

Grado en Ingeniería Informática
2022-2023

Trabajo Fin de Grado

“Sistema de Recomendación de Moda basado en Atributos Multimodales”

Gonzalo Llosá Cea

Tutores

Miguel Ángel Patricio Guisado

Carlos Rodríguez - Pardo

Madrid, septiembre



Esta obra se encuentra sujeta a la licencia Creative Commons **Reconocimiento - No Comercial - Sin Obra Derivada**

RESUMEN

La industria de la moda es una de las más potentes en nuestro país y esta se ha ido adaptando al auge tecnológico que hemos vivido en las últimas décadas. Un gran porcentaje de las ventas de este sector se realizan de manera online y es por ello la importancia de un sistema de recomendación.

Un sistema de recomendación de moda para las tiendas online resulta muy beneficioso, tanto para propietarios como para clientes. Los primeros aumentan sus ventas mientras que los segundos encuentran más fácil y rápido productos que, en muchas ocasiones, ni siquiera hubieran podido encontrar.

Para solucionar este problema se hace uso de una amplia base de datos en donde las prendas vienen etiquetadas por sus diferentes atributos. Después de realizar un estudio sobre las tecnologías que pueden resolver este problema, se eligen diferentes arquitecturas para resolver las diferentes partes que componen el objetivo principal. Para la segmentación de imágenes, con el fin de conseguir sus máscaras, se hace uso de la arquitectura U2-Net. Para el algoritmo de recomendación, se extraen los vectores de características de las imágenes haciendo uso de la arquitectura VGG16, para posteriormente calcular la similitud coseno entre estos vectores. Además se añade la posibilidad de elegir los atributos que se quieren presentes en la recomendación de la prenda.

Una vez se realiza este modelo se desarrolla una aplicación web que es capaz de servirlo para que los usuarios puedan utilizarlo de una manera fácil e intuitiva. El modelo se sirve a través de una API, haciendo uso de *Flask*, a la aplicación web hecha con *React*.

También se lleva a cabo un estudio de la efectividad del sistema, analizando los porcentajes de similitud entre recomendaciones y la calidad de estas.

Este proyecto tiene gran relevancia en el mundo real. Puede llegar a causar un gran impacto tanto en clientes de negocios de venta online de ropa como en los propios clientes.

Palabras clave: Red convolucional, visión por computadora, segmentación de imágenes, interfaz gráfica.

ABSTRACT

The fashion industry is one of the most powerful sectors in our country, and it has been adapting to the technological boom that we have experienced in recent decades. A significant percentage of sales in this sector occur online, underscoring the importance of a recommendation system.

A fashion recommendation system for online stores proves highly advantageous, benefiting both proprietors and customers. The former witness an increase in sales, while the latter find it easier and quicker to discover products that, in many instances, they might not have come across otherwise.

To address this issue, an extensive database is employed, wherein garments are labeled with their distinct attributes. After conducting a study on technologies capable of resolving this problem, diverse architectures are selected to tackle the various components constituting the primary objective. For image segmentation, aimed at obtaining their masks, the U2-Net architecture is utilized. Concerning the recommendation algorithm, feature vectors are extracted from the images using the VGG16 architecture, subsequently computing the cosine similarity between these vectors. Furthermore, the option to select the attributes to be present in the garment recommendation is incorporated.

Once this model is developed, a web application is created to facilitate user-friendly and intuitive utilization. The model is served through an API, utilizing *Flask*, for the web application built with *React*.

An assessment of the system's effectiveness is also conducted, analyzing similarity percentages between recommendations and their quality.

This project holds significant relevance in the real world, potentially yielding a substantial impact on both the clientele of online clothing businesses and the customers themselves.

Keywords: Convolutional network, computer vision, images segmentation, graphic interface. Red convolucional, visión por computadora, segmentación de imágenes, interfaz gráfica.

ÍNDICE GENERAL

1. INTRODUCCIÓN	1
1.1. Motivación	1
1.2. Objetivos	2
1.3. Marco regulador	2
1.4. Entorno socioeconómico	2
1.5. Estructura del documento	3
2. ESTADO DEL ARTE	5
2.1. Visión por computadora	5
2.2. Redes de neuronas convolucionales	6
2.3. Extracción de características	7
2.4. Segmentación de imágenes	8
2.5. Trabajos similares	8
3. DISEÑO DEL SISTEMA	10
3.1. Conjunto de datos	10
3.2. Modelo	11
3.2.1. Segmentación.	11
3.2.2. Algoritmo de recomendación	14
3.3. API	15
3.4. Interfaz	17
3.5. Tecnologías empleadas	22
4. RESULTADOS	25
4.1. Sin máscaras	25
4.2. Con máscaras	26
4.3. Uso de atributos	29
5. CONCLUSIONES	32
6. TRABAJOS FUTUROS	34
7. PLANIFICACIÓN Y PRESUPUESTOS	35
7.1. Planificación	35

7.2. Presupuestos	36
7.2.1. Coste del personal	36
7.2.2. Coste del equipo	36
7.2.3. Costes directos	37
7.2.4. Costes indirectos	37
7.2.5. Costes totales	37
BIBLIOGRAFÍA	38

ÍNDICE DE FIGURAS

1.1	Peso de las ventas moda online sobre el total de los ingresos generados por las ventas de ropa en España de 2012 a 2022	1
2.1	Detección de caras con el software OpenCV	5
2.2	Red de Neuronas	6
2.3	Estructura VGG16	7
3.1	Ejemplo de imagen en el conjunto de datos	10
3.2	Ejemplo de imagen antes y después de aplicar la máscara	12
3.3	Arquitectura U2-Net	13
3.4	Ejemplo de creación de máscaras con la arquitectura U2-Net	14
3.5	Estructura algoritmo VGG16	14
3.6	Diagrama del funcionamiento del modelo	15
3.7	Ejemplo de petición al <i>endpoint</i> “/recommendations”	16
3.8	Ejemplo de petición al <i>endpoint</i> “/image/<id>”	17
3.9	Estructura del proyecto	18
3.10	Componente attributesSearch	18
3.11	Componente header	18
3.12	Componente imageUploader	19
3.13	Pantalla inicial, subida de imagen	20
3.14	Selección del número de recomendaciones	20
3.15	Selección de atributos	21
3.16	Barra de carga mientras la API procesa la solicitud	21
3.17	Resultado del proceso. Recomendaciones.	22
4.1	Recomendaciones sin uso de máscaras	25
4.2	Recomendaciones sin uso de máscaras	26
4.3	Recomendaciones con uso de máscaras	27
4.4	Recomendaciones con uso de máscaras	28
4.5	Gráfica de Similitud Coseno	29

4.6 Recomendaciones con selección del atributo “blazer”	30
4.7 Recomendaciones con selección de los atributos “blazer” y “loose (fit)” .	30
4.8 Recomendaciones con selección de los atributos “blazer”, “loose (fit)” y “geometric textile pattern”	31

ÍNDICE DE TABLAS

3.1	Especificaciones del hardware utilizado	23
4.1	Tabla de similitud de recomendaciones	26
4.2	Tabla de similitud de recomendaciones	26
4.3	Tabla de similitud de recomendaciones	27
4.4	Tabla de similitud de recomendaciones	28
4.5	Medias de similitud coseno de imágenes	29
4.6	Tabla de similitud de recomendaciones	30
4.7	Tabla de similitud de recomendaciones	31
4.8	Tabla de similitud de recomendaciones	31
7.1	Planificación del proyecto	36
7.2	Planificación del proyecto	36
7.3	Costes materiales	37
7.4	Costes directos	37
7.5	Costes directos	37

LISTA DE ACRÓNIMOS

API	Application Programming Interface
PIB	Producto Interior Bruto
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
CPU	Central Processing Unit
GPU	Graphics Processing Unit
IDE	Integrated development environment
UI	User Interface

1. INTRODUCCIÓN

1.1. Motivación

La industria de la moda siempre ha sido una de las más potentes en nuestra sociedad. Esta industria se ha ido adaptando al auge que han tenido las tecnologías informáticas y es que, en el año 2022, las ventas de ropa online supusieron el 21,1 % de las ventas totales en nuestro país.

