



**Escuela Superior
de Ingeniería y Tecnología**
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Curso 2020-2021

Catalogación automática de
colecciones digitales basado en el
contenido visual

*Automated cataloging of digital
collections based on the visual content*

Nicolás Hernández González

La Laguna, 30 de junio de 2021

Dña. **María Elena Sánchez Nielsen**, con N.I.F. 42848599J profesora Titular de Universidad adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora

C E R T I F I C A

Que la presente memoria titulada:

“Catalogación automática de colecciones digitales basado en el contenido visual”

ha sido realizada bajo su dirección por D. **Nicolás Hernández González**,
con N.I.F. 54108335F.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 30 de junio de 2021

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento-
NoComercial-CompartirIgual 4.0 Internacional.

Resumen

Este trabajo tiene como objetivo el desarrollo e implementación de un sistema que haga uso de la API de *Face*, correspondiente a los servicios cognitivos de *Microsoft Azure*, con el fin de analizar y clasificar las imágenes de diferentes colecciones digitales, todo ello basado en el reconocimiento facial.

Tras su desarrollo, el sistema resultante permite a los usuarios llevar a cabo consultas de búsquedas, tales como: mostrar personas que cumplan diferentes criterios de búsqueda (género, edad, emoción, etcétera), mostrar las imágenes donde aparezca una persona en concreto, localizar las imágenes donde aparezcan personas con un rostro similar a una persona específica y mostrar aquellas personas que aparecen con más frecuencia con relación a una persona específica.

Con el fin de evaluar tanto el resultado de las consultas como su rendimiento, se ha llevado a cabo una validación del sistema empleando colecciones digitales de un entorno real, donde se analizan los resultados de correcta identificación y la detección de falsos positivos, así como el tiempo de ejecución de las consultas.

Palabras clave: reconocimiento facial, Azure, Face, consulta.

Abstract

This work aims to develop and implement a system that makes use of the *Face* API, which corresponds to the cognitive services of *Microsoft Azure*, in order to analyze and classify images from different digital collections, all based on facial recognition.

After its development, the resulting system allows users to perform search queries, such as: displaying people that meet different search criteria (gender, age, emotion, etc.), displaying images where a specific person appears, locating images where people with a face similar to a specific person appear, and displaying those people that appear more frequently in relation to a specific person.

In order to evaluate both the result of the queries and their performance, a validation of the system has been carried out using digital collections of a real environment, where the results of correct identification and the detection of false positives, as well as the execution time of the queries, are analyzed.

Keywords: facial recognition, Azure, Face, query.

Índice General

Capítulo 1	Introducción	1
1.1	Objetivos.....	2
Capítulo 2	Antecedentes	3
2.1	Reconocimiento facial	3
2.1.1	Introducción al tema.....	3
2.1.2	Ámbitos de uso	5
2.1.3	Legalidad y ética.....	7
2.2	Microsoft Azure	8
2.2.1	Software as a Service.....	9
2.2.2	Infrastructure as a Service	9
2.2.3	Platform as a Service	10
2.2.4	Servicios de datos y <i>Big Data</i>	10
Capítulo 3	Análisis	13
3.1	Base de datos.....	13
3.1.1	Adobe Bridge	14
3.1.2	Alternativas.....	15
3.2	API de reconocimiento facial	16
3.2.1	Computer Vision.....	17
3.2.2	Face.....	19
3.2.3	APIs alternativas para el reconocimiento facial.....	23
3.3	Entorno de desarrollo.....	24
3.3.1	Visual Studio.....	24
Capítulo 4	Diseño	25
4.1	Base de datos.....	26
4.2	Arquitectura	27
4.3	Uso de la API <i>Face</i>	28
4.3.1	Modo de empleo.....	28
4.3.2	Estructuras de datos	29
4.3.3	Funciones	30

4.4	Interfaz de Usuario.....	31
4.4.1	Vista principal.....	32
4.4.2	Vista de consulta personalizada	33
4.4.3	Vista de selección de rostro.....	33
4.4.4	Vista de galería	34
4.4.5	Vista de galería de acompañantes	34
4.5	Sub procesos	35
4.5.1	Consulta simple y personalizada	35
4.5.2	Búsqueda de una persona determinada.....	39
4.5.3	Búsqueda de gente similar.....	42
4.5.4	Búsqueda de acompañantes.....	44
4.6	Dificultades e inconvenientes.....	51
Capítulo 5	Pruebas	52
5.1	Funciones de registro.....	52
5.2	Adquisición de imágenes	53
5.3	Tiempo de ejecución	54
5.4	Detección	57
5.4.1	Búsqueda de características.....	58
5.5	Reconocimiento.....	58
5.5.1	Verificación.....	58
5.5.2	Búsqueda de personas similares	60
5.6	Falsos positivos	60
5.7	Análisis de resultados.....	62
Capítulo 6	Presupuesto	64
Capítulo 7	Conclusiones y líneas futuras	65
Capítulo 8	Summary and Conclusions.....	66
Capítulo 9	Bibliografía.....	67

Índice de ilustraciones

Ilustración 1. Funcionamiento de una API.....	16
Ilustración 2. Resultado de Computer Vision API	18
Ilustración 3. Face API en Azure	19
Ilustración 4. Puntos de referencia de una cara.....	19
Ilustración 5. Ejemplo de verificación.....	20
Ilustración 6. Ejemplo de imagen etiquetada en Bridge.....	27
Ilustración 7. Arquitectura del sistema.....	27
Ilustración 8. Información de recurso Azure.....	28
Ilustración 9. Vista Principal	32
Ilustración 10. Menú de consulta personalizada	33
Ilustración 11. Vista de selección de rostro.....	34
Ilustración 12. Galería de resultados	34
Ilustración 13. Galería de acompañantes.....	35
Ilustración 14. Ejemplos de colecciones 1 y 4	54
Ilustración 15. Román Rodríguez Rodríguez.....	60
Ilustración 16. Imágenes donde se detectaron gafas de buceo.....	62
Ilustración 17. Rostros etiquetados como similares.....	63

Índice de Tablas

Tabla 1. Características de Azure Cognitive Services [23]	11
Tabla 2. Características de Computer Vision [24].....	17
Tabla 3. Modelos de detección [29].....	22
Tabla 4. Atributos faciales detectados por el sistema.....	29
Tabla 5. Tipos de consultas simples.....	36
Tabla 6. Resumen de colecciones digitales.....	53
Tabla 7. Tiempos de ejecución para búsqueda de personas	54
Tabla 8. Tiempo de detección de rostros en segundos	55
Tabla 9. Tiempo de búsqueda de personas similares en segundos.....	55
Tabla 10. Duración de la operación de filtrado en segundos.....	56
Tabla 11. Duración de la operación de verificación en segundos.....	56
Tabla 12. Número de rostros detectados según el M.D.....	57
Tabla 13. Gráfico de índice de acierto en operaciones de verificación	58
Tabla 14. Gráfico de acierto en verificación para la colección 4.....	59
Tabla 15. Resultados de búsqueda de personas similares.....	60
Tabla 16. Resultados y falsos positivos para el M.D. 3.....	61
Tabla 17. Herramientas y licencias.....	64
Tabla 18. Horas invertidas y coste.....	64

Capítulo 1

Introducción

Cuando hablamos de reconocimiento facial, hablamos de una tecnología que en sus inicios fue considerada como un tópico más de la ciencia ficción, para luego convertirse en algo rutinario e incluso una comodidad para muchos; a la par que ser capaz de infundir temor debido a la posibilidad de ser empleada como herramienta de control.

El mejor ejemplo que puede demostrar este hecho es el cómo las empresas investigan, desarrollan o simplemente adoptan dicha tecnología para mantener los niveles de competitividad que exigen los nuevos estándares marcados por la Cuarta Revolución Industrial [1]; donde las propiedades intelectuales, así como la conectividad y automatización se vuelven un activo y un aspecto clave respectivamente. No obstante, no son únicamente las grandes entidades quienes tienen la capacidad de trabajar con esta tecnología, y esto se debe a que en la actualidad se pueden encontrar herramientas para trabajar en ella sin una dificultad mayor que realizar una búsqueda en cualquier navegador.

Sin embargo, no debe olvidarse que si bien es posible desarrollar aplicaciones que empleen reconocimiento facial sin una excesiva dificultad, esto se debe principalmente a las herramientas creadas por empresas para que se emplee su tecnología de reconocimiento facial, en muchos casos mediante librerías. Gracias a esto, podremos emplear dichas técnicas, pero sin conocer los entresijos de los procesos que lo hacen posible; lo cual no es necesariamente un aspecto negativo.

Dicho esto, este trabajo se centra en la alternativa desarrollada por la gigante tecnológica *Microsoft*, que no es otra que *Face*, una API perteneciente a los servicios cognitivos de *Microsoft Azure*. Este servicio facilita el desarrollo de software basado en el reconocimiento facial, sin necesidad de implementar los algoritmos correspondientes, con la posibilidad de emplear su procesamiento en la nube o utilizar contenedores para entornos locales donde se priorice la seguridad de los datos y una baja latencia. Además, *Microsoft* se vale su inversión millonaria en el ámbito de desarrollo e investigación en ciberseguridad para hacer alarde de la seguridad de los datos y proyectos almacenados en su plataforma frente a las amenazas externas.

1.1 Objetivos

El propósito de este trabajo no es otro que el de estudiar y evaluar el comportamiento de la API *Face* de *Microsoft Azure* a la hora de detectar y reconocer rostros, así como en la extracción de diferentes características o atributos faciales de estos.

Para cumplir con dicho objetivo, en primer lugar, se llevará a cabo la elaboración de una base de datos que tenga la capacidad de gestionar y administrar diferentes colecciones digitales.

Posteriormente, se desarrollará un sistema que empleará la API de *Azure* para llevar a cabo operaciones de búsqueda tales como:

- Mostrar personas en base a diferentes características tales como la emoción que presentan (felicidad, enfado, tristeza, etc.) o la edad.
- Mostrar aquellas imágenes donde aparezca una persona en concreto.
- Mostrar las imágenes donde aparezcan personas con un rostro similar a una persona específica.
- Mostrar aquellas personas que aparecen con más frecuencia con relación a una persona específica.

Por último, con el fin de evaluar el comportamiento del sistema, se llevará a cabo un análisis de los resultados de las consultas, así como del rendimiento, realizando las pruebas sobre una colección digital en un entorno real.

Capítulo 2

Antecedentes

2.1 Reconocimiento facial

2.1.1 Introducción al tema

El reconocimiento facial como tal, es una actividad que los seres humanos podemos llevar a cabo sin mucho esfuerzo, pero con un índice de éxito notable; no obstante, la naturaleza de este proceso es tanto el sujeto de numerosos debates como objetivo de estudios en lo referente a su comprensión e implementación como sistema. Las investigaciones se han centrado principalmente en determinar el tipo de dominio que mejor controlan los humanos, si específico o general, y en cómo éstos desarrollan su destreza para con el reconocimiento facial, no tanto así como en el hecho de cuestionar dicha maestría en sí misma.

Sobre esta temática se ha estudiado la notable diferencia de rendimiento en cuanto al reconocimiento facial de los humanos entre rostros que nos resultan familiares frente a aquellos que no [2], en experimentos donde a los participantes se le pedía emparejar imágenes de personajes que no les eran familiares, esto resultaba en una tarea considerablemente más ardua que en rostros conocidos. Junto a otros experimentos, se consiguieron resultados que indicaban una diferencia en el modo en que percibimos rostros de aquellos que no son conocidos de los que no, e incluso, se deja en cuestión la posibilidad de que se empleen mecanismos perceptuales de diferencias cualitativas durante el reconocimiento.

Dejando a lado el rango de destreza de los humanos, el aspecto que más nos interesa es aquel relacionado con la computación. En lo que a esto respecta, en más de 20 años de investigación y desarrollo, aún está en duda que los sistemas automáticos puedan alcanzar el nivel de rendimiento humano, en especial en aquellos entornos donde las condiciones no son ideales.

En cuanto al procedimiento, podemos afirmar que los métodos empleados se enfocan en la extracción de las características de los rostros a analizar, aunque la forma de proporcionar el rostro varía con el método. Por ejemplo, en el caso de tener una imagen 2D, una de las técnicas más relevantes para extraer características es la de *Face-GLOH-signature*, cuya fiabilidad en resultados no se ve afectada por la escala, rotación u orientación de las imágenes, a diferencia de otras técnicas cuyos descriptores precisan de imágenes alineadas [3]. Este descriptor se define como un histograma 3D de localización y orientación, donde la localización y el ángulo están cuantificados mediante una cuadrícula de coordenadas

logarítmicas polares y el empleo de ocho orientaciones respectivamente; cada plano de orientación representa la magnitud del gradiente correspondiente a una orientación dada. Uno de los méritos más destacables que supone emplear este descriptor, es el de reducir la dimensionalidad de la imagen, aspecto que mejora el rendimiento de los métodos basados en aprendizaje automático, los cuales se vuelven prohibitivos para los descriptores tradicionales a causa de la conocida como la maldición de la dimensión¹.

Dicho esto, las técnicas no pueden ser iguales si se utiliza como fuente un rostro 3D; cuyo proceso está basado en la construcción de un mapa 2D a través de un mapeo esférico, con lo cual se obtiene un mapa bidimensional que mantiene las relaciones espaciales entre las características faciales. La comparación entre dos mapeados se ejecuta calculando el ángulo entre cada par de coordenadas según el color de cada píxel, para dar lugar a un nuevo mapeado con los valores de éstos representados con colores normalizados, para luego analizar su histograma a fin de poder medir la similitud entre los mapas originales [4].

Esta metodología puede llevarse a la práctica empleando redes neuronales prealimentadas², donde las neuronas, estructuradas en forma de capas, son entrenadas mediante el cálculo y proyección de los eigenvectores calculados mediante los métodos de extracción de características Análisis de Componentes Principales (*PCA*, por sus siglas en inglés) o Análisis Discriminante Lineal (*LDA*) [5]. El método *PCA*, también llamado *eigenfaces*, es una técnica basada en la apariencia que codifica la información en un espacio lineal ortogonal, mientras que el método *LDA*, igualmente denominado como *fisherfaces*, aunque también está basado en la apariencia, codifica la información en un espacio lineal separable cuyas bases no son necesariamente ortogonales. Según el procedimiento empleado, estas estructuras se denominan *PCA-NN* y *LDA-NN* para los métodos de *eigenfaces* y *fisherfaces* respectivamente; ambas estructuras han demostrado obtener un mayor ratio de reconocimiento que sus homólogos tradicionales basados en el uso de la distancia euclídea como clasificador.

Ahora bien, no sería correcto tratar como temas independientes el cómo las personas y los sistemas llevan cabo el reconocimiento facial, puesto que existen estudios realizados con el fin de profundizar en la temática del reconocimiento facial en rostros que familiares y aquellos que no desde el punto de vista computacional, mencionada con anterioridad. Por ejemplo, se ha estudiado el rendimiento de una red neuronal profunda convolucional con el de una persona, donde la red sólo obtiene unos resultados ligeramente peores para los reconocimientos de rostros no familiares [6]. También se observa el cómo la habilidad de las personas de obtener múltiples puntos de referencia y de realizar comparaciones detalladas de las características le supuso una ventaja sobre la red, la cual computaba una

¹ La maldición de la dimensión se refiere a varios fenómenos que surgen al analizar y organizar los datos en espacios de alta dimensión que no se dan en entornos de baja dimensión.

² Una red neuronal prealimentada es una red neuronal artificial donde las conexiones entre las unidades no forman un ciclo, la información se mueve en una única dirección: adelante.

representación perceptual de cada imagen, independientemente de la imagen a comparar. Como conclusión se obtiene que, si bien las idiosincrasias de los rostros deben aprenderse, no representan la variabilidad dominante, sino que es la variabilidad genérica la que debe ser aprendida para mejorar el reconocimiento de los rostros no familiares. Dicho esto, no es la única aproximación al tema puesto que se podría clarificar en mayor medida qué rasgo es genérico y cual es idiosincrático, además de emplear el análisis de imágenes combinado con un apoyo *top-down* para cohesionar diferentes imágenes que pertenezcan a la misma persona, aspecto que se podría considerar básico en los modelos [7], lo cual serviría como réplica a lo dicho anteriormente, aunque tenga puntos en común.

Podemos concluir que la tecnología del reconocimiento facial es parte de aquellas tecnologías que suponen una gran dificultad para los sistemas, a la par que un escaso esfuerzo para los seres humanos, no obstante, la continua investigación y desarrollo, así como sus posibilidades de uso, no invitan a pensar otra cosa más que en un futuro con técnicas más refinadas que permitan alcanzar, o incluso superar, la capacidad humana en este ámbito.

2.1.2 Ámbitos de uso

La tecnología de reconocimiento facial actualmente está presente en multitud de campos presentando funciones diversas más allá de la ya conocida identificación de rostros en seguridad. Los ámbitos de uso abarcan no sólo empresas y entidades, sino incluso cualquier persona que quiera valerse del reconocimiento facial para realizar algo tan rutinario como desbloquear el teléfono móvil o detectar sonrisas a la hora de sacar una foto.

Educación

En el ámbito educativo, cada vez con mayor frecuencia se encuentran ofertas de educación a distancia, lo cual se ha visto incrementado de forma exponencial debido a la pandemia y con cada vez más estudiantes que la demandan. Y es precisamente en este entorno donde la información sobre la metodología de evaluación del aprendizaje se vuelve escasa, lo cual genera toda una problemática que afecta negativamente a la valoración y credibilidad de las titulaciones otorgadas mediante este tipo de educación. Todo esto supone una oportunidad para emplear la tecnología de reconocimiento facial para solventar, o al menos mitigar, los aspectos negativos a la par que mejorar la calidad educativa.

Como ejemplos, podemos encontrar el estudio para el cual se realizó una prueba piloto con 67 estudiantes de máster de la Universidad a Distancia de Madrid [8], donde el reconocimiento facial permite a los profesores comprobar si el alumno es quien realiza las actividades, con el fin de que el alumno interactuase el mayor tiempo posible dentro de la actividad; como resultado se obtuvieron resultados positivos, con puntuaciones entre 5,54 y 6,15 en una escala Likert de siete puntos. Alternativamente, existe la posibilidad de emplear el reconocimiento facial en el control de asistencia [9], para determinar la presencialidad del alumnado, mientras se evalúan las diferentes metodologías de su implementación.

Marketing

En el mundo de la mercadotecnia y la publicidad, podríamos decir que el uso de la publicidad personalizada basada en nuestra información se ha vuelto la práctica estándar de la industria, puesto que se ha demostrado que su simpleza y alcance la vuelven una opción más rentable que los medios convencionales.

La publicidad personalizada es posible gracias a la información obtenida de sus usuarios por parte de las empresas que, mediante la elaboración de perfiles, puede inferir qué tipo de anuncio se adapta a cada uno, dando como resultado una publicidad más eficiente. Precisamente en la obtención de datos es donde el reconocimiento facial tiene cabida, pues el rostro humano puede considerarse como un dato más para obtener los hábitos de compra o demás preferencias asociadas a cada usuario; de tal forma este tipo de sistema ya se puede encontrar como patente [10].

Salud

En el marco de la salud también se benefician de las ventajas tecnológicas en la interminable proceso de elaborar nuevos tratamientos o de mejorar los existentes, puesto que no es un sector ajeno al deber de adecuarse a las necesidades de sus usuarios; lo cual se ve acrecentado en tiempos de pandemia que precisa de nuevos métodos y lo hace con urgencia.

El reconocimiento facial puede resultar útil de diferentes maneras en este sector, por ejemplo, para evitar el fraude médico y la suplantación de identidad, algo que puede impedir a los afectados el acceso a la atención médica en el futuro, podría solventarse mediante la implementación de esta tecnología a fin de evitar estos actos mediante una mejor identificación de los pacientes [11]. En su uso como tratamiento, podemos destacar el uso del reconocimiento facial para determinar la topografía facial de los músculos del rostro, puesto que difieren no solo entre hombres y mujeres, sino incluso para individuos del mismo sexo; por ello, la tecnología resulta útil tanto en las operaciones quirúrgicas en el rostro, como a la hora de predecir los cambios en las características y expresiones faciales que puedan producirse tras una cirugía correctiva [12].

Seguridad

Sin duda alguna, el ámbito de la seguridad es al que está mayormente ligado la tecnología de reconocimiento facial, ya sea como identificación o como parte de la biometría, es el sector más pujante a la hora de promover la investigación y desarrollo de esta tecnología, que se estima que mueva en torno a los 3.100 millones de dólares para 2022, sustentado por las incipientes necesidades de seguridad y la proliferación de la biometría tradicional.

Para ejemplificar hasta qué punto ha avanzado la tecnología para este ámbito, nos podemos valer de una tesis de la Universidad Politécnica de Valencia [13], donde se demuestra el cómo es posible elaborar un sistema de videovigilancia con un índice de acierto que ronda el 80%, que si bien se ha utilizado una determinada base datos como fuente, es decir, se realizó en un entorno controlado, demuestra que el desarrollo de estos sistemas está sumamente normalizado incluso fuera de las grandes corporaciones del sector.

También es interesante analizar cómo reacciona el ciudadano ante estas medidas de seguridad; por ejemplo, en un estudio llevado a cabo en Chile [14], se tienen en cuenta tanto rasgos de personalidad como una variable que mide la percepción de utilidad, así como las preocupaciones en lo referente a la utilidad de la tecnología en cuestión. Como conclusiones podemos destacar la valoración positiva de la utilidad de la tecnología cuanto mayor es la preocupación por la privacidad, y la posibilidad de obtener una mayor aceptación en aquellos países donde el nivel de inseguridad sea mayor.

