenta con una CPU de 4 núcleos y hasta 8 GB de RAM, por lo que puede manejar tareas generales de computación de forma efectiva. Sin embargo, con 21.8 GFLOPs de rendimiento, su capacidad de procesamiento sería algo limitada para su uso con la IA.
- NCS2: Tiene un rendimiento de 1050 GFLOPs en la precisión Int8, por lo que su rendimiento en aplicaciones de IA es muy bueno.
- Google Coral Dev Board: Esta es un sistema con un chip de aceleración de tensor (TPU) para inferencia de IA de alta velocidad, el cual ofrece 4 GFLOPs en precisión FP16.

7 https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

8 https://www.intel.es/content/www/es/es/products/sku/140109/intel-neural-compute-stick-2/specifications.html

9 https://coral.ai/products/dev-board/

10 Fuentes de Figura 5:

- (NVIDIA Jetson Nano Developer Kit (945-13450-0000-100) : Amazon.es: Informática, 2023.)
- (Raspberry Pi 4 Modelo B (4GB) : Amazon.es: Informática, 2023.)
- (Intel NCSM2485.DK, 2023.)
- (Coral Dev Board, 2023)

Tabla 2: Comparación de Rendimiento computacional

Dispositivo	**GFLOPs FP32**
NVIDIA Jetson Nano	472
Raspberry Pi 4 Model B	21.8
Intel NCS2	1050 (Int8)
Google Coral Dev Board	4 (FP16)

2.3.2. Versatilidad

La versatilidad se refiere a la capacidad del sistema para manejar diferentes tipos de cargas de trabajo y adaptarse a diversas necesidades de desarrollo.

- NVIDIA Jetson Nano: Puede ejecutar una variedad de modelos de *Deep Learning*, incluyendo Tensorflow, PyTorch, Caffe, entre otros, ofreciendo una gran versatilidad a la hora de trabajar con él.
- Raspberry Pi 4 Model B: Soporta una amplia gama de sistemas operativos y lenguajes de programación, aunque su soporte para algoritmos de *Deep Learning* es limitado.
- NCS2: En este caso, el NCS2 está optimizado para los modelos de inferencia de OpenVINO Toolkit de Intel, lo que hace que su versatilidad se vea algo limitada.
- Google Coral Dev Board: En este caso, solo puede ejecutar entornos desarrollados en TensorFlow Lite.

Tabla 3: Comparación de Versatilidad

Dispositivo	**Frameworks soportados**
NVIDIA Jetson Nano	TensorFlow, PyTorch, Caffe, entre otros
Raspberry Pi 4 Model B	TensorFlow Lite
Intel NCS2	OpenVINO
Google Coral Dev Board	TensorFlow Lite

2.3.3. Eficiencia energética

Para aplicaciones embebidas, la eficiencia energética es esencial ya que a menudo operan con recursos energéticos limitados y su funcionamiento debe seguir siendo eficaz.

- NVIDIA Jetson Nano: Tiene un consumo de energía de entre 5 y 10 vatios, lo que es excelente considerando su alto rendimiento.
- Raspberry Pi 4 Model B: Tiene un consumo energético de entre 3.5 y 7 vatios.
- NCS2: Tiene un consumo de 1 vatio, lo que hace que sea la más reducida de la lista.
- Google Coral Dev Board: En este caso, tenemos un consumo de entre 2 y 3 vatios.

Tabla 4: Comparación de Eficiencia Energética

Dispositivo	**Consumo de energía**
NVIDIA Jetson Nano	5 - 10W
Raspberry Pi 4 Model B	3.5 - 7W
Intel NCS2	1W
Google Coral Dev Board	2 – 3W

2.3.4. Coste

Es importante analizar la inversión mínima necesaria para poder desarrollar un sistema de control de acceso en cada uno de estos sistemas de *hardware* embebidos, es decir, analizar el coste que supondría la obtención de todo el *hardware* necesario para poder desarrollar en ellos. Tener en

consideración que los valores establecidos son los que corresponden a la fecha de 03/06/2023, por lo que estos pueden variar a lo largo del tiempo.

- NVIDIA Jetson Nano: Como se menciona en el apartado [Coste del desarrollo], el coste total del *hardware* necesario para desarrollar este sistema en el NVIDIA Jetson Nano sería de 245€.
- Raspberry Pi 4 Model B: Aproximadamente 210€ hacen falta para disponer de este equipo. El hardware cuesta 115€ pero hay que sumarle el precio de la cámara, el cable de alimentación y demás cosas. (PcComponentes, 2023)
- NCS2: El *hardware* solamente cuesta 115€, pero en este caso, se necesitaría un ordenador al que ser conectado. Por eso, el precio puede ascender por encima de los 500€.
- Google Coral Dev Board: El precio base de este *hardware* es de 125€. Añadiendo componentes como la cámara, la memoria microSD y el cable de alimentación acaba rondando los 200€.

Tabla 5: Comparación de los costes

Dispositivo	**Coste Base (€)**	**Accesorios adicionales**	**Coste Total (€)**
NVIDIA Jetson nano	150	Cable de alimentación, tarjeta microSD, cámara	245
Raspberry Pi 4 Model B	115	Cámara, cable de alimentación, tarjeta microSD, etc.	210
Intel NCS2	115	Necesita ser usado con una placa de desarrollo o PC	500+
Google Coral Dev Board	125	Cámara, cable de alimentación, tarjeta microSD	200

2.3.5. Resultados y conclusión

Cada uno de los sistemas embebidos mencionados anteriormente tiene sus ventajas dependiendo del proyecto en el que uno quiera trabajar, ya que, cada uno se centra en ser óptimo en campo diferente al resto.

Después de revisar y comparar cuidadosamente los distintos dispositivos, se ha decidido que el NVIDIA Jetson Nano es la mejor opción para llevar a cabo el prototipo de control de acceso con el modelo de reconocimiento facial integrado.

Hay varias razones para llevar a cabo esta decisión. En primer lugar, el rendimiento computacional. Si bien el Intel NCS2 puede proporcionar más GFLOPs en el formato Intel8, el Jetson Nano, gracias a la arquitectura CUDA mejorada, ofrece un rendimiento superior en la práctica para algoritmos de *Deep Learning*, lo que le permite aprovechar su rendimiento y potencia de cálculo para las aplicaciones de IA.

Además, NVIDIA Jetson Nano proporciona soporte para varios *frameworks* de IA, incluidos TensorFlow, PyTorch [11] y Caffe [12]. Esta versatilidad es una gran ventaja, puesto que no restringe su uso a un único *framework* y permite seleccionar el que mejor se ajuste a las necesidades específicas del proyecto en cuestión, aunque todos los sistemas embebidos suelen ser bastante complicados de configurar por la falta de información.

En términos de eficiencia energética, el NVIDIA Jetson Nano opera en el rango de consumo de energía de 5 a 10 vatios. Es un consumo ligeramente superior al de otros dispositivos, pero teniendo en cuenta su alto rendimiento, representa una relación eficiencia-energía muy competitiva.

Finalmente, la comunidad y el soporte alrededor de NVIDIA Jetson Nano son grandes y activos, lo que significa que hay muchos recursos y ayuda disponibles para resolver la mayoría de los problemas que uno pueda llegar a tener durante su desarrollo en él. Aunque es importante mencionar que todo el *software* libre, al ser creado por la comunidad, se dependerá de la información que se pueda encontrar y lo estable que sean las diferentes versiones de esta.

En resumen, NVIDIA Jetson Nano ofrece la combinación ideal de rendimiento computacional, eficiencia energética y soporte de la comunidad, lo que la convierte en la elección perfecta para un sistema de control de acceso basado en los requerimientos de este proyecto.

11 https://pytorch.org/

12 https://caffe.berkeleyvision.org/

3. Estudio Económico

3.1. Descripción del mercado actual

Los sistemas de control de accesos corporativas han cambiado mucho en la última década, esto en gran medida es por las nuevas necesidades de medidas más sólidas y por los avances en tecnología. En el corazón de este cambio se encuentra la emergencia de sistemas de control de acceso basados en inteligencia artificial, los cuales ofrecen un nivel de seguridad y eficiencia que los sistemas tradicionales basados en tarjetas de identificación o contraseñas no pueden igualar.

En la actualidad, el mercado de control de acceso basado en reconocimiento facial está en pleno auge. Esto es debido al gran creciente interés general en la seguridad y la identificación personal en diferentes industrias (ŞahiN, 2023). Además, la digitalización generalizada y esta necesidad están impulsando la adopción de estas tecnologías en una gran cantidad de sectores. Estas empresas buscan soluciones más sofisticadas y seguras para proteger sus recursos y garantizar así que solo las personas autorizadas tengan acceso a áreas restringidas, además de facilitar mucho su correcta interacción a los usuarios que vayan a utilizar el sistema.

Este mercado es cada vez más competitivo y está compuesto por una variedad de actores, que van desde startups tecnológicas hasta empresas establecidas con décadas de experiencia en seguridad y tecnología. Esto es algo inevitable, a medida que la tecnología mejora y se vuelve más accesible, el número de competidores en el espacio del reconocimiento facial sigue creciendo. También, este creciente interés en estas soluciones tecnológicas atrae muchas inversiones significativas en el sector, lo que podría acelerar aún más la innovación y el desarrollo de nuevas y mejoradas tecnologías (Chen & Liu, 2019).

Las aplicaciones de estas tecnologías de reconocimiento facial son diversas y van desde la seguridad empresarial, el control de acceso en eventos, hasta aplicaciones gubernamentales como la identificación de ciudadanos. Estos métodos de identificación son particularmente seguros y efectivos particularmente en entornos empresariales, donde mejoran las capacidades de monitorear y rastrear la entrada y salida de los empleados y también se pueden integrar con otros sistemas de seguridad para ofrecer un sistema completo de seguridad (Suman, 2023).

En resumen, la tendencia de utilizar tecnologías de reconocimiento facial para el control de acceso está ganando mucho impulso en el mercado actual. La creciente necesidad de seguridad y los beneficios inherentes que ofrecen estas soluciones en términos de rendimiento y eficiencia están impulsando la adopción de estas dentro de distintas entidades. Sin embargo, junto con esta creciente popularidad, también existen desafíos, como la privacidad y las consideraciones éticas, que deben manejarse con cuidado a medida que estas soluciones continúan evolucionando y adoptando una aplicación más amplia y sostenible.

3.2. Perspectivas de crecimiento del mercado

El mercado de sistemas de control de acceso basados en reconocimiento facial tiene perspectivas de crecimiento prometedoras en los próximos años. Este crecimiento se debe a una serie de factores clave, incluida la creciente demanda de soluciones de seguridad avanzadas, el crecimiento de la industria de la tecnología biométrica y los avances tecnológicos, como el *Deep Learning*.

En primer lugar, la creciente preocupación por la seguridad tanto en la industria como en la sociedad está impulsando la necesidad de mejores y más eficaces soluciones de control de acceso. Por lo tanto, se espera que el uso de la tecnología de reconocimiento facial siga creciendo en el futuro.

En segundo lugar, según diversos informes, la industria de la tecnología biométrica en su conjunto está experimentando un crecimiento significativo. El tamaño del mercado alcanzó los más de 43 mil millones de dólares en 2021 y se espera que registre un crecimiento anual compuesto (CAGR) del 14,8% durante el periodo de previsión (Emergen Research, 2022). También se espera que el mercado biométrico mundial crezca de 10.6 mil millones de dólares y se estima que alcance los 41.4 mil millones de dólares en 2025, una CAGR del 17,06% durante 2017-2025 (Clark, 2018).

En última instancia, se espera que los avances en tecnologías de *Deep Learning* que permitan la creación de sistemas de reconocimiento facial más precisos y confiables contribuyan a una mayor implementación de estas soluciones.

Además, se espera que el desarrollo y la implementación de tecnologías de *hardware* embebido continúen creciendo, proporcionando más oportunidades para implementar sistemas de reconocimiento facial en una variedad de contextos. En particular, la disponibilidad de dispositivos como el NVIDIA Jetson Nano, que brindan una potencia de procesamiento significativa, permite implementaciones más flexibles y rentables de estas tecnologías.

3.3. Demanda y oferta en el mercado

Para comprender el mercado de sistemas de control de acceso basados en reconocimiento facial, es crucial analizar tanto la demanda como la oferta de este tipo de soluciones. El objetivo de esto es obtener una visión más clara del estado actual del mercado para poder anticipar futuros posibles movimientos y tendencias.

3.3.1. Demanda

En los últimos años, la demanda de sistemas de control de acceso basados en reconocimiento facial ha crecido significativamente. Varios factores han contribuido a este crecimiento, incluida la mejora de la tecnología, el aumento de las preocupaciones de seguridad y una mayor digitalización en todas las industrias.

Este crecimiento de la demanda es especialmente relevante para el proyecto propuesto en este documento, ya que indica un mercado en crecimiento y la disposición de los usuarios y las empresas a adoptar soluciones similares. Es por este motivo que este proyecto se centra en desarrollar un sistema de control de acceso eficiente, confiable y seguro, reduciendo la dependencia de elementos físicos (Dubey, 2017).

Es importante también considerar el Gartnet Hype Cycle (2022 Gartner® Hype CycleTM for Digital Identity, 2022), que es una representación gráfica de la adopción y madurez de tecnologías específicas.

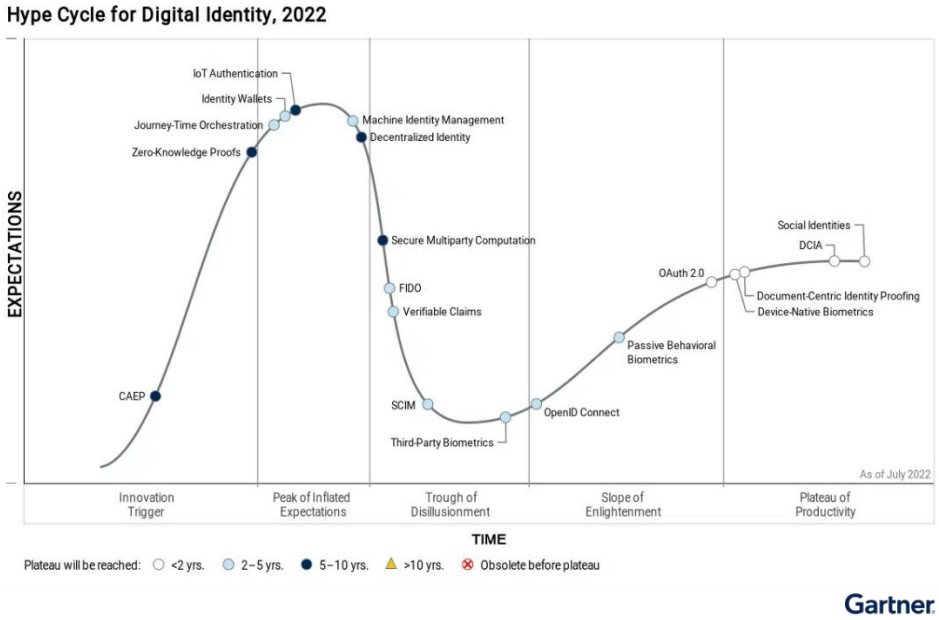


Figura 6: Gartnet Hype Cycle

ELABORACIÓN: EXTERNA [13]

El Gartner Hype Cycle se compone de cinco fases:

1. **Innovación y Disparo de Expectativas:** Cuando una nueva tecnología emerge, genera altas expectativas y entusiasmo, pero aún no está probada o ampliamente adoptada.
2. **Pico de Expectativas Infladas:** El interés y las expectativas alcanzan su punto máximo, pero a menudo se basan en casos de éxito aislados y no en adopción generalizada. Aquí es donde algunos proyectos fallan por no llegar a cumplir con las expectativas.
3. **Abismo de Desilusión:** La tecnología no cumple con las expectativas iniciales, y el interés comienza a disminuir.
4. **Pendiente de la Iluminación:** Las empresas y desarrolladores comienzan a entender mejor cómo aplicar la tecnología de manera efectiva. Las mejoras y avances comienzan a mostrar su verdadero potencial.
5. **Meseta de Productividad:** La tecnología se vuelve más ampliamente adoptada y su valor se extiende y acepta de manera más clara.

El reconocimiento facial actualmente ha pasado por el pico de expectativas y está en su camino hacia la meseta de productividad. Esto significa que la tecnología ha madurado y está siendo adoptada más ampliamente, lo que coincide con la creciente demanda de sistemas de control de acceso basados en reconocimiento facial.

Además, las preocupaciones de seguridad son mayores que nunca debido a la amenaza cada vez mayor de ataques cibernéticos y ataques físicos. Las empresas quieren siempre proteger sus activos y datos, lo que aumenta la necesidad de soluciones de seguridad más sólidas y fiables. En este contexto, los sistemas de control de acceso basados en reconocimiento facial se presentan como una solución efectiva (Mangata, 2022). Para este proyecto, esto enfatiza la

13 Fuente de figura 6: (Infocert.digital, 2023)

importancia de garantizar que el sistema desarrollado sea seguro y proteja los datos de los usuarios.

3.3.2. Oferta

En términos de oferta, hay varios proveedores de sistemas de control de acceso basados en reconocimiento facial en el mercado proporcionado por los [competidores directos], los cuales están siendo analizados próximamente en este documento. Estos proveedores difieren en tamaño, experiencia, capacidades técnicas y enfoque de mercado. Algunos proveedores se enfocan en proporcionar soluciones para industrias específicas, como la salud o la educación, mientras que otros ofrecen soluciones más generales que se pueden usar en todas las industrias (Kashmar, 2022).

Este panorama variado de proveedores también afecta a las decisiones de este proyecto. Es esencial que el sistema desarrollado sea competitivo en términos de características y funcionalidades, y que esté adecuadamente diferenciado para atender nichos específicos del mercado o necesidades particulares que no estén siendo satisfechas actualmente.

En conclusión, existe un equilibrio dinámico entre la oferta y la demanda en el mercado de sistemas de control de acceso. Esta dinámica afecta directamente las decisiones y el enfoque adoptado en este proyecto. Es importante no solo desarrollar una solución tecnológicamente avanzada (Duarte, 2016) sino también ser conscientes de las necesidades del mercado y asegurar así que el sistema ofrezca valor agregado para diferenciarse en este mercado tan competitivo.

3.4. Análisis competitivo

Es esencial examinar la competencia para comprender el entorno en el que es probable que se ejecute cualquier proyecto, incluso en aquellos como este, donde el objetivo no es crear un producto final para la venta. Sin embargo, también es fundamental saber quiénes están creando tecnologías similares y aprender de ellos.

Desde pequeñas empresas emergentes hasta grandes empresas de tecnología, el reconocimiento facial es un mercado amplio con numerosos competidores. Por este motivo, es fundamental diferenciar entre los competidores directos e indirectos.

3.4.1. Competidores Directos

Aquellos que ofrecen productos o servicios comparables, como sistemas de control de acceso mediante reconocimiento facial, son nuestros competidores directos.

Es más apropiado considerar empresas más pequeñas que están igualmente enfocadas en este tipo de tecnología, en lugar de corporaciones de renombre como Amazon [14] o Microsoft [15]. Oosto [16], TrueFace [17] y SAFR [18] son ejemplos de empresas que se especializan en soluciones de seguridad basadas en IA, incluido el reconocimiento facial. Estas empresas, a pesar de su presencia global, son más comparables con el tamaño y enfoque que tiene este proyecto.

14 https://www.amazon.es/

15 https://www.microsoft.com/es-es

16 https://oosto.com/

17 https://www.trueface.ai/

18 https://safr.com/

3.4.2. Competidores Indirectos

Un competidor indirecto es alguien que vende un producto o servicio que puede reemplazar (o potencialmente eliminar) la necesidad de un sistema de control de acceso basado en el reconocimiento facial (Downing et al., 2019). Estos incluyen sistemas basados en otros métodos de autentificación, como tarjetas de acceso, códigos PIN, reconocimiento de huellas dactilares, reconocimiento de iris, entre otros. Estas soluciones son proporcionadas por varias empresas, incluidas HID Global [19], Honeywell [20] y Johnson Controls [21].

Las empresas, ya sean directas o indirectas, sirven como punto de referencia para determinar las actividades actuales del mercado, los avances tecnológicos, la aceptación de los usuarios y los niveles de satisfacción de los clientes. Esto permite poder aprender de sus logros y errores, pudiendo aplicar estos aprendizajes en el proyecto.

Este proyecto tiene como objetivo desarrollar una solución de reconocimiento facial que se adapte a los requisitos específicos expuestos anteriormente, en lugar de competir en el mercado actual. La ventaja competitiva de este proyecto no está en el entorno o mercado competitivo, sino en su flexibilidad y capacidad para abordar las necesidades y desafíos específicos expuestos en este documento.

Analizar la competencia es crucial para identificar tendencias en el uso de esta tecnología y tener una idea de las características y funciones que más se valoran. Esto proporciona una comprensión más completa de los factores que deben tenerse en cuenta al diseñar el sistema de control de acceso.

3.5. Oportunidades y desafíos en el mercado

La creciente importancia de la inteligencia artificial y el *Deep Learning* en el mundo actual ha creado una variedad de oportunidades y desafíos en el desarrollo de sistemas de control de acceso que utilizan el reconocimiento facial. Estos elementos son cruciales para el progreso de cualquier proyecto, por eso es importante comprender su significado.

3.5.1. Oportunidades

Comenzaremos hablando de las oportunidades que se presentan en este campo. Los avances en tecnología de IA y *Deep Learning* constituyen una oportunidad invaluable. La precisión y la eficacia del reconocimiento facial han mejorado mucho gracias a los avances tecnológicos de los últimos años. Esto hace posible la creación de un sistema de reconocimiento facial que es altamente preciso y funciona perfectamente (Onyema et al., 2021). Además, ha habido un aumento en la capacidad del sistema para manejar situaciones desafiantes, como las condiciones de iluminación cambiantes, lo que ha mejorado la posible robustez de estos sistemas.

Además, la creciente demanda de seguridad mejorada en diversas industrias está impulsando la adopción de sistemas de control de acceso basados en tecnologías biométricas. Esto, se debe a la digitalización de los sectores y la necesidad de mejoras en los sistemas de seguridad a escala mundial. Es por este motivo, que el sistema propuesto en este proyecto es ideal como solución para las empresas que buscan mejorar la seguridad de sus instalaciones.