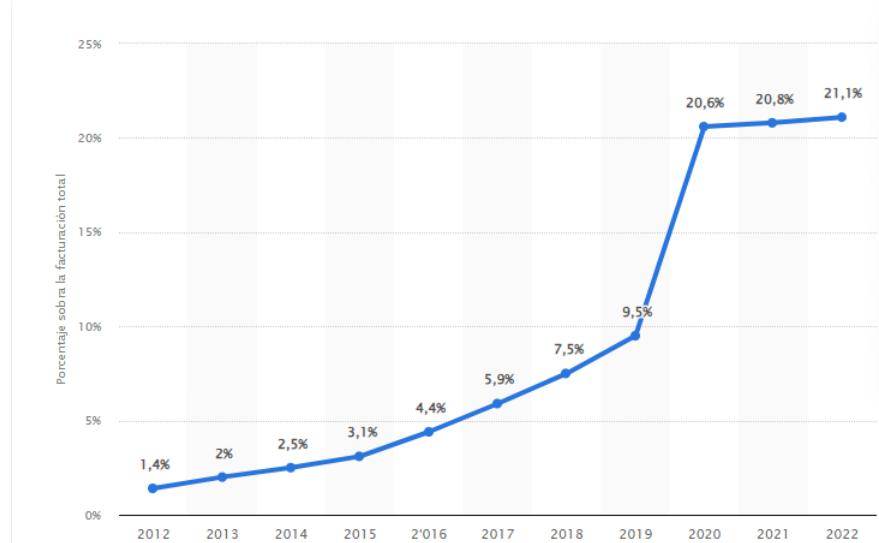


Fig. 1.1. Peso de las ventas moda online sobre el total de los ingresos generados por las ventas de ropa en España de 2012 a 2022 [1]

Es aquí donde entran en juego los sistemas de recomendación. Un sistema de recomendación de moda permite ayudar a las personas a encontrar prendas que se ajusten a sus gustos. Los clientes pueden mejorar su experiencia de compra y estar satisfechos con sus elecciones al recibir recomendaciones personalizadas y precisas. Esto resulta en un aumento de las ventas de los sitios web que se dedican a la moda. Un sistema de recomendación que esté basado únicamente en imágenes es útil, pero el hecho de introducir la posibilidad de especificar los atributos que deben estar en las prendas lo mejora aún más. Permitir a los usuarios obtener recomendaciones les ahorra mucho tiempo en búsquedas que, en muchas ocasiones, pueden resultar tediosas.

En conclusión, el hecho de crear un sistema de recomendación de moda basado en atributos multimodales resulta muy interesante tanto para propietarios de tiendas online, ya que mejoran sus ventas, como para usuarios, que tienen más facilidad a la hora de encontrar las prendas que buscan, ahorrando así una gran cantidad de tiempo.

1.2. Objetivos

El objetivo de este trabajo es el de crear una interfaz web que sea capaz de recomendar imágenes de ropa, con una cierta similitud, a partir de otra previamente dada por el usuario. Además se podrán especificar los atributos que se deseen en las imágenes de salida (por ejemplo: manga larga, ajuste suelto, etc.).

Siendo este el principal objetivo podemos diferenciar los siguientes sub-objetivos:

- Desarrollar un modelo de inteligencia artificial que sea capaz de recomendar imágenes en base a atributos multimodales y otra imagen.
- Crear una API que permita la comunicación entre el modelo previamente mencionado y una interfaz.
- Diseñar una interfaz intuitiva que permita a los usuarios cargar una imagen, seleccionar los atributos de interés y visualizar los resultados proporcionados por el modelo.

Estos objetivos pretenden resolver el problema planteado, derivando en una interfaz gráfica web que sea capaz de brindar a los usuarios la capacidad de obtener recomendaciones de prendas de ropa a partir de ropa que les guste y de los atributos que consideren.

1.3. Marco regulador

Para la realización del trabajo se hace uso de un conjunto de imágenes de la competición “iMaterialist (Fashion) 2020 at FGVC7” [2]. Estas imágenes contienen rostros de personas reales por lo que hay que tener en cuenta la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales [3], aún así el conjunto de datos ha sido citado de manera apropiada. Por la misma razón se ha de prestar atención a la Ley de Propiedad Intelectual [4]. Además si consideramos este como un trabajo de investigación habría que contemplar la Ley 14/2011, de 1 de junio, de la Ciencia, la Tecnología y la Innovación [5].

Las herramientas utilizadas para el desarrollo de este proyecto son de uso libre y gratuito, como son *Python*, Visual Studio Code, Tensorflow, Keras o React.

1.4. Entorno socioeconómico

Como ya se mencionó anteriormente en el apartado de la motivación, la moda es una de las mayores industrias a nivel global. Esta genera mucho interés y supone el 2,8 % del PIB español y es el 4 % del mercado laboral [6]. Es por ello que un sistema de recomendación de moda tiene un gran interés e impacto a nivel social. Algunos de los beneficios son

los siguientes, según el artículo de Alma Muñoz “Cómo los sistemas de recomendación pueden ayudarte a conseguir más ventas” [7].

- “Entre 15 % y 45 % de aumento de las conversiones”
- “25 % de media de incremento del valor promedio de compra.”
- “Alargamiento del ciclo de vida del cliente.”
- “Generación de fidelización del comprador.”

Además, Alma, menciona que “Amazon declara que alrededor del 35 % de sus ingresos se debe a recomendaciones de producto.”

En concreto, este producto pretende ser un sistema de recomendación de moda. Es por ello que podría ser útil para cualquier empresa que se dedique a la venta de ropa vía online por todos los beneficios que este aporta. Además, implementar este sistema en una web de venta de ropa supone una gran ventaja sobre el resto de los competidores.

Un sistema de recomendación también tiene una gran relevancia a nivel social. Los usuarios de estos pueden, no solo llegar a ahorrar grandes cantidades de tiempo, sino también conseguir encontrar prendas más afines a su gusto y personalidad. Esto resulta en una mayor satisfacción por parte del cliente, que necesitará invertir menos tiempo en encontrar mejores productos.

En conclusión, este proyecto aporta muchas ventajas económicas tanto a vendedores como a compradores, ahorrando tiempo y consiguiendo una mayor satisfacción con la compra. Más adelante, en su correspondiente apartado, se detallará la planificación y presupuesto.

1.5. Estructura del documento

El documento se presenta con una portada, a continuación podemos encontrar un resumen junto con las palabras clave. Después, podemos encontrar el índice general, el índice de figuras y la lista de acrónimos. De aquí en adelante el documento está dividido en siete marcados capítulos:

- **Introducción:** aquí podemos encontrar tanto la motivación para la realización del trabajo como los objetivos que este pretenden cumplir. Además se incluye el marco regulador y el entorno socioeconómico.
- **Estado del arte:** se elabora un análisis sobre el estado actual del objetivo del proyecto. Se analizan trabajos parecidos y el estado de las tecnologías empleadas. Además de argumentar como este proyecto se puede diferenciar de los trabajos ya realizados.

- **Diseño del sistema:** se explica como se ha llevado a cabo la solución del problema propuesto. Dividiéndolo en sus respectivos sub-objetivos ya planteados.
- **Resultados:** se presentan los resultados obtenidos, con ejemplos, de una manera visual. Además se analiza la calidad de estos a través de los porcentajes de similitud.
- **Conclusiones:** reflexión que valora si se han cumplido los objetivos propuestos.
- **Trabajos futuros:** se analizan las posibles continuaciones y mejoras que se pueden llevar a cabo a partir de este trabajo.
- **Planificación y presupuestos:** resumen a cerca de como se ha planificado el proyecto y explicación del cálculo de los presupuestos.

2. ESTADO DEL ARTE

En este capítulo se analizará el estado actual del proyecto, haciendo un recorrido por las tecnologías necesarias para su realización. Además se realizará un análisis de los proyectos similares junto con una argumentación a cerca de como este puede llegar a diferenciarse de ellos.

2.1. Visión por computadora

La visión por computadora es la disciplina del aprendizaje automático que trata con imágenes. A través de ella se intenta entender de forma computacional las imágenes. Entre otras cosas, la visión por computadora, se utiliza para el reconocimiento de objetos, la restauración de imágenes, el reconocimiento facial y un sin fin de tareas más. Dado que se trabaja con imágenes la importancia de esta disciplina en este proyecto es máxima.