2.1.3 Legalidad y ética

Como ocurre con la mayoría de las tecnologías, sus aspectos positivos no eliminan la necesidad de una regulación que protejan frente a los abusos que puedan resultar de su uso. En el contexto del reconocimiento facial esto afecta tanto a la privacidad como a la protección de datos.

Es un debate recurrente aquel basado en prescindir de la privacidad en favor de una mayor seguridad o incluso comodidad; no obstante, no debe obviarse la normativa vigente sobre protección de datos establecida en la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos de Carácter Personal y en el Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016 (Reglamento general de protección de datos), cuyo artículo 4-14) define como “datos biométricos” aquellos datos personales obtenidos a partir de un tratamiento técnico específico, relativos a las características físicas, fisiológicas o conductuales de una persona física que permitan o confirmen la identificación única de dicha persona, como imágenes faciales o datos dactiloscópicos.

De esta forma, el artículo 9-1 del Reglamento general de protección de datos manifiesta la prohibición del tratamiento de datos biométricos dirigidos a identificar de manera unívoca a una persona física, a no ser que el usuario sea consciente de que se le está sometiendo al reconocimiento facial, lo cual excluiría la vulneración de sus derechos por existir consentimiento expreso al tratamiento de la imagen y, por tanto, encontrarse dentro de las excepciones previstas, respectivamente, en el artículo 2-2 de la citada Ley Orgánica 3/2018, y en el artículo 9-2-a) del Reglamento general de protección de datos.

A pesar de todo lo dicho, no es el aspecto legal el único que de tenerse en cuenta a la hora emplear la tecnología, puesto que se pueden llevar a cabo abusos bajo el marco de la legalidad, lo cual pone de manifiesto la necesidad de actuar siempre considerando la ética para con aquellos que puedan resultar damnificados.

En referencia a esta época contemporánea, el artículo de Gloria López-Cleires y Álvaro Porras Soriano [15] analiza y critica esta situación de hiper-exposición, donde las redes sociales y los dispositivos móviles conforman un modelo de vigilancia difuso donde más que un vehículo para el intercambio de información, estos medios se emplean para la formación

de la identidad y su control. Como ejemplo se muestra el sistema de etiquetado de la red social Facebook, donde se ampara bajo el pretexto de evitar la suplantación para ocultar una invasión de la privacidad donde los usuarios son etiquetados sin su consentimiento.

Siguiendo esta línea, en el ámbito de la medicina, y como ya se ha mencionado previamente, la tecnología del reconocimiento facial se emplea tanto en la monitorización de pacientes como en los diagnósticos, lo cual supone una serie de implicaciones éticas que se deben tener en cuenta. En su artículo en *AMA Journal of Ethics*, Nicole Martinez-Martin [16] trata sobre los principales aspectos éticos que se deben valorar a la hora de emplear esta tecnología en el campo de la medicina; donde se destaca la importancia del consentimiento informado en cuanto al riesgo que genera el hecho de que toda esa información clínica adicional no esté completamente anonimizada. También es importante exigir un nivel de precisión que demuestre que puede resultar más beneficiosa que perjudicial, algo especialmente importante en el ámbito de la medicina, junto a garantizar la privacidad del paciente.

Relacionado con la situación actual, donde cualquier actividad está condicionada por la pandemia, es interesante ver el análisis de las consideraciones éticas que deberían tener los estados sobre el efecto negativo que ha tenido el uso de las mascarillas en la eficacia de las mediciones biométricas, además de poner en duda la eficacia de la predicción de comportamientos basándose en la observación y medición de emociones sobre las manifestaciones faciales y corporales, llevado a cabo por Cristhian Almonacid Díaz [17] para la revista *Veritas: revista de filosofía y teología*.

2.2 Microsoft Azure

Microsoft Azure, cuyo lanzamiento se remonta a febrero de 2010, es una marca general para los servicios de computación en la nube de *Microsoft*. Abarca una amplia, y aún creciente, gama de servicios que conforman los elementos principales de la computación en la nube.

Es importante tener en cuenta que, si bien no tiene una extensa historia, no debe tratarse a *Azure* como una tecnología novel, pues está basada en otras tecnologías de renombre de *Microsoft* tales como *Windows Server Hyper-V*, *Active Directory services*, *SQL Server*, *System Center* y demás.

Con una infraestructura que se identifica a nivel global, regional y zonal, las regiones que maneja *Azure* están basadas en países y divididas en zonas geográficas. Esto implica que no todos los servicios están disponibles en todas las regiones del mundo, ya que cada una tiene una disponibilidad específica.

Entre los usos y soluciones que aporta *Microsoft Azure* se destacan su empleo como *Software as a Service* (SaaS), *Infrastructure as a Service* (IaaS) y *Platform as a Service* (PaaS); todos ellos monetizados mediante el uso de suscripciones que permiten identificar y controlar cada recurso creado dentro de la plataforma. Cada recurso designa una entidad u objeto inteligente manejado por *Azure*, como son por ejemplo las máquinas virtuales o las cuentas de almacenamiento.

Como alguien que ofrece servicios, *Microsoft* precisa de cumplir los estándares de seguridad requeridos por la industria, para lo cual creó los centros de confianza o *Trust Center*. Este centro contiene toda la información relacionada en cuanto a seguridad, confianza y privacidad para con sus servicios, con el fin de despejar las dudas de sus usuarios para con dichos aspectos.

Esta breve introducción puede encontrarse más detallada, así como una muestra de los primeros pasos en la plataforma, en el libro *Microsoft Azure. Planning, Deploying, and Managing Your Data center in the Cloud* [18].

2.2.1 Software as a Service

El Software como Servicio o SaaS, es un modelo en el cual los clientes acceden a servidores donde se da soporte al software por parte de la compañía propietaria. La empresa es la encargada del mantenimiento de dicho software, el cual puede ser empleado en cualquier dispositivo, esté o no en dicha empresa; de esto se deduce que el almacenamiento y procesamiento es llevado cabo por la empresa distribuidora.

En el caso que nos ocupa destaca la herramienta *Office 365*, la cual proporciona a los clientes de *Microsoft* acceso a su software sin necesidad de tener una infraestructura propia. Para esto, *Microsoft* llevó a cabo la construcción de centros de datos cuya gestión cayó en manos de un equipo interno especial llamado *Global Foundation Services (GFS)* [18]; no es baladí mencionar que la escalabilidad, elasticidad y fiabilidad de *Office 365* como SaaS depende en gran medida de la infraestructura de *Azure*.

2.2.2 Infrastructure as a Service

La Infraestructura como Servicio o IaaS, se define como la capacidad del consumidor de aprovisionar procesamiento, almacenamiento, redes y otros recursos informáticos fundamentales, a fin de poder desplegar y ejecutar software arbitrario, lo que incluye sistemas operativos y aplicaciones. El consumidor no gestiona ni controla la infraestructura subyacente de la nube, pero tiene control sobre los sistemas operativos, el almacenamiento y las aplicaciones desplegadas, y posiblemente un control limitado de determinados componentes de red.

A fecha de Julio de 2020 [19], de los 170 servicios contabilizados de *Azure*, 55 de ellos son IaaS. Por citar uno, tenemos el empleo de máquinas virtuales que permiten tanto el uso de

equipos cuyas especificaciones de CPU y memoria son personalizables, como de diferentes niveles de servicio y escalabilidad para proveer recursos de cómputo bajo demanda.

2.2.3 Platform as a Service

Azure ofrece multitud de soluciones es un cuanto a su como de Plataforma como Servicio o PaaS. Por ejemplo, tenemos el servicio *Azure App*, que permite soporte para *back-end* móviles, APIs *RESTful* y la habilidad para montar y alojar aplicaciones web.

Por otro lado, está el servicio de base de datos de *Azure*, servicio de bases de datos *multitenant*³ que incluye soporte de red, *Hypervisor* y un hardware proporcionado mediante la nube con su propia gestión de servicios [20].

2.2.4 Servicios de datos y *Big Data*

Cuando se emplea *Azure* como PaaS para el tratamiento de datos, se hacen uso de los llamados servicios de datos de *Azure*. Se trata de un conjunto de soluciones específicas para el tratamiento de datos con el fin abordar la replicación, manipulación y almacenamiento de los diferentes tipos de datos, a escala y de forma segura. En este campo destacan *Azure SQL Database*, *Azure Synapse* y *Azure Cosmos DB* que, si bien no son las únicas, en su correspondiente capítulo del libro *Microsoft Azure. Planning, Deploying, and Managing the Cloud* [21], podemos ver que son las más prominentes en el despliegue de proyectos en la actualidad y entender el porqué de ello.

Junto a lo dicho, y también relacionado con el *Big Data*, se pueden añadir dos apartados de *Azure* también relevantes. Por un lado tenemos el *Machine Learning* y el *Deep Learning*, donde *Azure Machine Learning Studio* lleva proporcionando desde 2015 una plataforma para el desarrollo y despliegue de modelos de este ámbito; y por el otro están los Servicios Cognitivos, los cuales, como parte importante de este trabajo, merecen una sección propia a fin de obtener una visión más exhaustiva sobre los mismos.

Azure Cognitive Services

Sobre este tema, *Microsoft* recomienda la lectura del libro *O'Reilly: Building Intelligent Apps with Cognitive APIs* [22] elaborado por miembros del equipo de la plataforma de inteligencia artificial de *Microsoft*, por ello, la información mostrada a continuación ha sido extraída de dicha fuente, que puede consultarse para obtener una visión más completa del tópico en cuestión.

³ **Multitenant** o tenencia múltiple es el principio de arquitectura de software donde una sola instancia de la aplicación se ejecuta en el servidor, pero sirviendo a múltiples clientes.

Con un crecimiento constante de sus capacidades, los servicios cognitivos de *Microsoft* permiten el empleo de los últimos avances en IA sin necesidad de desarrollar e implementar modelos propios. Como puede verse en la **Tabla 1**, los servicios disponibles se agrupan en cinco categorías: visión, lenguaje, voz, decisión y búsqueda web. A pesar de pertenecer a diferentes categorías, no existen restricciones a la hora de llamar a los diferentes servicios de forma conjunta en el desarrollo de aplicaciones.

Visión	<i>Computer Vision API</i>	La API de visión puede analizar los datos visuales y extraer textos/objetos/análisis en forma de JSON.
	<i>Emotion API</i>	Esta API puede utilizarse para identificar emociones, donde cada rostro humano tendrá también un cuadro delimitador.
	<i>Face API</i>	Esta API puede utilizarse para la detección y el reconocimiento de rostros en una imagen.
	<i>Video API</i>	Esta API permite detectar rostros y emociones en un vídeo.
Lenguaje	<i>Bing Spell Check API</i>	Esta API puede utilizarse para proporcionar correcciones gramaticales y ortográficas contextuales.
	<i>Text Analytics API</i>	Esta API puede utilizarse para realizar análisis de sentimientos a partir de un texto o extraer frases clave.
	<i>Web Language Model API</i>	Esta API puede utilizarse para construir un modelo lingüístico utilizando el corpus a escala web recopilado por Bing.
	<i>Linguistics Analysis API</i>	Esta API puede utilizarse para identificar la estructura del texto.
Voz	<i>Bing Speech API</i>	Esta API le ayuda a habilitar la entrada de voz para sus aplicaciones.
	<i>Speaker Recognition API</i>	Esta API proporciona servicios de identificación y verificación de hablantes.
Decisión	<i>Recommendations API</i>	Esta API puede utilizarse para recomendar artículos a clientes en función de su actividad y del catálogo.
	<i>Entity-Linking Intelligence Service</i>	Esta API puede utilizarse para identificar entidades dentro de un párrafo y un contexto específico.
	<i>Academic Knowledge API</i>	Esta API puede utilizarse para interpretar las consultas de búsqueda relacionadas con la intención académica.
Búsqueda Web	<i>Bing Search API</i>	Esta API proporciona capacidades de búsqueda que pueden integrarse en cualquier aplicación.
	<i>Bing Auto-Suggest API</i>	Esta API se puede utilizar para proporcionar capacidades de auto sugerencia para consultas de búsqueda a los usuarios.

Tabla 1. Características de Azure Cognitive Services [23]

Los servicios cognitivos se encuentran disponibles globalmente y amparados bajo 75 certificaciones de la industria, además de contar con un constante esfuerzo para actualizar los modelos de cada servicio a fin de permanecer al día con las necesidades de sus usuarios. Además, existe la posibilidad de emplear modelos de datos propios en aquellos entornos donde se precise, por ejemplo, para reconocer nombres técnicos específicos de un sector o nombres de productos.

A la hora de trabajar con las mencionadas APIs, se pueden emplear las diferentes *SDK* disponibles en lenguajes populares como *C#*, *Python*, *Java*, *JavaScript* y *Go*; esto permite llamar a los métodos requeridos directamente desde el código de la aplicación mediante la adición de las librerías pertinentes. Otra forma de acceder a los servicios de forma directa es la llamada de su correspondiente *REST API URLs*, lo cual permite su uso el cualquier entorno o lenguaje con el que se trabaje.

La documentación de *Azure Cognitive Services* dispone de guías para iniciarse en cada característica con códigos de aplicaciones a modo de ejemplo, tanto como material de referencia para los métodos de empleo mencionados.

Capítulo 3

Análisis

Para cumplimentar los diferentes objetivos marcados al inicio del trabajo, se precisa de una serie de herramientas tanto para el desarrollo del sistema como para la administración de colecciones digitales que se emplearán en la validación de éste, donde se comparan no sólo sus prestaciones y características, sino también la tenencia de una licencia gratuita que resulta en un aspecto a favor frente a otras alternativas.

En primer lugar, el carácter de este trabajo requiere de diferentes colecciones digitales que contengan rostros humanos para poder llevar a cabo las pertinentes operaciones de reconocimiento facial; por ello es necesaria una herramienta capaz de gestionar dichas colecciones, así como poder administrar cada imagen individual.

Seguidamente, y en común con todo proyecto de desarrollo software, la elección del entorno empleado no es una decisión banal, puesto que las características del caso que nos ocupa devienen en unos requisitos que, como se verá posteriormente, suponen diferencias suficientes para decantar la elección.

Y por último, se encuentra la API de reconocimiento facial en la cual está basado el desarrollo. En primera instancia se marcó como objetivo el análisis de la API *Computer Vision*, para posteriormente decidir el cambio a la API *Face*, hecho motivado principalmente por la mayor cantidad de información, en comparación con el anteriormente citado *Computer Vision*, que este servicio es capaz de extraer a partir de la imagen de un rostro. Tampoco está de más llevar a cabo un breve repaso a las alternativas que existen para el desarrollo de aplicaciones de la misma índole.

3.1 Base de datos

Para poder emplear el sistema, es necesario tener al menos una colección de imágenes sobre la que trabajar, la cual debe poder ser gestionada y administrada no solo como preparación de cara a realizar consultas de búsqueda, sino también para poder validar los resultados obtenidos por el programa.

Con esto en mente, la búsqueda tiene como objetivo una herramienta especializada en tratar conjuntos de imágenes que cuente con la capacidad de almacenar información junto a la imagen en forma de metadatos, con el fin de contrastar los resultados que se obtienen mediante búsquedas de reconocimiento facial y búsquedas según dichos metadatos.

Como ya se ha mencionado, otra característica que debe tenerse en cuenta sería el plan

de pago, pues para este trabajo se valora de forma positiva que posea una licencia gratuita, frente al pago mensual o por compra de licencia como único método de empleo.

3.1.1 Adobe Bridge

Tras comprobar diferentes alternativas, se tomó la decisión de emplear la herramienta *Adobe Bridge* para la gestión y administración de las colecciones digitales.

Desarrollado en 2015 por la reconocida empresa de software estadounidense *Adobe*, fue lanzado como parte de *Adobe Creative Suite* en su versión CS2, con el objetivo de conectar los componentes de *Creative Suite* bajo un mismo formato de exploración de archivos.

Las funcionalidades que la han llevado a ser escogida para el desarrollo de este proyecto van desde la capacidad de organización y edición de imágenes mediante palabras clave, etiquetas y puntuaciones, hasta las opciones de búsqueda mediante potentes filtros y funciones avanzadas de búsqueda de metadatos.

Haciendo un análisis más detallado, en primer lugar, desde la interfaz se nos permite elaborar colecciones conformadas por imágenes que no están necesariamente en el mismo directorio, además de filtrar las imágenes según qué características. Adicionalmente, se dispone de varios métodos de visualización que van desde una vista general de la imagen y su información, a otras más específicas centradas en la imagen, sus metadatos o palabras clave asociadas.

De las capacidades de edición, las funciones que resultan de mayor interés para el proyecto son las de añadir información sobre el contenido de la imagen como palabras clave, que más tarde nos permitan filtrar las imágenes según las búsquedas que queramos realizar. Por ejemplo, podemos añadir la etiqueta del nombre de una persona determinada en aquellas imágenes donde aparezca, para luego comparar los resultados llevado cabo por el sistema de reconocimiento facial.

Sus funcionalidades van más allá de lo necesario para el proyecto, como son sus exportaciones a diferentes formatos, escalación de imágenes, renombramiento por lotes o su apilamiento automático de imágenes panorámicas; además de otros sistemas de etiquetado tales como puntuación o colores para marcar estados como “pendiente de revisión”. Al ser una herramienta de *Adobe*, sus características están pensadas para poder usarse en conjunto con otras herramientas de la empresa, como son *Photoshop* o *Illustrator*.

Por último, no está de más destacar que, si bien es un programa con licencia, *Bridge* es, a diferencia de otras herramientas de *Adobe*, completamente gratuita para aquellos usuarios registrados en su servicio de *Adobe Creative Cloud*. Esta herramienta puede emplearse en dispositivos que hagan uso de las versiones actuales de *Windows* y *MacOS*.

3.1.2 Alternativas

Si bien para este proyecto se ha empleado *Adobe Bridge* para la elaboración de la base de datos, también se han sopesado otros gestores que merecen la pena de mencionar como alternativas que se manejaron, a pesar de ser descartadas por uno u otros motivos.

Kalimages

Kalimages es un software creado para catalogar imágenes e indexar fotografías por palabras clave y otras anotaciones mediante la información directa del modelo *IPTC*⁴ desde sus respectivas carpetas. Es posible introducir metadatos en la colección según la especificación *IPTC* para facilitar las actividades de catalogación, búsqueda y publicación de las imágenes.

Diseñada para versiones antiguas de *Windows* (95, 98, Me, NT, 2000, XP, 2003), *Kalimages* tiene una licencia gratuita con funciones limitadas respecto a su versión comercial, cuyo precio es de 47\$.

ThumbsPlus version 10

Thumbsplus es un gestor de fotos y gráficos para su organización y búsqueda en *Windows*. Capaz de crear miniaturas y visualizar archivos gráficos, permite la edición, conversión, búsqueda, aplicación de filtros y demás operaciones para ofrecer una solución a aquellos que tengan gran cantidad de fotos, películas, fuentes u otros activos digitales que organizar, gestionar y editar.

ThumbsPlus está construido alrededor de una base de datos relacional, que puede ser compartida a través de una red. Soporta directamente las bases de datos *Microsoft Access* (por defecto) y *SQLite*.

Disponible tanto para las versiones actuales de *Windows* como para *Windows Server*, *ThumbsPlus* es un programa de pago con una versión de prueba gratuita. Para poder hacer uso de su versión completa se necesita adquirir una de sus licencias, por ejemplo, está la licencia estándar para uso personal por 59,95\$ o la profesional por 119,95\$.

Adobe Lightroom

Al igual que *Bridge*, otra herramienta perteneciente a *Adobe* es la de *Adobe Lightroom*, la cual dispone de herramientas organizativas, como el etiquetado, para catalogar imágenes y poder encontrarlas fácilmente. Además, permite su almacenamiento en la nube a fin de poder acceder a las imágenes desde cualquier dispositivo, junto a la posibilidad de emplear un sistema de búsqueda inteligente basado en el aprendizaje automático de *Adobe Sensei*.

⁴ El *IPTC* (International Press and Telecommunications Council), es una organización internacional creada para desarrollar y promover estándares en los intercambios de datos. *IPTC* definió un modelo global de datos llamado *IPTC-NAA Information Interchange Model* (*IPTC/IIM*) aplicable a todo tipo de datos.

Al igual que *Bridge*, *Lightroom* está pensado para poder emplearse junto a otras herramientas de *Adobe* gracias a su facilidad a la hora de exportar y compartir imágenes.

Adobe Lightroom cuenta con una prueba gratuita temporal disponible tanto en *Windows* como *MacOS* y dispositivos móviles, pero para su uso completo es necesario comprar una licencia basada en un pago mensual; por ejemplo, en su edición más básica, la cual incluye *Adobe Photoshop*, el precio es de 12,09€ mensuales.

Xnview

XnView es un software dedicado a la visualización, búsqueda y conversión de más de 500 archivos de imagen compatibles. Permitiendo múltiples modos de visualización, como las miniaturas, así como comparación entre imágenes, también dispone de diversos métodos de edición y creación, ya sea la aplicación de filtros o la exportación a alguno de los más de 70 formatos diferentes. Además, incorpora el soporte y edición de metadatos en formato *IPTC*.

El programa se encuentra disponible para *Windows*, *MacOS* y *Linux*, y es totalmente gratuito para su uso doméstico, en cambio, para su uso en empresas es necesario pagar su correspondiente licencia cuyo precios ascienden a partir de los 29.00 €.