19 https://www.hidglobal.com/es

20 https://www.honeywell.com/us/en

21 https://www.johnsoncontrols.com/

Otra oportunidad relevante es la adopción cada vez mayor de sistemas de *hardware* embebidos, como la NVIDIA Jetson Nano. Estos sistemas son altamente eficientes en términos de consumo y energía, lo que permite utilizar la tecnología de reconocimiento facial en áreas con recursos limitados (Wong, 2021).

3.5.2. Desafíos

Sin embargo, existen importantes desafíos que considerar. Entre ellos se encuentran las preocupaciones por la privacidad y las estrictas normas que rigen el uso de datos biométricos (Blanco, 2020). El uso de las tecnologías de reconocimiento facial puede implicar la recopilación de datos sensibles y confidenciales, y, por lo tanto, se encuentra sujeto a regulaciones rigurosas diseñadas para proteger la privacidad de las personas. Es igualmente desafiante navegar por estas regulaciones y garantizar el cumplimiento al desarrollar y aplicar esta tecnología.

Otro desafío es el requisito de grandes conjuntos de datos para entrenar el modelo de *Deep Learning*. Entrenar y validad modelos de IA y *Deep Learning* requiere una cantidad significativa de datos, lo que puede ser un desafío en términos de recopilación y almacenamiento de datos. Por este motivo existen técnicas para aumentar los datos existentes, mejorando así el desempeño de los distintos modelos (Sarasa, 2020).

Además, las dificultades que enfrenta la implementación de los sistemas de reconocimiento facial son muy variadas. El diseño de un sistema eficaz requiere la combinación adecuada de *hardware* y *software*, así como la consideración de factores como la iluminación, la posición facial y las diferencias de apariencia.

3.5.3. Enfoque del Producto Considerando Oportunidades y Desafíos

Teniendo en cuenta las oportunidades y desafíos identificados, el enfoque del proyecto propuesto en este documento tendrá como objetivo maximizar los beneficios de la resolución proactiva de los desafíos.

En primer lugar, para aprovechar los avances en inteligencia artificial y *Deep Learning*, el sistema se basará en modelos optimizados de redes neuronales para garantizar una alta precisión y eficacia del reconocimiento facial, incluso en condiciones difíciles. Esto también abordará el problema de la sensibilidad a los cambios de luz y otros factores del entorno.

Para satisfacer las crecientes necesidades de seguridad, el sistema se diseñará para satisfacer las necesidades necesarias para ello. Esto incluye funciones como la capacidad de integrarse con los sistemas de seguridad existentes, así como una interfaz de usuario intuitiva y accesible para garantizar una adopción perfecta por parte del usuario.

Finalmente, hay que mencionar que el enfoque adoptado busca proporcionar un sistema de reconocimiento facial de alto rendimiento y confiable que sea sensible a las preocupaciones de privacidad y que pueda ser aplicado en diversos sectores industriales de manera eficiente y rentable.

3.6. Nivel de Madurez Tecnológica (TRL)

3.6.1. Definición de TRL

Para poder identificar el estado de desarrollo que se ha realizado durante el TFG, he optado por el Nivel de Madurez Tecnológica (Technology Readiness Level, TRL por sus siglas en inglés). TRL se entiende como un sistema de medida que se utiliza para evaluar la madurez de una tecnología

específica. La NASA fue la primera en crear el TRL, que desde entonces se ha utilizado como estándar para evaluar el progreso de la tecnología desde la investigación básica hasta la implementación comercial (A Model Based on the Technology Readiness Level (TRL) Scale to Measure the Maturity Level of Research Projects That Can Become Spinoffs in Higher Education Institutions, 2021).

Los TRLs se dividen en 9 niveles, donde el nivel 1 se refiere a la investigación básica y el nivel 9 se aplica a la tecnología probada en condiciones reales y preparada para la implementación comercial. A continuación, se proporciona una breve descripción de cada nivel (Tzinis, 2021):

- **Investigación básica (TRL 1):**
  En esta etapa inicial, se estudian los fundamentos de la tecnología en consideración. Se trata de un estudio teórico destinado a determinar las propiedades y posibles aplicaciones. Esta investigación es de naturaleza exploratoria y tiene como objetivo avanzar en el conocimiento, sin una aplicación específica en mente.
- **Concepto de tecnología (TRL 2):**
  En este nivel, la atención se centra en la práctica de los principios descritos en TRL 1. Se llevan a cabo estudios analíticos y de simulación para explorar el potencial de la tecnología. Todavía está en la etapa de concepto, pero ha quedado claro cómo se puede desarrollar y utilizar en aplicaciones reales.
- **Prueba experimental de concepto (TRL 3):**
  Aquí el concepto de tecnología se traslada al laboratorio. Se realizan experimentos preliminares para verificar que el concepto funciona como se espera en un entorno controlado. Esto generalmente incluye la creación de prototipos y pruebas a pequeña escala para demostrar la viabilidad del proyecto en cuestión.
- **Validación de la tecnología en laboratorio (TRL 4):**
  La tecnología se desarrolla y se somete a fondo en el laboratorio. Esto incluye la optimización de prototipos y la evaluación de su rendimiento en diversas condiciones controladas. Los resultados de estas pruebas se utilizan para refinar el diseño del prototipo.
- **Validación de la tecnología en entorno relevante (TRL 5):**
  Los prototipos se prueban en entornos que se parecen mucho a las aplicaciones de la vida real. Esto permite determinar cómo funciona la tecnología en situaciones más realistas y posiblemente imprevistas. Este paso es importante para refinar y mejorar la tecnología antes de pasar a las pruebas en condiciones más extremas.
- **Demostración de la tecnología en entorno relevante (TRL 6):**
  En este nivel, la tecnología está representada en un entorno que refleja fielmente su uso final. Esto proporciona una estimación más confiable de cómo funcionará la tecnología en el mundo real y permite identificar y abordar cualquier problema restante antes de pasar a las etapas finales de desarrollo.
- **Demostración de la tecnología en entorno operacional (TRL 7):**
  En el TRL 7, esta tecnología se prueba en el entorno operativo real en el que se espera que opere. Esto significa exponer la tecnología a todas las condiciones cambiantes e imprevistas del entorno real. Esta es una prueba importante que se acerca a un producto casi finalizado.
- **Sistema completo y cualificado (TRL 8):**
  En esta etapa, la tecnología se integra con tolos los demás componentes necesarios para formar un sistema completo. Se realizan una serie de pruebas de calidad para garantizar que el sistema cumple con todos los requisitos funcionales y de seguridad necesarios para su correcta implementación.
- **Sistema probado y aprobado para uso comercial (TRL 9):**

En este último paso, la tecnología ha superado todas las etapas de desarrollo y pruebas, por lo que está lista para su implementación comercial. Esto significa que ha demostrado un rendimiento confiable y constante en condiciones reales, por lo que cumple con los estándares y regulaciones de la industria.

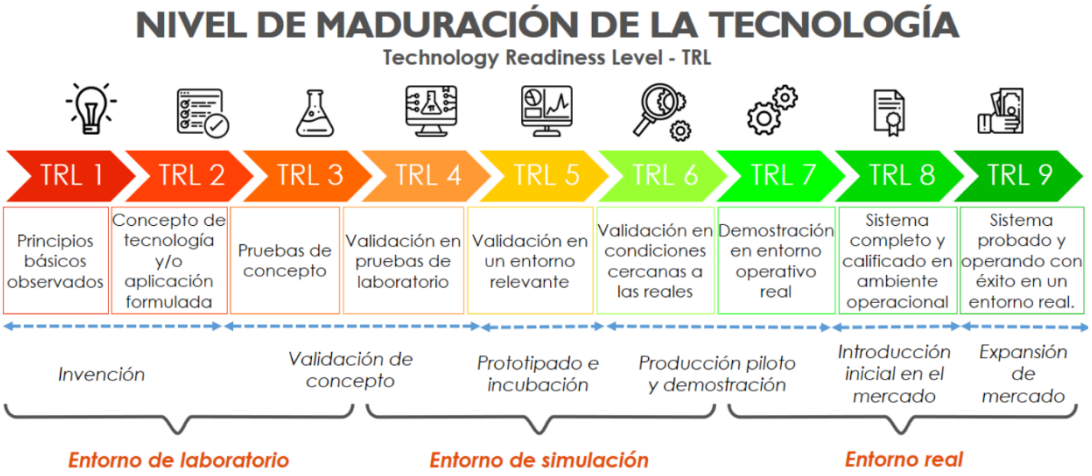


Figura 7: Nivel de Maduración de la Tecnología (TRL)

FUENTE: EXTERNA [22]

3.6.2. Nivel TRL actual y plan para alcanzar TRL 9

Por el momento, se estima que el nivel TRL 3-4 está asociado con el proyecto. En esta etapa, el *hardware* embebido NVIDIA Jetson Nano se ha utilizado para desarrollar y probar la tecnología del sistema de control de acceso mediante el reconocimiento facial, aunque esto ha sido en un entorno controlado de laboratorio. A este nivel, se puede afirmar que el sistema puede identificar rostros de manera efectiva y precisa. Sin embargo, no se ha probado en un entorno relevante, por lo que aún queda trabajo por hacer antes de poder clasificar el proyecto en un nivel superior.

Para poder alcanzar el nivel TRL 9, se deben superar varios hitos. Inicialmente, la tecnología debe mostrarse en un entorno aplicable, lo que podría implicar pruebas en una empresa u organización que requiera un *software* de control de acceso con estas características. El siguiente paso es exhibir la tecnología en una situación operativa tangible y luego vincularla con otros sistemas, lo que da como resultado un sistema que lo abarca todo. Por último, el sistema debería ser probado y aprobado para uso comercial.

Los obstáculos que deben superarse son sustanciales y requieren esfuerzos coordinados para desarrollar, probar, validar y cumplir con las reglamentaciones pertinentes. Sin embargo, el reconocimiento de estos problemas también permite un enfoque directo para mejorar la sofisticación tecnológica del sistema.

3.7. Análisis Económico

Una comprensión clara del coste total asociado y el potencial retorno de la inversión es crucial para que los proyectos de desarrollo tecnológico tomen consideraciones económicas. El análisis

22 Fuente de Figura 7: (Innova - Ciencia Asturias, n.d.)

económico incluirá costes de desarrollo, costes operativos y posibles ingresos en caso de comercializar el sistema.

3.7.1. Coste del Desarrollo

El desarrollo de este sistema de control de acceso basado en reconocimiento facial tiene algunos costes asociados. El gasto más importante es el *hardware* necesario para configurar el sistema, en este caso, el NVIDIA Jetson Nano, que actualmente está disponible a un precio estimado de 150€. Sin embargo, este gasto es solo una fracción del gasto total del desarrollo.

El coste de desarrollo de *software* puede variar enormemente dependiendo de la complejidad del sistema, la cantidad de funciones que se desarrollarán y el tiempo requerido para el desarrollo en cuestión. El *software* de este proyecto incorporará la implementación de algoritmos de reconocimiento facial y el diseño de una interfaz de usuario para administrar el sistema.

Además del coste de los componentes necesarios, es importante tener en cuenta las horas dedicadas a las distintas partes del proyecto. Añadiendo, los distintos miembros que componen el equipo que hace posible la creación del proyecto. Esto podría ayudarnos a obtener un coste aproximado mucho más realista de lo que costaría llevar a cabo este proyecto pagando un sueldo a todas las personas involucradas en él. (Talent.com, n.d.)

Tabla 6: Coste del desarrollo

Elemento	**Coste (€)**	**Horas de Desarrollo**	**Perfil del Miembro del Equipo**	**Sueldo medio por hora (€)**
NVIDIA Jetson Nano	150	-	-	-
Cable de alimentación	20	-	-	-
Webcam Logitech C920 HD	75	-	-	-
Licencias de *software*	0	-	-	-
Desarrollo				
Investigación y diseño	-	50	Ingeniero de sistemas	16.92
	-	30	Diseñador UX/UI	15.38
Implementación del prototipo	-	80	Programador Junior	10.77
		40	Programador Senior	17.69
Pruebas y validación	-	20	Ingeniero de pruebas	16.67
		15	Programador Junior	10.77
Documentación	-	70	Redactor Técnico	12.31
		30	Ingeniero de sistemas	16.92
Total	**245**	**335**	**-**	**-**

El coste total aproximado del desarrollo de este proyecto sería calculado de la siguiente manera:

$$Coste\ total = 245 + (50 \cdot 16.92) + (30 \cdot 15.38) + (80 \cdot 10.77) + (40 \cdot 17.69) + (20 \cdot 16.67) + (15 \cdot 10.77) + (70 \cdot 12.31) + (30 \cdot 16.92) = 4985.85€$$

3.7.2. Costes Operativos

Los gastos de mantenimiento de *hardware* y el *software*, el suministro de energía y la actualización o adaptación del sistema para satisfacer las necesidades cambiantes están incluidos e los costes operativos del sistema. Estos costes incluyen el mantenimiento de equipos, costes de mano de obra, etc. Estos costes son continuos y deben tenerse en cuenta en el análisis económico.

3.7.3. Potenciales Ingresos

Si bien este proyecto no se está desarrollando con la intención de comercialización, es importante considerar los ingresos potenciales que podrían resultar de dicho desarrollo. El precio de venta del sistema está sujeto a varios factores como los costes de producción, la demanda del mercado y los precios de la competencia. Sin embargo, una estimación inicial podría basarse en el coste de desarrollo y operación del sistema, con un margen adicional para beneficios.

3.7.4. Costes de Desarrollo hasta el TRL 9

Considerar los costes de desarrollar un sistema de control de acceso mediante reconocimiento facial hasta alcanzar el nivel TRL 9 es crucial a la hora de estudiar este prototipo. A partir de su estado actual de TRL 4, se prevén distintos gastos adicionales para cada fase.

Desarrollo del prototipo y pruebas iniciales (TRL 4 a 6)

Para pasar de TRL 4 a 6, es necesario que las partes de este sistema se puedan ensamblar en un prototipo funcional y demostrarlo en un entorno relevante. Esta fase implica costes adicionales para adquirir los componentes necesarios, como varias unidades del NVIDIA Jetson Nano para pruebas paralelas, y otros componentes esenciales para la integración del sistema. Además. Es imprescindible invertir en el desarrollo de *software* para integrar y probar el prototipo. Aunque se aprovechará el *software* gratuito y de código abierto para la implementación, todavía hay costes asociados con el tiempo de ingeniería. Se estima que esta fase requerirá alrededor de 250 horas de trabajo, con un coste total estimado de alrededor de 8000€.

Demostración en entorno real y ajustes (TRL 7 y 8)

En las fases TRL 7 y 8, se debe demostrar el sistema en un entorno operativo real o equivalente. Estas etapas pueden incurrir en gastos adicionales que incluyen acceso a instalaciones de prueba adecuadas, así como adaptaciones al sistema para cumplir con los requisitos específicos de esos entornos. Además, en esta fase, es crucial contar con un equipo multidisciplinario que incluya ingenieros de *hardware* y *software*, especialistas en pruebas, y expertos en seguridad. Se estima que se requerirán aproximadamente 200 horas de ingeniería, con un coste total estimado de 12500€.

Implementación Final (TRL 9)

Llegar a TRL 9, es decir, implementar y operar completamente el sistema en su estado final, es un paso importante que involucra varios aspectos y costes adicionales. Estos incluyen la producción en masa, implementación de infraestructuras, el mantenimiento del servicio, el soporte del cliente y la formación del personal que utilizará el sistema.

Primero, la producción en masa del equipo necesario implica costes de producción, incluida la adquisición de componentes a gran escala, lo que puede implicar negociar con los proveedores y optimizar los procesos de fabricación para reducir costes.

En segundo lugar, implementar la infraestructura necesaria para soportar el sistema, especialmente si se trata de un sistema de control de acceso a gran escala, lo cual puede requerir inversiones en servidores, sistemas de seguridad y redes.

Además, es muy importante contar con un equipo de soporte y mantenimiento para mantener su sistema funcionando sin problemas y para resolver cualquier problema que pueda surgir. Esto incluye personal de mantenimiento, así como un equipo de soporte técnico que pueda ayudar a los clientes.

Finalmente, es necesario capacitar al personal que trabajará con el sistema para garantizar un uso adecuado y eficiente. Esto puede incluir el desarrollo de materiales de capacitación, así como sesiones de capacitación.

Teniendo todos estos aspectos en cuenta, el coste total de llevar el sistema a TRL 9 puede variar considerablemente según el tamaño del proyecto. Sin embargo, el coste aproximado de un sistema de control de acceso de tamaño medio puede oscilar entre 40000 y 60000€ aproximadamente, incluida la producción, implementación, el soporte, mantenimiento y la formación del personal.

Tabla 7: Resumen de Costes ajustados por niveles TRL

TRL	**Descripción**	**Componentes y servicios clave**	**Coste estimado**
4-6	Ensamblaje de prototipo funcional	1x NVIDIA Jetson Nano, *software* gratuito, componentes de integración	~8.000€
7-8	Demostración en entorno operativo real o equivalente	Acceso a instalaciones de prueba, adaptaciones al sistema, personal para pruebas	~12.500€
9	Demostración en entorno operativo real o equivalente	Mejora y documentación del prototipo, mantenimiento, soporte, formación del personal	~40.000 – 60.000€

Costes de Personal

Para llevar el proyecto a un nivel TRL 9, es necesario contar con un equipo de expertos en diversas áreas. Los costes de personal son una parte significativa del presupuesto total del proyecto. A continuación, se presenta un resumen de los costes de personal estimados, como se detalló previamente:

Tabla 8: Costes de personal estimados

Personal	Número de personas	Salario anual por persona (€)	Coste total anual (€)
Ingenieros de *software*	2	50.000	100.000
Ingenieros de *hardware*	1	50.000	50.000
Especialistas en pruebas	1	40.000	40.000
Expertos en seguridad	1	60.000	60.000
Gestor de proyecto/producto	1	60.000	60.000
Abogado (GDPR y Prop. Intelectual)	1	70.000	70.000

3.8. Regulaciones y consideraciones éticas

Las tecnologías de reconocimiento facial, como cualquier otra tecnología biométrica, están sujetas a un conjunto de reglas y normas reglamentarias que deben tener en cuenta al diseñar e implementar cualquier sistema para el que estén destinadas. Además, hay una serie de cuestiones éticas importantes que se deben considerar para garantizar que estas tecnologías se utilicen de manera responsable, respetando los derechos y la privacidad de todos.

Cuando se trata de regulaciones, los sistemas de reconocimiento facial están sujetos a muchas leyes y regulaciones en diferentes jurisdicciones, por lo que es muy importante conocer y cumplir con todas ellas. Por ejemplo, en la Unión Europea, el Reglamento General de Protección de Datos (GDPR) es importante para cualquier sistema que recopile y procese datos personales, incluidos los datos biométricos (Calabrò, 2019).

Según el GDPR, los datos biométricos se consideran "datos personales sensibles" y están sujetos a normas de protección estrictas. En particular, la recopilación y el procesamiento de estos datos a menudo requieren el consentimiento expreso del individuo y se toman las medidas apropiadas para proteger estos datos y garantizar su confidencialidad y seguridad.

Desde el punto de vista ético, el uso de la tecnología de reconocimiento facial plantea una serie de cuestiones. Una de las más importantes es la privacidad. Si bien el reconocimiento facial puede ofrecer importantes beneficios de seguridad y conveniencia, también puede verse como una molestia si se usa de forma errónea. Por lo tanto, al implementar estas tecnologías, es fundamental tener en cuenta la privacidad de las personas y garantizar que los datos se utilicen de forma que se respete y proteja su privacidad (Bala, 2022).

Otra cuestión ética importante es la cuestión de la precisión y el sesgo en los sistemas de reconocimiento facial. Varios estudios han demostrado que algunos sistemas de reconocimiento facial pueden tener tasas de error más altas para ciertos datos demográficos, lo que podría conducir a un trato injusto o discriminatorio. Por lo tanto, es importante que cualquier sistema de reconocimiento facial implementado sea preciso y justo para todos los usuarios (Mehrabi, 2019).

En resumen, al diseñar un sistema de control de acceso es importante tener en cuenta tanto los estándares legales como las consideraciones éticas que se han planteado anteriormente. La construcción de un sistema real de autentificación biométrica representa un reto no solamente técnico, sino también legal, para poder obtener todas las homologaciones y certificaciones necesarias. Dicho esto, cabe remarcar que está fuera del alcance del proyecto su análisis.

4. Análisis y selección de algoritmos

4.1. Introducción al análisis de algoritmos para el reconocimiento facial

La inteligencia artificial y el *Deep Learning* actualmente están explorando el reconocimiento facial como un tema candente. Las aplicaciones de este tipo de tecnología son innumerables y, por tanto, su potencial en la sociedad es inmenso. Por esto mismo es fundamental identificar y familiarizarse con los algoritmos más efectivos y que mejor se adecuen a las circunstancias de este proyecto.

La selección del algoritmo de la aplicación está determinada en gran medida por el problema a tratar, por lo que puede haber muchos factores que determinen el más correcto en cada caso. Estos incluyen la precisión y eficiencia del algoritmo (Bisht, 2022) y su robustez ante variaciones en las condiciones de iluminación (Safdar, 2021), entre otros.

El tipo de algoritmo de reconocimiento facial que se utiliza puede ser básico, como entrenar una red neuronal convolucional (CNN) para clasificar rostros según la identidad de la persona, o complejo, con diferentes técnicas que se emplean para capturar diferentes aspectos de las imágenes. Los dos algoritmos discutidos en este documento caen en esta última categoría.

Estos algoritmos utilizan enfoques avanzados de *Deep Learning* para capturar las distintas características faciales. Ambos algoritmos utilizan *embeddings* (Terhorst, 2020), que son representaciones de baja dimensión de las caras que capturan información clave necesaria para su procesamiento. Sin embargo, difieren en la forma en que generan y utilizan estos *embeddings*, así como la arquitectura que utilizan.

En los próximos apartados, se proporcionará una descripción más detallada de estos dos algoritmos, de discutirán sus puntos fuertes y debilidades, y se presentará un análisis comparativo para justificar la selección del algoritmo más adecuado para el sistema presentado en este proyecto.

4.2. Métodos de comparación de embeddings

Un *embedding* es una representación vectorial de alta dimensión que captura la información y características esenciales de un objeto, es este caso, se estaría hablando de caras humanas. Una vez que se hayan generado los *embeddings* para distintas entradas, estas se comparan para determinar su semejanza. Esta comparación se basa en la calculación de la distancia entre dos *embeddings* en el espacio establecido, donde una distancia pequeña indica una alta semejanza, al contrario que una distancia grande, lo cual indica una baja semejanza.