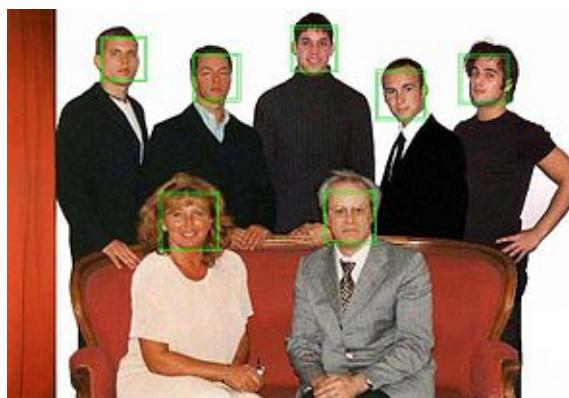


Fig. 2.1. Detección de caras con el software
OpenCV [8]

La visión por computadora ha tenido un gran avance durante los últimos años debido, principalmente, a las mejoras en hardware. Somos ahora capaces de realizar una mayor cantidad de cálculos en una cantidad mayor de datos gracias a un significativo aumento de la capacidad de cómputo. Estos progresos permiten que se investigue y profundice más en este campo.

Con la llegada de las redes neuronales se ha hecho posible la realización de tareas complejas como la clasificación de imágenes, la detección de objetos o la segmentación. Las redes de neuronas convolucionales se han convertido en el estándar para este campo de la ciencia. Estas son capaces de ver patrones visuales que a un humano, se le harían complicados, si no imposibles, de ver.

2.2. Redes de neuronas convolucionales

Las redes neuronales están compuestas por diferentes capas, que a su vez están compuestas por diferentes neuronas. Todas las redes tienen una capa de entrada, una o más capas intermedias y una capa de salida. Los nodos de las capas están conectados entre sí y estos pueden activar o no, dependiendo del resultado de su función asociada, el nodo al que están conectados.

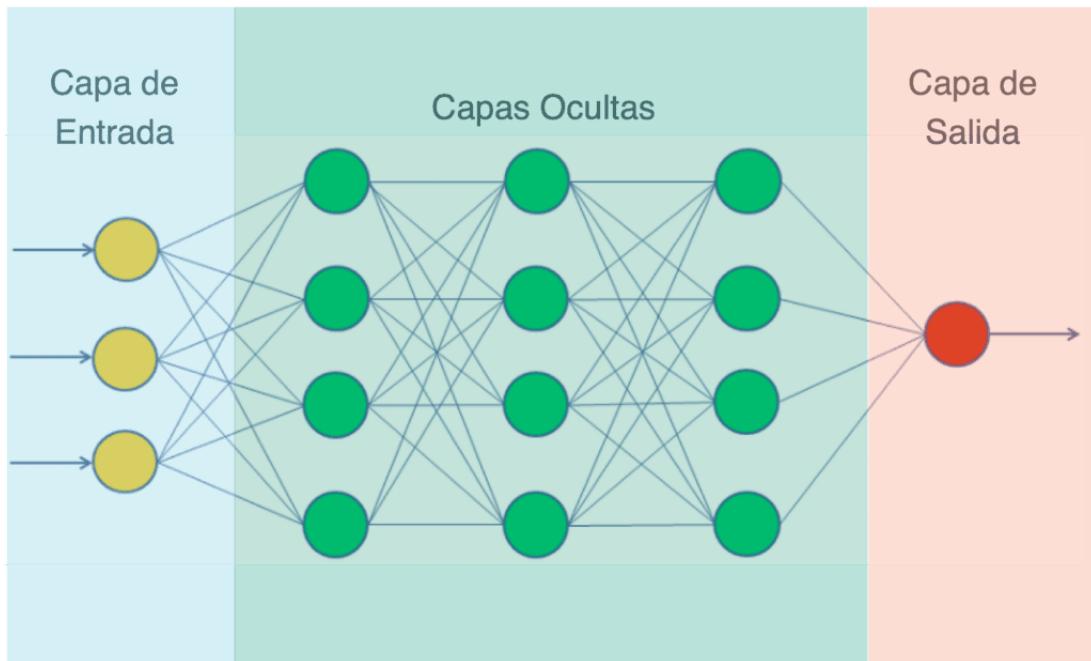


Fig. 2.2. Red de Neuronas [9]

Las redes de neuronas convolucionales son las que se utilizan para tareas de visión por computadora. Estas redes son mucho más eficientes a la hora de tratar con imágenes y esta es la razón por la que se usan en este campo del aprendizaje automático. En las primeras capas se identifican los elementos de la imagen a una escala grande, como pueden ser los bordes o los colores. A medida que se va profundizando en la red se van reconociendo partes más grandes de la imagen. Una vez se llega al final, la red es capaz de identificar el tipo de imagen. Las redes convolucionales se componen de tres capas:

- **Capa convolucional:** en esta capa se aplica el proceso conocido como convolución. Se aplica un filtro (kernel) con el fin de detectar las características que se buscan en la imagen.
- **Capa de agrupación:** en esta capa se aplica una reducción de dimensionalidad. Esta capa es muy importante ya que, aunque se pierde información, se gana en eficiencia.
- **Capa totalmente conectada:** en esta capa, a diferencia de las demás, todos los nodos están conectados a algún nodo de la capa anterior.

Existen varios tipos de redes de neuronas convolucionales, entre ellas: AlexNet [10], GoogleNet [11], ResNet [12], ZFNet [13] o VGGNet [14]. Para la realización de este proyecto se ha optado por usar la última.

2.3. Extracción de características

Para la extracción de características se ha optado por usar la arquitectura VGG16. Esta arquitectura fue propuesta por Karen Simonyan y Andrew Zisserman en el artículo “VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION” [14].

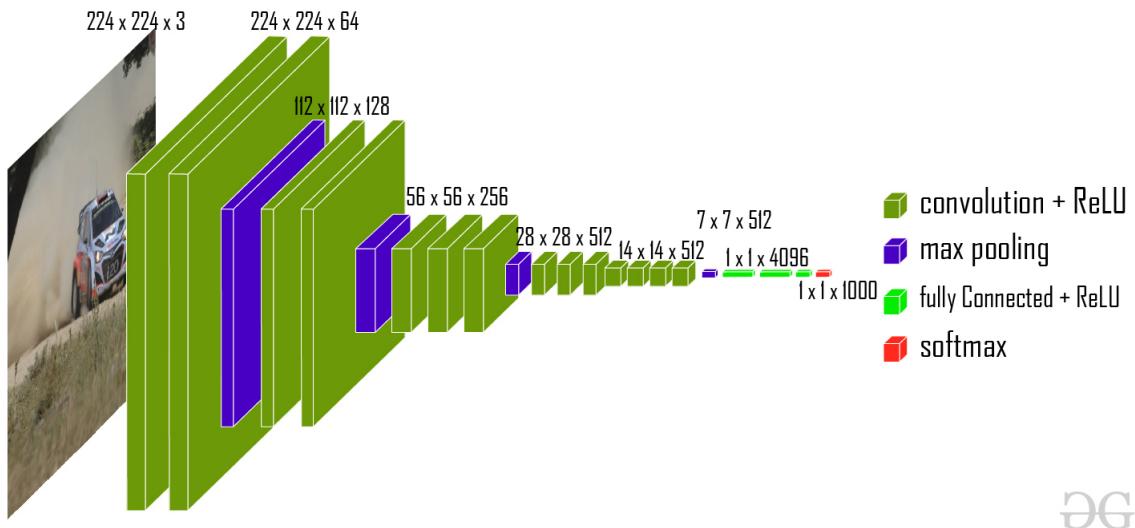


Fig. 2.3. Estructura VGG16 [15]

Este modelo consiguió un 92,7 % de aciertos en el set de datos “ImageNet” [16], que contiene más de catorce millones de imágenes. Es uno de los modelos más populares a la hora de trabajar con la extracción de características en imágenes. Es muy útil para conseguir clasificar imágenes, o en el caso de este proyecto, para calcular similitud entre ellas.

La arquitectura VGG16 se ha elegido por varias razones. En primer lugar, esta está ampliamente documentada y es fácilmente accesible a través de bibliotecas de *Python*. Aún siendo una arquitectura más simple que las demás opciones está contrastada como un modelo con un gran rendimiento en tareas de visión artificial, lo que la convierte en una elección sólida para un sistema de recomendación de imágenes. Además, esta arquitectura probó ser la más eficaz en el artículo “Image-based Product Recommendation System with Convolutional Neural Networks” [17].

2.4. Segmentación de imágenes

En cuanto a la segmentación de imágenes se refiere destaca U2-Net, que es la arquitectura que se ha elegido. Presentada por investigadores de la universidad de Canadá en el año 2020, esta arquitectura se ha expandido rápidamente en el ámbito de la visión artificial debido a su gran eficacia.