3.2 API de reconocimiento facial

Como se mencionó anteriormente, en esta contemporánea Cuarta Revolución Industrial, las propiedades intelectuales juegan un papel vital en la competitividad [1], por lo que son muchas las empresas que no dudan en invertir en investigación y desarrollo de nuevas tecnologías. Como resultado se obtienen gran cantidad de alternativas de uso para aquellos que tengan intención de emplear estas tecnologías. Mediante el empleo de APIs, entre otras formas, es posible hacer uso del producto de este arduo trabajo de estudio y desarrollo, eso sí, como propiedad intelectual que es, no se llega a conocer en profundidad su funcionamiento. Este proceso se puede observar simplificado en la *Ilustración 1*⁵.

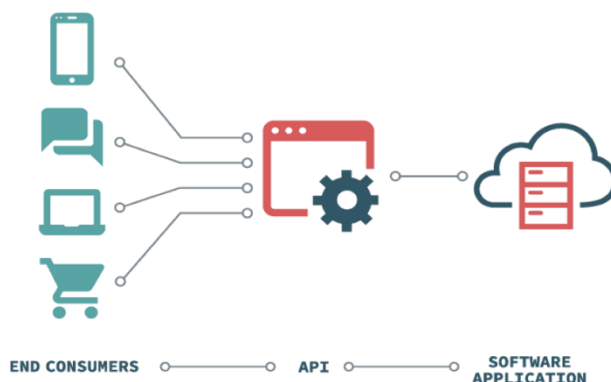


Ilustración 1. Funcionamiento de una API

⁵ Esta imagen fue obtenida de la página www.academiaweb.ca

En concreto, este trabajo tiene como finalidad estudiar y analizar el comportamiento de la API de *Microsoft Azure* para el reconocimiento facial.

Durante las fases iniciales del trabajo, se marcó como objetivo el estudio y evaluación de la API de *Microsoft Azure, Computer Vision*, para la búsqueda visual y comparación de rostros humanos, a la par que se podía evaluar técnicas alternativas. Y es precisamente en dicha etapa donde se optó por cambiar la API *Computer Vision* como núcleo del trabajo por la API *Face*, también de *Microsoft Azure*. Los motivos de esta alteración del anteproyecto original serán desgranados a las secciones de a continuación.

Junto a lo dicho, y como en el apartado anterior, no hay por qué obviar las otras alternativas que se pueden encontrar en la actualidad a la hora de desarrollar aplicaciones o sistemas que tengan el reconocimiento facial como una de sus funcionalidades.

3.2.1 Computer Vision

Perteneciente a los servicios cognitivos de *Microsoft Azure, Computer Vision* es un servicio que proporciona el acceso a algoritmos avanzados para el procesamiento de imágenes a fin de obtener información basada en las características visuales de interés. Sin necesidad de tener conocimientos de aprendizaje automático, es posible emplear los diferentes tipos de análisis disponibles listados en la *Tabla 2*.

Análisis de imagen	Analiza una imagen en busca de contenido para adultos, detección facial, etiquetado por características y colores dominantes.
Descripción de imagen	Genera una descripción de una imagen en un lenguaje entendible para humanos.
Obtener miniatura	Genera una miniatura de una imagen especificada.
Listar modelos de dominio específico	Obtiene una lista de los modelos de dominio específicos actualmente soportados.
OCR	Lleva a cabo un reconocimiento óptico de caracteres sobre una imagen y almacena el texto detectado.
Reconocer contenido de dominio específico	Analiza una imagen para recuperar contenido de dominio específico.
Reconocer escritura manual	Ejecuta un reconocimiento de escritura manual.
Etiquetación de imagen	Genera una lista de términos que son relevantes para el contenido de una imagen.

Tabla 2. Características de Computer Vision [24]

En cuanto a su desempeño a la hora de detectar rostros, Alessandro Del Sole muestra en su libro el funcionamiento de las diferentes características del servicio en una aplicación escrita en el lenguaje *c#* [25], concretamente el método de análisis de imágenes, donde es posible especificar el atributo *Faces* como una de las características visuales a

extraer. El resultado se obtiene en formato JSON, donde se indica la posición en la imagen, así como la estimación de sexo y edad, de cada rostro identificado en la imagen, junto al resto de características especificadas; como puede verse visualmente en la *Ilustración 2*⁶.



Ilustración 2. Resultado de Computer Vision API

Obviando el resto de información que se puede obtener de la imagen, tenemos como resultado una cantidad de información cuanto menos exigua del rostro, únicamente sabemos su posición y las estimaciones de sexo y edad. Esto resulta en una cantidad insuficiente de información a la hora de llevar a cabo operaciones más avanzadas de reconocimiento facial, y es que, aunque ciertamente el servicio permite extraer información variada de la imagen en sí misma, esto no guarda relación con el objeto de estudio.

Queda por decir que, como cualquier servicio de *Azure*, se garantiza la seguridad de este y su disponibilidad desde cualquier lugar gracias al procesamiento en la nube. Además de destacar la versatilidad de las soluciones que se pueden desarrollar, como por ejemplo controlar el número de personas que hay en un espacio, la distancia social y el uso de mascarillas en el contexto actual o, de forma general, se puede emplear para optimizar la distribución de tiendas y oficinas, así como acelerar procesos de pago.

En términos de precio, con su licencia gratuita las llamadas a la API están limitadas a 20 por minuto con un máximo de 5.000 mensuales para cualquier característica del servicio. Mientras, su licencia de pago limita a 10 llamadas de la API por segundo y donde según qué característica el precio ronda el euro por las mil llamadas mensuales, para disminuir progresivamente en intervalos de un millón, diez millones y cien millones de llamadas, aunque también depende de la región establecida.

⁶ Imágenes extraídas de las páginas web www.concepto.de y azure.microsoft.com/es-es/services/cognitive-services/computer-vision/

3.2.2 Face

La API Face, se muestra como una alternativa a la *Visión Artificial* o *Computer Vision* en el campo del reconocimiento facial y podría definirse como un servicio basado en la nube que ofrece los cálculos faciales más dinámicos. Se trata de una aplicación biométrica de programación especialmente dispuesta para identificar o analizar rostros humanos, diferenciando y examinando las estructuras faciales de la persona [26] y cuyo funcionamiento se muestra a grandes rasgos en la *Ilustración 3*⁷.

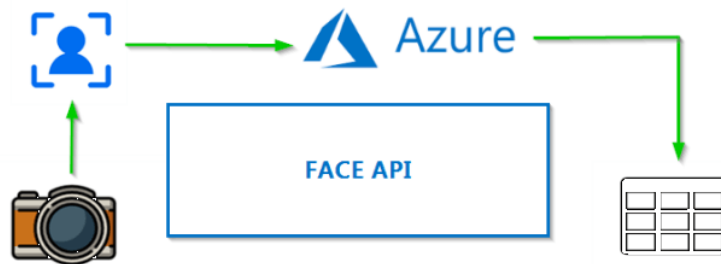


Ilustración 3. Face API en Azure

Podemos dividir sus funcionalidades principales en detección facial y reconocimiento de rostros y emociones. La primera de ellas se basa en percibir determinados lugares del rostro humano en una imagen y devolver el rectángulo dentro de la imagen que contiene dicho rostro, junto a otras características que se pueden inferir tale como los gestos, posible mediante el uso de *Machine Learning*. Estas características son: edad, exposición del rostro, género, postura, sonrisa, vello facial, detectados mediante la localización de los 27 puntos de referencia mostrados en la *Ilustración 4*⁸. Pudiendo detectar hasta 64 rostros con gran precisión en una única imagen, es posible obtener más información relacionada con el rostro, como puede ser el cabello o la presencia de gafas, a partir del área rectangular inicial mencionada.

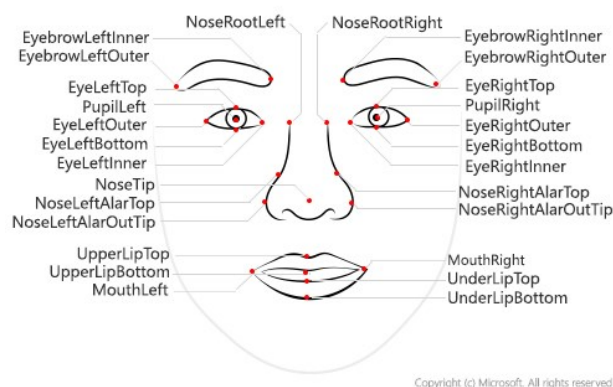


Ilustración 4. Puntos de referencia de una cara

⁷ Imagen extraída de la web www.c-sharpcorner.com

⁸ Imagen extraída de la web docs.microsoft.com/es-es/azure/cognitive-services/face/concepts/face-detection

En cuanto al reconocimiento de rostros y emociones, estamos hablando de una funcionalidad que podemos encontrar comúnmente en aplicaciones de seguridad, interfaces de usuario o en la examinación y organización de contenido digital. Las operaciones de reconocimiento emplean estructuras de datos que se almacenan en la nube y pueden ser referenciados mediante un identificador, el cual es único para cada suscripción. Las operaciones que pueden llevarse a cabo son:

- **Verificación:** esta operación toma dos identificadores diferentes y determina si pertenecen a la misma persona (ejemplo en *Ilustración 5*⁹).
- **Agrupación:** agrupación toma como entrada una matriz de identificadores y los devuelve, pero en diferentes matrices de menor tamaño a la original y agrupadas mediante aquellas caras que puedan ser similares.
- **Búsqueda de similares:** se toma un identificador y una lista de identificadores, tras lo cual se retorna una lista de menor tamaño con aquellas caras que son similares a la cara especificada.
- **Identificación:** esta operación toma uno o varios identificadores y un objeto que representa un grupo de personas, para retornar una lista de candidatos para cada rostro.



Ilustración 5. Ejemplo de verificación

Si hablamos de casos de uso, en la web de Azure podemos encontrar diversas empresas que emplean esta tecnología, no obstante, y a fin de tener idea más clara de sus posibilidades, podemos tomar como ejemplo el artículo escrito en el diario online IJETA [27] sobre el

⁹ Imagen extraída de azure.microsoft.com/es-es/services/cognitive-services/face/

desarrollo de una puerta inteligente para el hogar que, mediante el empleo de la API de *Microsoft Face*, restringe la entrada sólo a determinadas personas; si bien el resultado presenta limitaciones, se presenta una solución viable para entornos donde la seguridad es prioritaria. Para otro caso de uso donde la utilización de esta tecnología es menos común, como es el de estudio y análisis, resulta interesante ver cómo puede emplearse la API de *Face* para inferir la demografía de usuarios marginales en la red social *Twitter* [28]; donde se analizan las fotos de perfil, obviando aquellas imágenes de avatares por defecto y las que emplean imágenes de celebridades, para elaborar un entorno de trabajo que será comparado con los bancos de datos ya existentes. Como resultado se obtuvo un marco con unas predicciones que mostraron un alto nivel de acierto, junto a poner de manifiesto el cómo su filtrado de imágenes podría ser eficaz frente el problema de escasez de datos en las investigaciones sociales.

Si se busca su precio, encontramos un plan de pago muy similar al de *Computer Vision*, con un plan gratuito capaz de emplear todas las características, pero con limitaciones de 20 llamadas por minuto y 30.000 mensuales. Su plan de pago estándar, como resulta evidente, también dispone de todas las funcionalidades pero con unos límites más flexibles de 10 llamadas por segundo y con un precio menor a la unidad de euro cada mil llamadas, precio que se reduce si se llevan a cabo una cantidad de llamadas superior al millón; esta licencia también facilita el almacenamiento de caras en la nube a 0,009€ mensuales por mil rostros.

Modelos

El servicio *Face* emplea modelos de aprendizaje automático para realizar operaciones en imágenes de caras humanas. Gracias a los comentarios de los clientes y los avances en la investigación, se continúan elaborando mejoras en la precisión de estos modelos, las cuales se ofrecen como actualizaciones del modelo. En consecuencia, a la hora de emplear la API *Face* es necesario tener en cuenta qué modelo emplear para el reconocimiento y la detección, y es que, a la hora de identificar y extraer la información de los rostros de una imagen es necesario especificar qué modelos emplear en la identificación y reconocimiento de rostros.

Comenzando por el **modelo de reconocimiento**, *Azure Face* dispone de cuatro modelos hasta la fecha, los modelos *recognition_01* (publicado en 2017), *recognition_02* (publicado en 2019) y *recognition_03* (publicado en 2020) reciben soporte técnico continuo para garantizar la compatibilidad con versiones anteriores. El modelo *recognition_04* (publicado en 2021) es el modelo más preciso disponible actualmente. *Recognition_04* proporciona una precisión mejorada tanto para las comparaciones de similitudes como para las de coincidencia de personas, además mejora el reconocimiento de los usuarios inscritos que llevan mascarillas faciales (mascarillas quirúrgicas, mascarillas n95, mascarillas de tela). Para todos aquellos noveles en el uso del servicio, se recomienda emplear el modelo *recognition_04*, aunque en caso de no especificarse, las operaciones hacen uso del modelo *recognition_01*, ya que es el elegido por defecto.

En lo referente al **modelo de detección**, existen tres modelos: *detection_01*, *detection_02* (publicado en 2019) y *detection_03* (publicado en 2021). Nuevamente se recomienda emplear el modelo *detection_03*, es especial para los nuevos usuarios, pero la opción por defecto es el modelo *detection_01*. Entre sus diferencias, la más destacable sería que únicamente el modelo *detection_01* es capaz de devolver los atributos principales del rostro, mientras que el modelo *detection_02* no devuelve ningún atributo y el modelo *detection_03* solamente es capaz de retornar el atributo “*mask*”, para indicar si boca y nariz están cubiertos. Para obtener una vista más detallada de las características y diferencias entre modelos, consulte la **Tabla 3**.

<i>detection_01</i>	<i>detection_02</i>	<i>detection_03</i>
Opción predeterminada para todas las operaciones de detección de caras.	Publicado en mayo de 2019 y disponible de forma opcional en todas las operaciones de detección.	Publicado en febrero de 2021 y disponible de forma opcional en todas las operaciones de detección.
No está optimizado para caras pequeñas, de perfil o borrosas.	Precisión mejorada en caras pequeñas, de perfil o borrosas.	Mayor precisión, también en caras más pequeñas (64 x 64 píxeles) y orientaciones de caras giradas.
Devuelve los atributos principales de la cara si se especifican en la llamada de detección.	No devuelve atributos de la cara.	Devuelve el atributo " <i>mask</i> " si se especifica en la llamada de detección.
Devuelve puntos de referencia de la cara si se especifican en la llamada de detección.	No devuelve puntos de referencia de la cara.	No devuelve puntos de referencia de la cara.

Tabla 3. Modelos de detección [29]

Ética

Como se habló anteriormente en este escrito sobre la legalidad y ética de la tecnología de reconocimiento facial, se puede sacar a colación la postura de *Microsoft* sobre *Face* para con este tópico.

Al momento de redactar esta memoria, la información más reciente se trata de la negativa de *Microsoft* a vender su tecnología de reconocimiento facial a la policía estadounidense a la espera de una regulación federal, convirtiéndola en una más de entre las grandes firmas que se suman a las protestas en contra de la excesiva violencia llevada cabo por parte de las fuerzas del orden del país [30].

3.2.3 APIs alternativas para el reconocimiento facial

Como anteriormente se mencionó, al igual que el resto de las tecnologías, existen multitud de alternativas en formas de *SDK* para desarrollar aplicaciones con reconocimiento facial. En este apartado se verán de forma fugaz diversas herramientas, tanto licenciadas como *Open Source*, que podrían haberse utilizado para el mismo fin que *Azure Face*.

Face API

Alternativa *Open Source* específica para el reconocimiento facial desarrollada para ser usada en navegadores y sobre *Nodejs*. Dispone de diversos modelos para las diferentes operaciones a realizar, que incluyen detección y reconocimiento de rostros tanto como la estimación de edad, expresión y género.

Open Face

También *Open Source*, es una implementación en *Python* y *Torch* de reconocimiento facial sobre redes neuronales profundas basado en el escrito "*FaceNet: A Unified Embedding for Face Recognition and Clustering*" escrito por miembros de *Google* [31]. Es posible ejecutar la red sobre una CPU gracias a estar implementado en *Torch*.

IBM Watson Visual Recognition

Licenciada por la reconocida multinacional estadounidense *IBM*, *Watson Visual Recognition* pertenece a la plataforma en la nube *IBM Cloud* diseñado para el análisis de rostros, escenas y objetos. Con la posibilidad de elaborar un modelo propio o emplear uno preestablecido, sus características se ven sujetas a la licencia de pago a la que se esté suscrito.

Cloud Vision

Desarrollada por *Google* como parte de los servicios de su nube *Google Cloud*, esta API permite extraer valiosa información de cualquier imagen mediante el servicio *AutoML Vision* además del uso de modelos previamente entrenados para detectar emociones, interpretar texto y demás usando la *API de Vision*. El desglose en diferentes servicios de esta tecnología también implica una diferencia en el plan de pago de cada una, pero todos ellos coinciden en un precio basado en el uso de los recursos.

Amazon Rekognition

Amazon Rekognition, como su nombre indica, está desarrollada por la compañía *Amazon*, y se caracteriza por proporcionar análisis de contenido en imágenes y vídeos, así como análisis faciales de alta precisión y capacidades de búsqueda facial para detectar, estudiar y comparar rostros. El uso de las etiquetas personalizadas le permite al usuario utilizar la tecnología sin necesidad de experiencia en aprendizaje automático, ya que éstas se encargan del trabajo de modelado. Su precio varía según su uso y se clasifica en *Amazon Rekognition Image*, *Amazon Rekognition Video* y etiquetas personalizadas de *Amazon Rekognition*.

3.3 Entorno de desarrollo

Como es común para cualquier API actual, la API *Face* se encuentra disponible en múltiples lenguajes de programación para el desarrollo de aplicaciones con reconocimiento facial. Estos lenguajes no son otros que *Java*, *Python*, *Go* y *C#*, a los que se añaden la posibilidad de emplear el entorno *Nodejs* y la API de *REST*.

Por motivos que se explicaran posteriormente, la decisión final fue el desarrollo del sistema empleando el lenguaje de programación *C#*, el cual se define como un lenguaje multiparadigma desarrollado por la propia *Microsoft* como parte de su plataforma *.NET*¹⁰. *C#* ha sido aprobado como un estándar por la *ECMA* (ECMA-334) e *ISO* (ISO/IEC 23270) y cuenta con una sintaxis básica que deriva de *C/C++* y utiliza el modelo de objetos de la plataforma *.NET*, similar al de *Java*.

Teniendo en cuenta tanto el lenguaje de programación como las características propias del proyecto, se precisa de un entorno de trabajo que facilite el desarrollo del sistema mientras se ajusta las necesidades de este.

3.3.1 Visual Studio

La respuesta es acorde a la recomendación de la propia *Microsoft*, que no es otro que el IDE¹¹ desarrollado por la misma empresa, *Visual Studio*. La principal característica que motiva su uso, aparte del hecho de que en la documentación de *Face* sea usado como referencia a la hora de mostrar ejemplos de uso, es que está especialmente preparado para trabajar con los servicios de *Azure*, así como otras aplicaciones habilitadas para la nube; para ser más específico, la API de *Face* se encuentra fácilmente accesible como un paquete *NuGet*¹² para descargar desde el gestor de paquetes del IDE.

¹⁰ *.NET* es un framework de *Microsoft* que hace un énfasis en la transparencia de redes, con independencia de plataforma de hardware y que permite un rápido desarrollo de aplicaciones.

¹¹ Un entorno de desarrollo integrado, en inglés *Integrated Development Environment (IDE)*, es una aplicación que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.

¹² Mecanismo que define cómo se crean, hospedan y consumen paquetes en *.NET*

Capítulo 4

Diseño

Como objetivo principal de este trabajo, se precisa la elaboración de un sistema que permita al usuario llevar a cabo una serie de consultas sobre una colección digital para valorar los resultados y el rendimiento de la API; esto implica que se deben recabar imágenes que tengan relación entre sí, tanto como decidir el cómo serán etiquetadas en la base de datos, para poder contrastar los resultados obtenidos. Tras la recogida de datos, se procede a la elaboración del sistema como tal, con unos requisitos que casi podríamos afirmar que inducen al empleo del patrón Modelo-Vista-Controlador (MVC) para su elaboración, el cual está basado en la división de las partes que conforman una aplicación en modelo, vistas y controladores, permitiendo así la implementación independiente de cada elemento, facilitando la actualización y el mantenimiento del software de manera sencilla [32].

Con este patrón en mente, el funcionamiento básico del programa consta de las siguientes etapas: selección de un tipo de consulta desde la interfaz, selección de la colección o rostro pertinente según corresponda, procesamiento de la entrada en base a la consulta seleccionada y la posterior muestra de resultados. En aquellos casos donde se toma un rostro particular como entrada, se decidió emplear como destino la búsqueda, la colección a la que pertenece dicho rostro.

También es necesario mejorar la definición de las consultas disponibles en el sistema de cara a su desarrollo, para lo cual, y partiendo de lo definido en los objetivos del trabajo, obtenemos lo siguiente:

- Consultas basadas en las características faciales (sexo, edad, cabello, emoción, etc.), pudiéndose llevar a cabo de forma individual o de forma conjunta, donde se busca una persona que cumpla dos más requisitos especificados, por ejemplo, mujeres con gafas de lectura y de pelo castaño o rubio.
- Búsqueda de aquellas imágenes donde aparezca una persona específica.
- Búsqueda de aquellas imágenes donde aparezcan personas similares a otra en específico.
- Búsqueda y muestra de aquellas personas que aparecen junto a una persona en concreto, junto a su frecuencia con respecto al total de apariciones. El resultado es un listado donde se muestran aquellas personas que compartan imagen con el individuo especificado, añadiendo su frecuencia de aparición, para poder saber quiénes le acompañan con mayor asiduidad.