Hay varias formas de calcular estas distancias entre los *embeddings* (Nayak, 2022). Una opción común es calcular la distancia mediante la distancia euclidiana. Esta es la raíz cuadrada de la suma de las diferencias cuadradas entre los componentes correspondientes de los *embeddings*. Otra de las opciones sería utilizando la distancia del coseno, el cual mide el ángulo entre los dos *embeddings*. Por último, tenemos la distancia L1 (Manhattan), la cual se calcula tomando la suma de los valores absolutos del vector.

Esta elección de la medida dependerá de la tarea que se quiera llevar a cabo y de los datos con los que se vayan a trabajar, ya que estas juegan un papel crítico en el funcionamiento de las Redes Siamesas y otros algoritmos de *Deep Learning*.

Es importante remarcar que, a pesar de la dificultad de generar y comparar *embeddings*, una Red Siamesa es capaz de aprender automáticamente a realizar distintas tareas a través del entrenamiento, sin necesidad de una programación explícita para ello.

Distancia Euclidiana

La distancia euclidiana es una de las métricas más utilizadas (Wu, 2021). Es, en esencia, la longitud de la línea más corta entre dos puntos en un espacio euclidiano (como un espacio tridimensional ordinario). La fórmula general para calcular la distancia euclidiana entre dos puntos P y Q es:

$$D(P,Q) = \sqrt{(p1-q1)^2 + (p2-q2)^2 + \cdots + (pn-qn)^2}$$

Donde P = (p1, p2, …, pn) y Q = (q1, q2, …, qn) son dos puntos en un espacio n-dimensional.

Cada uno de estos puntos representa un vector de características, donde la distancia euclidiana mide la separación "geométrica" entre estos.

Distancia Coseno

A diferencia de la distancia euclidiana, que considera la separación geométrica entre dos puntos, la distancia coseno mide el ángulo entre dos vectores (Kirişci, 2022). Esta es una medida de orientación y no tiene en cuenta la magnitud de los vectores. La fórmula general para calcular la distancia coseno es:

$$D(P,Q) = 1 - \frac{P \cdot Q}{||P||\,||Q||}$$

Donde P y Q son dos vectores, "·" denota el producto punto de los vectores, y $||P||$ y $||Q||$ son las magnitudes de los vectores.

La distancia coseno varía entre -1 y 1. La distancia coseno de dos vectores con la misma dirección es 0, mientras que la distancia de correlación entre dos tipos de vectores diferentes con direcciones opuestas es 1. Cuando se trabaja con *Deep Learning*, normalmente se establece un espacio de dimensiones positivas, por lo que la distancia coseno oscilará entre 0 y 1.

La distancia coseno es útil cuando la magnitud de las diferencias entre vectores es menos importante que su orientación. Esto al hace especialmente popular en tareas como la similitud de documentos, donde la presencia o ausencia de términos específicos (orientación) es más importante que la influencia de esos términos (magnitud).

Distancia L1 (Manhattan)

La distancia L1, también conocida como distancia de Manhattan, es otra métrica importante que se utiliza para medir la similitud entre dos puntos en un espacio n-dimensional (Suwanda, 2020). A diferencia de la distancia euclidiana, que se calcula como la longitud de la línea más corta entre dos puntos, la distancia de Manhattan tiene en cuenta las diferencias absolutas entre sus coordenadas. Su nombre deriva de la distancia que debe recorrer un peatón para llegar de un punto a otro en ciudades con calles cuadriculadas, como la ciudad de Nueva York.

La fórmula general para calcular la distancia L1 es:

$$D(P,Q) = |p1 - q1| + |p2 - q2| + \cdots + |pn - qn|$$

Donde P = (p1, p2, …, pn) y Q = (q1, q2, …, qn) son dos puntos en un espacio n-dimensional.

La distancia de Manhattan entre dos puntos se representa como la suma de las variaciones verticales y horizontales, similar a caminar sobre un tablero de ajedrez con solo movimientos horizontales y verticales.

A diferencia de la distancia euclidiana, la distancia de Manhattan es menos sensible a los cambios repentinos en la diferencia entre los componentes individuales de los vectores. Esto la hace útil en ciertos escenarios donde las diferencias en una sola dimensión no deben tener un impacto desproporcionado en la distancia total.

4.3. Algoritmo de la Red Siamesa

La Red Siamesa es un enfoque de *Deep Learning* que se utiliza comúnmente para problemas de verificación o reconocimiento, donde el objetivo es comparar nuevas entradas con conjuntos conocidos y determinar si estas coinciden. Este algoritmo lleva el nombre "siamés" debido a su estructura, en la que se utilizan dos redes neuronales idénticas que comparten el mismo peso y parámetros para así funcionar en paralelo (Sun, 2022).

4.3.1. Funcionamiento del Algoritmo de la Red Siamesa

Como he mencionado anteriormente, la estructura de este algoritmo se basa en dos redes neuronales idénticas, donde cada una tomará una entrada diferente, y las salidas de estas se combinan para proporcionar una medida de la semejanza entre las entradas.

El principio fundamental del funcionamiento del algoritmo se basa en el concepto de "distancia" en un espacio de características. La red Siamesa transforma las entradas en vectores de características, también conocidos como *embeddings*, donde la distancia entre dos vectores se interpreta como una medida de semejanza. Por ello, un mayor grado de similitud entre dos entradas significará que sus *embeddings* están más cerca, dentro del espacio correspondiente de características. En el caso del reconocimiento facial, esta medida está directamente relacionada con le identidad de las personas.

Estas Redes Siamesas se entrenan mediante un proceso conocido como aprendizaje contrastivo. La red se entrena con pares de ejemplos, que se etiquetan para indicar su similitud o divergencia. La red aprende a mapear las entradas a *embeddings* en el espacio de características de tal manera que las entradas iguales se mapeen cerca unas de otras, mientras que las diferentes se mapeen más lejos unas de otras (Soleymani, 2020).

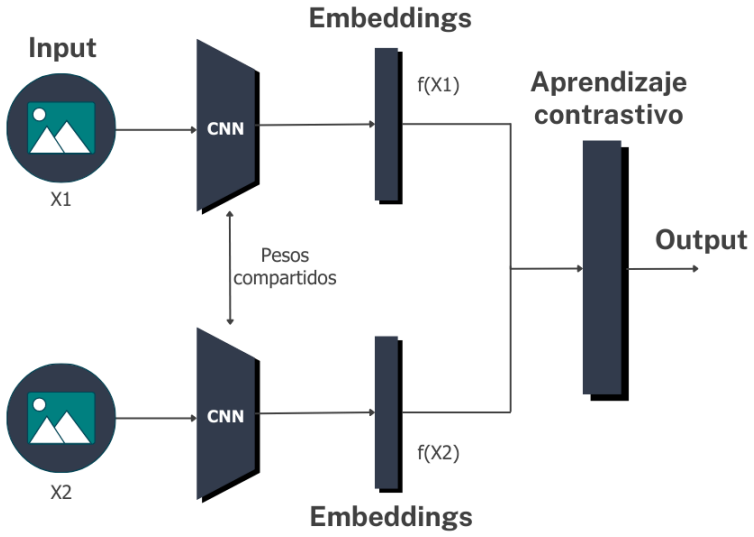


Figura 8: Arquitectura de las Redes Siamesas

FUENTE: ELABORACIÓN PROPIA

4.3.2. Generación y Comparación de embeddings en la Red Siamesa

Como he mencionado anteriormente, en el entrenamiento de una Red Siamesa mediante aprendizaje contrastivo, el objetivo es aprender una función de *embedding* en la que las imágenes similares se mapeen cerca mientras que las dispares se mapeen distantes entre ellas.

Una vez entrenada la red neuronal, los *embeddings* son generados. Al igual que se mencionó anteriormente, capa cada entrada, la red Siamesa genera un vector de características que es un punto en un espacio multidimensional. Esto se logra mediante una serie de transformaciones que implican convoluciones, pooling y otras operaciones típicas de las redes neuronales convolucionales (Koch, 2015).

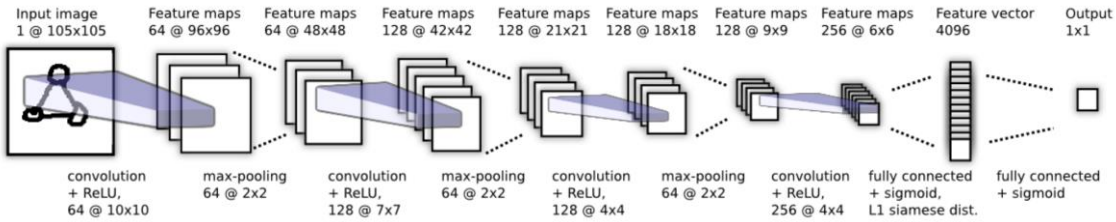


Figura 9: Arquitectura convolucional seleccionada

FUENTE: EXTERNA [23]

La función de pérdida del aprendizaje contrastivo desempeña un papel crucial en este proceso (Khosla, 2020). Esta función intenta asegurar que los *embeddings* de las imágenes similares estén cerca, mientras que las de las imágenes dispares estén separadas por un margen. Esto hace que la red aprenda una función de incrustación significativa.

23 Fuente de Figura 9: (Koch, 2015)

Una vez que los *embeddings* han sido generados, es hora de compararlos. En el caso de las Redes Siamesas, típicamente se utiliza la distancia [L1 Manhattan] para medir la similitud entre dos *embeddings*. Con esto, se puede diferenciar la distancia, haciendo que las distancias por debajo de un cierto umbral sean consideradas como iguales y las que superen ese umbral, se consideren desiguales.

4.3.3. Aplicación de la Red Siamesa al reconocimiento facial

Las redes Siamesas se han utilizado con éxito para el reconocimiento facial. En este proyecto, cada red toma una imagen de una cara como entrada y produce un *embedding* que captura sus características, las cuales serán comparadas para obtener una distancia entre ellas y determinar su similitud, como se ha mencionado anteriormente.

La ventaja de este enfoque es que puede manejar un gran número de identidades diferentes sin necesidad de reentrenar la red cada vez que se añada una nueva identidad.

4.3.4. Ventajas y desventajas de la Red Siamesa

Las redes Siamesas tienen varias ventajas. Son capaces de aprender una representación de las caras que es robusta ante variaciones en la postura y las diferentes expresiones faciales. Además, pueden gestionar una amplia gama de identidades y son computacionalmente eficientes, gracias a los mismos pesos y parámetros utilizados para ambas redes (Srihari, 2022).

Sin embargo, también hay algunas desventajas en el uso de estas. Elegir y usar pares de imágenes apropiados puede plantear dificultades de entrenamiento para las redes, además que estos entrenamientos suelen ser de una muy larga duración debido a la cantidad de operaciones que se deben realizar (Naseri, 2021). Además, aunque las redes Siamesas pueden generar *embeddings* de caras de alta calidad, estos no son directamente interpretables, lo que dificulta la comprensión y el diagnóstico del rendimiento del algoritmo.

Otro factor importante que considerar es que las redes Siamesas pueden ser sensibles a los cambios en la iluminación. Esto significa que, si la iluminación de la imagen de entrada difiere a la iluminación de las imágenes utilizadas en el entrenamiento, la precisión del sistema puede verse afectada (Xu, 2020).

En general, las redes Siamesas representan una opción sólida para el reconocimiento facial debido a su capacidad para obtener características faciales complejas y generar representaciones de las caras. No obstante, para determinar qué algoritmo es el más adecuado para el proyecto, es esencial tener en cuenta la variación de iluminación y otros posibles problemas.

Teniendo en cuenta los pros y los contras de las redes Siamesas, se decidió explorar la implementación de este algoritmo en el proyecto por varias razones principales. Primero, su capacidad de aprender representaciones de caras es muy robusta, especialmente cuando se tiene que lidiar con diferentes poses y expresiones faciales.

Además, la eficiencia computacional de las redes Siamesas la convierte en una opción atractiva para implementaciones en tiempo real y donde los recursos disponibles son limitados, lo cual encaja con los objetivos de este proyecto.

Los desafíos asociados con el entrenamiento y la sensibilidad a la iluminación son preocupaciones considerables. Por este motivo entre otros, en un principio se tenía pensado implementar este

algoritmo en el prototipo final de este proyecto, hasta que, debido a sus limitaciones, solo se quedó en un algoritmo implementado desde cero pero que no estará en el sistema final.

4.4. Algoritmo FaceNet

FaceNet es un algoritmo desarrollado por Google [24] que utiliza redes neuronales convolucionales para generar *embeddings* de imágenes de caras. Estos *embeddings*, como bien he mencionado anteriormente, son representaciones de alta dimensión que capturan características clave de las caras. Este algoritmo está diseñado principalmente para tareas de reconocimiento facial y verificación de identidad. El enfoque de FaceNet es distinto de los métodos convencionales que se enfocan en optimizar las puntuaciones de coincidencia entre pares de imágenes de caras, ya que busca adquirir una representación en un espacio de *embeddings* donde las distancias se correlacionan directamente con la similitud facial.

4.4.1. Funcionamiento del Algoritmo FaceNet

FaceNet busca aprender una representación de características, o *embedding*, del rostro de tal manera que la distancia entre dos imágenes de la misma persona resultaría pequeña, mientras que la de dos *embeddings* de personas diferentes sería grande.

Este algoritmo utiliza una arquitectura de red neuronal convolucional (CNN) profunda. Durante el entrenamiento, este algoritmo no aprende a clasificar imágenes en identidades específicas. En cambio, se entrena mediante una función de pérdida de tripletes (triplet loss).

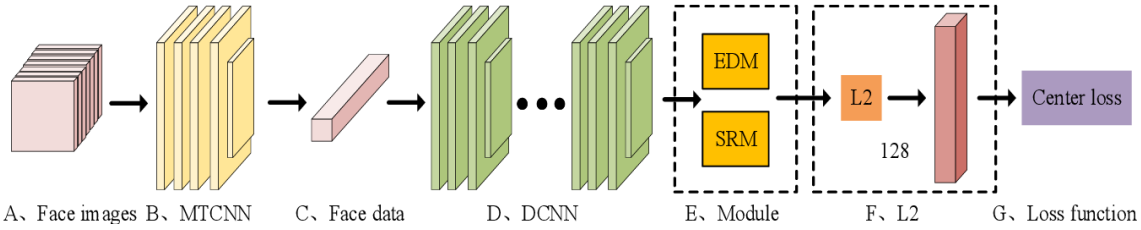


Figura 10: Arquitectura del modelo FaceNet

FUENTE: EXTERNA [25]

La detección de caras es el primer paso esencial en el funcionamiento de FaceNet. Consiste en identificar y localizar las caras presentes en una imagen. Con FaceNet, se suelen utilizar detectores de caras de alta precisión, como el detector MobileNet [26], para asegurar que las caras sean detectadas con precisión incluso en diferentes condiciones de luz y orientaciones diversas (Li, 2022).

Una vez que se detecta la cara, el siguiente paso es alinear la cara. Este proceso implica ajustar y transformar la cara detectada para que la cara tenga una orientación estándar. La alineación generalmente implica encontrar puntos de referencia clave de la cara, y usarlos para transformar la cara en una orientación frontal y tamaño estandarizado. Con esto se garantiza la correcta representación de las caras antes de pasarlas por la red neuronal, para una correcta generación de los *embeddings* (Kangwanwatana, 2022).

24 https://www.google.com/

25 Fuente de figura 10: (Gu, 2021)

26 https://mobilenet.uma.es/

4.4.2. Generación y Comparación de embeddings en FaceNet

Como he mencionado anteriormente, FaceNet genera *embeddings* de imágenes mediante el entrenamiento de una red neuronal con una función de optimización específica llamada función de tripletes. Esta función es esencial para poder garantizar que las representaciones vectoriales de las imágenes similares estén cerca entre sí, mientras que las dispares estén más alejadas (Hermans, 2017).

Un triplete consiste en tres elementos: Una imagen ancla (A), una imagen positiva (P) que es de la misma clase que la ancla, y una imagen negativa (N) que es de una clase diferente. Con clase se está haciendo referencia a la identidad de la persona de la imagen.

La función de pérdida de triplete intenta minimizar la separación entre la imagen ancla y la imagen positiva, mientras maximiza la distancia entre la imagen ancla y la imagen negativa. Esto se logra asegurándose de que la distancia entre la imagen ancla y la positiva más un margen (generalmente denotado como alfa) sea menor que la distancia entre la imagen ancla y la imagen negativa. Matemáticamente, esta función se describe como:

$$Triplet\ Loss = \max\left(||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + \alpha, 0\right)$$

Donde $f(A)$, $f(P)$, y $f(N)$ son los *embeddings* de las imágenes ancla, positiva y negativa, y ||.|| denota la norma euclidiana. La función de pérdida busca minimizar este valor.

La elección de tripletes durante el entrenamiento es crítica. Si los tripletes son demasiado fáciles, el modelo no aprende características útiles, mientras que, si son demasiado difíciles, el modelo puede tener dificultades para coincidir.

Una vez creados los *embeddings*, estos son comparados teniendo en cuenta la distancia entre los dos vectores. Para ello, la [distancia euclidiana] es comúnmente utilizada (Oo, 2019). Con esto, se puede determinar si dos imágenes de caras humanas pertenecen a la misma persona o no. Si la distancia entre dos *embeddings* es pequeña, significará que ambas imágenes pertenecen a la misma persona, mientras que, si la distancia es grande, sugiere que las imágenes representan caras de personas diferentes.

En la práctica, se establece un umbral de distancia. Este umbral será el que determine si dos imágenes son consideradas de la misma persona o no. Es decir, si la distancia entre dos *embeddings* es inferior al umbral, se considera que representan a la misma persona, de lo contrario, se consideran identidades diferentes.

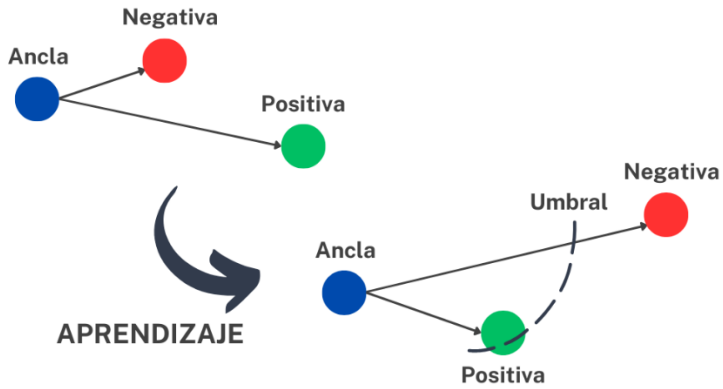


Figura 11: Minimización de distancia mediante la función de pérdida de tripletes

FUENTE: ELABORACIÓN PROPIA

4.4.3. Aplicación de FaceNet al reconocimiento facial

FaceNet, como algoritmo de *Deep Learning*, ha sido muy utilizado en el campo del reconocimiento facial debido a su excepcional rendimiento y precisión. Existen muchas áreas sonde destaca, incluidas la seguridad, la autenticación biométrica, la asistencia por reconocimiento facial…

La implementación en cámaras de seguridad ha sido una aplicación crítica. Esto es debido a la precisión que FaceNet tiene a la hora de generar *embeddings* de las caras, facilitando así la identificación de individuos sospechosos en investigaciones de seguridad pública.

También se utiliza en sistemas de autentificación biométricas, como el desbloqueo facial en dispositivos móviles y sistemas de control de acceso.

Se utiliza también en entornos educativos y corporativos donde en lugar de depender de métodos manuales, los sistemas de asistencia basados en FaceNet pueden identificar automáticamente a los empleados o estudiantes cuando ingresan al recinto y registrar su asistencia.

4.4.4. Ventajas y desventajas de FaceNet

Una de las ventajas más importantes de FaceNet es su alta capacidad para generar *embeddings* muy precisos de las caras. Esto se traduce en la capacidad que tiene para distinguir entre diferentes individuos con un alto rango de acierto, lo cual es esencial para aplicaciones que requieran de mucha precisión (Goel et al., 2021).

Otra ventaja notable es la robustez que tiene ante variaciones en las características faciales. El algoritmo es capaz de administrar de manera efectiva varias modificaciones, como cambios en la iluminación, las expresiones faciales y los ángulos de la cabeza. La adaptabilidad de FaceNet a las condiciones del mundo real lo convierte en un activo valioso para las distintas aplicaciones (Anaya, 2019).

Adicionalmente, la incorporación de la función de pérdida de tripletas es un factor que mejora significativamente la calidad de los *embeddings*.

Sin embargo, también tiene ciertas desventajas. Una de ellas es el requisito de disponer de tripletas de entrenamiento, que incluyen una imagen de ancla, una imagen positiva y una

negativa. Esto puede complicar el preprocesamiento y la organización de los datos, lo que puede implicar algún desafío.

A su vez, la selección de imágenes durante el entrenamiento es otra desventaja. Escogerlas y que estas no sean desafiantes para el algoritmo puede resultar en un modelo que no está bien generalizado y que puede no funcionar de manera óptima.

Dado el análisis de las ventajas y desventajas, se consideró la implementación de este algoritmo como una opción altamente viable para el proyecto. Su capacidad de general *embeddings* precisos y su robustez frente a variaciones de características faciales lo hacen particularmente adecuado para un sistema de reconocimiento facial destinado a ser eficaz en entornos del mundo real.

4.5. Comparación de algoritmos

Una vez explicados el funcionamiento y tanto las ventajas como las desventajas de cada uno de los algoritmos, es hora de realizar una comparación detallada entre ellos con el objetivo de seleccionar el más adecuado para el proyecto propuesto en este documento.

4.5.1. Criterios de comparación

Es esencial establecer criterios claros y objetivos a la hora de comprar los distintos algoritmos. Por ello, los criterios seleccionados para la comparación son:

- Precisión: Como de precisa es la capacidad del algoritmo para identificar de forma correcta las caras, teniendo en cuenta diferentes condiciones de iluminación, orientación y expresiones.
- Robustez: Cómo el algoritmo maneja las diferencias en iluminación, ángulos de la cara y otras variaciones en las imágenes.
- Velocidad de procesamiento: El tiempo que toma el algoritmo en procesar una imagen y generar resultados. Esto es crítico para aplicaciones en tiempo real.
- Facilidad de implementación: Cómo de sencillo es aplicar el algoritmo en un sistema y personalizarlo para cumplir los requerimientos necesarios.