U2-Net consigue una segmentación muy precisa haciendo uso de capas convolucionales y deconvolucionales, llegando de esta manera a captar características a un nivel profundo. Tiene una gran capacidad para conservar algunos de los detalles más pequeños, lo que lo hace ideal para la segmentación de imágenes de moda.

U2-Net es una arquitectura líder en cuanto a la segmentación de imágenes. Además su uso resulta fácil en *Python*, y por estas razones se ha optado por hacer uso de ella.

2.5. Trabajos similares

Hay muchos trabajos previos similares a este. Entre ellos destacan:

- “Image-based Product Recommendation System with Convolutional Neural Networks” [17]. En este artículo se describen las técnicas utilizadas para realizar un sistema de recomendación para tiendas online basado en la similitud entre imágenes. Se prueban los algoritmos SVG, AlexNet y VGG. Tras evaluar los tres este último resulta ser el más preciso. Una vez obtenidos los vectores de las imágenes se calcula la distancia coseno entre estos para calcular la similitud entre ambas. Este trabajo se relaciona fácilmente con el propuesto, ya que resuelve el problema haciendo uso del mismo modelo de clasificación y el mismo algoritmo de recomendación.
- “Image-Based Service Recommendation System: A JPEG-Coefficient RFs Approach” [18]. Este artículo consta de dos fases. En la primera tratan de clasificar las imágenes según el tipo de producto mientras que en la segunda se crea un sistema de recomendación que trata de encontrar los productos más similares a otro dado. Para la segunda fase, que es la que aplica en nuestro caso, se usan coeficientes JPEG para sacar las características de las imágenes. Después se calcula la distancia euclíadiana entre estas. Este otro trabajo propone una manera diferente de resolver el objetivo a la utilizada en nuestro caso.

Este trabajo se diferencia en varios aspectos de los anteriores. En primer lugar, se añade el uso de máscaras para obtener recomendaciones que no se vean influenciadas por factores ajenos a la prenda en sí. Además, no solo se busca una recomendación en base a una imagen inicial, si no que también se permite al usuario seleccionar los atributos que quiere que tengan las imágenes recomendadas. A esta principal diferencia se suma

el hecho de que este proyecto no se queda solo en un modelo, también cuenta con una interfaz gráfica que permite a los usuarios hacer uso de este. Este trabajo hace uso de arquitecturas de redes convolucionales ya existentes y combina métodos ya explorados para crear un sistema de recomendación, pero no propone ningún avance desde el punto de vista de estas arquitecturas.

3. DISEÑO DEL SISTEMA

Este apartado pretende abordar el diseño del sistema, detallando como las distintas partes de este funcionan y trabajan en conjunto para resolver el problema propuesto. En primer lugar se describe el conjunto de datos empleado y posteriormente el modelo de inteligencia artificial. Después, la API y como esta sirve para comunicar el modelo con la interfaz gráfica. Esta última se presenta a través de imágenes. Finalmente, se procede a enumerar todas las tecnologías empleadas.

3.1. Conjunto de datos

Como se mencionó en el apartado “Marco Reguador” en este trabajo se hace uso del conjunto de datos “iMaterialist (Fashion) 2020 at FGVC7” [2]. Este conjunto de datos cuenta con varios archivos que se explican a continuación.

En primer lugar tenemos dos conjuntos de imágenes, “train” y “test”. El conjunto “train” cuenta con 43.793 imágenes y “test” con 3200. Ambos conjuntos contienen estas imágenes nombradas por un identificador.



Fig. 3.1. Ejemplo de imagen en el conjunto de datos

Además hay un archivo llamado “train.csv” que contiene los siguientes campos:

- **imageId**: identificador único para las imágenes. Para cada imagen hay varias entradas en el archivo, tantas como segmentaciones tenga la imagen
- **encodedPixels**: estos pixeles son la segmentación de la imagen

- **height**: altura de la imagen
- **width**: ancho de la imagen
- **classId**: identificador de la clase a la que pertenecen
- **attributesIds**: lista de identificadores de atributos que contiene este segmento de la imagen

La segmentación es de gran utilidad, ya que sirve para realizar las máscaras de las imágenes. También se destaca el campo **attributesIds** ya que gracias a este podemos filtrar las imágenes a gusto del usuario. Estos atributos vienen descritos en un archivo llamado “label_descriptions.json”. Cada atributo tiene una descripción y una supercategoría asociadas a su identificador único.

3.2. Modelo

3.2.1. Segmentación

En primer lugar se deben preprocesar las imágenes, el objetivo del trabajo es hacer recomendaciones de prendas similares, por lo tanto todo lo que no sea una prenda en la imagen puede influir negativamente en la recomendación. Por ejemplo, se podrían recomendar prendas con el fondo de la imagen similar, imágenes que tengan una misma marca de agua o imágenes en las que los modelos que las visten tengan parecidos físicos, entre otras cosas. Estos aspectos no deberían influir en la recomendación por lo que lo primero que se realiza son las máscaras de las imágenes. Para ello se utiliza el atributo **encoded-Pixels**. Con estos pixeles, que son una segmentación de las prendas, se puede obtener una imagen que consista únicamente en ropa. A continuación se muestra un ejemplo de una imagen antes y después de aplicarle la máscara.



Fig. 3.2. Ejemplo de imagen antes y después de aplicar la máscara

Como se puede observar en la figura, la máscara ha quitado todo excepto la prenda. Todas estas máscaras son pre-procesadas y almacenadas, ya que serán las que se usen para el modelo.

Esto no es suficiente, ya que el usuario proporciona una imagen que no está en la base de datos, por lo que no se tiene la segmentación de esta. Debido a este motivo es necesario hacer uso de un modelo de segmentación para generar la máscara de la imagen dada por el usuario. Hay un gran número de arquitecturas que podrían resolver este problema. A continuación se describen algunas de ellas:

- **U-Net [19]**: su nombre viene dado porque su arquitectura tiene forma de U, combina capas de convolución y deconvolución para conseguir la segmentación de las imágenes.
- **SegNet [20]**: utiliza capas de **pooling** para conseguir el objetivo de segmentar las imágenes.
- **Mask R-CNN [21]**: esta arquitectura es una combinación de **Faster R-CNN**, que sirve para la detección de objetos, con la generación de máscaras.
- **DeepLab [22]**: esta arquitectura utiliza la convolución dilatada para lograr el objetivo de segmentar las imágenes.
- **U2-Net [23]**: esta arquitectura es una extensión de **U-Net** que incluye mejoras sustanciales, mejora la precisión y el rendimiento.

Finalmente se elige hacer uso de la arquitectura **U2-Net**, que cuenta con todas las ventajas de **U-Net** pero teniendo una mayor precisión y rendimiento. La precisión es aún mayor cuando actúa sobre imágenes de alta resolución, lo que la convierte en una gran candidata para la segmentación de imágenes de moda.

U2-Net nació en el año 2020 en la universidad de Alberta, Canadá, con el artículo “U2-Net: Going Deeper with Nested U-Structure for Salient Object Detection” [23]. Este artículo fue galardonado en 2020 como el mejor en reconocimiento de patrones.