Junto a esto, y con propósito de realizar pruebas para su evaluación, se añade la posibilidad de seleccionar los modelos de detección y reconocimiento antes de cada búsqueda, además de guardar la actividad de los procesos en registros donde se muestren los pasos que

se han llevado a cabo y sus respectivas duraciones. Y como lo permite la API, se podrá ajustar el índice mínimo de confianza que determinará cuando dos rostros pertenecen al mismo individuo. A fin de reducir el número de comparaciones, y por ende de llamadas a la API, se optimizan los procesos correspondientes mediante operaciones de filtrado, a fin de eliminar la necesidad de comparar rostros con una similitud menor a un umbral determinado.

En este capítulo se describen los desarrollos de la base de datos, arquitectura, elementos a emplear de la API, la interfaz y la implementación del sistema, así como los principales inconvenientes encontrados durante el desarrollo de estas etapas, donde estos últimos no guardan relación con los resultados obtenidos en la fase de evaluación.

4.1 Base de datos

Como ya se ha mencionado, y con motivo de evaluar el funcionamiento de la aplicación, es necesaria la elaboración de una base de datos con colecciones de entornos reales. Para ello, se recabaron imágenes clasificadas dentro de cuatro colecciones de diferentes tamaños: una primera colección basada en los actuales miembros del Gobierno de Canarias, una segunda con diferentes imágenes perteneciente a la liga de baloncesto profesional española, la ACB, y una tercera con fotografías pertenecientes a los Juegos Olímpicos de Río de Janeiro de 2016. La cuarta colección se trata de una variación de la primera.

Como primera colección tenemos alrededor de 70 imágenes enfocadas en los diferentes miembros del Gobierno de Canarias, con imágenes que van desde actos hasta ruedas de prensa, pasando por otras actividades de promoción. Para esta colección se han etiquetado tanto a los miembros principales del gabinete, como otras características que pueden extraerse de los presentes, como la presencia de gafas, sexo de los presentes, vello facial y demás. La cuarta colección comparte temática, pero se diferencia en la presencia de mascarillas en casi la totalidad de los rostros; en total cuenta con 100 imágenes.

La segunda, contabilizada en 280 imágenes, se compone principalmente de instantáneas realizadas durante los encuentros pertenecientes a la liga ACB 2020/2021, donde están etiquetados los jugadores presentes en las mismas, así como el entrenador de cada equipo.

En la última y de mayor tamaño, 1080 imágenes para ser exactos, encontramos las fotografías llevadas a cabo durante los numerosos eventos deportivos que se dieron lugar en la metrópolis brasileña. En esta ocasión tenemos etiquetados a diversas figuras de renombre que participaron en los Juegos, así como otras características únicas que no se dan lugar en el resto de las colecciones, como es la presencia de gafas de buceo.

Empleando la herramienta *Bridge* para ello, se nos permite realizar búsquedas sobre este etiquetado que hemos realizado, no solo desde la aplicación sino incluso desde el explorador de archivos del sistema operativo *Windows*. Estas búsquedas no guardarán relación con las

llevadas a cabo por el sistema más allá de la comparativa entre ambas, para esclarecer el tanto porcentaje de acierto y la detección de falsos positivos. Podemos comprobar en la **Ilustración 6**¹³ cómo queda reflejada la información de etiquetado en el programa de gestión, junto al resto de datos propios de la imagen.



Ilustración 6. Ejemplo de imagen etiquetada en Bridge.

La herramienta Bridge como tal no lo exige, sin embargo, todas las colecciones se han organizado en sus correspondientes directorios, ya que así es como serán tratadas por el sistema a la hora de su procesamiento.

4.2 Arquitectura

Como puede apreciarse en la **Ilustración 7**, el sistema lo conforman una primera parte de procesamiento local, compuesto por los elementos desarrollados empleando el patrón MVC, y una segunda donde el procesamiento, API mediante, se lleva a cabo de forma remota.

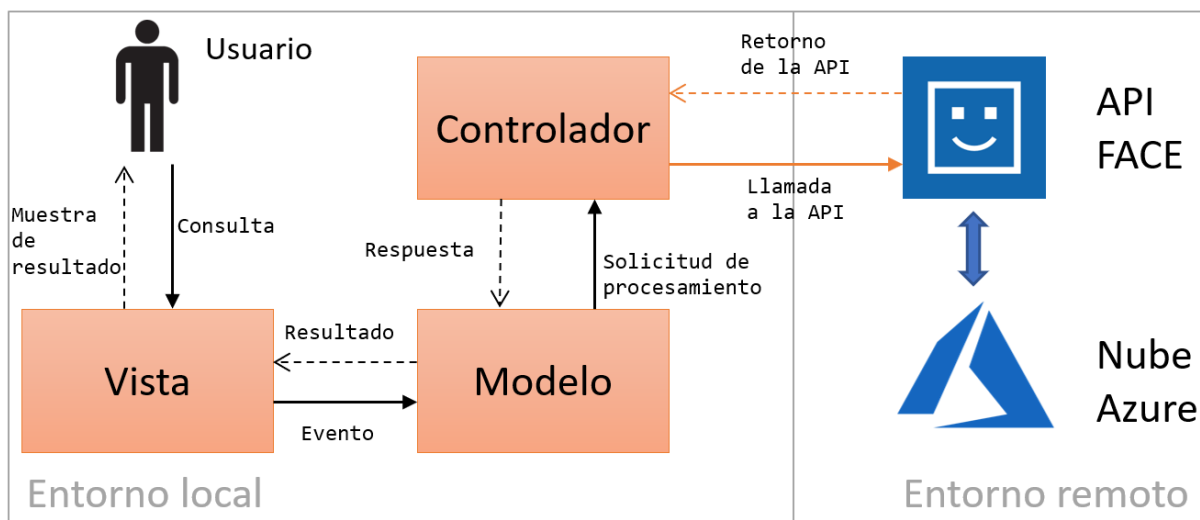


Ilustración 7. Arquitectura del sistema

¹³ Toda imagen en la que aparezcan integrantes del Gobierno de Canarias, a excepción de las ilustraciones 15 y 17, pertenece a la página oficial de Facebook del organismo, www.facebook.com/PRES.Gobcan/photos

Todas aquellas tareas donde se procesa el contenido de la imagen así como las operaciones de reconocimiento facial, como son llevadas a cabo por la API, se ejecutan en el servicio de la nube de Azure y, por ende, en un entorno remoto. El resto de los procesos del sistema, que van desde la organización y muestra de resultados hasta la selección y lectura de las diferentes colecciones digitales, se llevan a cabo de forma local.

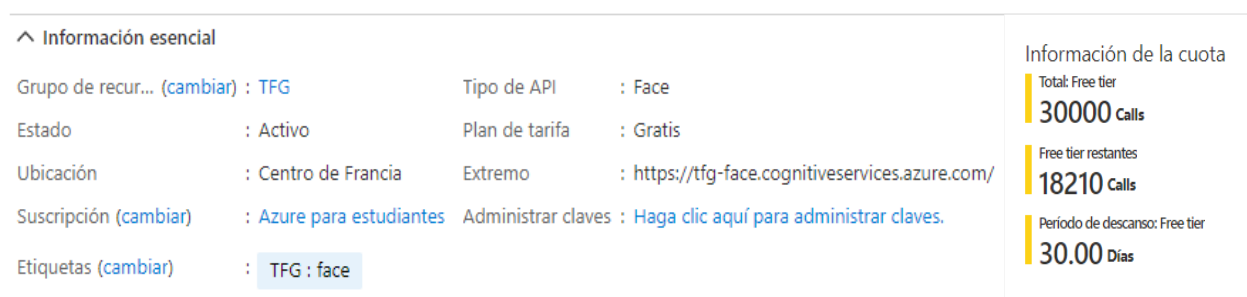
4.3 Uso de la API *Face*

En este apartado se verán los fundamentos del funcionamiento de la API *Face* mediante el empleo de la *SDK* de *c#* en el entorno de desarrollo de *Visual Studio*, así como sus estructuras de datos propias y demás funciones que se requerirán a lo largo del desarrollo del sistema.

4.3.1 Modo de empleo

La API como tal no se encarga de llevar a cabo las operaciones de reconocimiento facial de las que dispone el servicio, sino que actúa como intermediario entre el sistema local y un recurso externo alojado en la nube de *Azure*. Esto implica que únicamente instalando el paquete correspondiente no es posible comenzar a utilizar esta tecnología, primero se necesita la creación de un recurso en la plataforma *Azure*.

Sin entrar en detalles de este proceso de creación, dentro de *Azure* se necesita instanciar un recurso de *Cognitive Services* que emplee la API *Face*. Tras seleccionar diferentes aspectos como la región, tarifa y grupo de recursos, obtendríamos una vista similar a la de la *Ilustración 8*¹⁴. Es en esta vista del recurso donde obtendremos la **clave** y **punto de conexión** empleados en el proceso de autenticación de la API desde el lado del cliente, que para nuestro caso sería el sistema de catalogación.



^ Información esencial	
Grupo de recur... (cambiar) : TFG	Tipo de API : Face
Estado : Activo	Plan de tarifa : Gratis
Ubicación : Centro de Francia	Extremo : https://tfg-face.cognitiveservices.azure.com/
Suscripción (cambiar) : Azure para estudiantes	Administrar claves : Haga clic aquí para administrar claves.
Etiquetas (cambiar) : TFG : face	

Información de la cuota

Total: Free tier
30000 Calls

Free tier restantes
18210 Calls

Periodo de descanso: Free tier
30.00 Dias

Ilustración 8. Información de recurso Azure

¹⁴ Imagen perteneciente al sitio web de Microsoft Azure www.portal.azure.com, precisa de iniciar sesión y la creación de un recurso para obtener esta vista

4.3.2 Estructuras de datos

El empleo de la API conlleva el uso de unas estructuras de datos propias del servicio, por ello se explicarán aquellas que necesarias para el desarrollo del sistema además de realizar un aparte para los datos de entrada.

En primer lugar, se encuentra *DetectedFace*, una representación facial única obtenida en el proceso de detección de rostros. Como atributos principales tenemos el identificador, una cadena que actúa como una identificación única para cada rostro, y los atributos, que corresponden a las características que pueden extraerse del rostro de manera opcional. Las características empleadas por el sistema se encuentran en *Tabla 4*.

Edad	Edad estimada en años de una cara determinada.
Emoción	Lista de emociones con una confianza de detección para la cara determinada. Las puntuaciones de confianza están normalizadas y las puntuaciones de todas las emociones suman uno.
Vello Facial	Presencia estimada de vello facial y longitud de la cara determinada.
Sexo	Sexo estimado de la cara determinada.
Gafas	Si la cara determinada tiene gafas.
Cabello	Tipo de pelo de la cara. Este atributo muestra si el pelo está visible, así como la calvicie y el color de pelo que se detecten.
Maquillaje	Si la cara está maquillada.
Sonrisa	Expresión sonriente de la cara determinada.

Tabla 4. Atributos faciales detectados por el sistema.

Seguidamente tenemos *VerifyResult*, que, como su nombre indica, es la respuesta a una operación de verificación. Está compuesto por un valor correspondido entre cero y uno que representa la confianza de similitud en que dos rostros pertenezcan a la misma persona, además de un valor de tipo booleano, que será verdadero por defecto cuando la confianza sea igual o superior a 0,5.

Similar al anterior, está el *SimilarFace*, que corresponde a la respuesta dada por la operación de búsqueda de personas similares para un determinado rostro. Sus atributos son un valor de confianza de similitud nuevamente limitado entre 0 y 1, y el identificador único del rostro objetivo de la comparación.

Datos de entrada

La API *Face* no acepta cualquier tipo de archivo de imagen como entrada, sino que está limitado a los archivos *JPEG*, *PNG*, *GIF* (el primer fotograma) y *BMP*. Además, se recomienda que dichas imágenes tengan un tamaño no mayor a 6 megabytes, lo cual no es un requisito en sí, pues acepta imágenes de tamaños superiores, pero los resultados obtenidos serán mejores si se cumple dicha recomendación.

Como apunte, para toda imagen con una resolución menor a 1920 x 1080 píxeles, el tamaño de rostro mínimo detectable es de 36 x 36 píxeles, el cual aumenta proporcionalmente al tamaño de la imagen a partir de la resolución citada hasta un máximo de 4096 x 4096. Las caras fuera del intervalo de tamaño de 36 x 36 a 4096 x 4096 píxeles no se detectarán.

Las operaciones de detección y reconocimiento podrían verse afectadas por los desafíos técnicos presentes en las imágenes de entrada, que incluyen casos tales como la iluminación extrema y los obstáculos que bloqueen los rostros de forma parcial.

4.3.3 Funciones

Antes de explicar los procesos del sistema, es necesario conocer cuáles son las llamadas de la API implicadas en los mismos. Con esto en mente, a continuación se explican los métodos, parámetros y retornos de aquellas llamadas que serán mencionadas en el pertinente apartado de **Sub procesos**.

Comenzamos con *Authenticate*, llamada mediante la cual se obtiene como resultado un objeto cliente autenticado en el servicio de *Azure*. Como parámetros solicita un *endpoint*¹⁵ en formato de cadena, que contiene la URL a la instancia de *Face* dentro de *Azure*, así como la clave de acceso para la instancia, también en formato de cadena.

Seguidamente, tenemos la llamada *DetectWithStreamAsync*, que permite la detección de rostros en una imagen local de forma asíncrona. Este proceso tiene su variante para procesar imágenes mediante URL.

- Parámetros: de todos los parámetros que permite, en nuestro caso solo necesitamos nombrar cuatro. Primero está el flujo de la imagen de la imagen a analizar. En segundo lugar, y de forma optativa, el listado de características que queremos extraer del rostro, omitir este parámetro no devolverá ninguna de las características opcionales. En tercer y cuarto lugar está el modelo de detección y reconocimiento respectivamente, se puede realizar cualquier combinación de modelos que se precise, no obstante, cuando se quieran extraer las características adicionales, el modelo de detección ha de ser obligatoriamente el modelo *detection_01*.

¹⁵ Los **endpoints** son las URLs de un API o un backend que responden a una petición.

- Retorno: se obtiene una lista de objetos *DetectedFace* para cada uno de los rostros detectados en la imagen. En caso de que se hayan solicitado, se encontrarán disponibles para su acceso los atributos faciales.

Luego encontramos *Verify*, método para comparar si dos rostros pertenecen a la misma persona. La calidad de imagen y del rostro (imagen frontal, sin obstáculos, etc.) mejorará la calidad y precisión del proceso.

- Parámetros: como entrada se toman los identificadores de los rostros a comparar.
- Retorno: se devuelve un único objeto *VerifyResult* con el resultado de la operación.

Por último, tenemos *FindSimilarAsync*, que se emplea para comparar la similitud entre un rostro o grupo de ellos para con un rostro específico.

- Parámetros: de todos los disponibles, nosotros precisamos de tres parámetros de entrada. En primer y segundo lugar respectivamente están el identificador del rostro objetivo y una lista de los identificadores de los rostros a comparar. Y el tercero es el modo de trabajo, que tiene dos valores posibles:
 - *matchPerson*: este modo emplea unos umbrales internos para detectar si dos rostros pertenecen a la misma persona, pero no con la misma precisión que tendría el proceso de *Verify*. Es el modo por defecto y es posible la devolución de una lista vacía si ningún rostro supera los umbrales.
 - *matchFace*: ignora los umbrales mencionados para devolver una lista completa de todos los rostros ordenados según su similitud, incluso si ésta es baja.
- Retorno: una lista de objetos *SimilarFace*, cuyo número total dependerá del modo de trabajo especificado.

Para las llamadas de verificación y comparación de rostros similares, el modelo de reconocimiento empleado durante la detección debería ser el mismo para los objetivos de la comparación.

4.4 Interfaz de Usuario

Representando a la parte de Vista dentro del patrón MVC, el funcionamiento del sistema conlleva el uso de diferentes interfaces que permiten al usuario interactuar con el mismo a

fin de poder realizar las consultas, así como vislumbrar los resultados de éstas. A continuación, veremos cada una de las vistas que conforman el sistema junto a la explicación de su papel dentro del mismo.

En este apartado no se incluyen las interfaces propias del sistema operativo que se dan como resultado de diversas operaciones tales como la selección de archivos o directorios, debido a que éstas no son vistas personalizadas, sino interfaces predefinidas del sistema operativo.

4.4.1 Vista principal

Esta interfaz representa en núcleo sobre el cual se sustentan el resto de las vistas, pues en ella se encuentran todos los accionadores de eventos que dan lugar a la creación de resto de interfaces, además de ser el contenedor sobre el que se muestran todas las vistas de resultados.

Como se observa en la *Ilustración 9*, la vista se divide en tres secciones:

- Sección superior: un menú que contiene diversas categorías desplegadas, donde cada una invoca una consulta diferente definida por su propio nombre. En el caso de consultas con registro, y de forma adicional, se dan las opciones de activar o no las operaciones de filtrado y abrir el directorio donde se almacenan los registros.
- Sección intermedia: la parte central será aquella que contenga a las vistas de resultados.
- Sección inferior: junto a una barra de progreso y un mensaje del estado actual de las operaciones, se encuentran las opciones de modificación del valor mínimo de confianza en las operaciones de verificación junto a los modificadores de modelo de detección y reconocimiento.

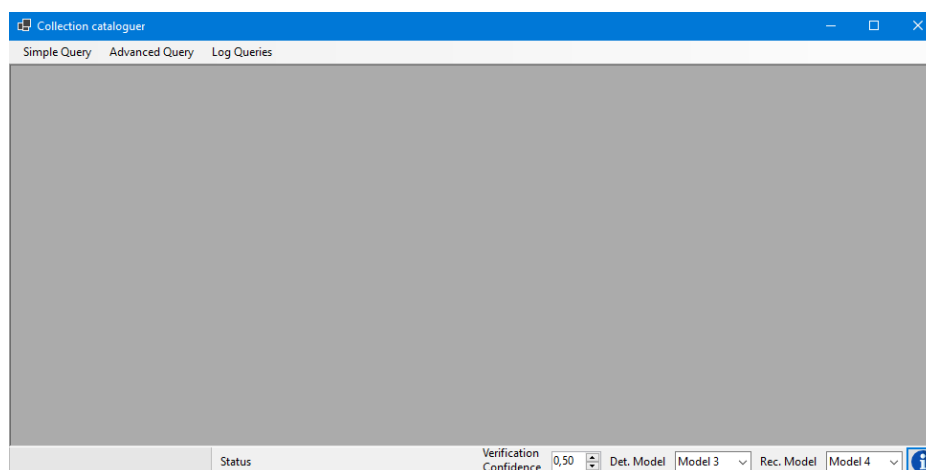


Ilustración 9. Vista Principal

4.4.2 Vista de consulta personalizada

El usuario puede realizar consultas simples de características desde el menú, donde se buscarán personas que cumplan un requisito específico, por ejemplo, imágenes con una persona sonriendo; pero, para aquellas consultas donde se busque una persona que cumpla no uno sino varios requisitos, el usuario deberá emplear una interfaz específica para este fin.

Como se aprecia en la *Ilustración 10*, esta vista no es otra cosa que selector de aquellos rasgos o atributos faciales que se desean buscar. El sistema buscará una persona que cumpla, para cada atributo seleccionado, al menos una de las características, además, se mostrará un aviso al usuario cuando cometa algún error en la selección de características, cuyas posibilidades son elegir un tipo de atributo, pero no seleccionar ninguna de las opciones disponibles, o introducir erróneamente el intervalo en la sección de edad.

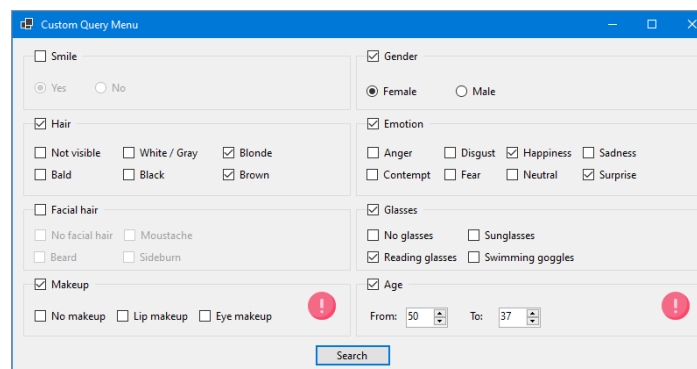


Ilustración 10. Menú de consulta personalizada

4.4.3 Vista de selección de rostro

Esta interfaz, mostrada en la *Ilustración 11*, se emplea únicamente en las consultas donde se precisa la selección de un rostro en concreto y el usuario ha seleccionado una imagen donde aparecen dos o más de ellos. Con un menú desplegable y la remarcación del rectángulo que contiene al rostro a modo de retroalimentación, esta interfaz insta al usuario especificar el rostro a buscar dentro de la imagen antes de comenzar los procesos pertinentes. Para las imágenes con un único rostro, esta interfaz se omite al seleccionar ese rostro como el objetivo de la búsqueda.