4.5.2. Resultados de la comparación

Basándose en los criterios de comparación establecidos, se pueden observar las siguientes tendencias:

- Precisión: FaceNet tiende a tener una precisión bastante superior en comparación con la Red Siamesa, especialmente en conjuntos de datos grandes y diversos.
- Robustez: FaceNet generalmente es más robusto ante variaciones de la imagen. Por el contrario, la precisión de la Red Siamesa en condiciones diferentes de luz suele dar bastantes problemas.
- Velocidad de procesamiento: La Red Siamesa puede ser más rápida en ciertos escenarios debido a su estructura más sencilla, mientras que FaceNet, debido a su estructura más profunda, requiere de algo más de tiempo para generar los *embeddings*.
- Facilidad de implementación: Implementar FaceNet desde cero podría ser bastante más complicado que la implementación de una Red Siamesa. Sin embargo, uno de los beneficios de FaceNet es que dispone de muchos modelos pre-entrenados y de código abierto. Teniendo esto en cuenta, la implementación de FaceNet se ve claramente facilitada, haciendo que llegue a ser más fácil de implementar que la Red Siamesa, aunque su estructura sea más compleja.

4.6. Conclusión sobre la selección de algoritmos

Existen múltiples algoritmos y arquitecturas disponibles para el reconocimiento facial. Algunas de ellas son: Eigenfaces, Fisherfaces, Local Binary Patterns Histograms (LBPH), DeepFace, Redes Siamesas, FaceNet, MobileNet, VGGFace y ArcFace.

La selección de las redes Siamesas y FaceNet para este proyecto fue dictada por su popularidad y efectividad en el reconocimiento facial. Además, ambos algoritmos son los métodos más avanzados para el *Deep Learning*, lo cual está alineado con los objetivos del este proyecto de utilizar tecnología de vanguardia.

Vale la pena señalar que se tomó la decisión de implementar las redes Siamesas desde cero. Esto se realizó con el objetivo de adquirir un entendimiento más profundo de cómo funcionan los algoritmos de reconocimiento facial basados en *Deep Learning* y obtener experiencia práctica en la implementación de estos sistemas.

Después de analizar las características y capacidades de dos algoritmos seleccionados, FaceNet y las redes Siamesas, se decidió optar por FaceNet como el algoritmo principal de reconocimiento facial para ese proyecto. Esta elección se basó en varios factores clave.

FaceNet ha demostrado ser un poco más preciso que las redes Siamesas y, en un sistema de control de acceso, la precisión es primordial. FaceNet también muestra una mayor robustez a las condiciones cambiantes. Esto es muy importante en aplicaciones prácticas donde el sistema necesita adaptarse y operar eficientemente en diferentes entornos.

Otra ventaja de FaceNet es la disponibilidad de modelos pre-entrenados. Si bien implementar FaceNet desde cero puede ser difícil, el acceso a modelos de código abierto simplifica y acelera enormemente el proceso de implementación.

FaceNet también destaca en términos de escalabilidad, siendo capaz de manejar eficientemente grandes conjuntos de datos. Esto es útil si el sistema necesita procesar una gran base de datos de caras en el futuro.

El apoyo de la comunidad es otro factor que inclina la balanza a favor de FaceNet. Debido a que el algoritmo se usa tan ampliamente, hay disponible una gran cantidad de recursos, documentación y soporte.

Es cierto que FaceNet puede requerir más poder computacional en comparación con las redes Siamesas, pero esta compensación es justificada por las mejoras en precisión y robustez que FaceNet proporciona. Además, al utilizar la NVIDIA Jetson Nano como *hardware* embebido, se dispone de suficiente capacidad de procesamiento para manejar las demandas de FaceNet de manera eficiente.

En resumen, FaceNet fue seleccionado como el algoritmo principal para este proyecto debido a su precisión superior, robustez, escalabilidad, y el amplio soporte de la comunidad. Sin embargo, la implementación de las redes Siamesas sirvió como una valiosa experiencia de aprendizaje y una oportunidad para comprender más a fondo los fundamentos de los algoritmos de reconocimiento facial.

5. Análisis, Diseño e implementación del sistema

5.1. Introducción

En este capítulo, nos enfocaremos en el diseño e implementación del sistema de control de acceso basado en reconocimiento facial. Abordaremos el análisis del sistema, los requisitos funcionales que se relacionan con el software, y los requisitos no funcionales, como la robustez, estética y usabilidad. Python se ha seleccionado como el lenguaje de programación para su implementación y TensorFlow como *framework* para crear y entrenar los algoritmos de aprendizaje automático. El objetivo es demostrar un sistema que sea confiable en términos de seguridad y al mismo tiempo efectivo, asequible y fácil de utilizar, sobre todo, para la persona encargada del manejo del sistema.

5.2. Análisis del Sistema

En la fase inicial, es imprescindible comprender el contexto en el que el sistema operará y las necesidades que debe satisfacer. El sistema tiene como objetivo principal mejorar la seguridad en entornos empresariales. Esto implica permitir o denegar el acceso a las instalaciones en función de la identificación facial. El sistema debe ser capaz de procesar imágenes, identificar caras y verificar la identidad de manera eficiente y precisa.

5.2.1. Requisitos Funcionales

En cuanto a los requisitos funcionales, es fundamental que el *software* tenga características específicas que permitan satisfacer las necesidades del sistema. El reconocimiento facial es una de las principales funcionalidades, y el sistema debe ser capaz de identificar caras con alta precisión. Además, debe haber un componente de gestión de usuarios que permita el registro de nuevos usuarios, y que pueda almacenar y administrar los datos relacionados con la identificación facial. También es importante tener una interfaz de usuario intuitiva, que facilite la interacción con el sistema y permita el registro y gestión de los usuarios, así como el monitoreo de eventos de acceso. En cuanto a la seguridad, es crucial garantizar que los datos de los usuarios y cualquier otra información sensible, estén seguros. Por último, la integración con el *hardware* embebido seleccionado es esencial, y el software debe ser capaz de interactuar con los módulos de cámara y otros componentes del sistema.

5.2.2. Requisitos no Funcionales

En relación con los requisitos no funcionales, estos son aspectos que tienen que ver con cómo debe ser el sistema, en lugar de qué debe hacer. Uno de estos aspectos es la robustez. El sistema debe ser sólido y capaz de operar de manera confiable en diferentes condiciones, como variaciones de la iluminación y ángulos de la cara. En cuanto a la usabilidad, es importante que el sistema sea fácil de usar, no requiriendo un extenso entrenamiento o conocimiento técnico por parte de los usuarios finales. La estética también es un factor importante, la interfaz de usuario debe tener un diseño atractivo y profesional que sea acorde con un entorno empresarial. Además, el tiempo de respuesta es un factor crítico, donde el sistema debe ser capaz de procesar la información y tomar decisiones de acceso en un tiempo razonablemente corto para no entorpecer el flujo de personal. Finalmente, la escalabilidad es un aspecto clave para permitir que el sistema pueda acomodar un creciente número de usuarios sin degradar su rendimiento.

5.2.3. Análisis de Casos de Uso

El sistema de gestión y control de usuarios es un componente integral de la aplicación de control de acceso. Esta sección asegura que solo los usuarios autorizados tengan acceso a ciertas áreas o funcionalidades. Este sistema incluye varias funcionalidades como el registro y autentificación de usuarios, gestión de permisos y se encarga de manejar la interacción entre usuarios y el sistema. Además, este módulo es el que permite la creación de nuevos usuarios al sistema.

5.2.4. Interacción entre usuarios y el sistema

La interacción entre usuarios y el sistema es facilitada por una interfaz de usuario intuitiva. Los usuarios autenticados pueden interactuar con el sistema de acuerdo con los permisos asociados a su rol.

Es importante visualizar las interacciones del sistema entre los diferentes usuarios (actores) y ver así los diferentes aspectos de un sistema. Para ello, crear un diagrama de casos de uso es idóneo.

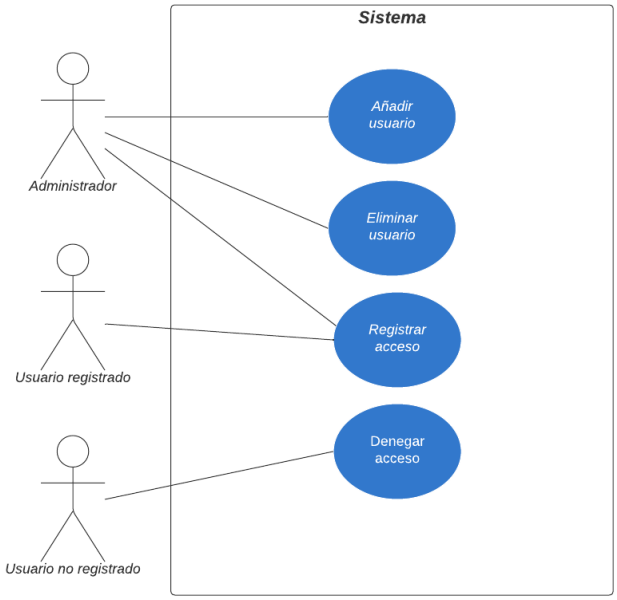


Figura 12: Diagrama de casos de uso UML

FUENTE: ELABORACIÓN PROPIA

5.2.5. Registro y autenticación de usuarios

El primer paso para que un usuario interactúe con el sistema es registrarse y autenticarse mediante él. Durante el registro, los usuarios tendrán que introducir su nombre y darán su consentimiento para registrar imágenes suyas en el sistema para permitir su posterior identificación.

Una vez registrados, los usuarios deben autenticarse para acceder al sistema. Esta autenticación se realiza mediante el reconocimiento facial, el cual mediante las imágenes que el usuario ha introducido, permitirá al sistema identificar si el usuario en cuestión está o no dentro del sistema.

5.2.6. Gestión de permisos y roles

En este sistema podemos clasificar a los diferentes usuarios en 3 categorías: Administrador, usuarios regulares y visitantes.

Administrador

El administrador tiene el control total sobre el sistema. Esto incluye la capacidad de añadir, eliminar, monitorear los usuarios y la actividad de estos. Este usuario será el único en saber la contraseña que requiere entrar en la sección del administrador, por lo que nadie más tiene acceso a estas configuraciones del sistema.

Usuario Registrado

Llamamos usuarios registrados a estos individuos que están actualmente dentro del sistema, es decir, que son capaces de ser identificados por el sistema actualmente. No pueden modificar la configuración del sistema ni administrar otros usuarios. Sin embargo, pueden ver el historial de acceso del sistema, ya que se tratan de personas autorizadas en el sistema.

Usuario no Registrado

Estos son usuarios no autorizados en el sistema. En el estado actual, estos no tienen ningún tipo de permisos en el sistema por lo que lo único que pueden hacer es activar la cámara para intentar identificarse, aunque esto no vaya a funcionar. Por otro lado, siempre van a poder acudir al administrador del sistema para que en caso de tener que autorizarlos, encargarse del proceso.

5.3. Diseño del Sistema

En esta sección, el enfoque cambia hacia el desarrollo de la aplicación que se ejecutará en el *hardware* embebido y que será el responsable del correcto funcionamiento del reconocimiento facial y el control de acceso.

5.3.1. Implementación de la interfaz de usuario

La interfaz de usuario se pensó para que fuera lo más fácil e intuitiva posible. Por este mismo motivo, se diseñó una interfaz basada en ventanas, lo cual hace que el manejo de estas y la adaptación al espacio que tengamos sean lo más personalizables posibles.

Se realizó un diagrama con las diferentes ventanas que esta tendría y se diseñó en su totalidad antes de realizar ninguna implementación para asegurar su correcta funcionalidad y cerciorar su correcta integración con el sistema a desarrollar y las especificaciones requeridas del sistema.

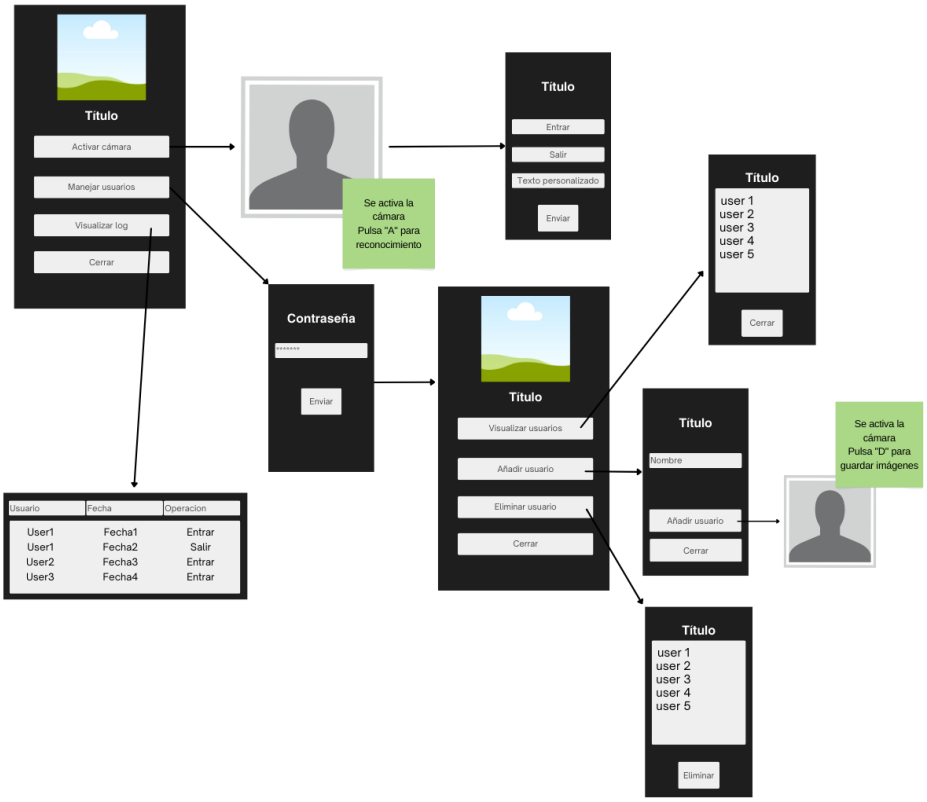


Figura 13: Diagrama de flujo de la interfaz de usuario

FUENTE: ELABORACIÓN PROPIA

Al ejecutar el programa, se abrirá la primera ventana, la cual mostrará varias opciones que el usuario va a poder seleccionar. Estas opciones dependerán de los permisos que tengas en el sistema, aunque eso lo discutiremos próximamente.

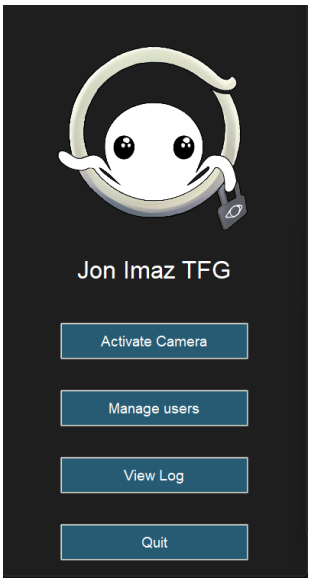


Figura 14: Interfaz de la ventana principal

FUENTE: ELABORACIÓN PROPIA

La intención principal de esta interfaz era que fuera minimalista y simple, mostrando solamente opciones totalmente necesarias y evitando mostrar demasiada información al usuario al mismo tiempo. Entre las opciones podemos encontrar la opción de activar la cámara, la cual es la opción principal y la encargada del reconocimiento facial de los distintos usuarios que estén registrados en el sistema.

Una vez se ha detectado a una persona registrada en el sistema, este va a disponer de varias opciones a escoger. Estas opciones incluyen la de entrar y salir, aunque por si en algún caso alguien quisiera añadir algo que no fuera ninguna de ellas por alguna circunstancia, hay una tercera opción que permite al usuario introducir lo que desee.

Hablando de usuarios, la siguiente opción estará restringida por una contraseña de administrador, ya que aquí será donde se van a poder añadir, eliminar y visualizar los usuarios en el sistema.

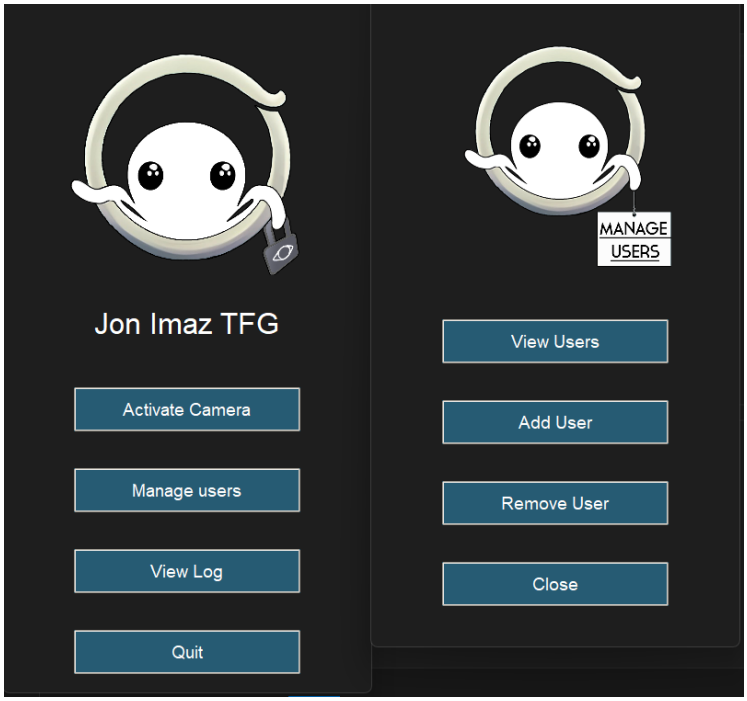


Figura 15: Interfaz de la ventana de manejo de usuarios

FUENTE: ELABORACIÓN PROPIA

Esta ventana está pensada para ser abierta únicamente por el usuario encargado de administrar el sistema. Esta persona será la encargada del funcionamiento correcto de los datos en el sistema. Para ello, ahora se disponen de varias opciones:

- Visualizar usuarios: Al presionar en este botón, se abrirá una ventana nueva. Esta, mostrará en un formato de lista todos los usuarios registrados en el sistema. Esto es muy útil para facilitar al encargado de la gestión del sistema comprobar qué usuarios hay, y comprobarlo de forma sencilla e intuitiva.

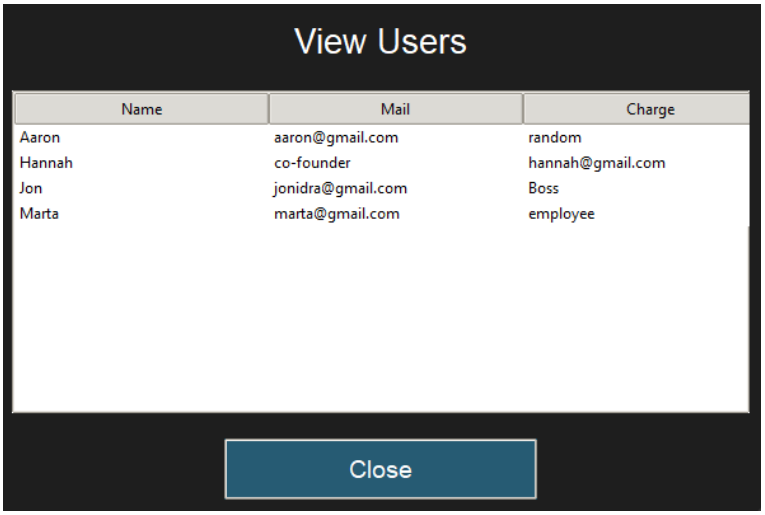


Figura 16: Interfaz de la ventana para visualizar usuarios

FUENTE: ELABORACIÓN PROPIA

- Añadir usuario: Esta opción permitirá la inserción de una nueva identidad que el sistema de reconocimiento facial va a reconocer. Para ello, lo único que hará falta será introducir el nombre, mail y puesto del nuevo usuario y presionar en el botón "Añadir usuario".

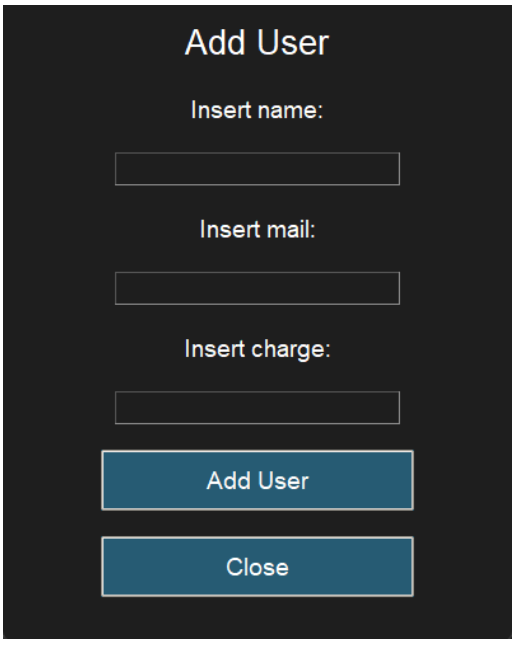


Figura 17: Interfaz de la ventana de añadir usuario

FUENTE: ELABORACIÓN PROPIA

Una vez presionado el botón para añadir el usuario, una nueva ventana se abrirá mostrando la cámara. La nueva persona va a tener que posicionarse delante de esta y tendrá que presionar la tecla "D" del teclado para comenzar a almacenar muestras de su cara. Es recomendable que en este proceso el usuario mueva su cabeza para así obtener diferentes ángulos de la cara.

- Eliminar usuario: Por último, nos encontramos con la opción de eliminar usuarios. Dado que a esta opción solamente puede acceder el administrador del sistema mediante su contraseña, no será necesaria ninguna autentificación adicional para llevar a cabo estos procesos. Una vez presionado el botón, se abrirá una nueva ventana mostrando una lista con los distintos usuarios que tenemos en el sistema. Si presionamos ante alguno de ellos, y presionamos el botón de eliminar usuario, este será totalmente eliminado del sistema, haciendo que el sistema deje de reconocer a esa persona en cuestión.