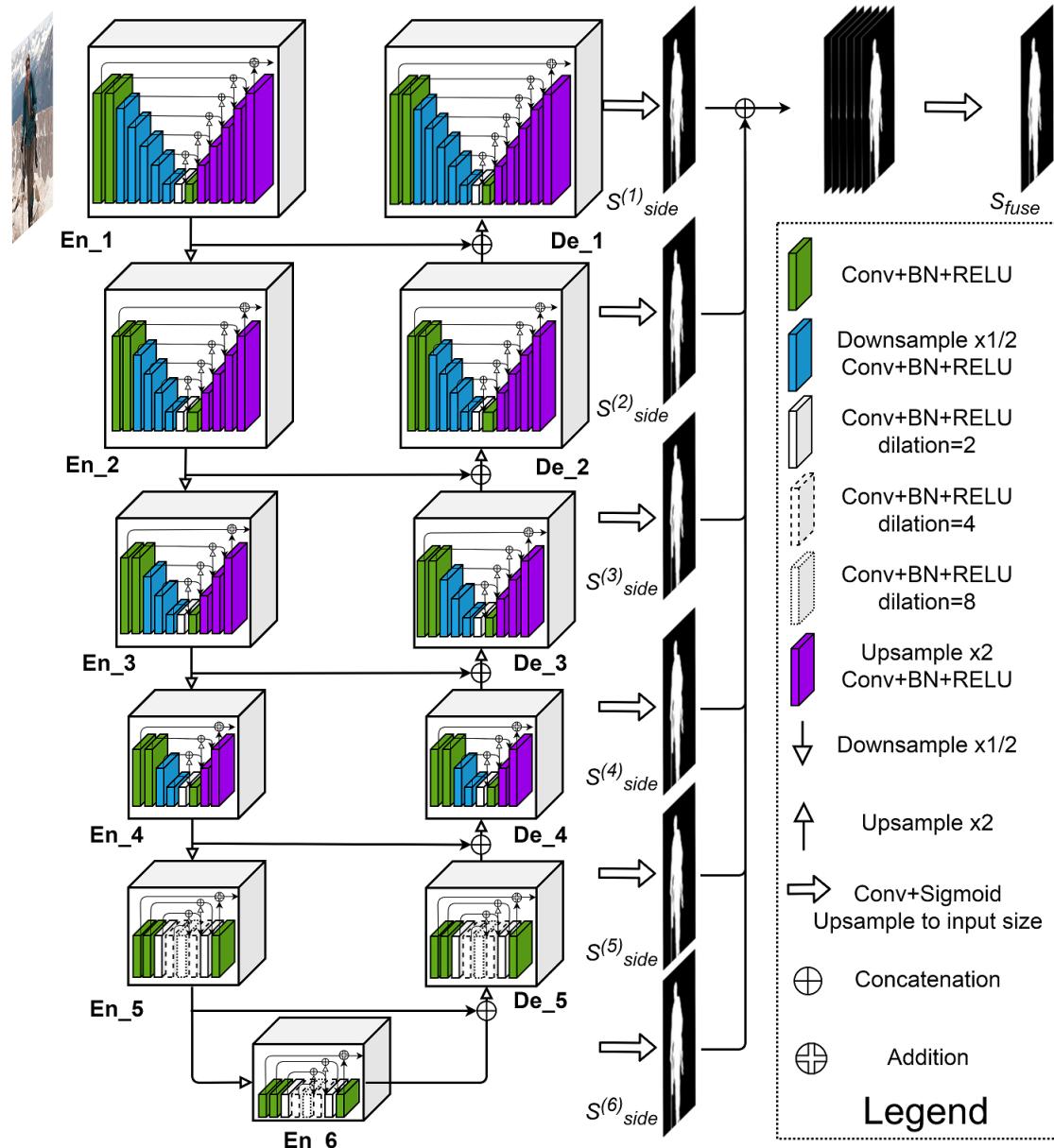


Fig. 3.3. Arquitectura U2-Net [24]



Fig. 3.4. Ejemplo de creación de máscaras con la arquitectura U2-Net [24]

3.2.2. Algoritmo de recomendación

El siguiente paso es extraer las características de todas las imágenes. Para ello se hace uso de una red de neuronas convolucional. Como ya se mencionó anteriormente se ha optado por elegir el modelo VGG16. Este modelo tiene una estructura fácil de entender y es un modelo más profundo que sus predecesores, como puede ser AlexNet, lo que permite extraer características de mayor complejidad.

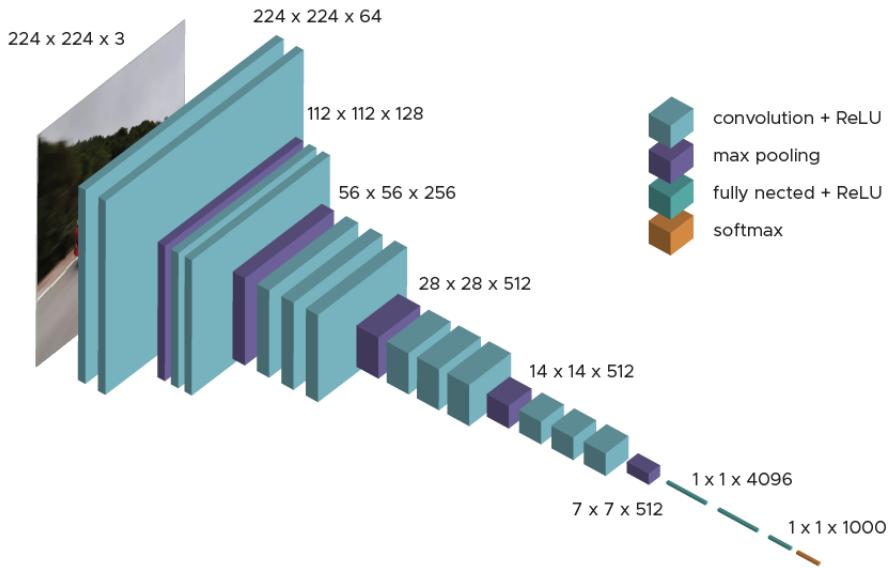


Fig. 3.5. Estructura algoritmo VGG16 [25]

El modelo hace uso de la arquitectura VGG16 y extrae el vector de características de todas las máscaras. Estos vectores son almacenados en una matriz. Cuando el usuario proporciona una imagen, se genera su máscara con el procedimiento mencionado anteriormente. Después se obtiene también su vector de características y se calcula la similitud coseno con los vectores de todas las demás imágenes. Finalmente se filtra el conjunto de datos por las prendas que contengan los atributos seleccionados por el usuario y se de-

vuelven las N imágenes con la mayor similitud, donde N es el número de imágenes que el usuario desea obtener. El siguiente diagrama resume el funcionamiento del modelo.

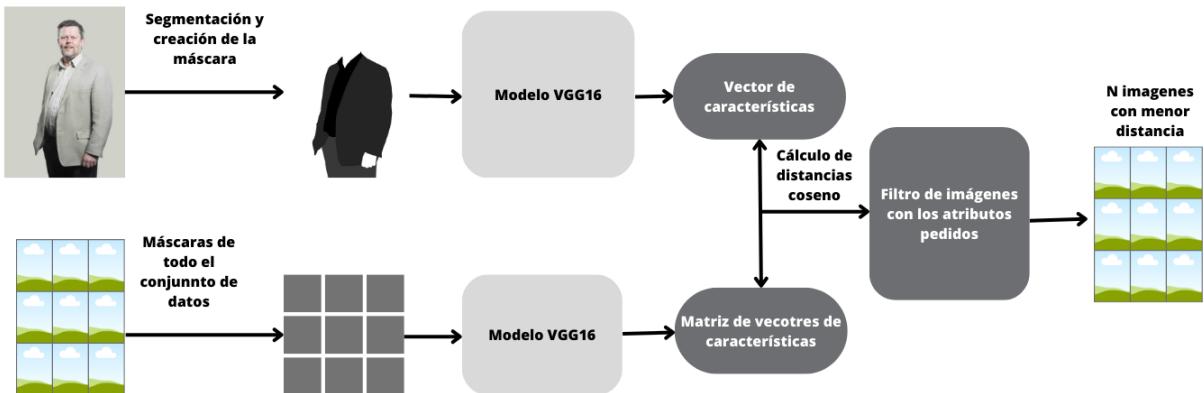


Fig. 3.6. Diagrama del funcionamiento del modelo

3.3. API

Con el fin de poder crear una interfaz gráfica fácil de usar para un usuario común es necesario tener una API que sirva el modelo. Una API es una forma de conectar diferentes aplicaciones para lograr que trabajen juntas. La API permite a la interfaz gráfica poder consumir el modelo de inteligencia artificial.

Se ha optado por hacer esta API en *Python*, ya que facilita el desarrollo al estar el modelo también hecho con este mismo lenguaje. La librería elegida para construir esta API es *Flask*, un *framework* diseñado para poder hacer aplicaciones CRUD (*Create, Read, Update, Delete*). Una aplicación CRUD puede tener varios *endpoint*, rutas a las que enviar una solicitud. Cada una de estas rutas puede realizar las acciones de crear, leer, actualizar o borrar información en la API.

En el caso de nuestra API solo han sido necesarios dos *endpoints*.

- “/recommendations”:

Usa el método *POST*. Este *endpoint* espera recibir una imagen, un número de recomendaciones y una lista de identificadores de atributos opcional. Una vez recibe estos argumentos, la API hace uso del modelo y devuelve la lista de identificadores de las imágenes que, teniendo los atributos requeridos, tengan la mayor similitud.