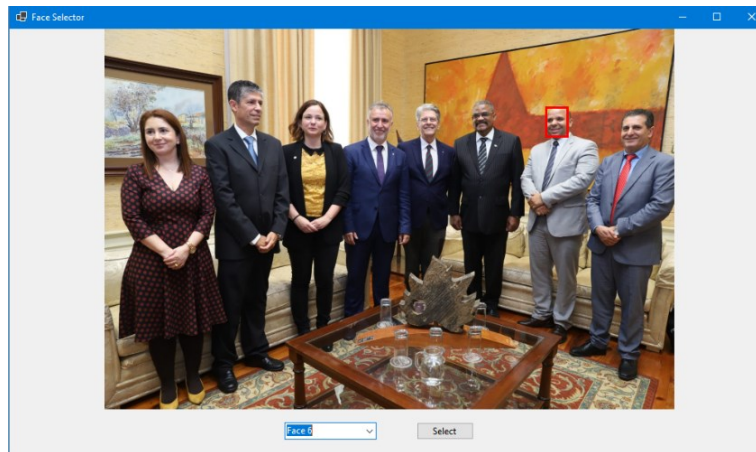


Ilustración 11. Vista de selección de rostro

4.4.4 Vista de galería

Interfaz para mostrar los resultados obtenidos de todas las consultas a excepción de la búsqueda de acompañantes. Tal y se puede observar en la *Ilustración 12*, es una galería con la que se puede interactuar mediante diferentes botones que permiten pasar entre las imágenes, eliminar la imagen actual de la galería, que no de la colección original, y resaltar aquel rostro que hizo que el sistema diera por válida la imagen. Se puede destacar que no es una vista escalable, es decir, su tamaño es fijo, y le presencia de un índice que indica el número de la imagen dentro del total de la galería, siendo este no editable.



Ilustración 12. Galería de resultados

4.4.5 Vista de galería de acompañantes

Esta última interfaz se usa de forma exclusiva para los resultados de las consultas de búsqueda de acompañantes. En primer lugar, se encuentra un recorte del rostro de aquella persona cuyos acompañantes estamos buscando, entendiendo acompañantes como persona que comparten imagen con dicha persona, y como segundo y último elemento un listado con los rostros de estos acompañantes junto a su frecuencia de aparición correspondiente. Esta

vista puede apreciarse en la *Ilustración 13*, que se realizó empleando una colección de tamaño reducido.

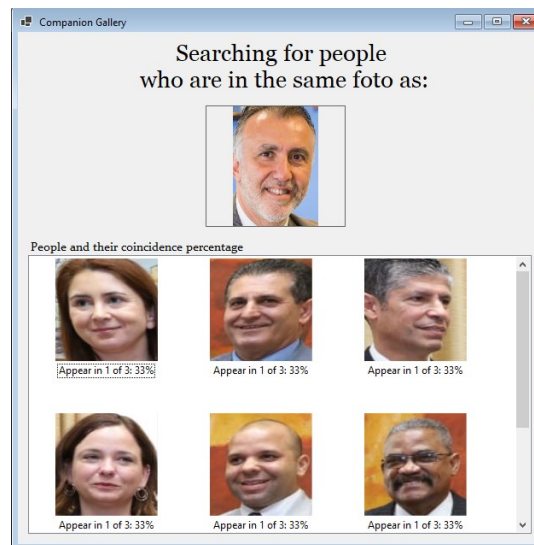


Ilustración 13. Galería de acompañantes

4.5 Sub procesos

Los procesos de detección y reconocimiento implicados en las consultas, de forma general, implican la captura del evento para la consulta seleccionada, la entrada de datos, su procesamiento y la posterior muestra de los resultados. Dentro del sistema estas fases pueden dividirse en los siguientes subprocesos: adquisición y preparado de datos, preprocesado, procesado y muestra de resultados; estos pasos varían según la consulta. En los siguientes apartados se explicarán las tareas llevadas a cabo por estos subprocesos según qué consulta, así como la lógica de su implementación dentro del sistema.

Pero antes es necesario mencionar el cómo se crea el cliente que llevará a cabo las llamadas de la API. Al iniciar el sistema se lleva a cabo la autenticación del cliente para su conexión con el recurso en la nube *Azure* mediante un *endpoint* y clave determinados, estos valores se han almacenado como variables de entorno del sistema operativo siguiendo la recomendación de *Microsoft* para el desarrollo de aplicaciones. Si bien es un procedimiento necesario para poder emplear la API, este proceso no entraña mayor lógica que la explicada y es común para todo proyecto que pretenda emplear la tecnología de *Face*, salvo por las diferencias que pueda haber en cuanto al modo de almacenar los valores citados anteriormente.

4.5.1 Consulta simple y personalizada

Este tipo de consultas tienen un gestor de eventos independiente para manejar todas las peticiones de esta índole, las cuales se pueden observar en la *Tabla 5*.

Consulta	Valores posibles
Sonrisa (<i>smile</i>)	Sí o no.
Cabello (<i>hair</i>)	Oculto (por ejemplo, por un gorro) calvo, blanco/gris, negro, castaño o rubio.
Gafas (<i>glasses</i>)	Gafas de lectura, de sol, de buceo o sin gafas.
Emoción (<i>emotion</i>)	Enfado, desprecio, asco, felicidad, miedo, neutral, tristeza o sorpresa.
Género (<i>gender</i>)	Femenino o masculino.
Maquillaje (<i>makeup</i>)	Maquillaje de ojos, de labios o sin maquillaje.
Bello facial (<i>facial hair</i>)	Sin bello facial, bigote, barba o patillas.
Edad (<i>age</i>)	Intervalos 0-25, 26-50, 51-75, 76-100 o más de 100.
Personalizada (<i>custom</i>)	Cualquier combinación de los anteriores.

Tabla 5. Tipos de consultas simples.

Como la consulta personalizada engloba los métodos empleados en las consultas simples, esta sección se centra en dicha petición.

Adquisición y preparado de datos

Tras seleccionar esta consulta, se le solicita al usuario la selección de la correspondiente colección sobre la cual se llevará a cabo la búsqueda, lo cual se hace mediante el objeto *FolderBrowserDialog*, que emplea el selector de directorios del explorador de archivos de *Windows* y cuya ruta por defecto será el directorio “*Mis imágenes*”. Una vez obtenida la ruta de la colección, se llama al siguiente método:

```
List<String> GetImagesFromCollection(string route)
```

Este método analizará la ruta mandada como parámetro para retornar una lista de todos los archivos que tengan alguna de las siguientes extensiones: *.png*, *.jpg*, *.jpeg*, y *.bmp*. No se tendrán en cuenta todo archivo cuya extensión no coincida con alguna de las citadas.

Seguidamente, se le solicita al usuario, mediante su correspondiente interfaz, la introducción de los atributos a buscar, con lo cual se consiguen todos los datos necesarios para comenzar el procedimiento.

Preprocesamiento

Esta fase de preparación para el procesamiento de las imágenes se dedica a elaborar, a partir de los requisitos de búsqueda especificados, una lista de los métodos de verificación que serán necesarios durante el proceso, junto a una lista de dichos atributos de forma codificada. Además de crear la propia lista de objetos *FaceAttributes* con los atributos a retornar.

```
List<Func<FaceAttributes, string, bool>> searchProcedures = new
List<Func<FaceAttributes, string, bool>>();

List<string> searchParameters = new List<string>();
```

También se crean dos listas, una que contendrá la ruta de cada imagen verificada junto a otra con los rostros que se han validado en cada imagen. Ahora se explicará en qué consisten los métodos de verificación mencionados.

Función de comprobación

Estas funciones, una para característica extraíble, toma como entrada uno de los atributos del rostro como objeto *FaceAttributes*, junto a una cadena de texto que contiene aquella característica o característica que buscamos, para retornar verdadero en caso de que el rostro cumpla los requisitos, o falso en caso contrario. Su estructura es la siguiente:

```
bool checkGlasses(FaceAttributes attribute, string glassesType)
```

Para este caso, donde se comprueba la presencia de gafas, la cadena codificada es una ristra de ceros y unos que, dependiendo de su posición, hace referencia a un tipo de valor u otro, con un uno si esa característica se busca o si se ignora en caso de que sea un cero. Esta codificación se repite para el resto de las características, a excepción de sonrisa, sexo y edad, que, o bien expresan uno de los dos únicos valores posibles, o un rango numérico.

Procesamiento

El procesamiento de las imágenes consiste en la iteración de cada una de las imágenes de la colección en dos fases: detección y comprobación.

Detección

Esta primera fase implica la invocación de la clase Modelo para que éste realice la correspondiente llamada a la API. Podemos ver a continuación dicho método, donde se han omitido las secuencias *try catch* para mejorar la legibilidad.:

```
public async Task<IList<DetectedFace>> DetectFaceExtract(String imageRoute, IList<FaceAttributeType>
faceAttributes)
{
    IList<DetectedFace> detectedFaces = null;

    using (Stream imageFileStream = File.OpenRead(imageRoute))
    {
        detectedFaces = await Client.Face.DetectWithStreamAsync(imageFileStream,
            returnFaceAttributes: faceAttributes,
```

```

        detectionModel: DetectionModel.Detection01,
        recognitionModel: SelectedRecognitionModel
    );
}
return detectedFaces;
}

```

Como se puede observar, este método toma como entrada la ruta de una imagen y una lista de atributos faciales para llamar a la función de la API *DetectWithStreamAsync*, con los siguientes parámetros:

- El flujo abierto para leer el archivo de imagen.
- La lista de atributos que serán devueltos.
- El modelo de detección uno, ya que es el único modelo que devuelve atributos.
- El modelo de reconocimiento que se ha establecido desde la vista principal.

Este proceso retornará todos los rostros detectados en la imagen, con sus correspondientes atributos, en forma de lista.

Comprobación

Esta segunda fase consiste en una iteración a través de cada uno de los rostros detectados en la fase anterior. Para cada rostro, un bucle lleva ejecuta todos los métodos de comprobación contenidos en la lista *searchProcedures* con su correspondiente cadena de *searchParameters*. Si todas las condiciones retornan una respuesta positiva, esta imagen se añade a la lista de resultados positivos.

Muestra de resultados

En este punto, el sistema ha elaborado ambas listas que contienen los resultados de la colección, con lo que solo resta enviar estos resultados a la vista principal para ser mostrados de la siguiente manera:

```

void CreateGalleryChild(List<Image> imageList, IList<DetectedFace> faceList, String title)
{
    if (imageList.Count > 0)
    {
        ViewGallery newMDIChild = new ViewGallery(imageList, faceList, title);
        newMDIChild.MdiParent = this;
        newMDIChild.Show();
    }
    else
        MessageBox.Show(
criteria",
            "The programm couldn't found any coincidence in the collection with the specified
            "No coincidence found"
        );
}

```

Esta llamada, perteneciente vista principal, crea una nueva vista galería que actuará como “hija” de ésta. Para ello precisa como parámetros una lista de objetos *Image*, una lista de los rostros identificados por cada imagen y el título para la galería. Los rostros se emplearán para que el usuario pueda ver qué rostro se validó en cada imagen. En caso de que la lista de imágenes esté vacía, se le notificará al usuario de que no se encontró ninguna imagen en la colección que cumpliera los criterios especificados.

Desde el controlador, y antes de ejecutarse el método *CreateGalleryChild*, se llama a la función *GetImageList*, que convertirá la lista de rutas de imágenes una lista de objetos *Image*.

```
List<Image> GetImageList(List<String> routeList)
```

Tras la creación de la galería con las imágenes resultado, se da por terminado el proceso de la consulta.

4.5.2 Búsqueda de una persona determinada

Perteneciente a las denominadas “consultas avanzadas” dentro de la vista principal, y como tal, compartiendo gestor con el resto de las consultas de la misma categoría, este procedimiento puede resumirse en los siguientes pasos: selección de un rostro y adquisición de imágenes, verificación facial y muestra de resultados.

Adquisición y preparado de datos

La elección de esta consulta conlleva la selección una imagen mediante el objeto *OpenFileDialog*, dicha imagen contendrá el rostro de aquella persona que se desea buscar. Para las imágenes donde hay más de un rostro, se seleccionará el rostro específico mediante la interfaz específica para este fin. Como ya se ha mencionado, la colección donde se realizará la búsqueda en la misma colección a la que pertenece la imagen escogida, cuyas imágenes serán recabadas mediante *GetImagesFromCollection*.

Una vez seleccionado el rostro se procederá a llamar al método correspondiente de la consulta.

```
async Task SearchPerson(List<String> imageList, string initialImageRoute, DetectedFace faceToVerify)
```

Este método toma como entrada la lista de imágenes de la colección, junto al rostro que se desea buscar y la imagen en que se encuentra para llevar a cabo la consulta.

Preprocesamiento

Al igual que para las consultas las consultas personalizadas (consultar *Preprocesamiento*), se crean una lista de resultados para las rutas de las imágenes y para los rostros, pero con la particularidad de que la imagen seleccionada en el inicio se extrae de la lista de imágenes de la colección y se añade a la lista de resultados junto al rostro objetivo de las comparaciones, a fin de ahorrarnos una comparación cuyo resultado conocemos de antemano.


```

List<String> resultImageList = new List<String>();
IList<DetectedFace> resultFaceList = new List<DetectedFace>();

imageList.Remove(initialImageRoute);
resultImageList.Add(initialImageRoute);
resultFaceList.Add(faceToVerify);

```

Procesamiento

Para este proceso podemos establecer tres fases, donde una de ellas sólo se lleva a cabo bajo ciertas características: detección, filtrado y verificación. Al igual que las simples, las consultas avanzadas iteran sobre cada imagen de la colección para llevar cabo las operaciones pertinentes divididas en las fases comentadas.

Detección

Guarda similitud con su homólogo de las consultas personalizadas, pero debido a que no se precisa extraer atributos adicionales de los rostros, se emplea un método alternativo del modelo. Téngase en cuenta que se han omitido las cláusulas *try catch* para mejorar la legibilidad.

```

async Task<IList<DetectedFace>> DetectFaceExtract(String imageRoute)
{
    IList<DetectedFace> detectedFaces = null;
    using (Stream imageFileStream = File.OpenRead(imageRoute))
    {
        detectedFaces = await Client.Face.DetectWithStreamAsync(imageFileStream,
            detectionModel: SelectedDetectionModel,
            recognitionModel: SelectedRecognitionModel
        );
    }
    return detectedFaces;
}

```

A diferencia de cuando se extraen parámetros, este método únicamente precisa de la ruta la imagen como entrada para invocar al método de la API *DetectWithStreamAsync*, que guarda dos diferencias con su uso en la ocasión anterior. La primera de ella es la ausencia del listado de atributos faciales por motivos obvios, y la segunda es que el modelo de detección ya no es fijo, y al igual que el de reconocimiento, éste viene determinado por la elección hecha en la vista principal.

Filtrado (opcional)

Como se indica en el título, esta etapa no se lleva a cabo en todas las ocasiones, se ejecuta únicamente cuando se cumple la condición determinada por el número de rostros detectados en cada imagen, en concreto, se ejecuta cuando se excede el umbral dado por la constante *MAXFACESBEFOREFILTER*, que por defecto tiene un valor de 5.

```

if (detectedFaces.Count > MAXFACESBEFOREFILTER)
{
    detectedFacesId = await FilterSimilarFaces(detectedFaces, faceToVerify);
}
else
    detectedFacesId = FaceListToGuidList(detectedFaces);

```

La operación de filtrado *FilterSimilarFaces*, mostrada a continuación, se basa en el uso del método perteneciente al modelo *SearchSimilar*, que se emplea para comparar la similitud de un rostro para con una lista de ellos, con la peculiaridad de pasar como parámetro el modo de trabajo *MatchPerson*, el cual, como se explica en el apartado **Funciones**, conlleva el uso de unos baremos para comparar rostros que pertenezcan a un mismo individuo, pero con menor precisión que la operación de verificación.

```

async Task<IList<Guid?>> FilterSimilarFaces(IList<DetectedFace> faceList, DetectedFace faceToVerify)
{
    IList<Guid?> resultFaceIdList = new List<Guid?>();
    IList<Guid?> targetFaceIds = FaceListToGuidList(faceList);

    IList<SimilarFace> similarFaces = await Model.SearchSimilar(faceToVerify.FaceId.Value,
targetFaceIds, FindSimilarMatchMode.MatchPerson);

    foreach (SimilarFace similarFace in similarFaces)
        resultFaceIdList.Add(similarFace.FaceId);

    return resultFaceIdList;
}

```

Esta llamada *Model.SearchSimilar* se verá con detalle en la sección correspondiente a la consultad “Búsqueda de gente similar”.

El motivo de esta operación de filtrado es el de reducir el número de comparaciones en aquellas imágenes con un gran número de rostros en ella, a fin de reducir el impacto negativo de la limitación impuesta por la cuota gratuita. Además, se puede observar el uso de una función que convierte una lista de rostros de una lista de identificadores, debido a que las llamadas de la API emplean estos identificadores en lugar de los objetos *DetectedFace* en sí.

```
IList<Guid?> FaceListToGuidList(IList<DetectedFace> faceList)
```

Verificación

Esta última fase está destinada a comparar cada rostro detectado en la imagen con el rostro objetivo y, en caso de pertenecer a la misma persona, añadir dicha imagen a la lista de resultados.

```

foreach (Guid? faceId in detectedFacesId)
    if (await Model.Verify(faceToVerify.FaceId.Value, faceId.Value))
    {
        resultImageList.Add(imageRoute);
        resultFacelist.Add(detectedFaces.First(T => T.FaceId == faceId));
        break;
    }

```

Como se puede observar, un bucle recorre la lista de rostros candidatos y los compara mediante la llamada al método del modelo *Verify*, de cuyo resultado dependerá si se acepta o no la imagen.

```

async Task<Boolean> Verify(Guid faceGuid01, Guid faceGuid02)
{
    VerifyResult result = new VerifyResult(false, 0);
    result = await Client.Face.VerifyFaceToFaceAsync(faceGuid01, faceGuid02);
}

```

```
} return result.Confidence >= VerifyMinimunConfidence;
```

Este método del modelo toma como entrada dos identificadores para efectuar la llamada *VerifyFaceToFaceAsync* con ambos. El retorno será el resultado de la comparación de la confianza de similitud del retorno de la operación con el umbral *VerifyMinimunConfidence*, el cual puede ser editado desde la vista principal; para afinar el resultado empleamos este umbral propio el lugar del proporcionado por la API mediante el atributo *isIdentical*, que resulta verdadero si la confianza igual o supera el valor de 0,5. Nuevamente, se han omitido las cláusulas *try catch* para mejorar la legibilidad.

Muestra de resultados

La muestra de los resultados se lleva cabo de la misma manera que en el caso de la consulta personalizada, donde únicamente cambia el título de la vista. Para más detalles consulte la sección *Muestra de resultados*.

4.5.3 Búsqueda de gente similar

Continuando con las denominadas “consultas avanzadas” dentro del sistema, tenemos la consulta para buscar aquellas personas que son similares a una persona en específico. Con una complejidad menor respecto a las operaciones de búsqueda de personas específicas, debido a que no se precisan operaciones de filtrado, esta operación involucra menos llamadas a la API que sus semejantes.

Adquisición y preparado de datos

Este procedimiento se lleva cabo de igual manera que en las consultas de búsqueda de una persona determinada, para más información véase la sección pertinente, *Adquisición y preparado de datos*.

Una vez preparados los datos de entrada, el sistema está listo para proceder con el método.

```
async Task SearchSimilar(List<String> imageList, string initialImageRoute, DetectedFace faceToVerify)
```

Preprocesamiento

Como en la consulta anterior, se crean los recursos donde se almacenarán progresivamente los resultados, a la par que se insertan en éstos tanto la imagen seleccionada originalmente por el usuario y como rostro correspondiente, eliminando dicha imagen de la lista de comprobación. Para más detalles, consulte *Preprocesamiento*.

Procesamiento

Al igual que el resto de las consultas, se lleva a cabo una iteración sobre cada imagen de la colección donde se realiza la búsqueda, llevando a cabo las siguientes tres fases para cada una: detección, búsqueda y comprobación.

Detección

Para estos casos donde se pretende únicamente extraer la información del rostro sin ningún tipo de atributo adicional, se emplea el método del modelo descrito en la sección *Detección* del subproceso de búsqueda de una persona determinada.

Los modelos de detección y reconocimiento empleados serán los definidos por el usuario desde la vista principal. Una vez obtenidos los rostros, se transforma la lista de objetos *DetectedFace* a una de objetos *Guid*, es decir, solo de identificadores, para poder ser usada en las operaciones del modelo.

```
IList<DetectedFace> detectedFaces = await Model.DetectFaceExtract(imageRoute);
IList<Guid?> targetFaceIds = FaceListToGuidList(detectedFaces);
```

Búsqueda

Este paso guarda similitud con la que sería la operación de filtrado, pues se emplea el mismo método del modelo, con la salvedad de que el parámetro que hace referencia al modo de trabajo es diferente, pues para esta operación se emplea el modo *MatchFace*. Al emplear este modo, nos aseguramos de obtener una lista con todos y cada uno de los rostros a comparar, junto a su correspondiente índice de confianza de similitud, al contrario que en la operación de filtrado, donde el resultado es una lista sesgada de aquellos rostros que la API considera que pertenecen al mismo individuo.

```
Model.SearchSimilar(faceToVerify.FaceId.Value, targetFaceIds, FindSimilarMatchMode.MatchFace);
```

```
public async Task<IList<SimilarFace>> SearchSimilar(Guid faceGuid, IList<Guid?> faceGuidList,
FindSimilarMatchMode matchMode)
{
    IList<SimilarFace> result = null;

    result = await Client.Face.FindSimilarAsync(faceGuid, null, null, faceGuidList, default,
matchMode);

    return result;
}
```

El retorno obtenido será una lista de objetos *SimilarFace*, con el mismo número de elementos que la lista de rostros a comparar, los cuales estarán ordenados según su atributo *Confidence* de mayor a menor valor.