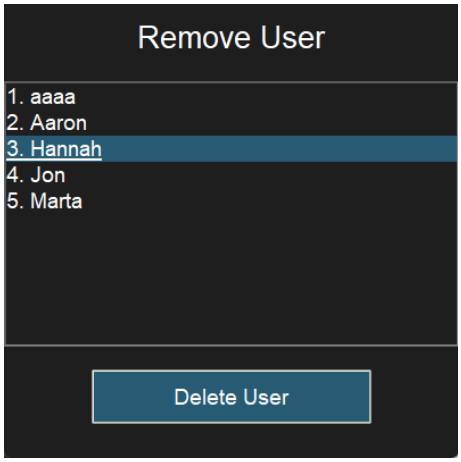


Figura 18: Interfaz de la ventana de eliminación de usuarios

FUENTE: ELABORACIÓN PROPIA

Por último, en la ventana principal podemos observar la opción para visualizar el registro de la actividad que se haya tenido en la compañía.

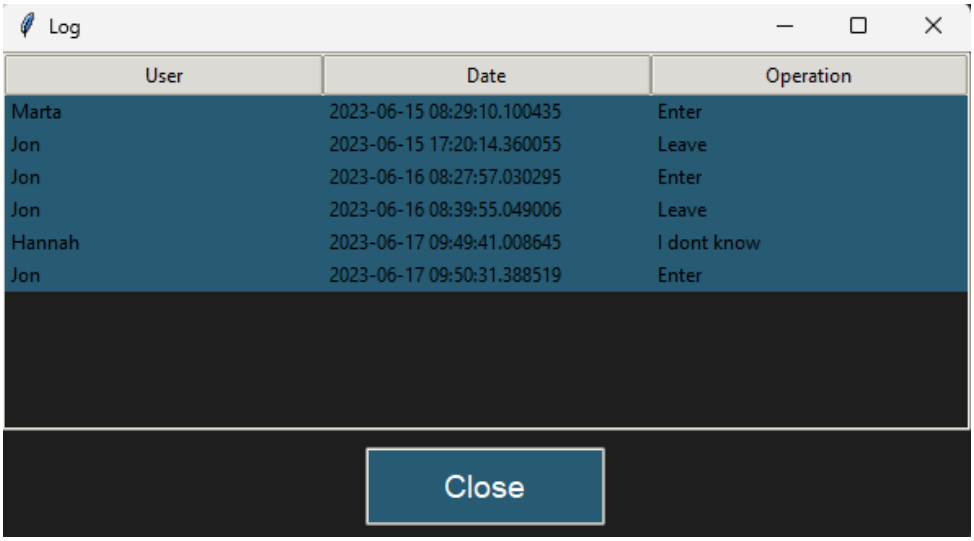


Figura 19: Interfaz de la ventana de actividad de los usuarios

FUENTE: ELABORACIÓN PROPIA

Hay que puntualizar que esta actividad se puede ordenar de forma que vamos a poder ver las actividades en el orden cronológico que queramos y, además, se va a poder filtrar por usuario,

para que la actividad de los usuarios esté en orden alfabético en vez de tenerlo de forma cronológica.

5.3.2. Diseño del sistema de control de acceso

Para integrar el sistema de reconocimiento facial que se ha implementado anteriormente y poder crear un sistema de control de acceso funcional hace falta diseñar correctamente todas las partes que compondrán este sistema.

Para ello, existen varias herramientas que facilitan ver el desarrollo de cualquier tipo de proyecto. Es muy importante pensar y clarificar todos los detalles antes de lanzarse al desarrollo de cualquier tipo de aplicación puesto que, de esta forma, se evitan posibles errores o cambios potenciales en el programa.

Para poder ser capaces de implementar el programa, primero hay que pensar y diseñar toda la actividad que se va a poder llevar a cabo al estar ejecutando la aplicación. Los diagramas de actividad sirven para describir los recursos utilizados, cómo estos son utilizados, y es una manera excelente de visualizar el flujo de acciones y decisiones dentro de la aplicación.

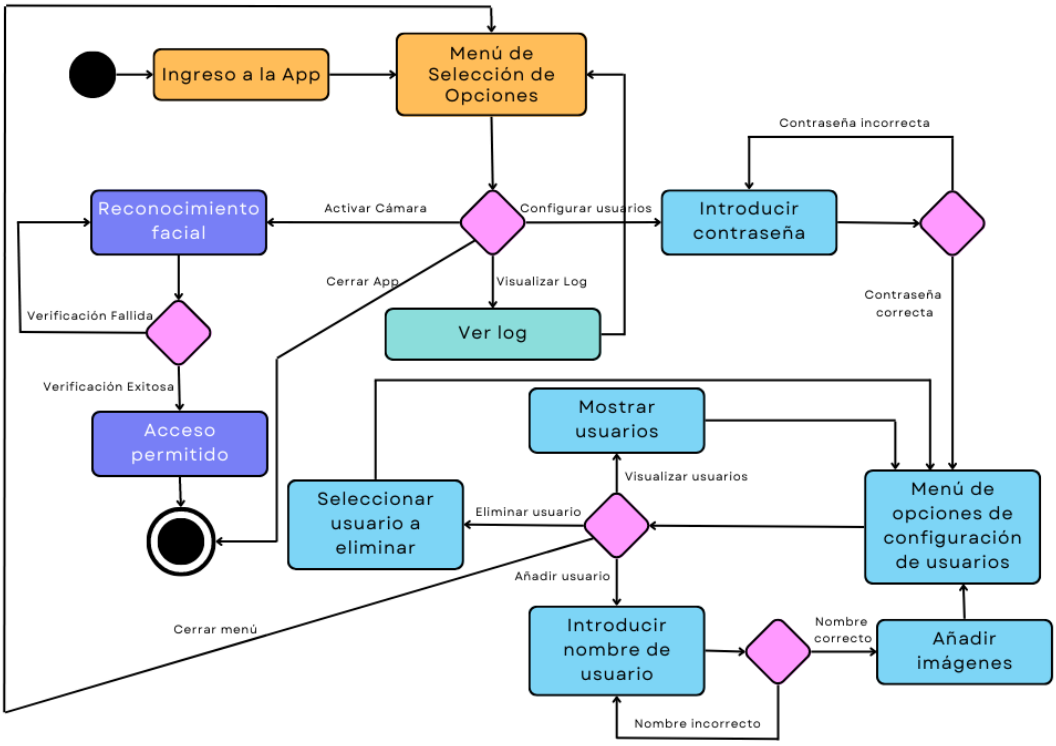


Figura 20: Diagrama de flujo UML

FUENTE: ELABORACIÓN PROPIA

Como se puede observar en el diagrama anterior, el usuario dispondrá de varias acciones a poder realizar durante la ejecución del programa. El sistema está diseñado para que solamente se pueda terminar la ejecución desde un mismo punto, haciendo que este sea más robusto y menos propenso a tener fallos o interrupciones innecesarias.

Además de ver la diferente estructura de las acciones que se pueden llevar a cabo dentro del sistema, es importante diseñar una buena relación entre las diferentes clases. Para ello, es importante realizar un diagrama de clases, los cuales ayudan a visualizar la estructura del programa desde un punto de vista más técnico.

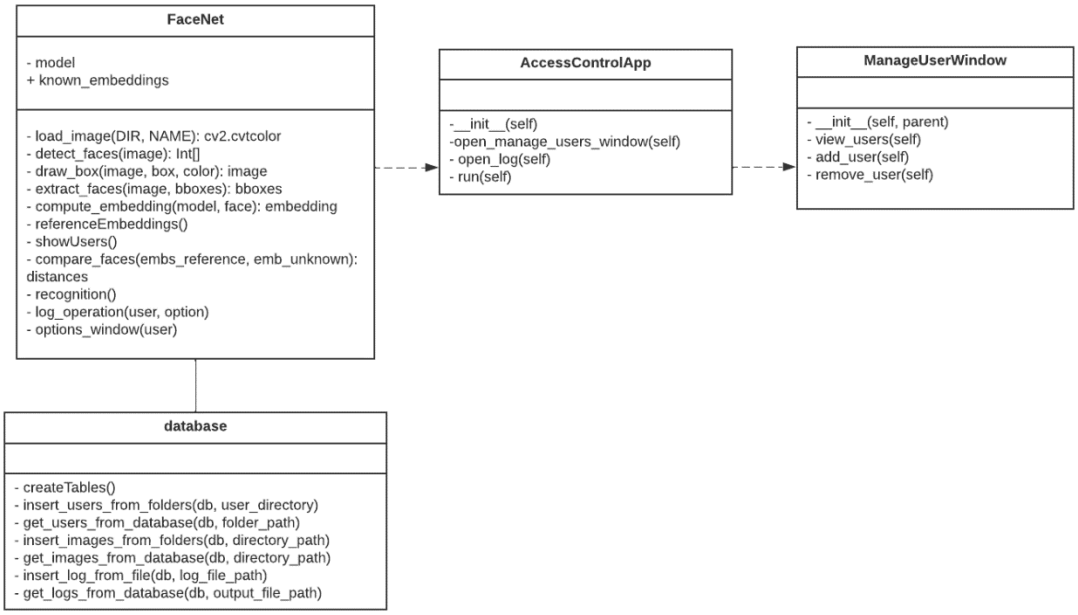


Figura 21: Diagrama de clases UML

FUENTE: ELABORACIÓN PROPIA

Como se puede observar en el diagrama anterior, el programa se divide en 4 clases grandes donde se dividen las tareas. Cada una de ellas se encargará de una función importante del programa, haciendo que, de esta forma la posible modificación del programa en un futuro sea bastante más fácil.

Cabe destacar que el programa entero se escribió en un documento cuaderno Jupyter con un formato .ipynb. En este formato no es nada común utilizar clases para dividir el proyecto ya que se trata de un formato modulable donde el código estará dividido por bloques que se ejecutan de forma independiente. Esto quiere decir, que tras tener el programa funcionando en el cuaderno de Jupyter, se tuve que migrar todo el código a un formato más tradicional y orientado a objetos en un archivo de Python con formato .py.

5.4. Desarrollo del Hardware Embebido

Este apartado se enfoca en el proceso de desarrollo del *hardware* embebido necesario para el sistema de control de acceso mediante el uso de reconocimiento facial. Para este sistema en específico, el *hardware* es una parte fundamental, por lo que en esta sección se detallarán los componentes utilizados, cómo estos fueron integrados y la configuración inicial de los mismos.

5.4.1. Selección de componentes

Para el desarrollo de este sistema se debieron tener varias cosas a tener en cuenta, y una de ellas fue escoger los componentes a utilizar para su desarrollo. Es importante considerar varios

factores, como el rendimiento, el coste, la compatibilidad con el *software* y la funcionalidad que se quiera implementar en el sistema para escogerlos de la manera más eficiente.

En el apartado [*Hardware* embebido] de este documento se realizó una comparación de posibles sistemas *hardware* embebidos que se podrían utilizar en este proyecto, y como bien se menciona en ese apartado, se utilizó el NVIDIA Jetson Nano como sistema *hardware* principal.

Es importante disponer de una cámara de buena calidad cuando se trata de un sistema que utiliza esta de forma indispensable para la obtención de datos y asegurar su correcto funcionamiento. Se tuvieron en cuenta factores como compatibilidad, tamaño, calidad de imagen y resolución. Teniendo estas cosas en cuenta, la utilización de una webcam era lo más apropiado debido a su gran compatibilidad con estos sistemas y a la gran calidad que ofrecen sin ocupar demasiado espacio.

5.4.2. Integración y ensamblaje del hardware

Primero, se introdujo el NVIDIA Jetson Nano en una caja metálica para su protección, ya que esta no deja de ser una placa de desarrollo que está al descubierto y que se debe proteger de posibles daños. También, es importante tener en cuenta que se trata de un componente que requiere de refrigeración, por lo que un chasis con una correcta ventilación sería lo más conveniente.

Una vez protegido el sistema embebido, este cuenta con muchos puertos, los cuales se usarán para conectar los diferentes dispositivos.

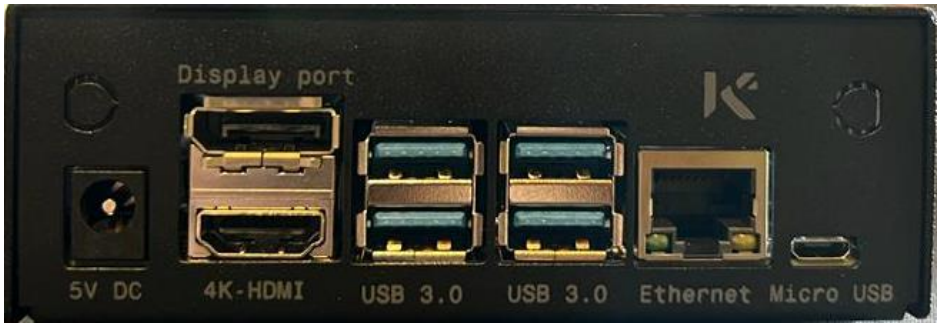


Figura 22: Puertos de NVIDIA Jetson Nano

FUENTE: ELABORACIÓN PROPIA

También tener en cuenta que, para obtener el máximo rendimiento del *hardware*, se dispone de un cable de alimentación de 5V, el cual debe ir conectado a la vez que un cable Micro USB para que este pueda funcionar de la forma óptima.

5.4.3. Configuración inicial y pruebas

Para la configuración inicial del sistema embebido, se tuvo que configurar el sistema operativo para que este estuviera en su última versión, para asegurarse de tener la máxima compatibilidad posible con las diferentes herramientas que se fueran a utilizar.

Lo más importante era poder crear un entorno donde poder ejecutar el programa escrito en Python mediante el entorno de trabajo de TensorFlow. Para ello, se investigaron varias formas

de instalar estos mediante programas de manejo de sistemas y entornos que fueran de distribución de código abierto.

Tener en cuenta que primeramente todo el programa se desarrolló en un sistema con Windows, y posteriormente se introdujo en el sistema embebido. Se hizo esto para asegurar la máxima comodidad a la hora de desarrollar el sistema.

En Windows, el programa más utilizado para ello es Conda [27], aunque por desgracia, esta solo está disponible para sistemas con una arquitectura x86-64, lo cual significa que no es compatible con el procesador ARM que se dispone en el NVIDIA Jetson Nano.

Se buscaron alternativas que fueran parecidas a esta y que fueran compatibles con el sistema embebido, aunque por desgracia, la poca información de la que se dispone puesto que todo el *software* libre es desarrollado mayoritariamente por la comunidad, y esta que no es demasiado grande por el momento. Estas alternativas incluyen Archiconda y Miniforge, las cuales resultaron en un fracaso por incompatibilidades entre distintas versiones de dependencias del sistema.

Por último, se instaló la versión de Python 3.7 en el sistema, y mediante ciertos cambios en las rutas de las variables de entorno del sistema operativo, se consiguió instalar una versión de TensorFlow que funcionara correctamente.

Por último, y tras mucho esfuerzo en la configuración por la falta de documentación, se instalaron las dependencias y librerías necesarias para asegurar el correcto funcionamiento de la cámara y las distintas partes de la aplicación.

Se realizaron pruebas de funcionamiento para asegurar que todo funcionara correctamente antes de proseguir con la implementación del sistema de control de acceso en el sistema embebido. Estas pruebas consistían en probar las diferentes funciones del sistema en trozos pequeños, asegurándose del correcto funcionamiento de todas las partes individualmente antes de juntarlo todo. Esto ayudó mucho a la hora de encontrar errores y encontrar posibles soluciones.

5.5. Almacenamiento en Base de Datos

El almacenamiento seguro de datos es una parte importante de cualquier sistema, y en el caso del sistema de control de acceso, es esencialmente importante debido a la naturaleza sensible de los datos involucrados. Una base de datos bien estructurada y diseñada garantiza un rendimiento óptimo, además de asegurar la seguridad del sistema y la integridad de los datos. A continuación, se detallan los pasos en el diseño, implementación e integración de la base de datos con el sistema.

5.5.1. Diseño de la base de datos

El diseño de la base de datos es un paso crucial ya que define cómo se almacenarán y gestionarán los datos. Se debe prestar atención especial a garantizar que el diseño respalde el rendimiento y la escalabilidad del sistema. Esta base de datos debe ser capaz de almacenar toda la información necesaria para el reconocimiento de los usuarios, la información de estos usuarios y los registros de acceso de la compañía.

27 https://docs.conda.io/en/latest/

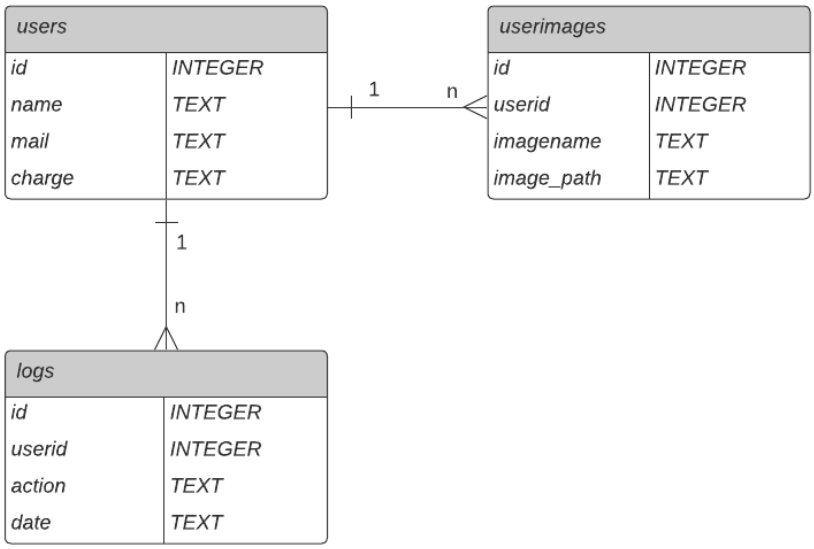


Figura 23: Diagrama Entidad-Relación de bases de datos

FUENTE: ELABORACIÓN PROPIA

En este diagrama podemos observar que la base de datos está dividida en 3 tablas: users, userimages y logs.

Users

Esta será la table encargada de almacenar todo lo relacionado con la información directa del usuario. Cada usuario tendrá un identificador asignado, además de un nombre, mail y el cargo del usuario en la compañía. Esta información sirve para identificar de forma más correcta a los diferentes usuarios y para tener alguna forma de distinguir a los diferentes usuarios dentro del sistema.

Userimages

Cada usuario tendrá varias imágenes asignadas a su identificador. Estas imágenes estarán almacenadas en esta tabla, la cual almacenará todas las rutas de las imágenes del sistema. Cada una de ellas estará asociado a un nombre y un identificador de usuario.

Se ha decidido almacenar las rutas de las imágenes debido al escaso rendimiento del que se dispone en el *hardware* embebido, esto hace que el rendimiento aumente. Las bases de datos están optimizadas para manejar texto y números, no archivos binarios como imágenes. Por este mismo motivo, al almacenar solo la ruta de la imagen se estará disminuyendo significativamente el tamaño de la base de datos.

Logs

Por último, tenemos la tabla encargada de almacenar todo lo relacionado a la actividad de los diferentes usuarios en el sistema. En ella, podemos encontrar el identificador que usaremos para diferenciar cada uno de los registros, además de que cada uno de ellos estará asociado a un identificador que estará asociado a un usuario del sistema. Además, también se guardarán las acciones que el usuario ha realizado y la fecha de la realización de esta.

5.5.2. Implementación de la base de datos

Para implementar la base de datos se ha utilizado SQLite [28] como sistema de gestión de bases de datos. Esta es una biblioteca desarrollada en el lenguaje C, y que proporciona una base de datos ligera. Esta elección se realizó debido a la potencia computacional limitada del *hardware* embebido a utilizar.

Hay varios motivos por los que se ha seleccionado esta librería, las cuales benefician considerablemente el proyecto. En primer lugar, el poco espacio que esta requiere debido a su pequeño tamaño y bajo consumo de recursos es de gran ayuda cuando optimizar los recursos es una prioridad, sobre todo, cuando se está trabajando con un sistema embebido.

Además, la simplicidad y facilidad de configuración de SQLite son notables. Al no requerir ningún proceso de servidor separado y se almacena toda la base de datos en un solo archivo, la administración y configuración de la base de datos se simplifica considerablemente.

5.5.3. Integración de la base de datos con el sistema

Integrar la base de datos con el sistema es un paso fundamental para asegurar que los datos de los usuarios, las imágenes, los registros y demás información se almacenen y gestionen eficazmente. La integración implica la comunicación entre la aplicación y la base de datos, lo que permite realizar operaciones como inserciones, actualizaciones, eliminaciones y consultas a las diferentes tablas que esta contiene.

Primero de todo, se tuvieron que crear las tablas en nuestra base de datos. Una vez tenemos las tablas, es importante tener ciertas configuraciones diseñadas para facilitar la integración con el programa. Algunas de estas integraciones consisten en funciones que automáticamente añaden, eliminan o modifican las tablas a nuestro placer mediante los archivos que localmente estén actualmente en el sistema. A su vez, debemos ser capaces de crear funciones que creen los archivos locales dependiendo del contenido de la base de datos.

```
#Database funcions
database.delete_table('access_control.db', 'userimages')
database.createTables()
database.insert_users_from_folders('access_control.db', 'knowns')
database.get_users_from_database('access_control.db', 'knowns')

database.insert_images_from_folders('access_control.db', 'database_test')
database.get_images_from_database('access_control.db', 'database_test1')

database.insert_log_from_file('access_control.db', 'log.csv')
database.get_logs_from_database('access_control.db', 'log.csv')
```

Figura 24: Funciones de base de datos

FUENTE: ELABORACIÓN PROPIA

Es importante que, durante este proceso, se manejen correctamente los errores y se garantice la integridad de los datos. Utilizar transacciones para agrupar operaciones que deben ser ejecutadas como una unidad puede ser muy útil para asegurar la consistencia de los datos.

28 https://www.sqlite.org/index.html

Por último, se debe asegurar el correcto funcionamiento de este mediante pruebas al sistema. Con esto, se debe comprobar que no se puede provocar ningún error en el sistema ya que esto sería nefasto para la empresa en la que esta estaría aplicada.

5.6. Implementación de Los Algoritmos de Reconocimiento Facial

En esta sección se va a profundizar en la implementación de los dos algoritmos de *Deep Learning* utilizados y analizados en este documento, explicando todos los pasos seguidos y el porqué de cada uno de ellos, argumentando la toma de decisiones aplicada en el proceso.