KEY	VALUE
<input checked="" type="checkbox"/> selectedAttributes	{"17": true}
<input checked="" type="checkbox"/> recomendationsNumber	3
<input checked="" type="checkbox"/> file	000c9b4926cd78edd4c19cbc6beba111.jpg

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```

1
2   "images": [
3     "409d0263a72abdad2af97d85131f9761.jpg",
4     "ba3bb072bf67724461ae89c2368c8be.jpg",
5     "aaf1eb0b7d382cf8bc102e3e9e0e883f.jpg"
6   ]
7

```

Fig. 3.7. Ejemplo de petición al *endpoint* “/recommendations”

- “/image/<id>”:

La API permite recuperar imágenes a partir de su identificador gracias a este *endpoint*, donde “id” es el identificador de la imagen que se desea obtener. Solo es necesario hacer una petición de tipo *GET* con un identificador válido en la ruta.

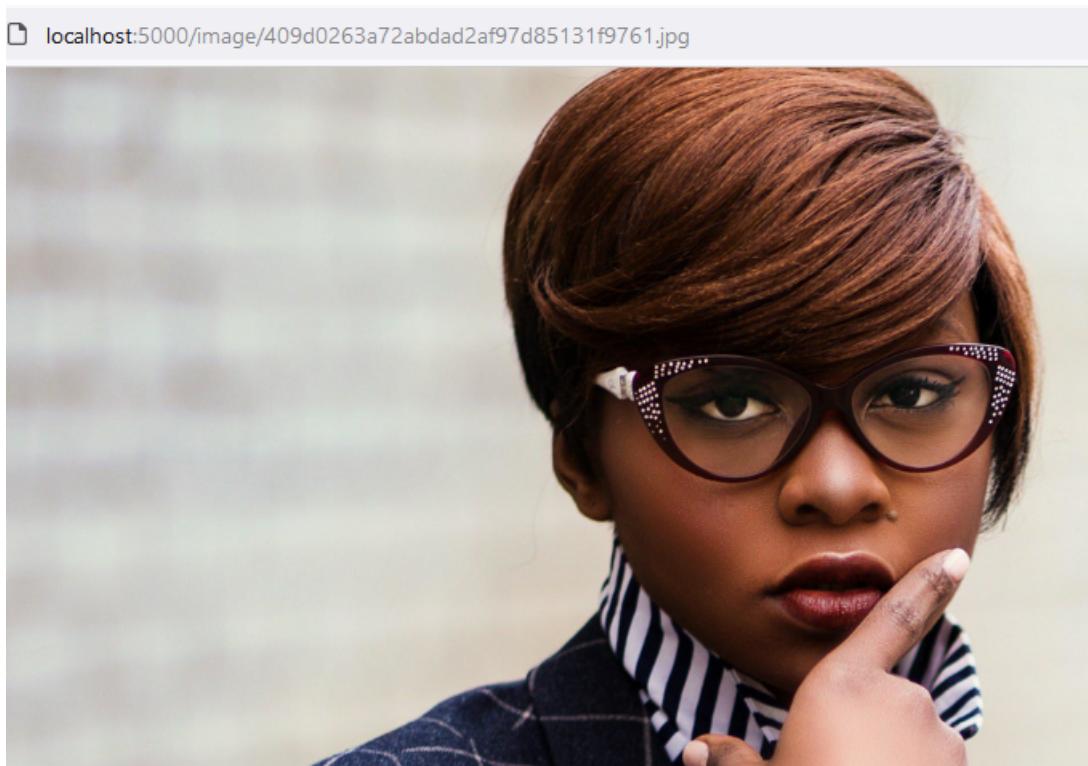


Fig. 3.8. Ejemplo de petición al *endpoint* “/image/<id>”

3.4. Interfaz

Para la interfaz gráfica se tuvieron en cuenta varias opciones. La primera de ellas fue *StreamLit*, una plataforma para desarrollar interfaces para proyectos de inteligencia artificial escrita en *Python*. Permite desplegar este tipo de aplicaciones de una manera muy sencilla, escribiendo muy pocas líneas de código. Por otro lado se pensó en usar *Gradio*, una plataforma que funciona de una manera muy similar a *StreamLit*. Ambas opciones son buenas pero al final se decidió hacer uso del *framework* de desarrollo *React*, del lenguaje *JavaScript*. Aunque las dos primeras herramientas eran interesantes para conseguir un desarrollo rápido, las opciones de interfaz quedaban limitadas.

El hecho de no poder hacer la interfaz completamente a mi gusto fue lo que me hizo decantarme por *React*, con el cual se puede hacer una web completa, con todas sus características. Además se hace uso de *TypeScript*, en lugar de *JavaScript*, ya que ofrece una mayor solidez gracias a su tipado estático. Esto ayuda a no cometer errores difíciles de depurar durante el desarrollo, aunque a veces pueda resultar tedioso el hecho de tener que marcar todos los tipos. El hecho de utilizar *React* resulta muy útil, ya que se puede organizar el código por componentes reutilizables en donde se junta el *HTML* con el *JavaScript*.

El proyecto tiene la siguiente estructura:

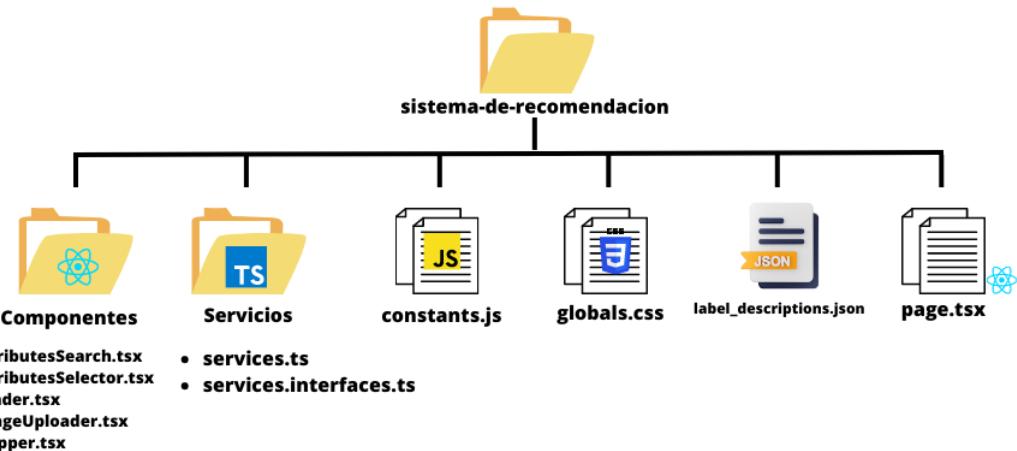


Fig. 3.9. Estructura del proyecto

- **attributtesSearch.tsx:** este componente sirve para buscar los atributos que el usuario quiera seleccionar para que aparezcan en sus recomendaciones. Cuenta con la lista completa de atributos (que son elementos seleccionables) y con un buscador, ya que hay 341 atributos diferentes.
- **attributtesSelector.tsx:** este componente contiene la búsqueda de atributos. Además cuenta con un botón para que, en caso de que el usuario así lo desee, sirve para obtener recomendaciones de prendas con todo tipo de atributos.

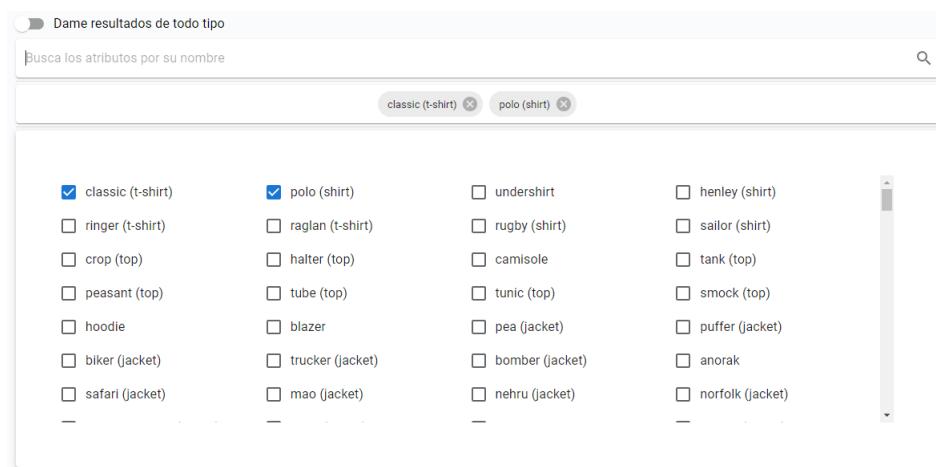


Fig. 3.10. Componente **attributesSearch**

- **header.tsx:** este es el encabezado de la página.



Fig. 3.11. Componente **header**

- **imageUploader**: este componente es una caja que permite al usuario subir la imagen que desea utilizar para obtener recomendaciones. Esta permite explorar sus archivos locales y subir un archivo que debe ser de tipo imagen.



Fig. 3.12. Componente **imageUploader**

- **stepper**: este componente sirve para pasar las pantallas del proceso de dar toda la información, ya que la aplicación consiste en una sola página.
- **services.ts**: este archivo se encarga de realizar las peticiones a la API.
- **services.interfaces.ts**: este archivo contiene las interfaces necesarias para los servicios.
- **constants.js**: en este fichero se almacenan las variables que son constantes a lo largo del proyecto.
- **globals.css**: en este fichero se guarda la configuración CSS que afecta a todo el proyecto.
- **label_descriptions.json**: este archivo se usa para asociar los identificadores de los atributos a sus descripciones.
- **page.tsx**: esta es la página principal, donde todos los componentes se unen para componer la aplicación.