Comprobación

Como ya se mencionó, debido a que la lista de objetos *SimilarFace* ya se encuentra ordenada, para comprobar si en la imagen hay alguien similar solo es necesario comprobar el primer elemento de ésta, que será el rostro con un mayor índice de confianza de similitud. Esta comparación está basada en un umbral constante dentro del sistema con el nombre de *SIMILARFACEMINCONFIDENCE*, a diferencia de la verificación, donde el umbral podía ser editado por el usuario.

```
if (similarFaces[0].Confidence > SIMILARFACEMINCONFIDENCE)
{
    resultImageList.Add(imageRoute);
    resultFaceList.Add(detectedFaces.First(T => T.FaceId == similarFaces[0].FaceId));
}
```

Podemos asumir que esta operación obtendrá resultados, como mínimo, de igual tamaño a la de búsqueda de personas concretas, ya que resulta obvio que la confianza de similitud para dos rostros de una misma persona será, cuanto menos, alto.

Muestra de resultados

De igual manera que la fase de adquisición y preparado de datos, esta consulta y su predecesora comparten la misma estructura a la hora de mostrar los resultados obtenidos por el proceso de detección y comparación que la consulta personalizada. Para ver en detalle el cómo se muestra el producto de este proceso, puede consultar la sección previa de **Muestra de resultados** de la consulta pertinente.

4.5.4 Búsqueda de acompañantes

La consulta basada en la búsqueda de una persona específica, junto a la medición de la frecuencia de aquellos individuos que aparecen en la misma imagen, se denomina dentro del sistema como “Búsqueda de acompañantes” y se trata de la consulta con mayor complejidad a nivel de número de operaciones de verificación y, por consiguiente, de llamadas a la API. Como último elemento de las denominadas consultas avanzadas, guarda similitud en varios aspectos con sus consultas hermanas, pero además emplea elementos únicos que la diferencian del resto de consultas del sistema como es, por ejemplo, su interfaz para mostrar resultados.

Adquisición y preparado de datos

Como ocurre en la consulta de búsqueda de gente similar, este proceso de recabación de datos se realiza de igual manera que en la primera de las consultas avanzadas explicadas, por ello puede consultar el apartado de **Adquisición y preparado de datos**, perteneciente a la sección de la “Búsqueda de una persona determinada”. A destacar como única diferencia, sería el propio método empleado encargado de llevar a cabo la consulta, pues donde en los casos anteriores sólo precisaba de la imagen inicial, el rostro seleccionado y el conjunto de imágenes que conforman la colección digital de origen, para esta consulta se precisa además el envío la lista con todos los rostros detectados en dicha imagen inicial, la cual es posible que contenga un único rostro.

```
async Task SearchCompanion(List<String> imageRouteList, string initialImageRoute,
DetectedFace faceToVerify, IList<DetectedFace> initialImageFaces)
```

Estos rostros serán tratados como los primeros acompañantes de la persona especificada.

Preprocesamiento

A diferencia de las consultas avanzadas detalladas previamente, en esta ocasión no se necesitarán almacenar las imágenes donde se hallen coincidencias, en cambio, los resultados serán guardados como recortes del rostro de cada acompañante junto con su número de apariciones, además del recorte de aquella persona especificada por el usuario como objetivo

de la búsqueda.

Seguidamente, se mostrarán todas le secciones de código previas al procesamiento de las imágenes, que van desde la creación de las estructuras de datos necesarias, hasta la elaboración de los recortes para cada uno de los rostros detectados en la imagen inicial.

```
Dictionary<DetectedFace, int> companionsMatchesAmount = new Dictionary<DetectedFace, int>();
Dictionary<DetectedFace, Image> companionsMatchesImages = new Dictionary<DetectedFace, Image>();

List<DetectedFace> companionsFaceList = new List<DetectedFace>();

int totalVerifiedImages = 1;
```

En primer lugar, se instancian dos diccionarios que albergarán el número de apariciones de un acompañante y el recorte de su correspondiente rostro respectivamente. Ambas estructuras estarán relacionadas entre sí mediante el empleo de objetos *DetectedFace* como clave; en un principio se valoró crear un objeto *Tuple* para almacenar en un mismo diccionario frecuencia y recortes, pero se descartó por la imposibilidad de editar estas tuplas más allá de crear una nueva con los valores actualizados.

Seguidamente se crea una lista de los compañeros preparada para las operaciones de filtrado, junto con un contador del número de imágenes en que aparece el individuo seleccionado inicialmente por el usuario.

```
using (FileStream stream = new FileStream(initialImageRoute, FileMode.Open, FileAccess.Read))
    actualImage = Image.FromStream(stream);

int height = actualImage.Height;
int width = actualImage.Width;

Rectangle faceRectangle = MakeRectangle(faceToVerify.FaceRectangle, height, width);
Image originalFace = CropImage(actualImage, faceRectangle);

initialImageFaces.Remove(faceToVerify);

foreach(DetectedFace face in initialImageFaces)
{
    faceRectangle = MakeRectangle(face.FaceRectangle, height, width);
    Image tmp = CropImage(actualImage, faceRectangle);

    companionsMatchesAmount.Add(face, 1);
    companionsMatchesImages.Add(face, tmp);
    companionsFaceList.Add(face);
}
```

En esta sección se crean los recortes, primero del rostro objeto de la búsqueda y luego del resto de rostros detectados en la imagen, si es que los hubiera; estos rostros se añaden tanto a los pertinentes diccionarios, como a la lista para filtrado mencionada con anterioridad.

Para llevar a cabo los recortes se invocan a dos funciones que se encargan de crear el rectángulo de que contiene al rostro dentro de la imagen para luego extraerlo dentro de la misma como un objeto *Image*.

```
Rectangle MakeRectangle(FaceRectangle faceCoordinates, int height, int width)
```

```
Image CropImage(Image img, Rectangle cropArea)
{
    Bitmap bmpImage = new Bitmap(img);
    return bmpImage.Clone(cropArea, bmpImage.PixelFormat);
}
```

La función *MakeRectangle* no contiene más lógica que expandir, cuando se pueda, el rectángulo que contiene el rostro para mejorar su visibilidad. Mientras, el método *CropImage* se aprovechará del rectángulo creado para establecer una nueva imagen a partir de él y la imagen original, mediante el método *Clone* del objeto *Bitmap*.

```
IList<DetectedFace> detectedFaces;
IList<Guid?> facesIdListToCompare;
Guid? verifiedFace;

List<DetectedFace> coincidenceFaceList = new List<DetectedFace>();
List<DetectedFace> noCoincidenceFaceList = new List<DetectedFace>();
```

Solo restan las estructuras de datos que almacenaran la información que se recabe en cada paso del procesado. A destacar las listas *coincidenceFaceList* y *noCoincidenceFaceList*, que determinan los rostros que contendrán a los acompañantes ya registrados y aquellos que no pudieron ser verificados respectivamente, que luego se incluirán como nuevos acompañantes.

Procesamiento

La lógica detrás del procesamiento en las consultas de búsquedas de acompañantes conlleva una complejidad mayor con respecto al resto que comparten la categoría de consultas avanzadas. Esto se debe a que, donde antes se llevaba a cabo una iteración por cada imagen a fin de comparar cada uno de los rostros, ya fuera mediante verificación o por búsqueda se similares, en la consulta de búsqueda de acompañantes éste sería sólo el primer paso, puesto que, en caso de encontrarse el rostro especificado, se debe comparar el resto de rostros de la imagen con la lista de acompañantes para comprobar su existencia en la misma y aumentar su números de comparencias para las coincidencias o, en caso contrario, añadir una nueva entrada para el rostro a modo de acompañante.

Por lo dicho, podemos identificar tres fases en este procesamiento: detección de persona objetivo, comparación de acompañantes y almacenamiento de los resultados. Como en casos anteriores, este conjunto de fases se itera sobre cada imagen de la colección.

Detección de persona objetivo

```
detectedFaces = await Model.DetectFaceExtract(imageRoute);
verifiedFace = null;

//Filter to avoid excessive Verify operations
if (detectedFaces.Count > MAXFACESBEFOREFILTER)
    facesIdListToCompare = await FilterSimilarFaces(detectedFaces, faceToVerify);
else
    facesIdListToCompare = FaceListToGuidList(detectedFaces);
foreach (Guid? faceId in facesIdListToCompare)
    if (await Model.Verify(faceToVerify.FaceId.Value, faceId.Value))
    {
        verifiedFace = faceId;
        break;
    }
```

Esta primera sección se basa en detectar la presencia del rostro seleccionado por el usuario en la imagen, se emplea un código altamente similar al usado en la consulta de “Búsqueda de una persona determinada”, con la salvedad de que, en caso de que la verificación reporte una respuesta positiva, no se almacena la información como resultado, sino que ese rostro se guarda dentro de la variable *verifiedFace*, de cuyo valor depende que se continúe con el proceso o se descarte la imagen actual.

Comparación de acompañantes

Este bloque se divide en dos partes, cuya ejecución es excluyente la una para con la otra, es decir, en caso de ejecutarse la primera no se ejecutaría la segunda y viceversa. Esta división viene motivada por el esfuerzo de reducir al máximo el número de llamadas a la API, con el objetivo de limitar el impacto negativo de la licencia gratuita en el rendimiento del sistema. La diferencia entre dichas partes radica en que lista es comparada con cuál, si comparamos los rostros detectados en la imagen con la lista de compañeros o viceversa, a fin de comparar la lista de menor tamaño con la mayor y así poder filtrar los rostros para reducir el número de posibilidades; es decir, esta división se lleva cabo para decidir de qué forma es más efectiva la operación de filtrado.

```
if (detectedFaces.Count > 1)
{
    //Remove from the list the face previously verified
    detectedFaces.Remove(detectedFaces.First(T => T.FaceId == verifiedFace));

    bool foundCoincidence;

    using (FileStream stream = new FileStream(imageRoute, FileMode.Open, FileAccess.Read))
        actualImage = Image.FromStream(stream);

    height = actualImage.Height;
    width = actualImage.Width;

    coincidenceFaceList.Clear();
    noCoincidenceFaceList.Clear();

    ...
}
```

En primer lugar, nos aseguramos de que en la imagen no haya únicamente el rostro de la persona objetivo ya verificada. Seguidamente se elimina de la lista el rostro verificado y se preparan las estructuras de datos necesarias, como son la imagen y sus dimensiones para los recortes, además de vaciar las listas empleadas para guardar a los rostros que fueron verificados como acompañantes ya registrados y los que no.

```
if (detectedFaces.Count < companionsFaceList.Count)
{
    List<DetectedFace> companionsMatchedList = new List<DetectedFace>();

    foreach (DetectedFace detectedFace in detectedFaces)
    {
        foundCoincidence = false;

        //Filter operation
        if (companionsFaceList.Count > MAXFACESBEFOREFILTER)
            facesIdListToCompare = await FilterSimilarFaces(companionsFaceList, detectedFace);
```



```

else
    facesIdListToCompare = FaceListToGuidList(companionsFaceList);

foreach (Guid? faceGuidToCompare in facesIdListToCompare)
    if (await Model.Verify(detectedFace.FaceId.Value, faceGuidToCompare.Value))
    {
        foundCoincidence = true;
        DetectedFace tmp = companionsFaceList.Find(T => T.FaceId == faceGuidToCompare);
        companionsFaceList.Remove(tmp);
        companionsMatchedList.Add(tmp);
        break;
    }

if (!foundCoincidence)
    noCoincidenceFaceList.Add(detectedFace);
}

foreach (DetectedFace matchedCompanion in companionsMatchedList)
{
    companionsFaceList.Add(matchedCompanion);
    coincidenceFaceList.Add(matchedCompanion);
}
}

```

Esta sección corresponde al caso donde la lista de acompañantes es mayor que el número de rostros detectados en la imagen, para lo cual primero creamos una lista que albergará todos aquellos acompañantes que se han detectado en la imagen. Seguidamente, se itera sobre la lista de rostros detectados, comenzando por la operación de filtrado sobre la lista de acompañantes, en el caso de que su tamaño supere el umbral marcado por la constante *MAXFACESBEFOREFILTER*; se prosigue con la iteración sobre el resultado de la lista filtrada o la original según corresponda.

En el consiguiente bucle se realiza una verificación entre cada acompañante y el rostro detectado de la iteración actual, en caso de respuesta afirmativa, se guarda mediante la variable *foundCoincidence* que la verificación dio positivo y se elimina temporalmente de la lista dicho acompañante, a la par que se dan por terminadas las operaciones de verificación. Una vez finalizado el bucle, si la verificación no dio ningún resultado positivo, el rostro actual se guarda en la lista de no coincidencias. Cuando se han analizado todos los rostros de la imagen actual, se itera nuevamente sobre la lista de acompañantes verificados en la imagen, para restablecer la lista de acompañantes que se redujo temporalmente y añadir éstos a la lista de coincidencias.

Con esto concluye la sección para aquellos casos donde haya menos acompañantes registrados que rostros detectados en la imagen, por lo tanto, resta la sección que contempla el caso contrario.

```

else
{
    foreach (DetectedFace companionFace in companionsFaceList)
    {
        foundCoincidence = false;

        if (detectedFaces.Count > MAXFACESBEFOREFILTER)

```

```

        facesIdListToCompare = await FilterSimilarFaces(detectedFaces, companionFace);
    else
        facesIdListToCompare = FaceListToGuidList(detectedFaces);
    foreach (Guid? faceToCompare in facesIdListToCompare)
        if (await Model.Verify(companionFace.FaceId.Value, faceToCompare.Value))
        {
            foundCoincidence = true;
            detectedFaces.Remove(detectedFaces.First(T => T.FaceId == faceToCompare));
            break;
        }
    if (foundCoincidence)
        coincidenceFaceList.Add(companionFace);
}
noCoincidenceFaceList = detectedFaces.ToList();
}

```

Para estos casos, el proceso, si bien similar, guarda diferencias con su homólogo más allá de que lista se compara con cual. Como se puede observar, primero se inicia un bucle que itera sobre cada acompañante listado, y se comienza comprobando si se lleva a cabo o no el filtrado de los rostros detectados. Seguidamente, se itera sobre la lista de rostros detectados, ya esté filtrada o no, y se verifica cada uno con el acompañante de la secuencia actual. Cuando la verificación retorna una respuesta positiva, es decir, un valor *true*, y al igual que en el caso anterior, se guarda mediante la variable *foundCoincidence* que la verificación dio positivo, a la par que se retira de la lista de rostros detectados aquel que produjo la coincidencia.

Cuando se termina la iteración de rostros detectados, si se produjo una coincidencia, el acompañante de la sección actual se guarda en la lista de coincidencias. Una vez finalizada todas las verificaciones de acompañantes, aquellos de los rostros detectados que quedaron sin verificar, se añaden a la lista de no coincidencias.

Almacenamiento de los resultados

Una vez procesada la imagen, independientemente de la sección empleada, se obtienen una lista con aquellos acompañantes que produjeron una coincidencia y los rostros detectados que quedaron sin verificar.

```

foreach (DetectedFace face in coincidenceFaceList)
    companionsMatchesAmount[face] = companionsMatchesAmount[face] + 1;
foreach (DetectedFace face in noCoincidenceFaceList)
{
    //Create the crop
    faceRectangle = MakeRectangle(face.FaceRectangle, height, width);
    Image tmp = CropImage(actualImage, faceRectangle);

    //Add it to the dictionary
    companionsMatchesAmount.Add(face, 1);
    companionsMatchesImages.Add(face, tmp);
    companionsFaceList.Add(face);
}

```

Como tenemos dos listas con rostros, debemos recorrer ambas realizando las acciones pertinentes según corresponda. Para la lista de coincidencias, simplemente se incrementa en uno la entrada del diccionario de frecuencias para cada rostro. Y para aquellos que no se pudieron verificar, primero se elabora el recorte con el rostro a partir de la imagen y luego se crean las entradas en ambos diccionarios, con un uno como valor de frecuencia y con el recorte elaborado como imagen; tras lo cual se añaden a la lista de acompañantes que se emplea en la verificación.

Muestra de resultados

Finalizado el procesamiento de cada una de las imágenes de la colección, solo resta mostrar los resultados del proceso al usuario.

```
List<Image> imageList = new List<Image>();
List<String> percentageList = new List<String>();
//Elaborate the Image and Percentages list from the dictionaries
    foreach (DetectedFace face in companionsFaceList)
    {
        imageList.Add(companionsMatchesImages[face]);

        percentageList.Add($"Appear in {companionsMatchesAmount[face]} of {totalVerifiedImages}:
" + (companionsMatchesAmount[face] * 100 / totalVerifiedImages).ToString() + "%");
    }

    View.CreateCompanionGalleryChild(imageList, originalFace, percentageList, "Search
companions");
```

Antes de elaborar la interfaz, primero se rellenan dos listas a partir de los diccionarios elaborados:

- Una lista con las imágenes recortadas de los rostros a partir del diccionario de recortes.
- Una lista de cadenas de texto que muestren la frecuencia de aparición en porcentaje.

Con ambos listados completos, se procede a realizar la invocación del método perteneciente vista principal *CreateCompanionGallery*.

```
void CreateCompanionGalleryChild(List<Image> imageList, Image image, List<String> percentage, String
title)
{
    if (imageList.Count > 0)
    {
        ViewCompanionGallery newMDIChild = new ViewCompanionGallery(imageList, image,
percentage, title);

        // Set the Parent Form of the Child window.
        newMDIChild.MdiParent = this;

        // Display the new form.
        newMDIChild.Show();
    }
    else
        MessageBox.Show(
            "The programm couldn't found any companion in the collection for the selected",
            "No companions found" );
}
```

Este proceso, similar a aquel que crea la galería para el resto de las consultas, instancia una vista del tipo *ViewCompanionGallery* bajo la indicación de que tiene como “padre” a la vista principal. No obstante, esta vista no se creará en el caso de que no se haya podido encontrar ningún acompañante en la colección para el individuo seleccionado. Esta vista está conformada por dos elementos principales, un objeto *PictureBox*, que contendrá el rostro que seleccionó el usuario, y un objeto *ImageList* que mostrará cada recorte de los acompañantes, seguido de su frecuencia como información sobre la imagen.

Una vez mostrada la interfaz con el resultado al usuario, la consulta “Búsqueda de acompañantes” se da por finalizada.

4.6 Dificultades e inconvenientes

A la hora de diseñar el sistema, se han tenido que solventar o mitigar dos inconvenientes principales que se explicarán brevemente, así como su curso de acción.

En primer lugar, si bien el sistema está desarrollado en el lenguaje *C#*, no fue ésta la primera opción elegida para este fin. En su etapa inicial, se decidió emplear el lenguaje Java para la implementación del trabajo, decisión motivada debido a la familiaridad con el mismo. No obstante, en etapas tempranas del desarrollo se encontró un problema en el método de autenticación de la API, el cual, tras una breve investigación, se pudo identificar como *bug* para dicha librería dentro del repositorio Maven, que provoca la imposibilidad de autenticar del cliente. Al momento de escribir este documento, el *bug* se encuentra registrado en el portal *Github* desde mayo de 2019 y continúa sin solución.

En segundo lugar, tenemos la limitación de la licencia gratuita en 20 llamadas por minuto. Como se ha mencionado de forma recurrente en esta memoria, para mitigar los efectos que pudiera tener esta limitación sobre el rendimiento del sistema, el diseño se orientó en torno a reducir cuanto fuera posible el número de llamadas a la API sin restar fiabilidad al sistema. Con esto en mente, se desarrolló la operación de filtrado en las operaciones de verificación, así como la división en secciones de la consulta “Búsqueda de acompañantes”, cuyo beneficio podrá verse en detalle en la posterior sección de pruebas. Obviamente, esto es un inconveniente para el rendimiento fácilmente omisible mediante una licencia de pago y no supone diferencia alguna en cuanto a la precisión y fiabilidad del sistema.

Estos serían los principales escollos que se dieron lugar durante el diseño y desarrollo del sistema que, mediante el cambio de lenguaje y la optimización del número de llamadas, se tuvieron que solventar más allá de otros problemas menores propios del desarrollo software que no merecen ser explicados más allá de esta breve mención.

Capítulo 5

Pruebas

Las pruebas aquí relatadas consisten en la ejecución del sistema sobre diferentes colecciones digitales compuestas por imágenes de entornos reales, con el objetivo de medir el comportamiento de éste en cuanto a la duración de los procesos y precisión de los resultados, mediante el empleo tanto de diferentes modelos de detección y reconocimiento, como de umbrales de confianza.

En primer lugar, se explicarán las denominadas funciones de registro, que no son otra cosa que modificaciones de los subprocesos explicados e implementadas a fin de elaborar un registro del proceso a medida que se lleva cabo la consulta, en el cual se almacenan los datos de ésta como son la duración de cada llamada a la API, los modelos elegidos y demás. También se lleva a cabo una estimación de los posibles puntos en que el programa demora la respuesta presuntamente debido a la limitación impuesta por la licencia gratuita, lo cual nos permitirá hacernos una idea cuál sería el rendimiento sin esta limitación.

Tras un breve repaso a las colecciones digitales empleadas para el desarrollo de las pruebas, explicadas en la sección *Base de datos*, se detallarán los resultados obtenidos en las pruebas con relación al tiempo de ejecución y el rendimiento de las diferentes consultas ligadas a los métodos propios de la API; esto último referencia tanto al índice de acierto como los errores en forma de falsos positivos. Se hará énfasis en la detección y reconocimiento de individuos empleando mascarillas, para medir el impacto que ha tenido dicho accesorio impuesto por la pandemia en la eficacia del reconocimiento facial, así como el desempeño de aquellos modelos creados por *Microsoft* en específico para este contexto.