5.6.1. Implementación de la Red Siamesa

Lo primero fue implementar una función que sirviera para obtener las caras de una imagen. Hay muchas formas distintas de hacer esto, por lo que se optó por la opción de implementar un clasificador de rostros frontales (haarcascade_frontalface_default), el cual se puede obtener de forma gratuita desde el GitHub del creador, donde hay muchas opciones dependiendo del tipo de clasificador que necesites almacenados en archivos .xml.

```
# Load pre-trained face detection model
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
```

Figura 25: Implementación de Haarcascade para caras frontales

FUENTE: ELABORACIÓN PROPIA

Ahora es el momento de crear las carpetas necesarias para la implementación correcta del entrenamiento de este modelo:

- Imágenes ancla
- Imágenes positivas
- Imágenes negativas

Una vez tenemos las carpetas creadas, es hora de introducir imágenes en ellas. Para las imágenes negativas, se utilizó un dataset público que contiene más de 13.000 imágenes de personas diferentes (Labeled Faces in the Wild), convirtiéndolo en una forma eficaz de añadir muchas imágenes negativas para el entrenamiento posterior del sistema. Para las imágenes positivas y de ancla, se creó una pequeña función que habilita la cámara mediante la utilización de OpenCV, y que, al pulsar diferentes teclas, estas imágenes se guardaban en diferentes carpetas.

```
#Connection with webcam
cap = cv2.VideoCapture(0)
while cap.isOpened():
    ret, frame = cap.read()

    #adjust frame size to 250x250px and move it
    frame = frame[120:120+250, 200:200+250, :]

    #Get anchor images
    if (cv2.waitKey(1) & 0XFF == ord('a')):
        imgName = os.path.join(ANCHOR_PATH, '{}.jpg'.format(uuid.uuid1()))
        cv2.imwrite(imgName, frame)
```

Figura 26: Recolección de imágenes para la Red Siamesa

FUENTE: ELABORACIÓN PROPIA

Una vez que ya tenemos las imágenes necesarias en nuestras carpetas, es momento de aplicar el modelo anteriormente implementado a las distintas imágenes. Para ello, se creó una función que fuera iterando entre las diferentes imágenes y recortando solo las partes de las caras en un formato de 250x250 píxeles. Es importante aplicar este modelo a todas las imágenes, ya que de esta forma se va a reducir mucha información que aparece en las imágenes, y que pueden alterar el modelo para que este no se entrene de forma efectiva.

Ahora es el momento de crear conjuntos de datos para las imágenes positivas como para las negativas, para después poder crear las particiones de entrenamiento y tests. Para ello, primero se debe crear una función para preprocesar las imágenes. En esta función se está transformando una imagen en su equivalente de numpy, es decir, cambiamos el tamaño de la imagen a un 100x100 píxeles y después dividimos esta imagen por el valor 255, cambiando el valor de los píxeles tradicionales que van del 0 al 255 y estableciéndolos del 0 al 1.

```
#Convert 0-255 values into 0-1 values
def preprocess(file_path):
    byte_img = tf.io.read_file(file_path)    # Read an image
    img = tf.io.decode_jpeg(byte_img)        # Load the image
    img = tf.image.resize(img, (100, 100))   # Preprocess & resize image to be 100x100px
    img = img / 255.0                        # Scale image to be between 0-1
    return img
```

Figura 27: Función de preprocesado

FUENTE: ELABORACIÓN PROPIA

Ahora se crearán grupos de datos, tanto positivos como negativos. Mediante la función de preprocesado anteriormente nombrada, crearemos conjuntos de datos donde tendremos, una imagen de ancla y, o bien una imagen positiva o una imagen negativa. Esto nos servirá para cuando tengamos que entrenar nuestro modelo, de este modo, los conjuntos de datos ya estarán preparados para ello. Estos conjuntos también serán agrupados por grupos a los que llamaremos *batch*, en el caso de este proyecto, el número de estos será de 16.

```
#Training partition
print(round(len(data)*.7))
train_data = data.take(round(len(data)*.7)) #Take the 70% of the data
train_data = train_data.batch(16)
train_data = train_data.prefetch(8)
```

Figura 28: Conjuntos de datos para el entrenamiento

FUENTE: ELABORACIÓN PROPIA

Lo mismo se hará con los datos que utilizaremos para hacer los tests, aunque utilizaremos los conjuntos de imágenes que no hemos cogido para los datos que son para el entrenamiento.

```
#Testing partition
test_data = data.skip(round(len(data)*.7))        #Skip the first 70% of the data (used for training)
test_data = test_data.take(round(len(data)*.3))   #Take the last 30% of the data
test_data = test_data.batch(16)
test_data = test_data.prefetch(8)
```

Figura 29: Conjuntos de datos para los tests

FUENTE: ELABORACIÓN PROPIA

Ahora es el momento de diseñar los *embeddings* que utilizaremos en estas Redes Siamesas. Para ello, se ha utilizado la arquitectura descrita en el artículo científico (Koch, 2015b) y mostrada anteriormente en la explicación de las Redes Siamesas.

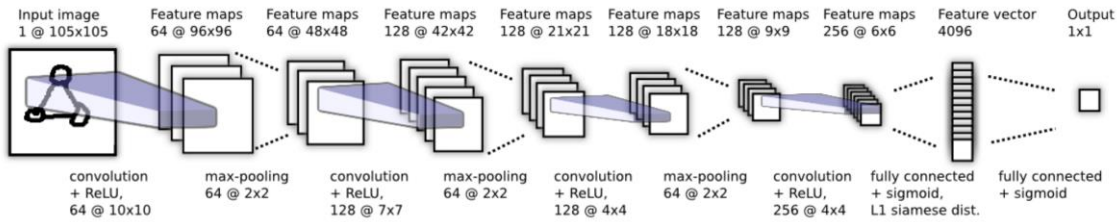


Figura 30: Arquitectura convolucional seleccionada

FUENTE: EXTERNA [29]

Para ello, realizaremos las convoluciones y *pooling* descritos en el artículo anteriormente mencionado. Para ello, se utilizarán librerías de TensorFlow que nos facilitarán el trabajo.

Se creará una función para crear estos *embeddings*. En ella, se utilizará un input de 100x100x3 datos. Para aplicar las convoluciones, se aplicarán utilizando las librerías de TensorFlow anteriormente mencionadas, Conv2D en este caso. De igual forma, para aplicar los *pooling*, se usarán las librerías MaxPooling2D.

Este proceso se repetirá 3 veces, aplicando siempre en el proceso los resultados de los bloques anteriores. Lo único que cambiará en estos bloques será el tamaño de datos con los que trabajaremos. Por último, aplicaremos una última convolución y los convertiremos en una sola dimensión mediante otras librerías de TensorFlow llamadas *Flatten* y *Dense*.

```
def make_embedding():
    input = Input(shape = (100, 100, 3))

    #block 1
    c1 = Conv2D(64, (10, 10), activation = 'relu')(input)
    m1 = MaxPooling2D(64, (2, 2), padding = 'same')(c1)
    #block 2
    c2 = Conv2D(128, (7, 7), activation = 'relu')(m1)
    m2 = MaxPooling2D(64, (2, 2), padding = 'same')(c2)
    #block 3
    c3 = Conv2D(128, (4, 4), activation = 'relu')(m2)
    m3 = MaxPooling2D(64, (2, 2), padding = 'same')(c3)
    #block 4
    c4 = Conv2D(256, (4, 4), activation = 'relu')(m3)
    f1 = Flatten()(c4)
    d1 = Dense(4096, activation = 'sigmoid')(f1)

    return Model(inputs=[input], outputs=[d1], name='embedding')
```

Figura 31: Embeddings en Red Siamesa

FUENTE: ELABORACIÓN PROPIA

29 Fuente de Figura 30: (Koch, 2015)

Ahora ha llegado el momento de crear la función que va a comparar estos *embeddings* en nuestro algoritmo de Redes Siamesas. Para ello, se definirá una función que obtendrá la distancia L1 Manhattan de estos *embeddings*, por lo que ya se podría obtener la distancia que separa a estos *embeddings*.

Ha llegado el momento donde ya tenemos todos los componentes necesarios para llevar a cabo la creación del modelo para una Red Siamesa. En este modelo se están añadiendo valores a estos *embeddings* que hemos creado anteriormente, tanto para las imágenes de input como para las imágenes de validación. Una vez tenemos los valores de estos, se obtiene la distancia entre ellos y se devuelve el modelo entero con todos los datos que este conlleva.

```
def make_siamese_model():
    input_image = Input(name = 'img_input', shape = (100, 100, 3))            #anchor image
    validation_image = Input(name = 'img_validation', shape = (100, 100, 3))   #Validation image

    #combine siamese distance components
    siamese_layer = L1Dist()
    siamese_layer._name = 'distance'
    distances = siamese_layer(embedding(input_image), embedding(validation_image))

    #Classification layer
    classifier = Dense(1, activation = 'sigmoid')(distances)

    return Model(inputs = [input_image, validation_image], outputs = classifier, name = 'SiameseNetwork')
```

Figura 32: Modelo de Red Siamesa

FUENTE: ELABORACIÓN PROPIA

Ahora, lo único que le falta al modelo para que sea funcional es su entrenamiento. Para ello, se va a crear la función de entrenamiento de cada *batch*. Como se ha mencionado anteriormente, cada *batch* contiene un conjunto de imágenes, por lo que se separarán los datos que pertenecen al ancla y los que pertenezcan a las imágenes positivas o negativas.

Ahora, utilizaremos esos nuevos datos para calcular las predicciones y las pérdidas. Para las predicciones, estaremos metiendo los datos de las imágenes a un modelo nuevo. Una vez tenemos todo esto, y con las pérdidas obtenidas, es hora de calcular y aplicar los gradientes para todos los datos que se puedan utilizar para entrenar dentro del modelo. Los gradientes se calculan para todos los diferentes pesos de los distintos modelos respecto a sus pérdidas. Esto se realiza para ajustar los pesos de las distintas neuronas durante el entrenamiento, optimizando de esta forma su eficacia y haciendo que el modelo sea más robusto.

```python
#Build Train Step Function
@tf.function #Compiling the function outside using graphs
def train_step(batch):
    with tf.GradientTape() as tape:
        #Get anchor and positive/negative image
        X = batch[:2] #Slicing the first 2 values
        #get label
        y = batch[2]

        #forward pass
        yhat = siamese_model(X, training = True)
        #calculate loss
        loss = binary_cross_loss(y, yhat)
    print(loss)

    #calculate gradients
    grad = tape.gradient(loss, siamese_model.trainable_variables)

    #calculate updated weights and apply to siamese model
    opt.apply_gradients(zip(grad, siamese_model.trainable_variables))


    return loss
```

Figura 33: Entrenamiento para cada batch

FUENTE: ELABORACIÓN PROPIA

Una vez tenemos el entrenamiento definido para cada *batch*, ahora es el momento de definir la función donde se va a iterar entre todos los *batch* que haya y entrenarlos mediante la función definida anteriormente. En esta función, se van a definir los *epochs*, es decir, iteraciones del entrenamiento. Este número será la cantidad de veces que el proceso se va a repetir hasta que el modelo se considere como entrenado.

```python
def train(data, EPOCHS):
    #Loop through epochs
    for epoch in range(1, EPOCHS+1):
        print('\n Epoch {}/{}'.format(epoch, EPOCHS))
        progbar = tf.keras.utils.Progbar(len(data))

        #Loop through each batch
        for idx, batch in enumerate(data):
            #Run train step
            train_step(batch)
            progbar.update(idx + 1)
```

Figura 34: Función de entrenamiento de las Redes Siamesas

FUENTE: ELABORACIÓN PROPIA

Una vez tenemos esto, solo faltaría entrenar el modelo de verdad. Por lo que, se definen la cantidad de *epochs* que se quieren, y se introduce en la función junto al conjunto de datos de entrenamiento que contiene los *batch* que van a ser entrenados.

Una vez tenemos el modelo entrenado, se definen ciertas métricas para saber la precisión y el *recall* de este modelo, que los usaremos para evaluar el modelo en cuestión. La precisión se refiere a la cantidad de verdaderos positivos divididos por la suma de los verdaderos positivos y falsos positivos, con esto se calcula la cantidad de instancias relevantes de entre todas las

detectadas. Por otro lado, el *recall* se refiere a la cantidad de verdaderos positivos divididos por la suma de verdaderos positivos y falsos negativos, dando un indicativo de la cantidad de valores relevantes han sido detectados.

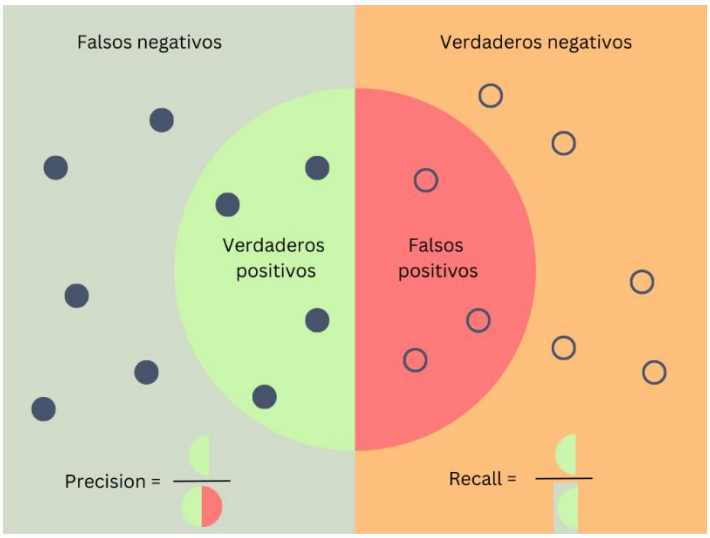


Figura 35: Precisión y Recall

FUENTE: ELABORACIÓN PROPIA

Ahora es el momento de guardar el modelo entrenado como un archivo .h5 en el ordenador. Esto nos sirve para no tener que andar entrenando el modelo cada vez que lo queramos utilizar, y cargarlo en el programa deseado mediante la función de cargado del modelo. Esto facilita mucho el uso de este modelo ya que, con ello, no haría falta tener la parte del código previa al entrenamiento en el código donde se quiera implementar, haciendo que este esté más limpio.

```
#SAVE MODEL
siamese_model.save('siamesemodel.h5')



#reload model
model = tf.keras.models.load_model('siamesemodel.h5', custom_objects = {'L1Dist':L1Dist, 'BinaryCrossentropy':tf.losses.BinaryCrossentropy})
```

Figura 36: Funciones para guardar y cargar del modelo

FUENTE: ELABORACIÓN PROPIA

Para poder verificar si una persona es reconocida o no, se creará una función que analice una imagen y la compare con el resto de las imágenes de las personas que están dentro del sistema.

Primero vamos a iterar por todas las imágenes de personas reconocidas en el sistema que tengamos. Ahora aplicaremos la función preprocess que hemos mencionado con anterioridad a ambas imágenes para obtener sus datos y modificarlos para ajustarlos a las características del modelo.

Después, se hará una predicción sobre las imágenes, lo que nos devolverá una distancia. Esta distancia se comparará con un valor predefinido que indicará la distancia máxima necesaria para poder considerar dos imágenes iguales, es decir, validar la identidad de una persona.

Por último, dividiremos este valor por la cantidad de imágenes positivas entre la cantidad total de imágenes positivas, para obtener una proporción de acierto. Otra vez, se compara este valor con otro valor predefinido, que si resulta ser menor que nuestro valor significará que la persona en cuestión ha sido reconocida.

```python
#Verification function
def verify(model, detection_threshold, verification_threshold):

    for dir in os.listdir(os.path.join('application_data', 'verification_images')):
        user = 'Uknown'
        results = []

        for image in os.listdir(os.path.join('application_data', 'verification_images', dir)):
            input_img = preprocess(os.path.join('application_data', 'input_image', 'input_image.jpg'))
            validation_img = preprocess(os.path.join('application_data', 'verification_images', dir, image))

            resultant = model.predict(list(np.expand_dims([input_img, validation_img], axis = 1)))
            results.append(resultant)

        detection = np.sum(np.array(results) > detection_threshold) #detection_threshold: Metric above a prediction is considered positive

        verification = detection / len(os.listdir(os.path.join('application_data', 'verification_images', dir)))
        verified = verification > verification_threshold #verification_threshold: Proportion of positive predictions / total positive samples
        if (verified == 1):
            user = dir
            return results, verified, user
    #print(results)
    return results, verified, user
```

Figura 37: Función de verificación de las redes Siamesas

FUENTE: ELABORACIÓN PROPIA

5.6.2. Implementación de FaceNet

Primero de todo, necesitamos crear una función que sirva para la lectura de las imágenes. Esto nos permitirá trabajar con ellas y modificarlas a nuestro gusto. Para ello, utilizaremos la librería OpenCV, que está diseñada para el procesamiento de imágenes.

```python
#Load image
def load_image(DIR, NAME):
    return cv2.cvtColor(cv2.imread(f'{DIR}/{NAME}'), cv2.COLOR_BGR2RGB)
```

Figura 38: OpenCV para leer imágenes

FUENTE: ELABORACIÓN PROPIA

También se implementó una función que permitirá dibujar un rectángulo encima de la imagen. Esta función no es algo indispensable para el desarrollo del algoritmo, aunque resulta muy visual realizar la comprobación de si alguna cara ha sido detectada al ver un rectángulo sobre las imágenes. En esta, se verifica si hay alguna cara en la imagen, almacenada en la variable box, y si esta contiene alguna cara, se dibujará un rectángulo en las coordenadas específicas de la imagen.

```python
def draw_box(image, box, color, line_width=6):
    if (box == []):
        return image
    else:
        cv2.rectangle(image, (box[0], box[2]), (box[1], box[3]), color, line_width)
    return image
```

Figura 39: Dibujar rectángulo sobre las caras

FUENTE: ELABORACIÓN PROPIA

Ahora, es el momento de implementar el modelo de detección de caras mediante MobileNet, una Red Convolucional especializada en la detección de objetos. En este caso, se va a implementar un modelo entrenado para la detección de caras humanas.

Este modelo pre-entrenado [frozen_inference_graph_face.pb], tiene un formato de grafo de TensorFlow por lo que para leer el contenido de este hará falta utilizar funciones específicas de TensorFlow.

Una vez lo tenemos importado, implementaremos la función que servirá para la detección de las caras en una imagen. Esta función tendrá como entradas una imagen y un umbral. Este umbral nos servirá para determinar si las distintas partes detectadas por el modelo son consideradas o no. MobileNet asigna valores de entre 0 y 1 de cada región de interés de la cara, lo cual nos indicará la probabilidad de que la región en concreto pertenezca a una cara humana.

```
def detect_faces(image, score_threshold=0.7):
    global boxes, scores
    (imh, imw) = image.shape[:-1]
    img = np.expand_dims(image, axis=0)

    #Initialize mobilenet
    sess = tf.compat.v1.Session(graph=mobilenet)
    image_tensor = mobilenet.get_tensor_by_name('image_tensor:0')
    boxes = mobilenet.get_tensor_by_name('detection_boxes:0')
    scores = mobilenet.get_tensor_by_name('detection_scores:0')

    #Prediction (detection)
    (boxes, scores) = sess.run([boxes, scores], feed_dict={image_tensor:img})

    #Readjust sizes of boxes, scores
    boxes = np.squeeze(boxes, axis=0)
    scores = np.squeeze(scores, axis=0)

    #Purge bounding boxes
    idx = np.where(scores >= score_threshold)[0]

    #Create bounding boxes
    bboxes = []
    for index in idx:
        ymin, xmin, ymax, xmax = boxes[index,:]
        (left, right, top, bottom) = (xmin*imw, xmax*imw, ymin*imh, ymax*imh)
        left, right, top, bottom = int(left), int(right), int(top), int(bottom)
        bboxes.append([left, right, top, bottom])


    return bboxes
```

Figura 40: Función de detección de caras

FUENTE: ELABORACIÓN PROPIA

En la función anterior, la variable "boxes" hace referencia a las regiones de la cara. La variable "scores" indica el valor asignado y la variable "bboxes" contiene las coordenadas de las regiones almacenadas.

Ahora que ya tenemos la detección de caras implementada, es hora de pasar a la implementación del reconocimiento facial mediante el algoritmo FaceNet. Primero de todo,

debemos descargar el modelo [FaceNet] e implementarlo en nuestro código. Para ello, se usará una función de TensorFlow.Keras llamado load model.

```
#FACENET
facenet = load_model('facenet_keras.h5')
```

Figura 41: Implementación de modelo FaceNet pre-entrenado

FUENTE: ELABORACIÓN PROPIA

Este modelo funciona con imágenes de entrada de 160x160 píxeles, en formato RGB. Por este mismo motivo, antes de aplicar las imágenes en cuestión al modelo, se va a crear una función intermedia que extraiga la porción de la imagen de la cara detectada y se redimensione al tamaño requerido.

```
#Extract faces
def extract_faces(image, bboxes, new_size=(160, 160)):
    cropped_faces = []
    for box in bboxes:
        left, right, top, bottom = box
        face = image[top:bottom, left:right]
        cropped_faces.append(cv2.resize(face, dsize=new_size))
    return cropped_faces
```

Figura 42: Función para ajustar imágenes para el modelo FaceNet

FUENTE: ELABORACIÓN PROPIA

Con todo lo anterior, ya sería suficiente para poder aplicar el modelo y crear el *embedding* necesario de la cara. Para ello, se creará una función, cuyo objetivo será crear un *embedding* de una cara pasándole una imagen y el modelo que se ha implementado. Será necesaria la normalización de la imagen restándole a cada píxel la media de todas ellas y dividiéndolo por la desviación estándar, el cual es un procedimiento estándar a la hora de usar Redes Neuronales:

$$face = \frac{(face - mean)}{std}$$

Donde face es el valor de cada píxel, mean es la media de los valores de todos los píxeles y std es la desviación estándar.

```python
def compute_embedding(model, face):
    face = face.astype('float32')

    mean, std = face.mean(), face.std()
    face = (face - mean) / std

    face = np.expand_dims(face, axis=0)

    embedding = model.predict(face)
    return embedding
```

Figura 43: Creación de embeddings en FaceNet

FUENTE: ELABORACIÓN PROPIA

Una vez tenemos los *embeddings* creados, es hora de compararlos. Para ello, primero debemos aplicar la función anterior a todas las imágenes que queremos reconocer en nuestro sistema y almacenar sus respectivos *embeddings* en una variable, para poder acceder a ellos más adelante. Una vez procesadas todas las imágenes de los usuarios reconocidos y calculados sus *embeddings*, es el momento de obtener una imagen extra, la cual servirá como comparación con todas las demás imágenes y se determinará si este usuario en concreto está siendo identificado o no mediante una comparación de las distancias de sus *embeddings*.