Nada más cargar la página podemos ver una ventana que nos pide subir una imagen. Una vez subida la imagen se puede presionar el botón de “siguiente”, el cual nos lleva a seleccionar el número de recomendaciones, a través de una barra de selección de números. Una vez más es posible presionar el botón de “siguiente”, esta vez nos llevará a la ventana final, la selección de atributos. Esta contiene la lista completa de los atributos seleccionables. Los atributos seleccionados quedan marcados en la parte superior, desde donde se puede eliminar la selección. Además hay un buscador dada el gran número de atributos que hay. Una vez ya está todo listo se puede pulsar sobre el botón “procesar”. Este botón manda una petición a la API que, haciendo uso del modelo, devuelve los identificadores de las imágenes con mayor similitud. Con estos identificadores se vuelve a hacer una petición por cada uno de ellos para obtener sus respectivas imágenes. Estas imágenes se muestran en una galería que se puede poner a tamaño completo. Si se desea repetir el proceso basta con pulsar el botón “resetear”.

A continuación se muestra, a través de imágenes, el funcionamiento descrito anteriormente.

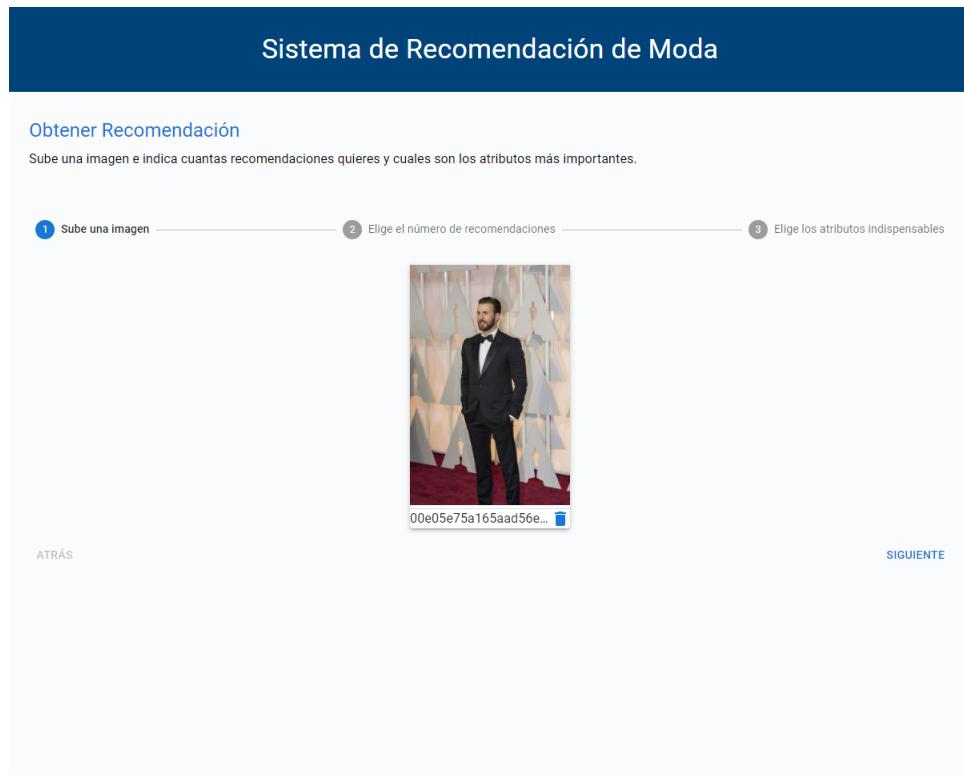


Fig. 3.13. Pantalla inicial, subida de imagen

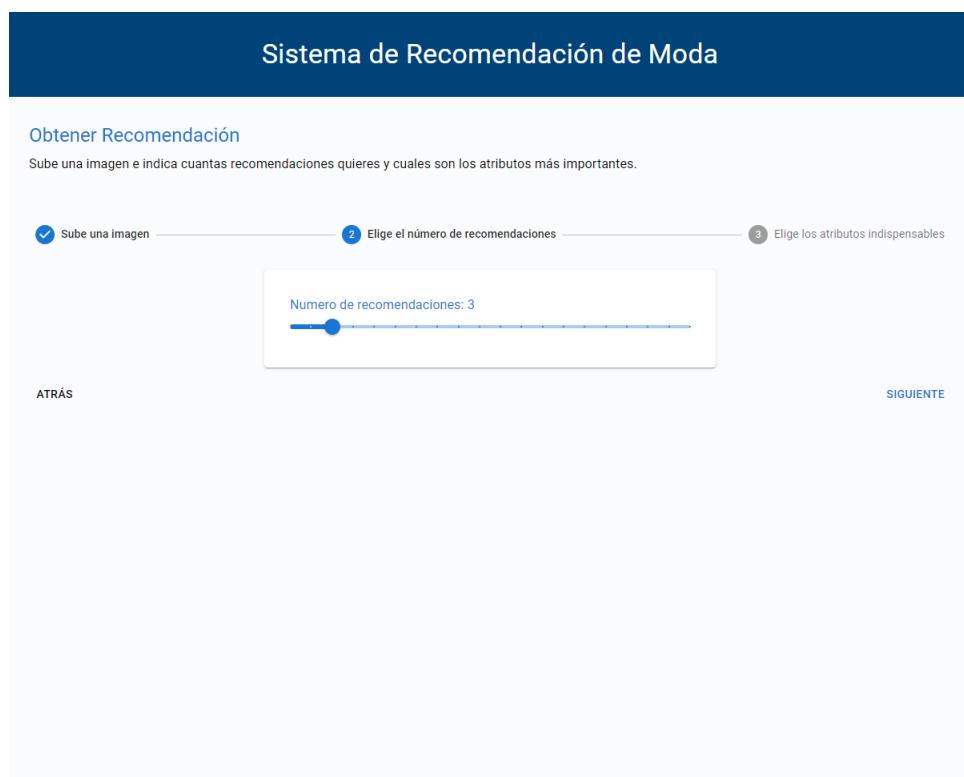


Fig. 3.14. Selección del número de recomendaciones

Sistema de Recomendación de Moda

Obtener Recomendación

Sube una imagen e indica cuantas recomendaciones quieres y cuales son los atributos más importantes.

1 Sube una imagen 2 Elige el número de recomendaciones 3 Elige los atributos indispensables

Dame resultados de todo tipo

Busca los atributos por su nombre

blazer

<input type="checkbox"/> classic (t-shirt)	<input type="checkbox"/> polo (shirt)	<input type="checkbox"/> undershirt	<input type="checkbox"/> henley (shirt)
<input type="checkbox"/> ringer (t-shirt)	<input type="checkbox"/> raglan (t-shirt)	<input type="checkbox"/> rugby (shirt)	<input type="checkbox"/> sailor (shirt)
<input type="checkbox"/> crop (top)	<input type="checkbox"/> halter (top)	<input type="checkbox"/> camisole	<input type="checkbox"/> tank (top)
<input type="checkbox"/> peasant (top)	<input type="checkbox"/> tube (top)	<input type="checkbox"/> tunic (top)	<input type="checkbox"/> smock (top)
<input type="checkbox"/> hoodie	<input checked="" type="checkbox"/> blazer	<input type="checkbox"/> pea (jacket)	<input type="checkbox"/> puffer (jacket)
<input type="checkbox"/> biker (jacket)	<input type="checkbox"/> trucker (jacket)	<input type="checkbox"/> bomber (jacket)	<input type="checkbox"/> anorak
<input type="checkbox"/> safari (jacket)	<input type="checkbox"/> mao (jacket)	<input type="checkbox"/> nehru (jacket)	<input type="checkbox"/> norfolk (jacket)
—	—	—	—

ATRÁS PROCESAR

Fig. 3.15. Selección de atributos

Sistema de Recomendación de Moda

Obtener Recomendación

Sube una imagen e indica cuantas recomendaciones quieres y cuales son los atributos más importantes.

1 Sube una imagen 2 Elige el número de recomendaciones 3 Elige los atributos indispensables

RESETEAR

Fig. 3.16. Barra de carga mientras la API procesa la solicitud

Sistema de Recomendación de Moda

Obtener Recomendación

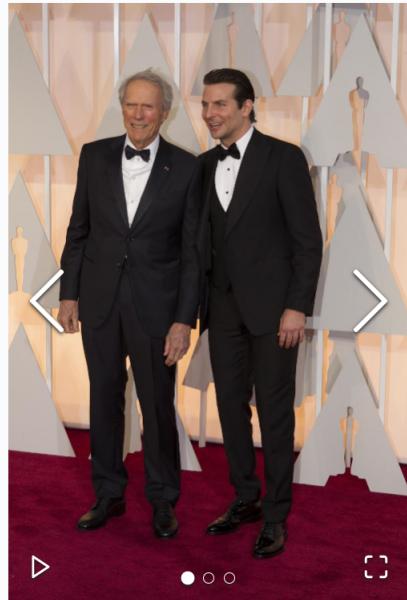
Sube una imagen e indica cuantas recomendaciones quieras y cuales son los atributos más importantes.

Sube una imagen

Elige el número de recomendaciones

Elige los atributos indispensables

Imagen original



[RESETEAR](#)

Fig. 3.17. Resultado del proceso. Recomendaciones.

3.5. Tecnologías empleadas

Para la realización de este trabajo ha sido necesario el uso tanto de hardware como de software. En la siguiente tabla se muestra el hardware utilizado.

Equipo	RAM	CPU	GPU
Ordenador de sobremesa	16 GB	AMD RYZEN 53500X	AMD Radeon RX 5500 XT
HP ENVY 13	8 GB	INTEL i7	NVIDIA GeForce MX150

TABLA 3.1. ESPECIFICACIONES DEL HARDWARE UTILIZADO

Ambos ordenadores tienen como sistema operativo *Windows*, 10 y 11 respectivamente. Se ha hecho uso del IDE *VScode*, de código abierto, para el desarrollo de tanto la API como la interfaz gráfica.

En el caso de la API se ha usado el lenguaje de programación *Python* junto con las siguientes librerías:

- **Flask:** *framework* minimalista que sirve para desarrollar aplicaciones web. En este caso sirve para comunicar el modelo con la web.
- **Pickle:** utilizada para cargar a memoria el modelo con la matriz de características extraídas de las imágenes.
- **OS:** utilizada para trabajar con los archivos del sistema. Recuperar, crear y eliminar imágenes en él.
- **JSON:** manejo de objetos de tipo JSON. Utilizada para comunicarse con la aplicación web.
- **Numpy:** cálculos optimizados con las matrices que representan las imágenes. Utilización de algunas de sus funciones matemáticas.
- **Keras:** utilizado para hacer uso de la arquitectura VGG16.