Una vez terminadas las pruebas, y con los resultados en mano, podemos llevar a cabo el análisis de éstos para extraer las conclusiones pertinentes, como pueden ser el saber qué par de modelos se desempeña mejor con gente que emplea mascarillas o en qué situaciones puede resultar de interés alterar el umbral de confianza de 0,5.

5.1 Funciones de registro

Dentro del sistema podemos encontrar las denominadas funciones de registro que, como su nombre indica, son similares a las consultas pero con la inclusión de funciones de registro que almacenan el proceder paso a paso de éstas. Estas funciones tienen como única función probar el sistema, por lo cual, y a diferencia de sus consultas homólogas, las operaciones de filtrado se convierten en un paso opcional, es decir, es posible su deshabilitación.

```
if (detectedFaces.Count > MAXFACESBEFOREFILTER && filterEnabled)
```

Como se puede observar, además de comprobar el número de rostros, existe un segundo comprobante para indicar si el filtrado debe llevarse a cabo o no.

```

async Task CustomQueryLogRercord(List<String> imageUrl, String collectionRoute){}

async Task SearchPersonLogRecord(List<String> imageUrl, string initialImageRoute, DetectedFace
faceToVerify, String collectionRoute){}

async Task SearchSimilarLogRecord(List<String> imageUrl, string initialImageRoute, DetectedFace
faceToVerify, String collectionRoute){}

async Task SearchCompanionLogRecord(List<String> imageRouteList, string initialImageRoute, DetectedFace
faceToVerify, IList<DetectedFace> initialImageFaces, String collectionRoute){}

```

Estas cuatro funciones llevan cabo de igual manera las operaciones de búsqueda de características, búsqueda de individuos concretos, búsqueda de gente similar y búsqueda de acompañantes, respectivamente. Como tales, estas funciones tienen un manejador de eventos independientes y durante su desempeño se registra no sólo el proceder del método, sino que además guardan el tiempo para cada llamada a la API, midiendo las posibles pausas por exceder la limitación de llamadas por minuto, y contabilizando el número y tipo de llamadas. En el resumen final se encuentra un resumen de estos aspectos, que incluye los modelos elegidos, si el filtro está activo o no, la duración total del procedimiento, el número de imágenes resultado y el número de cada tipo de llamada a la API junto a su duración media.

Estos archivos de registro tienen como extensión *.txt* y se almacenan en un directorio que se crea en la carpeta “*Documentos*” al iniciar el sistema. Los registros se nombran según el tipo de consulta, seguido de la fecha y hora de su ejecución.

5.2 Adquisición de imágenes

Como breve recordatorio, en la *Tabla 6*. Resumen de colecciones digitales se muestran los datos de las colecciones empleadas en las pruebas, todas ellas se tratan de imágenes basadas en entornos reales. Podemos ver ejemplos de estas colecciones en la *Ilustración 14*.

	Nº de imágenes	Temática
Colección 01	70	Miembros del Gobierno de Canarias
Colección 02	380	Liga de baloncesto profesional española
Colección 03	1080	Juegos Olímpicos Río de Janeiro 2016
Colección 04	100	Igual a 01, pero centrada en el uso de mascarillas

Tabla 6. Resumen de colecciones digitales

Cabe destacar la presencia de rostros con ángulos que dificultan su reconocimiento, así como de pequeño tamaño, ligeramente obstruidas o con una iluminación que dista de ser ideal; la presencia de estas imágenes se ve motivada por la comprobación del sistema en entornos no ideales.



Ilustración 14. Ejemplos de colecciones 1 y 4

5.3 Tiempo de ejecución

A continuación, se evaluarán los tiempos de ejecución tanto para las consultas al completo, como para cada llamada a la API de forma individual.

En lo que respecta al tiempo de ejecución total de la consulta, tomaremos como referencia la consulta “*Búsqueda de una persona determinada*”, puesto que implica los tres tipos de llamadas a la API empleados en el sistema. Donde la búsqueda de características y de similares emplea únicamente una y dos llamadas por imagen respectivamente, esta consulta emplea llamadas según el número de rostros detectados, mientras que la consulta de búsqueda de acompañantes varía según no solo los rostros por imagen, si no por la propia lista de acompañantes que se incrementa a medida que examina la colección.

En la *Tabla 7* se muestra el tiempo de ejecución medio para esta consulta en cada una de las colecciones mediante el empleo del modelo de detección (MD) 3 y modelo de reconocimiento (MR) 4, los cuales son, a priori, el par más preciso. Para la estimación del tiempo sin restricción de llamadas por minuto, se calculó mediante la sustracción de tiempo basado en el número de pausas llevadas a cabo por el sistema cada 20 llamadas en un minuto.

	Con filtro	Sin filtro	Estimación ideal
Colección 01 (70)	00:10:35	00:19:24	00:01:40
Colección 02 (380)	01:11:20	02:06:10	00:08:32
Colección 03 (1080)	03:32:01	06:55:45	00:23:78
Colección 04 (100)	00:14:28	00:27:01	00:02:13

Tabla 7. Tiempos de ejecución para búsqueda de personas

Los resultados mostrados se dividen en las categorías de filtrado activo, sin filtrado y la estimación de tiempo ideal, es decir, sin la limitación impuesta por la licencia gratuita. Para esta medición se empleó como umbral para filtrado más de cinco rostros en la imagen y, como puede observarse, se consigue una reducción de tiempo en torno al 44% tiempos cuando se activa este procedimiento. De media, el sistema tarda un tiempo que ronda el cuarto de hora en una colección de 100 imágenes y supera la hora en 380 incluso con el filtrado de rostros activado, no obstante, como se mencionó anteriormente en el informe, el tiempo total de consulta se puede reducir drásticamente hasta alcanzar los tiempos estimados en la tercera columna de la tabla con el mero hecho de emplear una licencia de pago en sustitución de la gratuita, por lo cual podríamos considerar que la estimación es de hecho un tiempo más acertado de cara al rendimiento real de la API.

Seguidamente, veamos la duración real de cada llamada a la API donde la restricción no es influyente, para considerar el tiempo de ejecución en base a los modelos empleados.

	M.R. 1	M.R. 2	M.R. 3	M.R. 4
M.D. 1	0,41	0,42	0,59	0,56
M.D. 2	1,01	1,04	1,34	1,25
M.D. 3	1,22	1,27	1,60	1,51

Tabla 8. Tiempo de detección de rostros en segundos

En la **Tabla 8** vemos los tiempos de ejecución para las operaciones de detección de rostros en la imagen según los modelos de detección y de reconocimiento. Los datos muestran que el modelo de detección 1 es notablemente más rápido a la hora de identificar rostros en una imagen, frente a los modelos 2 y 3 que se diferencian en apenas veinte centésimas. Para los modelos de reconocimiento, la diferencia no resulta tan notoria puesto que su uso principal es para con las operaciones de comparación de rostros, pero llama la atención que el modelo 4 sea ligeramente más rápido en la identificación que el modelo anterior, cuando se puede observar una tendencia incremental según más novedoso sea el modelo.

	M.D. 1	M.D. 2	M.D. 3
M.R. 1	0,070	0,075	0,077
M.R. 2	0,068	0,079	0,077
M.R. 3	0,069	0,076	0,080
M.R. 4	0,068	0,077	0,078

Tabla 9. Tiempo de búsqueda de personas similares en segundos

	M.D. 1	M.D. 2	M.D. 3
M.R. 1	0,080	0,092	0,089
M.R. 2	0,079	0,089	0,085
M.R. 3	0,079	0,087	0,084
M.R. 4	0,075	0,082	0,088

Tabla 10. Duración de la operación de filtrado en segundos

En la *Tabla 9* y la *Tabla 10* se muestran los tiempos de las operaciones de búsqueda de personas similares y de filtrado respectivamente. En donde nuevamente nos encontramos con un ligero incremento según más novedoso sea el modelo, pero que, para ambos, es cuanto menos despreciable al hablar de rendimiento. Además, podemos observar que, si bien las operaciones de filtrado son operaciones de búsqueda de similares empleando umbrales para detectar rostros de un mismo individuo, estos umbrales suponen apenas un incremento de centésimas de segundo en tiempo de ejecución, lo cual no puede considerarse como un impacto sustancial ni remarcable.

	M.D. 1	M.D. 2	M.D. 3
M.R. 1	0,057	0,060	0,058
M.R. 2	0,059	0,062	0,059
M.R. 3	0,057	0,057	0,058
M.R. 4	0,057	0,057	0,058

Tabla 11. Duración de la operación de verificación en segundos

Para las operaciones de verificación referenciadas en la *Tabla 11*, podemos encontrar unos tiempos menores que en las operaciones anteriores, lo cual es normal debido a que estas operaciones comparan rostros uno a uno y mientras que la búsqueda de personas similares compara uno con varios. En cuanto a las diferencias de modelos, nuevamente se obtienen cifras muy similares entre sí, que si bien es destacable cómo el modelo de reconocimiento 2 obtiene unas cifras mayores que sus homólogos para cualquier modelo de detección, aunque la diferencia es de milésimas.

5.4 Detección

Para hablar de la capacidad de detección de rostros de la API en imágenes, tomaremos como muestra los rostros detectados por cada modelo de detección en las colecciones 1 y 4, indicados en la *Tabla 12*, recordando que, como característica diferencial, la cuarta colección muestra a gente llevando mascarillas, con 70 y 100 imágenes respectivamente.

	M.D. 1	M.D. 2	M.D. 3
Colección 1	232	521	543
Colección 4	53	445	434

Tabla 12. Número de rostros detectados según el M.D.

En primer lugar, observemos el **modelo de detección 1**, cuyo aspecto positivo es la capacidad de extraer atributos del rostro; en comparación con los modelos más modernos tiene una capacidad de detección mucho menor y es que, como bien avisa de ello *Microsoft* en la documentación de la API, no está optimizado para rostros pequeños ni de perfil, y son precisamente este tipo de rostros los que producen la diferencia en número de detecciones para con sus semejantes. Otro aspecto que impide la detección para este modelo es la obstrucción del rostro, lo cual queda de manifiesto en la gran diferencia en número de detecciones en la colección 4, donde las mascarillas sobre el rostro es norma.

Para los **modelos 2 y 3**, tenemos dos modelos de una precisión considerablemente superior respecto al primer modelo, capaces de detectar mayor cantidad de rostros en condiciones no ideales, como son los borrosos, de perfil, obstruidos o pequeños. Con respecto a las diferencias entre ambos, el modelo 3 y más reciente, tiene una capacidad mayor de detección para rostros que estén girados y de tamaños reducidos, lo cual puede suponer una mayor detección, si bien no de gran magnitud, como se muestra en la diferencia de rostros en la colección 1; no obstante, podemos ver una capacidad de detección mayor del modelo 2 para con la colección de gente con mascarillas, las pruebas demostraron que este aumento marginal, de apenas el 2,5%, se debía a rostros incompletos, donde parte del rostro fuera quedaba de la imagen, o con una orientación excesiva que difícilmente permitía ver el rostro.

A destacar que los modelos más modernos, 2 y 3, detectaban rostros que no pertenecían exclusivamente a personas reales, pues se dieron casos donde se detectaron caras en cuadros y estatuas. Esto supondría un problema por ejemplo si se contabilizaran como acompañantes en la consulta pertinente, pero cabe destacar que se detectaban rostros incluso pertenecientes a una videoconferencia dentro de la imagen, lo cual en sí mismo era un acierto. Esta capacidad de detección puede suponer un inconveniente al aumentar el número de comparaciones o incluso un beneficio, ya que, por ejemplo, se podrían incluso detectar coincidencias en retratos si así se quisiera.

5.4.1 Búsqueda de características

Esta consulta se encuentra ligada al modelo de detección 1, ya que es el único capaz de extraer atributos de los rostros, y por tanto se ve limitado a la capacidad de detección de éste.

Respecto a su índice de acierto y ligado al modelo de detección, cuanto mayor sea el tamaño del rostro y su posición frontal, mayor en consecuencia será la capacidad de la API para detectar estos atributos. En estas condiciones ideales es posible obtener un alto índice de acierto, donde la detección de emociones estará supeditada a la expresividad de cada individuo, mientras que otros atributos más visuales como el cabello obtienen resultados más fácilmente corroborables. Para rostros no ideales pueden darse fallos de apreciación, como es el confundir un llamativo maquillaje de ojos con unas gafas y viceversa; y remarco, estos errores se dan especialmente es los rostros lejanos o pequeños.

5.5 Reconocimiento

A continuación, se verán los índices de acierto en las operaciones de reconocimiento según cada modelo de reconocimiento, y en qué manera puede influir sobre éstos el modelo de detección.

5.5.1 Verificación

Los datos que serán presentados a continuación se tomaron empleando como umbral de confianza el valor de 0,5, valor por defecto de la API para afirmar que dos rostros pertenecen al mismo individuo. Se tendrán en cuenta los diferentes pares de modelos de detección y reconocimiento sobre cada colección, contabilizando únicamente los aciertos registrados respecto del total de imágenes a buscar en la colección, no así los falsos positivos.

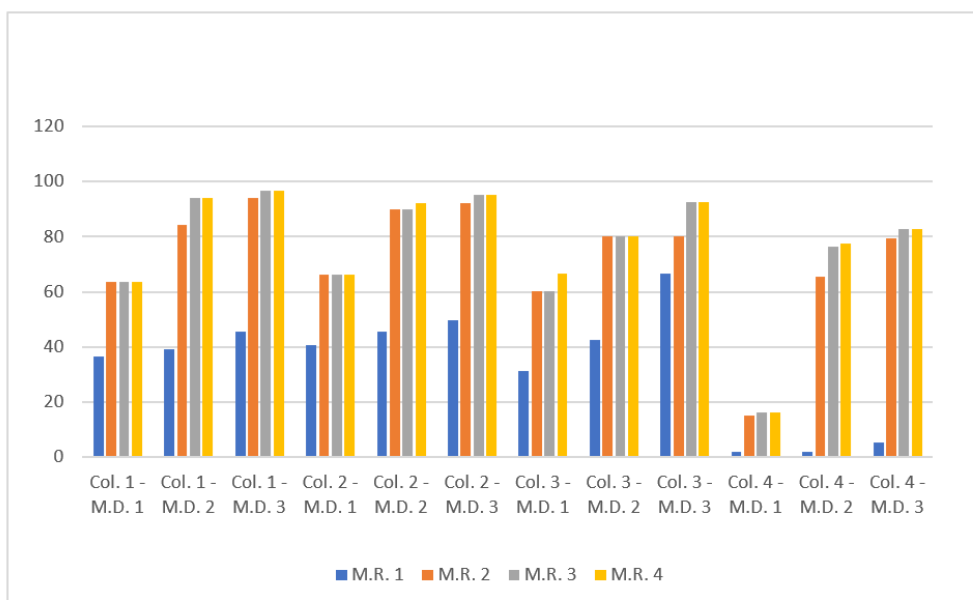


Tabla 13. Gráfico de índice de acierto en operaciones de verificación

A primeras de cambio, la *Tabla 13* pone de manifiesto una diferencia considerable en el acierto del modelo de detección 1 para con sus semejantes, pero como esto está ligado a la capacidad de detección, nos centraremos en las diferencias para con los modelos de reconocimiento.

Los datos muestran que el modelo de **reconocimiento 1** ronda el cincuenta por ciento de acierto a la hora de reconocer rostros, y no supera el 10 cuando tratamos con mascarillas, demostrando que este modelo, en cuanto a rendimiento, es ciertamente inferior a sus homólogos más recientes.

El **modelo de reconocimiento 2**, consigue unos resultados nada desdeñables, pero que nunca supera siquiera al modelo 3, las pruebas pusieron de manifiesto que los rostros de perfil o ladeados suponían casi la totalidad de los fallos en verificación.

En cuanto a los **modelos más recientes, 3 y 4**, para cada colección y modelo de detección los resultados son parejos o con una ligera superioridad para el modelo 4, pero ambos siempre rondando o superando un acierto del 90%. Para ver con mayor claridad la diferencia entre ambos modelos, veamos a continuación los resultados obtenidos al aumentar el umbral de confianza del 0,5 al 0,7 sobre la colección 4 que, al contener gente con mascarillas, debería suponer una ventaja para el modelo de reconocimiento 4, especializado en ello.

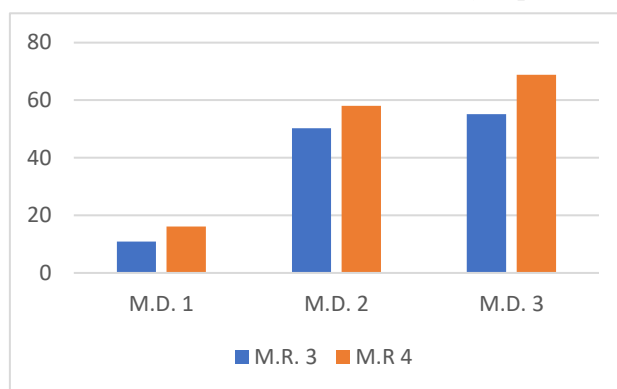


Tabla 14. Gráfico de acierto en verificación para la colección 4

Como vemos en la *Tabla 14*, ciertamente el modelo de reconocimiento 4 acrecienta su ventaja respecto a su homólogo al incrementar el nivel de confianza, es decir, determina con mayor exactitud que dos rostros pertenecen al mismo individuo y, por ende, la precisión de su reconocimiento es mayor, aunque la precisión del modelo 3 no es nada desdeñable.

A destacar, para estas pruebas se han empleado individuos de características diferentes, como son el sexo, color de piel y la presencia de accesorios como gafas; los resultados obtenidos no han mostrado ninguna diferencia reseñable que haga pensar que el reconocimiento varía en base a estos aspectos, no así como otros aspectos ya mencionados que sí influyen en los resultados, como son las condiciones adversas en la imagen, mala orientación o iluminación, que provocan que los rostros no sean reconocibles por la API, y la presencia de otros rostros que sean visiblemente similares que sean detectados como falsos positivos.; éstos últimos, como se verá más adelante, dejan de incidir en los resultados al aumentar el umbral de confianza.

5.5.2 Búsqueda de personas similares

En cuanto a los resultados de la búsqueda de similares, resulta complicado hablar de precisión o acierto de los resultados puesto que, si bien para la API es una cuestión de cálculos matemáticos, para las personas esta cuestión usualmente está basada a apreciaciones personales. Para comentar los resultados, tomemos como ejemplo las coincidencias en la colección 1 para un individuo concreto, las cuales siempre serán, como mínimo, de igual número que en las operaciones de verificación.

	M.R. 1	M.R. 2	M.R. 4	M.R. 4
M.D. 1	12	15	15	15
M.D. 2	25	24	23	23
M.D. 3	27	25	24	24

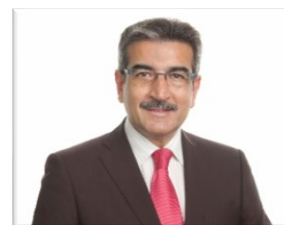


Tabla 15. Resultados de búsqueda de personas similares

Ilustración 15. Román Rodríguez Rodríguez

Para los resultados obtenidos en la **Tabla 15**, cuyo rostro objetivo no es otro que el vicepresidente del Gobierno de Canarias, Román Rodríguez Rodríguez, mostrado en la **Ilustración 15**¹⁶.

Como en casos anteriores, el modelo de detección 1 lastra el número de resultados con respecto a los modelos 2 y 3. Analizando los resultados obtenidos, los modelos de reconocimiento 3 y 4, su mayor precisión provoca que se localicen menos individuos similares al ser más estricto.

Para el modelo de reconocimiento 2, los resultados mostraron una tendencia a señalar como similares a los rostros de pequeño tamaño, mientras que el modelo 1 provocaba coincidencias con rostros que poseían características comunes tales como gafas o bigote, para este caso concreto.

Al observar los resultados en diferentes colecciones se reveló que, además de gente que visualmente podría ser similar, los rostros de perfil tenían más posibilidades de ser etiquetados como tal, especialmente en los modelos de detección 3 y 4. Mientras que los modelos de reconocimiento 1 y 2, como ya se comentó, tenían coincidencias mayores para atributos faciales comunes y para rostros de tamaño reducido, respectivamente.

5.6 Falsos positivos

A la hora de hablar de índice de acierto, no pueden dejarse de lado la otra cara de los resultados. Más allá de aquellos rostros que sí pertenecían a cierto individuo y no fueron calificados como tal, están aquellos rostros que se valoraron como coincidencias y no era el caso, los denominados como falsos positivos.

¹⁶ Imagen perteneciente a la web del Parlamento de Canarias, www.parcn.es

Estos falsos positivos, como se ha ido comentando, se dan principalmente bajo dos preceptos. El primero, que en la colección haya rostros de una similitud notoria para con el objetivo de la búsqueda, esta similitud puede verse acrecentada bajo ciertas condiciones tales como rostros lejanos o de perfil; y segundo, se decide emplear un umbral de confianza bajo para aumentar el índice de verificaciones exitosas, lo cual bien puede suponer que se verifican rostros antes descartados, los cuales no necesariamente pertenecen al individuo objetivo.