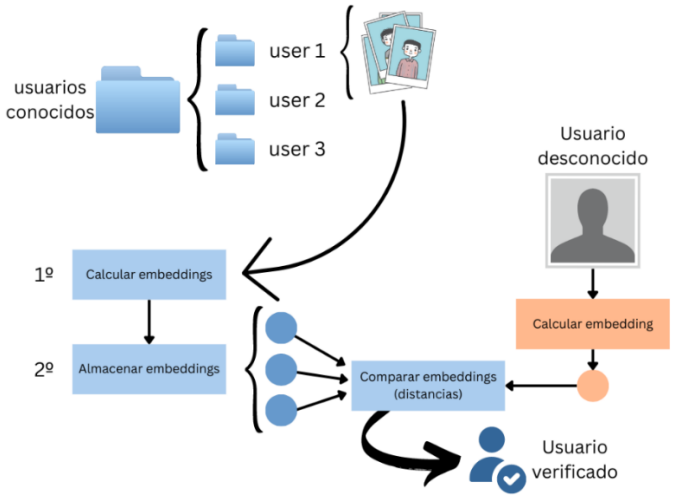


Figura 44: Estructura de la comparación de embeddings

FUENTE: ELABORACIÓN PROPIA

Una vez tengamos todos los datos listos, ahora es posible comparar si estas imágenes son similares a nuevas entradas de imágenes en nuestro sistema. Esta nueva imagen pasará por los mismos procesos de procesamiento que las imágenes que tenemos en el sistema, es decir, la forma de calcular su *embedding* es exactamente igual al resto.

```
image = cv2.cvtColor(cv2.imread(f'{DIR_UNKNOWNS}/{name}'), cv2.COLOR_BGR2RGB)
bboxes = FaceNet.detect_faces(image)
faces = FaceNet.extract_faces(image, bboxes)

#Calculate embeddings for each face
img_with_boxes = image.copy()

for face, box in zip(faces, bboxes):
    for user in FaceNet.known_embeddings:
        #print(f'Comparing with {user}')
        emb = FaceNet.compute_embedding(FaceNet.facenet, face)
```

Figura 45: Proceso de creación de embedding

FUENTE: ELABORACIÓN PROPIA

Una vez tenemos su *embedding* calculado, se realizará una comparación de estas. Este *embedding* será uno a uno comparado con todos los anteriormente guardados en el sistema, y si la distancia entre ellas es menor a un umbral establecido, la nueva imagen se considerará identificada por el sistema.

```
for user in FaceNet.known_embeddings:
    #print(f'Comparing with {user}')
    emb = FaceNet.compute_embedding(FaceNet.facenet, face)

    _, recognition = FaceNet.compare_faces(FaceNet.known_embeddings[user], emb)

    if any(recognition):
        print(f'    match, is {user}')
```

Figura 46: Comparación de embeddings FaceNet

FUENTE: ELABORACIÓN PROPIA

6. Resultados y validación

La etapa de resultados y validación es crucial para evaluar la eficacia y precisión del sistema de control de acceso desarrollado. En esta sección, se detallan las pruebas realizadas, así como un análisis exhaustivo de los resultados obtenidos.

6.1. Pruebas y validación del sistema

Para garantizar que el sistema funcione de manera efectiva y precisa, se realizaron varias pruebas y validaciones. Estas pruebas abarcan diversos aspectos del sistema, desde el rendimiento y precisión del reconocimiento facial hasta la confiabilidad de los datos almacenados.

1. **Pruebas de Rendimiento del Reconocimiento Facial:** Se realizaron pruebas para evaluar la velocidad y precisión del reconocimiento facial. Esto incluyó pruebas con diferentes condiciones de luz, variaciones en los ángulos de la cara, ocultamiento de ciertas partes de la cara y demás. También, se sometió al sistema a diferentes pruebas para determinar cómo de bien funciona ante situaciones críticas, tales como pruebas de reconocimiento para usuarios con pocas imágenes en el sistema.
2. **Pruebas de Interfaz de Usuario:** Se llevó a cabo la evaluación de la interfaz de usuario para garantizar que sea intuitiva y fácil de usar mediante pruebas de usabilidad.

6.2. Análisis de los resultados

Después de realizar las pruebas y validaciones, es importante analizar los resultados para determinar si el sistema cumple con los requisitos y expectativas establecidas.

6.2.1. Análisis de Rendimiento del Reconocimiento Facial

Tras la reciente pandemia provocada por la COVID-19, se estableció la obligación del uso de mascarillas. Esta puede ser una buena oportunidad para poner a prueba el sistema, ocultando de esta forma parte de la cara y verificando si sigue funcionando o no.

Figura 47: Imagen utilizada para la prueba con mascarilla

FUENTE: ELABORACIÓN PROPIA

Esta prueba fue exitosa, el sistema es capaz de reconocer a los distintos usuarios aun teniendo parte de las caras parcialmente ocultas.

Ahora se quiere verificar el funcionamiento del sistema mediante el uso de cambios de la iluminación y aplicando rotaciones en la cabeza del usuario a identificar.

Figura 48: Imágenes utilizadas para la prueba de rotaciones de cabeza y cambios de iluminación

FUENTE: ELABORACIÓN PROPIA

Las imágenes mostradas son algunas que se han utilizado para realizar esta prueba. El sistema reacciona bien a estos cambios en las imágenes de entrada y consigue realizar un buen y consistente reconocimiento facial.

Por último, ha querido probar la cantidad de imágenes registradas necesaria para el reconocimiento de una persona nueva. Para ello, se ha creado un usuario nuevo con una sola imagen en el sistema, y se ha probado a su identificación.

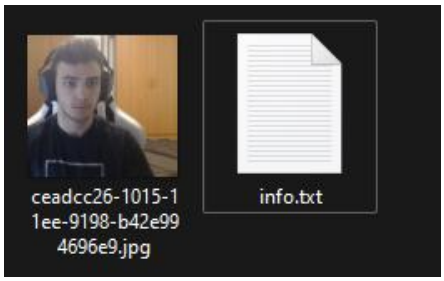


Figura 49: Carpeta contenedora del usuario creado para la prueba

FUENTE: ELABORACIÓN PROPIA

Tras probar el reconocimiento facial, se saca la conclusión de que una imagen es suficiente para la identificación de un usuario en el sistema. Está demostrado que cuantas más imágenes tengas, más sólida va a ser la identificación que realizará el sistema puesto que dispondrá que muchos más *embeddings* con los que comparar la imagen, por lo que la precisión será mucho mayor, aunque en esta prueba se quería obtener el número mínimo de imágenes necesaria para poder ser identificado por el sistema.

Ahora es el momento de poner a prueba los dos sistemas de reconocimiento facial. Se va a realizar una serie de experimentos con números de imágenes progresivas almacenadas, con esto se quiere observar cómo cada algoritmo se comporta con cada número de imágenes almacenadas. Para cada número de imágenes se realizarán 15 pruebas positivas and 15 negativas por cada tipo de imagen de entrada al sistema, es decir, se simulará que el usuario que quiere identificarse está tomando diferentes situaciones ante la cámara. Los tipos de imágenes serán clasificados como: Normal de frente, rotaciones de cabeza, cambios de luz y con mascarilla. Los resultados de las pruebas estarán en la siguiente tabla:

Tabla 9: Prueba FaceNet con Diferentes Números de Imágenes y Tipos de Entrada

	1 imagen	3 imágenes	5 imágenes	10 imágenes	20 imágenes	30 imágenes	100 imágenes
Normal de Frente							
Positivos Verdaderos	13	14	15	15	15	15	15
Falso Negativo	2	1	0	0	0	0	0
Negativos Verdaderos	15	15	15	15	15	15	15
Falso Positivo	0	0	0	0	0	0	0
Rotación de cabeza							
Positivos Verdaderos	7	8	10	13	14	15	15
Falso Negativo	8	7	5	2	1	0	0
Negativos Verdaderos	15	15	15	15	15	15	15
Falso Positivo	0	0	0	0	0	0	0
Cambios de luz							
Positivos Verdaderos	9	10	13	15	15	15	15
Falso Negativo	6	5	2	0	0	0	0
Negativos Verdaderos	15	15	15	15	15	15	15
Falso Positivo	0	0	0	0	0	0	0
Con mascarilla							
Positivos Verdaderos	3	6	8	12	13	14	15
Falso Negativo	12	9	7	3	2	1	0
Negativos Verdaderos	15	15	15	15	15	15	15
Falso Positivo	0	0	0	0	0	0	0

Ahora es el momento de calcular los valores de Precisión y *Recall* mediante estos datos:

Tabla 10: Cálculo FaceNet de Precisión y Recall para los Diferentes Casos

	1 imagen	3 imágenes	5 imágenes	10 imágenes	20 imágenes	30 imágenes	100 imágenes
Normal de Frente							
Precisión	1	1	1	1	1	1	1
Recall	0.87	0.93	1	1	1	1	1
Rotación de cabeza							
Precisión	1	1	1	1	1	1	1
Recall	0.46	0.53	0.67	0.87	0.93	1	1
Cambios de luz							
Precisión	1	1	1	1	1	1	1
Recall	0.6	0.67	0.87	1	1	1	1
Con mascarilla							
Precisión	1	1	1	1	1	1	1
Recall	0.2	0.4	0.53	0.8	0.87	0.93	1

Como se puede observar en la tabla con los valores asociados al algoritmo FaceNet, la precisión en todos los casos es de 1. Esto significa que el algoritmo descarta las imágenes y prioriza en no dar falsos positivos, por lo que todas las imágenes obtenidas serán relevantes. Esto, en un sistema de control de acceso como este, es algo fundamental ya que de esta forma evitaremos una posible intrusión de alguna persona externa.

Ahora, se creará un gráfico para mostrar el *Recall* para cada uno de los casos, ya que. como hemos analizado antes, todas las imágenes obtenidas son relevantes. Con el *Recall* estaremos calculando la cantidad de imágenes relevantes que están siendo obtenidas por el sistema.

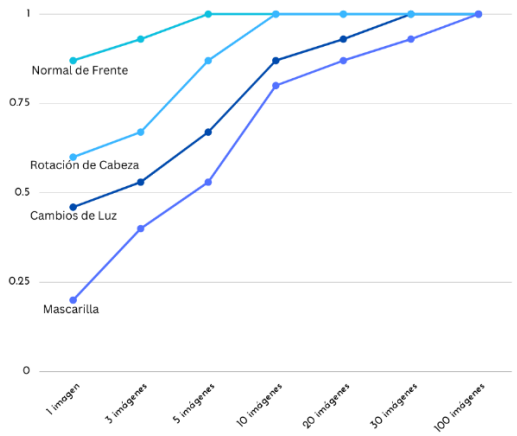


Figura 50: Gráfico de Recall para Cada Tipo de Imagen en FaceNet

FUENTE: ELABORACIÓN PROPIA

Al observar el gráfico, se observa una tendencia ascendente de *Recall* en todos los casos hasta llegar a un cierto punto donde todos ellos alcanzan el valor 1. Esto significa que la cantidad de imágenes relevantes que el sistema va a obtener será mayor si disponemos de más imágenes de muestra de ese mismo usuario almacenados.

Se puede observar cómo a partir de las 10 imágenes de muestra para cada usuario, los valores no descienden del 0.8 en ninguno de los casos. Con estos valores el sistema ya sería lo bastante preciso para poder funcionar con precisión y eficacia, aunque añadiendo más imágenes se obtendría un sistema más robusto.

Ahora se va a repetir el mismo proceso para el algoritmo de las redes Siamesas.

Tabla 11: Prueba Redes Siamesas con Diferentes Números de Imágenes y Tipos de Entrada

	1 imagen	3 imágenes	5 imágenes	10 imágenes	20 imágenes	30 imágenes	100 imágenes
Normal de Frente							
Positivos Verdaderos	3	5	6	8	11	12	12
Falso Negativo	14	10	9	7	4	3	3
Negativos Verdaderos	7	9	9	11	11	12	12
Falso Positivo	8	6	6	4	4	3	3
Rotación de cabeza							
Positivos Verdaderos	2	3	3	4	5	6	6
Falso Negativo	13	12	12	11	10	9	9
Negativos Verdaderos	10	10	11	11	12	12	13
Falso Positivo	5	5	4	4	3	3	2
Cambios de luz							
Positivos Verdaderos	3	4	5	6	8	11	11
Falso Negativo	14	11	10	9	7	4	4
Negativos Verdaderos	8	8	9	10	10	11	11
Falso Positivo	7	7	6	5	5	4	4
Con mascarilla							

Positivos Verdaderos	0	0	0	0	0	0	0
Falso Negativo	15	15	15	15	15	15	15
Negativos Verdaderos	15	15	15	15	15	15	15
Falso Positivo	0	0	0	0	0	0	0

Ahora es el momento de calcular los valores de Precisión y *Recall* mediante estos datos:

Tabla 12: Cálculo Redes Siamesas de Precisión y Recall para los Diferentes Casos

	1 imagen	3 imágenes	5 imágenes	10 imágenes	20 imágenes	30 imágenes	100 imágenes
Normal de Frente							
Precisión	0.27	0.45	0.5	0.67	0.73	0.8	0.8
Recall	0.2	0.33	0.4	0.53	0.73	0.8	0.8
Rotación de cabeza							
Precisión	0.29	0.38	0.5	0.5	0.63	0.67	0.75
Recall	0.13	0.2	0.2	0.27	0.33	0.4	0.4
Cambios de luz							
Precisión	0.3	0.36	0.45	0.54	0.61	0.73	0.73
Recall	0.2	0.27	0.33	0.4	0.53	0.73	0.73
Con mascarilla							
Precisión	0	0	0	0	0	0	0
Recall	0	0	0	0	0	0	0

Como se puede observar en la tabla con los valores asociados al algoritmo de las redes Siamesas, a diferencia del algoritmo FaceNet, hacen falta muchas más muestras para poder llegar a obtener un reconocimiento facial mínimamente estable, aunque en este caso, nunca llegaremos a obtener un reconocimiento facial tan sólido como en el FaceNet.

Ahora se realizará un gráfico mostrando tanto la precisión como el *Recall* de las redes Siamesas para analizar las tendencias que esta tiene:

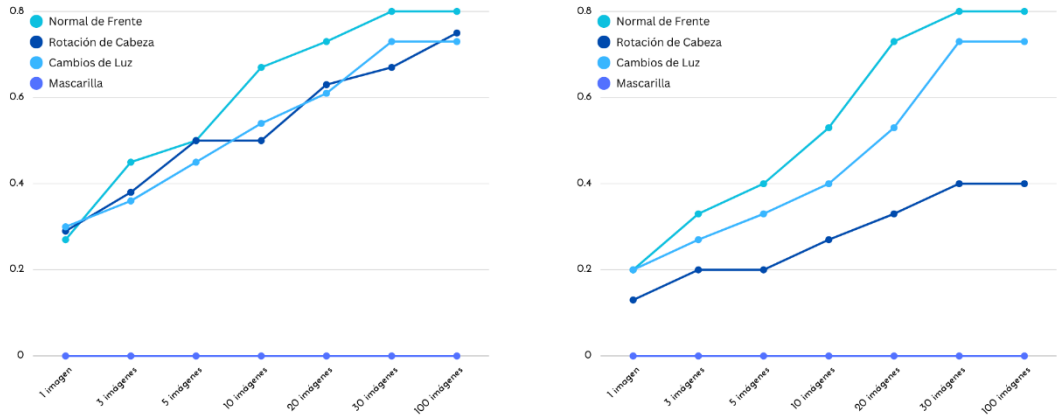


Figura 51: Gráfico precisión y Recall de las Redes Siamesas

FUENTE: ELABORACIÓN PROPIA

Como se puede observar en los gráficos anteriores, donde el de la izquierda representa el gráfico asociado a los valores de la precisión y el de la derecha al de *Recall* del algoritmo de la red Siamesa. En ambos se puede observar una tendencia ascendente también, aunque en el caso de la mascarilla, este algoritmo no funciona correctamente cuando las distintas partes de la cara están ocultas.

En ambos casos, los valores máximos alcanzados son de 0.8, lo cual está bien para un sistema de reconocimiento facial. La parte negativa de este algoritmo es que hay falsos positivos detectados por el sistema, lo cual queremos evitar ante todo en un sistema de control de accesos.

Hay que mencionar que las pruebas se han realizado hasta las 100 imágenes, aunque no hay un límite de imágenes para cada usuario. Esto dependerá de la memoria disponible y los requerimientos de la empresa en cuestión.

Por último, cabe mencionar que el reconocimiento, en todos los casos, ha requerido menos de un segundo para llevarse a cabo. No hay variaciones de tiempo cuando se está intentando poner al límite el sistema, por lo que se puede corroborar que el funcionamiento del reconocimiento facial es sólido y eficiente.

6.2.2. Análisis de la Experiencia de Usuario

En esta prueba, se quiso evaluar la experiencia de usuario de nuestro sistema. Se llevó a cabo una sesión de prueba con un grupo de personas que no estuvieron involucradas en el desarrollo de la aplicación. Esto fue clave para obtener percepciones imparciales sobre la usabilidad y funcionamiento del sistema. Se pidió a los participantes que interactuaran con la interfaz y que realizaran ciertas tareas específicas para ver como estos se comportaban.

Una vez completaron las tareas, se solicitó a los participantes que proporcionaran sus comentarios y sugerencias sobre sus experiencias. Algunos de los comentarios:

Participante 1: "En general, la aplicación fue fácil de usar, pero me tomó un tiempo darme cuenta de cómo guardar mi actividad."

Participante 2: "Me gustó la rapidez con la que la cámara me reconoció, aunque no me gustó demasiado la espera al registrarme en el sistema."

Participante 3: "El diseño es limpio y moderno, pero creo que un tutorial o una guía de inicio rápido sería útil para los nuevos usuarios."

A partir de estos comentarios, se realizó un análisis para identificar áreas de mejora y posibles cambios a implementar:

- Mejora de la Navegación:
  Se podría considerar rediseñar algunos iconos o botones con el fin de proporcionar una estructura de navegación más limpia o que sea más intuitiva.

- Inclusión de Materiales de Ayuda:
  Se podría realizar la creación de un tutorial o una guía para los nuevos usuarios del sistema. De esta forma, se podría evitar una formación necesaria para los usuarios que vayan a utilizar este sistema.

- Adición de más Elementos Visuales:
  Además de los elementos visuales ya existentes, la adición de más de ellos podría ser clave para mantener al usuario informado en todo momento con lo que está pasando. Por ejemplo, cuando un nuevo usuario es añadido y se tienen que obtener los *embeddings* de todas las imágenes del sistema otra vez, la espera puede resultar confusa puesto que en la interfaz no se muestra ningún elemento visual que diga esta información.

7. Conclusiones y trabajos futuros

7.1. Conclusiones

El principal objetivo del proyecto propuesto es desarrollar un sistema de control de accesos mediante el uso de inteligencia artificial eficiente e integrarlo en un sistema embebido de recursos limitados. A través de un análisis integral, una cuidadosa selección de tecnología y un riguroso proceso de prueba y desarrollo, se han logrado cumplir todos los objetivos propuestos en este documento en el apartado [Fases de la Metodología].

Una de las primeras tareas fue analizar el estado actual real reconocimiento facial, lo que permitió entender las tecnologías y algoritmos existentes y elegir los más adecuados para el proyecto. Al comparar soluciones que difieren en robustez y requisitos de *hardware*, fue posible tomar una decisión fundamentada sobre qué tecnología implementar.

El desarrollo de la aplicación para un sistema embebido presentó desafíos únicos, particularmente debido a las limitaciones de recursos y a la falta de información en la web. Sin embargo, mediante la selección de herramientas apropiadas, se consiguió un sistema eficiente capaz de realizar el reconocimiento facial de varias personas de manera precisa.

Además, se ha desarrollado un sistema de gestión de usuarios y control de ellos, lo cual es importante para la escalabilidad del sistema en entornos del mundo real. El almacenamiento de bases de datos fue otra área crítica que se abordó con éxito.

Cabe señalar que una de las dificultades encontradas durante el desarrollo del proyecto fue la adquisición del *hardware* necesario. Hubo una escasez de *stock* en el mercado, lo que dificultó enormemente la obtención de los componentes necesarios para el sistema. Además, la entrega y el proceso de los pedidos fueron excepcionalmente lentos, lo que contribuyó a un retraso en la fase de desarrollo y prueba. Esta es una gran limitación a tener en cuenta al planificar proyectos similares, especialmente en un contexto donde la cadena de suministro puede ser frágil.

Como nota personal, he de mencionar que a lo largo de este proyecto he adquirido una gran cantidad de conocimientos y habilidades. Analizar diferentes algoritmos y tecnologías para la identificación facial fue una experiencia de aprendizaje enriquecedora. No solo proporcionó una comprensión profunda sobre sus fundamentos teóricos, sino que también ofreció una visión práctica de cómo implementar estos algoritmos de manera efectiva.

Además, trabajar en un sistema embebido, con el cual no estaba familiarizado al comienzo del proyecto, fue especialmente valioso. Aprendí sobre las limitaciones y desafíos que un desarrollo de estas características conlleva y cómo superarlos apropiadamente.

7.2. Trabajos futuros

7.2.1. Limitaciones y posibles mejoras

A pesar de los logros del proyecto, hay varias limitaciones y áreas que podrían beneficiarse de mejoras futuras:

- **Optimización del rendimiento:** Aunque el sistema funciona eficientemente en un entorno embebido, siempre hay espacio para mejoras en términos de velocidad y precisión. Esto podría lograrse mediante la implementación de algoritmos más avanzados o mediante la optimización de los ya existentes.

- **Escalabilidad:** El sistema ha sido diseñado para funcionar con recursos limitados, lo que puede representar un desafío cuando se necesite escalar el sistema para manejar más usuarios. Investigar y aplicar técnicas de escalabilidad sería beneficioso.
- **Seguridad:** Si bien se ha implementado el almacenamiento seguro de datos, la seguridad es un área en constante evolución. Mantenerse actualizado y mejorar la seguridad a medida que surgen nuevas amenazas y técnicas es esencial.
- **Interfaz de usuario más avanzada:** El sistema de gestión de usuarios podría beneficiarse de una interfaz más intuitiva y rica en características, lo que permitiría a los administradores gestionar de manera más eficiente los usuarios y datos del sistema.
- **Integración con otras tecnologías:** integrar el sistema de reconocimiento facial con otras tecnologías, como sistemas de control de acceso o bases de datos en la nube, podría extender su utilidad y aplicabilidad en diferentes escenarios.
- **Gestión de la cadena de suministros:** Una posible área de mejora para proyectos futuros sería desarrollar estrategias más sólidas para la gestión: de la cadena de suministro, especialmente en lo que respecta a la obtención de *hardware*. Esto podría incluir la identificación de múltiples proveedores, la realización de pedidos con suficiente antelación, y tener alguna reserva en caso de retrasos o escasez de stock.

Esta experiencia ha subrayado la importancia de ser adaptable y estar dispuesto a aprender en un campo en constante evolución como el reconocimiento facial y los sistemas embebidos. El conocimiento y las habilidades adquiridas durante este proyecto serán invaluables en futuras iniciativas y desarrollos.

8. Bibliografía

2022 Gartner® Hype CycleTM for Digital Identity. (2022, August 22). Callsign. https://www.callsign.com/knowledge-insights/gartner-hype-cycle-for-digital-identity

A model based on the Technology Readiness Level (TRL) scale to measure the maturity level of research projects that can become spinoffs in Higher Education Institutions. (2021, September 29). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9619667

Accelerating Decision Making Support in Biometric Assistant for Remote Temperature Measures. (2007, August 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/4290930?denied=

Alarcón, J. A. O., López, J. R. R., & Palma, H. H. (2017). Importancia de la seguridad de los trabajadores en el cumplimiento de procesos, procedimientos y funciones. dialnet. https://dialnet.unirioja.es/descarga/articulo/6713605.pdf

Albalawi, S. (2022). A Comprehensive Overview on Biometric Authentication Systems using Artificial Intelligence Techniques. https://www.semanticscholar.org/paper/A-Comprehensive-Overview-on-Biometric-Systems-using-Albalawi-Alshahrani/81780b939acfa06798a3ff77deeddfe773f9c018

Anaya, J.A. (2019). Evaluación del rendimiento de Facenet. https://www.semanticscholar.org/paper/Evaluaci%C3%B3n-del-rendimiento-de-Facenet-Anaya/bc9ac5ddc52f1987ec13f0a561653d5a1131b4e5

Antika, F. P. (2023). Literature Review: Factors Affecting Marketing Strategy, Market Size, Technology, and End Users on Market Segmentation and Competitive Space. https://www.semanticscholar.org/paper/Literature-Review%3A-Factors-Affecting-Marketing-and-Antika/5f78b7bf49816ef5885f557ea3d57df157784869

Arslan, B. (2017). Machine Learning Methods Used in Evaluations of Secure Biometric System Components. https://www.semanticscholar.org/paper/Machine-Learning-Methods-Used-in-Evaluations-of-Arslan-%C3%9Clker/b546ce1612871cceb24aba50f7a124abb7603427

Bala, R. (2022). Challenges and Ethical Issues in Data Privacy. Igi Global, 12(2), 1–7. https://doi.org/10.4018/ijirr.299938

Bisht, E.A., Prabhat, P., & Kumar, S. (2022). Comparing Performance And Computational Efficiency Of Face Recognition Approaches. *2022 International Conference on Advances in Computing, Communication and Materials (ICACCM)*, 1-7. https://www.semanticscholar.org/paper/Comparing-Performance-And-Computational-Efficiency-Bisht-Prabhat/032b423054d52846f9306f23bdd07eccee35a5c8

Blanco, D. M. (2020). Los sistemas de control de la jornada laboral basados en datos biométricos: Un análisis crítico desde la privacidad. https://www.semanticscholar.org/paper/Los-sistemas-de-control-de-la-jornada-laboral-en-Un-Blanco-Fern%C3%A1ndez/f64464a1054c5e7baabf9f19654632f352284aea

Bravo, C., Ramirez, P., & Arenas, J. P. (2018). Aceptación del Reconocimiento Facial Como Medida de Vigilancia y Seguridad: Un Estudio Empírico en Chile. Información Tecnológica, 29(2), 115–122. https://doi.org/10.4067/s0718-07642018000200115

Calabrò, A. (2019). Integrating Access Control and Business Process for GDPR Compliance: A Preliminary Study. https://www.semanticscholar.org/paper/Integrating-Access-Control-and-Business-Process-for-Calabr%C3%B2-Daoudagh/ec3bafd9ff151515a594128f56d180ac876c0122

Chen, X., & Liu, Y. (2019). The Impact of Information Technology Investment on Firm Performance. DEStech Transactions on Computer Science and Engineering, msota. https://doi.org/10.12783/dtcse/msota2018/27543

Clark, M. (2018, August 14). Global Biometric Market Analysis: Trends and Future Prospects. Bayometric. Retrieved April 24, 2023, from https://www.bayometric.com/global-biometric-market-analysis/

Coral Dev Board. (2023). Coral | Mouser. https://www.mouser.es/new/google-coral/edge-tpu-dev-board/

Downing, S. W., Kang, J., & Markman, G. D. (2019). What You Don't See Can Hurt You: Awareness Cues to Profile Indirect Competitors. Academy of Management Journal, 62(6), 1872–1900. https://doi.org/10.5465/amj.2018.0048

Duarte, T. (2016). Biometric access control systems: A review on technologies to improve their efficiency. https://www.semanticscholar.org/paper/Biometric-access-control-systems%3A-A-review-on-to-Duarte-Piment%C3%A3o/eba0e1691c23c3ed7bafe92bc861b5969759a7c0

Dubey, D. (2017). Echelon Based Pose Generalization of Facial Images Approaches. https://www.semanticscholar.org/paper/Echelon-Based-Pose-Generalization-of-Facial-Images-Dubey/8ac4bd493190ecea0020e046e0f9270ad16c9cbb

Emergen Research, https://www.emergenresearch.com/. (2022). Biometrics Market Trend | Industry Forecast 2021-2030. Emergen Research. Retrieved April 23, 2023, from https://www.emergenresearch.com/industry-report/biometrics-market

Goel, R., Mehmood, I., & Ugail, H. (2021). A Study of Deep Learning-Based Face Recognition Models for Sibling Identification. Sensors, 21(15), 5068. https://doi.org/10.3390/s21155068

Gu, F. (2021). Face Verification Technology Based on FaceNet Similarity Recognition Network. https://www.semanticscholar.org/paper/Face-Verification-Technology-Based-on-FaceNet-Gu-Lu/84b1b21a72f9d1d2eead9a42058793b6333d60d8

Hermans, A., Beyer, L., & Leibe, B. (2017). In Defense of the Triplet Loss for Person Re-Identification. *ArXiv, abs/1703.07737*. https://www.semanticscholar.org/paper/In-Defense-of-the-Triplet-Loss-for-Person-Hermans-Beyer/a42758b4943c7599925e8c8415ee9b8078ff57ad

Infocert.digital. (2023, May 30). 2021 Gartner® Hype CycleTM Identity and Access Management Technologies - Infocert.digital. Infocert.Digital. https://infocert.digital/analyst-reports/2021-gartner-hype-cycle-identity-and-access-management-technologies/

Innova - Ciencia Asturias. (n.d.). Ciencia Asturias. Retrieved May 16, 2023, from https://ciencia.asturias.es/innovaci%C3%B3n

Intel NCSM2485.DK. (2023). Mouser Electronics. https://www.mouser.es/ProductDetail/Intel/NCSM2485.DK?qs=byeeYqUIh0OB4GXNqgW8aw%3D%3D

Kangwanwatana, S., & Sucontphunt, T. (2022). Improve Face Verification Rate Using Image Pre-Processing and FaceNet. *2022 7th International Conference on Business and Industrial Research (ICBIR)*, 426-429. https://www.semanticscholar.org/paper/Improve-Face-Verification-Rate-Using-Image-and-Kangwanwatana-Sucontphunt/c9cf157aa3cc1020deaa38823cc9d3cbba052a34

Kashmar, N. (2022). HEAD Access Control Metamodel: Distinct Design, Advanced Features, and New Opportunities. https://www.semanticscholar.org/paper/HEAD-Access-Control-Metamodel%3A-Distinct-Design%2C-and-Kashmar-Adda/baa6431b21f3abf58e05ab0594bcf70fe0def82b

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., & Krishnan, D. (2020). Supervised Contrastive Learning. *ArXiv, abs/2004.11362*. https://www.semanticscholar.org/paper/Supervised-Contrastive-Learning-Khosla-Teterwak/38643c2926b10f6f74f122a7037e2cd20d77c0f1

Kirişci, M. (2022). New cosine similarity and distance measures for Fermatean fuzzy sets and TOPSIS approach. *Knowledge and Information Systems, 65*, 855 - 868. https://www.semanticscholar.org/paper/New-cosine-similarity-and-distance-measures-for-and-Kiri%C5%9Fci/6de03178f6411b8e34c2577e47b0a40dae2cd227

Kit de desarrollo de NVIDIA Jetson Nano. (n.d.). NVIDIA. Retrieved June 1, 2023, from https://www.nvidia.com/es-es/autonomous-machines/embedded-systems/jetson-nano-developer-kit/

Koch, G.R. (2015). Siamese Neural Networks for One-Shot Image Recognition. Koch, G. R. (2015). Siamese Neural Networks for One-Shot Image Recognition. https://www.semanticscholar.org/paper/Siamese-Neural-Networks-for-One-Shot-Image-Koch/f216444d4f2959b4520c61d20003fa30a199670a

Ku, M., Hwang, J. S., Oh, B., & Park, J. W. (2020). Smart Sensing Systems Using Wearable Optoelectronics. Advanced Intelligent Systems, 2(3), 1900144. https://doi.org/10.1002/aisy.201900144

Ladipo, P. (2022). Market Segmentation and Competitive Advantage in Nigerian Telecommunications. https://www.semanticscholar.org/paper/Market-Segmentation-and-Competitive-Advantage-in-Ladipo-Dixon-Ogbechi/a43c031e3098ca21617425bc190955550990d231

Li, Y., Yuan, G., Wen, Y., Hu, E., Evangelidis, G., Tulyakov, S., Wang, Y., & Ren, J. (2022). EfficientFormer: Vision Transformers at MobileNet Speed. *ArXiv, abs/2206.01191*. https://www.semanticscholar.org/paper/EfficientFormer%3A-Vision-Transformers-at-MobileNet-Li-Yuan/dd1139cfc609c2f3263d02e97176d5275caebc0a

Liao, M., Agnihotri, D., & Zhong, X. (2022). "Paying with my face" – Understanding users' adoption and privacy concerns of facial recognition payment. Proceedings of the Human Factors and Ergonomics Society . . . Annual Meeting, 66(1), 731–735. https://doi.org/10.1177/1071181322661480

Machine and Deep learning in Biometric Authentication: A Review. (2023, May 5). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/10141692

Mangata, B. B. (2022). Modeling and implementation of an automatic Access control system for secure permises using facial recognition. https://www.semanticscholar.org/paper/Modeling-and-implementation-of-an-automatic-Access-MANGATA-Reagan/7bdbc80ad60bbe5899944fac53ce16343e484cb1

Mehrabi, N. (2019, August 23). A Survey on Bias and Fairness in Machine Learning. arXiv.org. https://arxiv.org/abs/1908.09635

Naseri, M.N., & Agrawal, A.P. (2021). Impact of Transfer Learning on Siamese Networks for Face Recognition with Few Images Per Class. *2021 Asian Conference on Innovation in Technology (ASIANCON)*, 1-5. https://www.semanticscholar.org/paper/Impact-of-Transfer-Learning-on-Siamese-Networks-for-Naseri-Agrawal/325440a47bedd31fda8bf25c63b5e4a88707e195

Nayak, S., Bhat, M., Subba Reddy, N.V., & Ashwath Rao, B. (2022). Study of distance metrics on k - nearest neighbor algorithm for star categorization. Journal of Physics: Conference Series, 2161. https://www.semanticscholar.org/paper/Study-of-distance-metrics-on-k-nearest-neighbor-for-Nayak-Bhat/d23f670b91595b1ae5c71fb7794328ce65f644d7

NVIDIA Jetson Nano Developer Kit (945-13450-0000-100) : Amazon.es: Informática. (2023). https://www.amazon.es/NVIDIA-Jetson-Nano-Kit-desarrollo/dp/B084DSDDLT

Nzegha, A. F., Fendji, J. L. E. K., Thron, C., & Tayou, C. T. (2020). Overview of Deep Learning in Facial Recognition. In Studies in computational intelligence. Springer Nature. https://doi.org/10.1007/978-3-030-37830-1_6

Omar, R. (2020). A Comparative Study of Network Access Control and Software-Defined Perimeter. https://www.semanticscholar.org/paper/A-Comparative-Study-of-Network-Access-Control-and-Omar-Abdelaziz/b78a849af1d44b59f28d864f7ca39deee351d71f

Onyema, E. M., Shukla, P. K., Dalal, S., Mathur, M. N., Zakariah, M., & Tiwari, B. (2021). Enhancement of Patient Facial Recognition through Deep Learning Algorithm: ConvNet. Journal of Healthcare Engineering, 2021, 1–8. https://doi.org/10.1155/2021/5196000

Oo, S.L., & Oo, A.N. (2019). ASEAN Child Face Recognition System with FaceNet. https://www.semanticscholar.org/paper/ASEAN-Child-Face-Recognition-System-with-FaceNet-Oo-Oo/a305cf41eec82785da0ad427516b3f4752c9156f

PcComponentes. (2023, January 29). Raspberry Pi 4 Modelo B 2GB | PcComponentes.com. https://www.pccomponentes.com/raspberry-pi-4-modelo-b-2gb

Quisilema, Y. (2017). PBR-erry : dispositivo low cost de seguridad palmar biométrico. https://www.semanticscholar.org/paper/PBR-erry-%3A-dispositivo-low-cost-de-seguridad-palmar-Quisilema-Wladimir/bebc886798ce96e5cbbaf47b4a72f0a3f4a53cc5

Raspberry Pi 4 Modelo B (4GB) : Amazon.es: Informática. (2023). https://www.amazon.es/Raspberry-Pi-4595-Modelo-GB/dp/B09TTNF8BT

Safdar, F. (2021). A Comparison of Face Recognition Algorithms for Varying Capturing Conditions. https://www.semanticscholar.org/paper/A-Comparison-of-Face-Recognition-Algorithms-for-Safdar/06a47453ff9f47bd21f282bdb2d3e93bb4e8536a

ŞahiN, Y. (2023). An Enterprise Data Privacy Governance Model: Security-Centric Multi-Model Data Anonymization. dergipark.org.tr. https://doi.org/10.29137/umagd.1272085

Sarasa, C. C. G. (2020). Técnicas de aumento de datos para imágenes aéreas y evaluación de rendimiento en modelos de deep learning. https://www.semanticscholar.org/paper/T%C3%A9cnicas-de-aumento-de-datos-para-im%C3%A1genes-a%C3%A9reas-y-Sarasa-Pab%C3%B3n/b98331d9c8a4a2a72ca54b77e9402b35bccacaad

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. https://doi.org/10.1109/cvpr.2015.7298682

Selvam, V. S. D. (2020b, May 8). Human Error in IT Security. arXiv.org. https://arxiv.org/abs/2005.04163

Shelton, J., Dozier, G., Bryant, K., Adams, J., Popplewell, K., Abegaz, T., Purrington, K., Woodard, D. L., & Ricanek, K. (2011). Genetic based LBP feature extraction and selection for facial recognition. https://doi.org/10.1145/2016039.2016092

Silvestrini, S., & Lavagna, M. (2022). Deep Learning and Artificial Neural Networks for Spacecraft Dynamics, Navigation and Control. Drones, 6(10), 270. https://doi.org/10.3390/drones6100270

Soleymani, S., Chaudhary, B., Dabouei, A., Dawson, J., & Nasrabadi, N. M. (2021, February). Differential morphed face detection using deep siamese networks. In Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part VI (pp. 560-572). Cham: Springer International Publishing. https://link.springer.com/chapter/10.1007/978-3-030-68780-9_44

Srihari, P., & Harikiran, J. (2022). Skeleton Based Human Activity Prediction in Gait Thermal images using Siamese Networks. *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, 1163-1170. https://www.semanticscholar.org/paper/Skeleton-Based-Human-Activity-Prediction-in-Gait-Srihari-Harikiran/dcd8c9d2bd55adccc814e6a3c9858f8761e06127

Suman, S. (2023). Authentication System for Enterprise Security. https://www.semanticscholar.org/paper/Authentication-System-for-Enterprise-Security-Suman/987a8f48e7133fa709df1776642c7772b24f9a5b

Sun, G. (2022). RF Transmitter Identification Using Combined Siamese Networks. *IEEE Transactions on Instrumentation and Measurement, 71*, 1-13. https://www.semanticscholar.org/paper/RF-Transmitter-Identification-Using-Combined-Sun/69627b521e2822dce6fd1c655c2e4fdc658195d3

Suwanda, R., Syahputra, Z., & Zamzami, E.M. (2020). Analysis of Euclidean Distance and Manhattan Distance in the K-Means Algorithm for Variations Number of Centroid K. *Journal of Physics: Conference Series, 1566*. https://www.semanticscholar.org/paper/Analysis-of-Euclidean-Distance-and-Manhattan-in-the-Suwanda-Syahputra/e74cceadd848b39fe7090b53458a34046dd7fb4b

Talent.com. (n.d.). Búsqueda de empleo en Talent.com | Encuentra vacantes disponibles cerca de ti. https://es.talent.com/es/

Terhorst, P., Kolf, J.N., Damer, N., Kirchbuchner, F., & Kuijper, A. (2020). SER-FIQ: Unsupervised Estimation of Face Image Quality Based on Stochastic Embedding Robustness. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5650-5659. https://www.semanticscholar.org/paper/SER-FIQ%3A-Unsupervised-Estimation-of-Face-Image-on-Terhorst-Kolf/8ff988530e3329bd6ab00dc5eef635a1bc5812ca

Tzinis, I. (2021). Technology Readiness Level. NASA. https://www.nasa.gov/directorates/heo/scan/engineering/technology/technology_readiness_level

Wong, K. C. K. (2021). Enhancing Low-Energy Facial Recognition Devices and Abilities To Improve Security: A Practical Study. https://www.semanticscholar.org/paper/Enhancing-Low-Energy-Facial-Recognition-Devices-and-Wong-Hunter/197c9752b05a210db75edfc5692515905d33e1b6

Wu, H., Cao, Y., Wei, H., & Tian, Z. (2021). Face Recognition Based on Haar Like and Euclidean Distance. *Journal of Physics: Conference Series, 1813*. https://www.semanticscholar.org/paper/Face-Recognition-Based-on-Haar-Like-and-Euclidean-Wu-Cao/0af45eda3eddb9a320aaa9634a997cc08a7e116a

Wu, Y. (2023). Edge-AI-Driven Framework with Efficient Mobile Network Design for Facial Expression Recognition. https://www.semanticscholar.org/paper/Edge-AI-Driven-Framework-with-Efficient-Mobile-for-Wu-Zhang/b09775dbc4c9a1f4a21095d8a6e0668804772031

Xu, X., Zhang, L., Duan, C., & Lu, Y. (2020). Research on Inception Module Incorporated Siamese Convolutional Neural Networks to Realize Face Recognition. *IEEE Access, 8*, 12168-12178. https://www.semanticscholar.org/paper/Research-on-Inception-Module-Incorporated-Siamese-Xu-Zhang/8d965fbe99b5bdd71d764f502b239aae5dde89ec

Yang, W., Wang, S., Zheng, G., Chaudhry, J. A., & Valli, C. (2018). ECB4CI: an enhanced cancelable biometric system for securing critical infrastructures. The Journal of Supercomputing, 74(10), 4893–4909. https://doi.org/10.1007/s11227-018-2266-0

Zhang, X. (2022, June 6). Compilation and Optimizations for Efficient Machine Learning on Embedded Systems. arXiv.org. https://arxiv.org/abs/2206.03326

Anexo I – Propuesta del Proyecto

Nombre alumno:	Jon Imaz Dravasa
Titulación:	Ingeniería Informática
Curso académico:	2022-2023

1. TÍTULO DEL PROYECTO

Sistema de control de acceso para empresas mediante el uso de reconocimiento facial.

2. DESCRIPCIÓN Y JUSTIFICACIÓN DEL TEMA A TRATAR

Los sistemas biométricos, en especial la identificación facial, permiten obtener mayor seguridad que los sistemas actuales basados en llave física. La utilización de las tecnologías de Deep Learning permiten obtener mejores rendimientos, cuando parte de la cara está oculta o por ejemplo con mascarilla, permitiendo conseguir resultados más robustos. Además, la utilización de hardware embebido es un reto actual por las reducidas capacidades de cálculo.

El proyecto consiste en la creación del prototipo de un sistema de reconocimiento facial orientado a la seguridad de la entrada del personal de las empresas mediante algoritmos de Deep Learning u otros en el estado del arte, en un sistema hardware embebido con reducidas capacidades de cálculo.

3. OBJETIVOS DEL PROYECTO

Los diferentes objetivos a obtener durante el proyecto son:

- Analizar el estado del arte y algoritmos para la identificación facial.
- Evaluar la mejor solución basándonos en robustez y requisitos de hardware
- Desarrollo de una aplicación para un sistema embebido de reconocimiento facial
- Permitir el reconocimiento de varias personas diferentes.
- Desarrollar un sistema de gestión y control de los distintos usuarios.
- Almacenar de forma segura los datos en una base de datos.
- Todo el sistema estará integrado en un sistema embebido con recursos limitados.

4. METODOLOGÍA

La metodología se fijará en las primeras reuniones con el tutor.

5. PLANIFICACIÓN DE TAREAS

Las tareas se irán definiendo de acuerdo a los objetivos del proyecto y se irán adecuando para tener una fase de implementación óptima.

6. OBSERVACIONES ADICIONALES

Se propuso la idea de ser dirigido por Rafael Del Hoyo.

Anexo II – Material Adicional

Como anexo a esta memoria se adjunta el código correspondiente a este proyecto. Se ha incluido una carpeta que incluye tres archivos y un directorio. Los archivos serán los documentos donde se almacenan todo el código fuente utilizado en este proyecto. Por otro lado, el directorio contendrá todo lo necesario para una correcta ejecución del sistema de control de acceso, con todos los directorios necesarios, imágenes y base de datos incluidos en su interior.

El código fuente ha sido comentado para una mejor y más fácil comprensión de este.