	M.R.	Confianza	Resultados	Falsos Positivos
Colección 2	3	0,50	5 / 5	6
		0,75	5 / 5	0
	4	0,50	5 / 5	5
		0,75	5 / 5	0
Colección 3	3	0,50	19 / 21	6
		0,75	16 / 21	0
	4	0,50	19 / 21	5
		0,75	17 / 21	0

Tabla 16. Resultados y falsos positivos para el M.D. 3

Para demostrar lo mencionado, podemos ver los datos de la **Tabla 16**, que muestra los resultados para la búsqueda de un individuo concreto de cada colección, empleando el modelo de detección 3 y los modelos de reconocimiento 3 y 4. Los resultados muestran que se consiguen identificar completamente o con ligeros fallos todos los individuos buscados con una confianza de 0,5, no obstante, también se detectan varios falsos positivos, los cuales se caracterizaban por ser en su mayoría imágenes de perfil o lejanas, junto a algún individuo realmente similar. Para eliminar estos falsos positivos se dispone a elevar el umbral de confianza de 0,5 a 0,75, lo cual provoca el descarte total de los falsos positivos manteniendo para la colección 2 todas las identificaciones correctas, no obstante, la colección 3 precisaría reducir el umbral a fin de no eliminar también algunos de los resultados correctos. Podemos deducir que los falsos positivos detectados sobrepasaban el umbral de 0,5 por un margen relativamente escaso, debido a su no detección tras el aumento del umbral.

De forma adicional, para examinar los casos donde se pretende reducir el umbral de confianza y así captar resultados descartados para el valor de por defecto, se probaron dos umbrales de menor tamaño: 0,4 y 0,25.

Para el 0,4, los resultados obtenidos mostraban un incremento de los resultados correctos de entre el 5 y el 10 por ciento, sin que se detectaran nuevos falsos positivos de manera significativa. Podemos hablar de un umbral que produce un ligero aumento en las verificaciones, con un riesgo escaso de que esto conlleve a la identificación de falsos positivos.

Con 0,25, se obtienen nuevos resultados acertados de en torno al 14 por ciento, es decir, no existe gran diferencia con el umbral de 0,4, no obstante, la aparición de nuevo falsos positivos se puede dar por asegurada, ya que cada prueba realizada con este umbral arrojaba varios nuevos falsos positivos a los resultados.

5.7 Análisis de resultados

Habiendo mostrado y examinado los datos mostrados por el sistema, en esta sección se detallarán aquellas conclusiones que pueden extraerse de éstos para comprobar que configuración puede resultar más eficaz dependiendo del entorno en que quiera emplearse.

Cuando hablamos de tiempo de ejecución, podemos ver que todo problema en cuanto al excesivo tiempo global de cada consulta se puede solventar con la licencia de pago, y que los modelos de reconocimiento no producen diferencias de tiempo significativas en las operaciones de reconocimiento, mientras que, para las operaciones de detección, el modelo de detección 1 consume la mitad de tiempo que sus semejantes, lo cual podríamos asegurar que está ligado a su menor grado de precisión en comparación con éstos.

Para el reconocimiento de rostros, los modelos de detección 2 y 3 se muestran ampliamente superiores al primer modelo, especial en lo que se refiere a la detección de rostros de perfil o pequeños. Sin embargo, hay que tener en cuenta que su mayor detección implica que también se identifiquen cuadros, videoconferencias e incluso estatuas como rostros, esto conlleva aumentar el número de comparaciones, por norma general, de forma innecesaria. Cabe destacar que el modelo 2 capta particularmente bien rostros obstruidos o incompletos, lo cual no es necesariamente algo positivo, puesto que este tipo de rostros serán difíciles de verificar por el sistema en las operaciones de comparación.

En la detección de características nos vemos limitados por el modelo de detección 1 y su escasa capacidad de identificar rostros en condiciones no ideales, no obstante, para estos rostros podemos llevar a cabo la búsqueda de las características deseadas con relativa precisión, más allá de errores donde se llega a confundir un extenso maquillaje con gafas o viceversa y similares, especialmente en las imágenes donde el rostro sea de tamaño reducido, como se muestra en la *Ilustración 16*¹⁷.

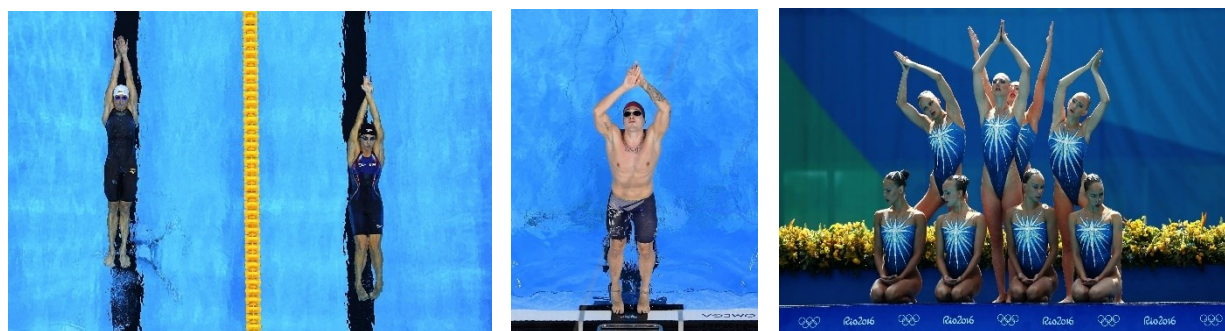


Ilustración 16. Imágenes donde se detectaron gafas de buceo

Prosiguiendo con las operaciones de verificación, tras comparar el rendimiento de los modelos de reconocimiento y su influencia en los resultados, resulta evidente el cómo los modelos más recientes consiguen un índice de acierto mayor respecto a sus predecesores, independientemente del modelo de detección empleado. En cuanto a qué modelo sería

¹⁷ Las imágenes mostradas fueron extraídas de la cuenta oficial de los Juegos Olímpicos en la red social Facebook: www.facebook.com/olympics

recomendable emplear, los resultados no muestran una diferencia significativa entre los citados modelos 3 y 4, por norma general, se encuentran igualados o el modelo 4 se impone por una ventaja mínima, ahora bien, este último modelo se diferencia del resto en que *Microsoft* lo elaboró con el reconocimiento de gente con mascarillas en mente; lo cual pudo comprobarse el ejecutar búsquedas sobre una colección centrada en este aspecto, donde quedó patente que los resultados obtenidos por este modelo mostraban una confianza mayor que los obtenidos por el modelo 3.

Continuado la búsqueda de gente similar, y teniendo en cuenta que ya se mencionó lo arbitrario que puede resultar para una persona juzgar la similitud entre dos individuos, podemos extraer varias conclusiones según qué modelo de reconocimiento se emplee. Por un lado, están los modelos 3 y 4 que, debido a su mayor precisión, tienden a producir una lista de resultados muy similar a la lista de verificación, incluso a pesar de que se emplea una comparación notablemente menos rigurosa. Y luego están los modelos 1 y 2, que arrojan resultados más reseñables, puesto que no se clasifica únicamente un mayor número de rostros como similar, sino que estas similitudes tienen cierta tendencia a compartir atributos faciales, tales como bello facial o la presencia de gafas. Este hecho, que ocurre con mayor frecuencia para el modelo de reconocimiento 1, puede verse en la *Ilustración 17*¹⁸, que muestra dos rostros etiquetados como similares.



Ilustración 17. Rostros etiquetados como similares

Por último, queda el comportamiento del sistema para con los denominados falsos positivos, para el cual se nos presentan dos escenarios, aumentar el umbral de confianza para eliminar falsos positivos, o cuanto podemos rebajarlo antes de que aparezcan nuevos.

Para el primero, cuando nos encontramos con ya de por sí varios falsos positivos con el umbral de 0,5, se puede observar que el aumento de este al 0,75 los elimina prácticamente en su totalidad, no obstante, debería ser aconsejable reducirlo a un término intermedio para no descontar de igual manera los resultados correctamente reconocidos previamente.

Y cuando queremos aumentar el número de aciertos a costa de reducir el umbral, se ha observado que en torno a 0,4 obtenemos una mejoría en el reconocimiento sin empeorar significativamente en cuanto a falsos positivos. Bajar más aún el umbral hasta 0,25 resulta contraproducente, puesto que obtenemos un incremento en los aciertos insustancial en comparación al aumento de nuevos falsos positivos.

¹⁸ Imágenes extraídas de www.parcán.es (izquierda) y www.laprovincia.es (derecha).

Capítulo 6

Presupuesto

Para el desarrollo de este trabajo se han empleado exclusivamente programas y licencias de carácter gratuito, esto mismo queda reflejado en la *Tabla 17*, donde se enumeran dichas herramientas junto a sus respectivas licencias o planes de pago.

Herramienta	Licencia
Visual Studio 2019	Edición gratuita <i>Community</i> , para uso individual
Adobe Bridge	Programa de carácter gratuito
Azure	Subscripción: <i>Azure para estudiantes</i>
Face API	Plan de tarifa: Gratis

Tabla 17. Herramientas y licencias

Para la asignatura de Trabajo de Fin Grado de 12 Créditos ECTS, que según la equivalencia de 25 horas por crédito supone un total de 300 horas, y teniendo en cuenta el salario medio de un ingeniero informático en España en 2021 de en torno a 36.500 € brutos por año [33], y con las 1.800 horas anuales estipuladas en el *Convenio colectivo estatal de empresas de consultoría y estudios de mercado y de la opinión pública*, se calculan 20,28 € por hora. La estimación de horas y coste puede verse en la *Tabla 18*.

	Horas invertidas	Coste
Búsqueda bibliográfica y examinación de alternativas	12	243,36 €
Desarrollo de basa de datos	54	1.095,12 €
Desarrollo del sistema	153	3.102,84 €
Pruebas y evaluación	36	730,08 €
Elaboración de memoria	31	628,68 €
Total	286	5.800,08

Tabla 18. Horas invertidas y coste

Capítulo 7

Conclusiones y líneas futuras

En este trabajo se expone el desarrollo de un sistema basado en la API *Face* de *Microsoft Azure*, además de la elaboración de un conjunto de colecciones de imágenes basadas en entornos reales que compondrán la base de datos sobre la que se llevarán a cabo las pruebas del sistema, las cuales se centrarán en el desempeño de los diferentes modelos de detección y reconocimiento de los que dispone esta tecnología y que están explicados en este escrito.

En lo que refiere a tiempos de ejecución, los resultados obtenidos muestran que, para la detección de rostros, el modelo de detección 1 conlleva un tiempo de procesamiento de entorno al medio segundo, mientras que los modelos 2 y 3 obtienen tiempos de en torno al doble y al triple de esa cifra. Para las operaciones de reconocimiento, cualquier modelo de reconocimiento obtiene tiempos inferiores a la décima de segundo; al evaluarlos por pares de detección y reconocimiento, se observa que emplear los modelos más modernos reduce el tiempo de procesamiento, en particular los tiempos de detección con modelos de reconocimiento más recientes. El análisis en detalle de los resultados lleva a la conclusión de que los modelos de detección modernos conllevan un tiempo mayor debido a su mejor capacidad de detección, mientras que los modelos de reconocimiento reducen los tiempos debido a su mejor optimización.

La precisión de los resultados en las pruebas de detección de características mostró estar directamente relacionada a las capacidades de detección del modelo de detección 1, es decir, los resultados empeoran de forma drástica para aquellos rostros que no están en condiciones ideales, como son las imágenes de perfil, parcialmente cubiertas o borrosas, sin ningún tipo de influencia según el método de reconocimiento. Por contraparte, en las operaciones donde se llevan a cabo verificaciones rostro a rostro, los modelos de reconocimiento más recientes obtienen mejores resultados que sus homólogos; acompañar a éstos de los últimos modelos de detección consigue resultados precisos incluso en rostros que distan de estar en condiciones ideales, en particular, para la presencia de mascarillas, el par de modelos de detección y reconocimiento más modernos muestran resultados razonablemente más precisos y de mayor confianza que cualquier otro. Las pruebas no arrojaron pruebas sustanciales de que características tales como sexo o color de piel influyan en la precisión de los resultados.

Con vistas al futuro del desarrollo, pueden explorarse dos vías. La primera sería seguir profundizando en las funcionalidades de la API, puesto que este sistema no emplea todas las capacidades de ésta. Por ejemplo, la funcionalidad conocida como *agrupación*, la cual precisa de entrenar un grupo de rostros para seguidamente calificar uno nuevo con éstos, con el fin de identificar al nuevo individuo dentro del grupo de aquellos que ya son conocidos. Por otro lado, valiéndonos de que es un servicio de la nube *Azure*, se puede estudiar la posibilidad de elaborar más de un recurso de la API *Face* para implementar un sistema que procese las imágenes de forma paralela y no secuencial, que es el modo de trabajo del sistema presentado en este informe. Aunque la posible mejoría se limita a tiempo de ejecución y no a la precisión.

Capítulo 8

Summary and Conclusions

At its final stage, the system has implemented all the queries defined at the beginning of this Project, which includes search of attributes, search of a specific person, search people similar to a specific person, and listing the people who accompany a specific person, besides the possibility to alternate between the different detection and recognition models available, as well as change the verification confidence threshold for testing purposes.

Once examined the duration of each individual API call, we can deduce that the processing of an image, which includes detection and verification calls, has a duration around two seconds, this time allows us to process a large amount of images in a matter of minutes, or even faster if we use less accurate models, which can reduce the duration to half at the expense of reducing the precision of the results; it is a complex and recurrent decision to choose efficiency or accuracy, in case that someone considers that two seconds per image is an excessive time. These times mentioned before, are independent of the selected license of the API Face, but the time per query is negatively affected by the limitation of the free license.

In terms of precision, the most recent models demonstrate satisfactory results even for distant faces or in profile, which doesn't happen in the older recognition models. With the recognition models 3 and 4, we are talking of an accuracy about 90% in verification operations and with the possibility of reducing the false positives by the increment of the verification confidence threshold, which at certain levels removes almost completely the false positives with a minimum impact over the previous results. However, the high index of detection could affect negatively to the system due to the most recent detection models can detect faces in statues and portraits, and increment the number of comparisons, specially in the company search where all the faces in the image can be stored as companions that will be compared in the rest of images.

In conclusion, the API Face is an alternative that must be considered both by efficiency and precision, where images with faces which conditions that are distant of being ideals could be recognized satisfactorily in a relative short time. Particularly, those images where there are people wearing a mask, the most recent models offer a response with an accuracy around 80 percent, which is a really good precision for faces partially occluded. And we mustn't forget that, with the cloud processing of *Azure*, the work necessary to identify and verify faces is carried out remotely, so we don't need more than a stable internet connection to work with the API resource, although exists the possibility of using Docker to carry out the processing locally.

Capítulo 9

Bibliografía

- [1] C. Liu, «International Competitiveness and the Fourth Industrial Revolution,» *Entrepreneurial Business and Economics Review*, vol. 5, n^o 4, p. 111–133, 2017.
- [2] A. Young y A. Burton, «Are We Face Experts?,» *Trends in Cognitive Sciences*, vol. 22 (2), pp. 100-110, 2018.
- [3] M. S. Sarfraz, O. Hellwich y Z. Riaz, «Feature Extraction and Representation for Face Recognition,» de *Face Recognition*, Universidad eslovaca de tecnología en Bratislava, Eslovaquia, IntechOpen, 2010, pp. 01-20.
- [4] A. F. Abate, S. Ricciardi y G. Sabatino, «3D Face Recognition in a Ambient Intelligence Environment Scenario,» de *Face Recognition*, Universidad de Zagreb, Croacia, IntechOpen, 2007, pp. 1-14.
- [5] A. Eleyan y H. Demirel, «PCA and LDA Based Neural Networks for Human Face Recognition,» de *Face Recognition*, Zagreb, IntechOpen, 2007, pp. 93-106.
- [6] Blauch, N. M, M. Behrmann y D. C. Plaut, «Computational Insights into Human Perceptual Expertise for Familiar and Unfamiliar Face Recognition.,» *Cognition*, vol. 208, 2021.
- [7] A. W. Young y A. M. Burton, «Insights from computational models of face recognition: A reply to Blauch, Behrmann and Plaut,» *Cognition*, vol. 208, 2021.
- [8] A. García-Barrera, I. García-Magariño y F. D. Guillén-Gámez, «Retos Y Posibilidades Del Software De Reconocimiento Facial Como Herramienta Para Asegurar La Calidad Educativa En Los Entornos Virtuales De

Aprendizaje.» *EduTec*, vol. 53, nº 53, 2015.

- [9] Á. López, H. H. Guillermo y J. Ignacio, «Desarrollo De Una Aplicación Para Ordenadores Para La Detección Y Reconocimiento Facial De Alumnos Para El Posterior Control De Asistencia,» Gandía, 2016.
- [10] J. Alonso Soto, «Procedimiento para la obtención de publicidad personalizada a partir de reconocimiento facial». España Patente ES2359796 (A1), 27 Mayo 2011.
- [11] D. A. La Madrid Arroyo, M. H. Barriga Rivera y L. M. Canaval Sánchez, «Modelo Tecnológico de Reconocimiento Facial para la Identificación de Pacientes en el Sector Salud,» Universidad Peruana de Ciencias Aplicadas (UPC), Lima, 2019.
- [12] «Quantitative anatomical analysis of facial expression using a 3D motion capture system: Application to cosmetic surgery and facial recognition technology,» *Clinical Anatomy*, vol. 28 (6), pp. 735-744, 2015.
- [13] N. Cañego Navio, J. I. Herranz Herruzo y J. Pelegrí Sebastiá, «Sistema de identificación de personas mediante reconocimiento facial aplicado a videovigilancia,» Gandía, 2017.
- [14] C. J. Bravo, P. E. Ramírez y J. Arenas, «Aceptación del reconocimiento facial como medida de vigilancia y seguridad: un estudio empírico en Chile,» *Información tecnológica*, vol. 29 (2), pp. 115-122, 2018.
- [15] G. López - Cleries y Á. Porras Soriano, «EL TAG COMO CELDA DE VIGILANCIA. Una visión crítica de los metadatos y sistemas de reconocimiento facial a través de las prácticas artísticas,» *ANIAV - Revista de Investigación en Artes Visuales*, vol. 2, pp. 99-107, 2018.
- [16] N. Martinez-Martin, «What Are Important Ethical Implications of Using Facial Recognition Technology in Health Care?,» *AMA J Ethics*, vol. 21 (2), pp. 180-187, 2019.
- [17] C. Almonacid Díaz, «Consideraciones teóricas y éticas del reconocimiento facial de las emociones en contexto de pandemia,» *Veritas: Revista De*

Filosofía Y Teología, nº 46, pp. 55-75, 2020.

- [18] M. Copeland, J. Soh, A. Puca, M. Manning y D. Gollob, «Microsoft Azure and Cloud Computing,» de *Microsoft Azure. Planning, Deploying, and Managing Your Data center in the Cloud*, Apress, 2015, pp. 03-26.
- [19] J. Soh, M. Copeland, A. Puca y M. Harris, «Overview of Azure Infrastructure as a Service (IaaS) Services,» de *Microsoft Azure. Planning, Deploying, and Managing the Cloud.*, Apress, 2020, pp. 43-55.
- [20] M. C. A. P. M. H. Julian Soh, «Overview of Azure Platform as a Service,» de *Microsoft Azure. Planning, Deploying, and Managing the Cloud*, Apress, 2020, pp. 43-55.
- [21] J. Soh, M. Copeland, A. Puca y M. Harris, «Azure Data Services,» de *Microsoft Azure. Planning, Deploying, and Managing the Cloud*, Apress, 2020, pp. 369-409.
- [22] C. Hoder y A. Raman, *Building Intelligent Apps with Cognitive APIs*, O'Reilly Media, 2020.
- [23] M. S. y M. R., «Azure Cognitive Services,» de *Developing Bots with Microsoft Bots Framework*, Berkeley, Apress, 2018, pp. 233-260.
- [24] D. S. A., «Getting Started with the Computer Vision API,» de *Microsoft Computer Vision APIs Distilled*, Berkeley, Apress, 2018, pp. 5-15.
- [25] A. D. Sole, «Invoking the Computer Vision API from C#,» de *Microsoft Computer Vision APIs Distilled*, Apress, 2018, pp. 17-42.
- [26] A. Verma, D. Malla, A. K. Choudhary y V. Arora, «A Detailed Study of Azure Platform & Its Cognitive Services,» de *International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, Faridabad, India, 2019.
- [27] K. Maheshwari y N. N, «Facial Recognition Enabled Smart Door Using Microsoft Face API,» *International Journal of Engineering Trends and Applications (IJETA)*, vol. 4 (3), pp. 1-4, 2017.
- [28] P. Kostakos, A. Pandya, O. Kyriakouli y M. Oussalah, «Inferring demographic data

of marginalized users in twitter with computer vision APIs,» de *European Intelligence and Security Informatics Conference*, Karlskrona, Suecia, 2018.

- [29] Y. Liu y olprod, «Microsoft Docs,» Microsoft, 03 Mayo 2021. [En línea]. Available: <https://docs.microsoft.com/es-es/azure/cognitive-services/face/face-api-how-to-topics/specify-detection-model>. [Último acceso: 15 Junio 2021].
- [30] J. Greene, «Microsoft won't sell police its facial-recognition technology, following similar moves by Amazon and IBM,» *The Washington Post*, 11 Junio 2020. [En línea]. Available: <https://www.washingtonpost.com/technology/2020/06/11/microsoft-facial-recognition/>. [Último acceso: 14 Abril 2021].
- [31] F. Schroff, D. Kalenichenko y J. Philbin, «FaceNet: A Unified Embedding for Face Recognition and Clustering,» de *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, EEUU, 2015.
- [32] Y. D. González y Y. F. Romero, «Patrón Modelo-Vista-Controlador,» *Telemática*, vol. 11, n^o 1, pp. 47-57, 2012.
- [33] Jobted, 2021. [En línea]. Available: <https://www.jobted.es/salario/ingeniero-inform%C3%A1tico>. [Último acceso: 18 Junio 2021].