- **Tensorflow:** utilizado para cargar imágenes y transformarlas a listas.
- **Scipy:** uso de su función optimizada para calcular la distancia coseno entre vectores, ya que ofrece un rendimiento superior y se reducen los tiempos de espera.
- **Pandas:** utilizado para usar la estructura *Dataframe* que aporta una gran velocidad a la hora de trabajar con grandes cantidades de datos, como es el caso.
- **Matplotlib:** utilizado para generar gráficas que sirvieron para analizar el conjunto de datos.

La interfaz gráfica por otra parte ha sido desarrollada con *TypeScript* junto con las siguientes librerías:

- **React:** este es el *framework* principal utilizado en la interfaz gráfica. Aporta una gran facilidad para poder reutilizar código y no repetirlo. Además cuenta con multitud de librerías que resultan de gran utilidad.

- **Material UI:** librería para el *framework React* que contiene elementos gráficos de UI. Además aporta una gran usabilidad a la web.
- **Axios:** librería que sirve para realizar las peticiones web a la API.
- **React image gallery:** componente de *React* que renderiza una galería de imágenes.

4. RESULTADOS

En este apartado se analizan los resultados obtenidos a través de ejemplos de entrada y salida. Además se comparan los resultados con y sin máscara y con y sin el uso de atributos.

4.1. Sin máscaras

En un inicio el sistema de recomendación funcionaba sin máscaras. Este sistema funcionaba correctamente pero tenía ciertas desventajas:

- En ocasiones la recomendación se fijaba en el color de fondo de las imágenes en lugar de las prendas en sí.
- Tenía en cuenta los modelos que vestían las prendas.
- Recomendaba prendas con el mismo color.
- Tenía en cuenta marcas de agua en las imágenes.

Aun así se obtienen recomendaciones con un alto nivel de similitud coseno. A continuación se muestran algunos ejemplos de entrada y salida con sus correspondientes porcentajes de similitud.

Imagen original



Recomendaciones



Fig. 4.1. Recomendaciones sin uso de máscaras

Recomendación	Similitud Coseno
Primera recomendación	0,7575
Segunda recomendación	0,7542
Tercera recomendación	0,7465

TABLA 4.1. TABLA DE SIMILITUD DE RECOMENDACIONES



Fig. 4.2. Recomendaciones sin uso de máscaras

Recomendación	Similitud Coseno
Primera recomendación	0,7951
Segunda recomendación	0,7784
Tercera recomendación	0,7723

TABLA 4.2. TABLA DE SIMILITUD DE RECOMENDACIONES

Estas recomendaciones ejemplifican el problema presentado anteriormente. Todas ellas tienen una marca de agua en la esquina inferior derecha, todas ocurren en un desfile de moda, todas están vistiendo el color negro. Este hecho no impide que se obtengan porcentajes de similitud bastante altos, aun así el hecho de no hacer uso de las máscaras presenta los problemas descritos anteriormente.

4.2. Con máscaras

En este apartado se llevará a cabo un análisis similar al realizado en el anterior, pero esta vez haciendo uso de las máscaras. El hecho de utilizar máscaras reporta varios beneficios, entre ellos:

- Eliminar las recomendaciones basadas en el fondo de la imagen.
- Eliminar las recomendaciones basadas en el color de la prenda.
- Eliminar las recomendaciones basadas en el modelo que viste las prendas.
- Eliminar recomendaciones basadas en marcas de agua que pueda tener la imagen.

Es decir, las máscaras permiten al sistema recomendar únicamente en base a las prendas, sin tener en cuenta otros factores propios de las imágenes. A continuación, algunos ejemplos.



Fig. 4.3. Recomendaciones con uso de máscaras

Recomendación	Similitud Coseno
Primera recomendación	0,8856
Segunda recomendación	0,8854
Tercera recomendación	0,8657

TABLA 4.3. TABLA DE SIMILITUD DE RECOMENDACIONES



Fig. 4.4. Recomendaciones con uso de máscaras

Recomendación	Similitud Coseno
Primera recomendación	0,8899
Segunda recomendación	0,8803
Tercera recomendación	0,8766

TABLA 4.4. TABLA DE SIMILITUD DE RECOMENDACIONES

Es fácil observar que los resultados de hacer uso de la máscara son claramente beneficiosos para el modelo de recomendación de imágenes. En primer lugar, las prendas recomendadas no tienen el mismo color que la original, tampoco la misma marca de agua o fondo. Además, los modelos no tienen ningún parecido entre sí. Otro indicador de la mejora de la recomendación es el porcentaje de similitud, que sube en torno al 10 %. Y, aunque puede tratarse de un aspecto subjetivo, las prendas son más parecidas entre sí, aunque no tengan el mismo color.

Similitud Coseno Media (%)

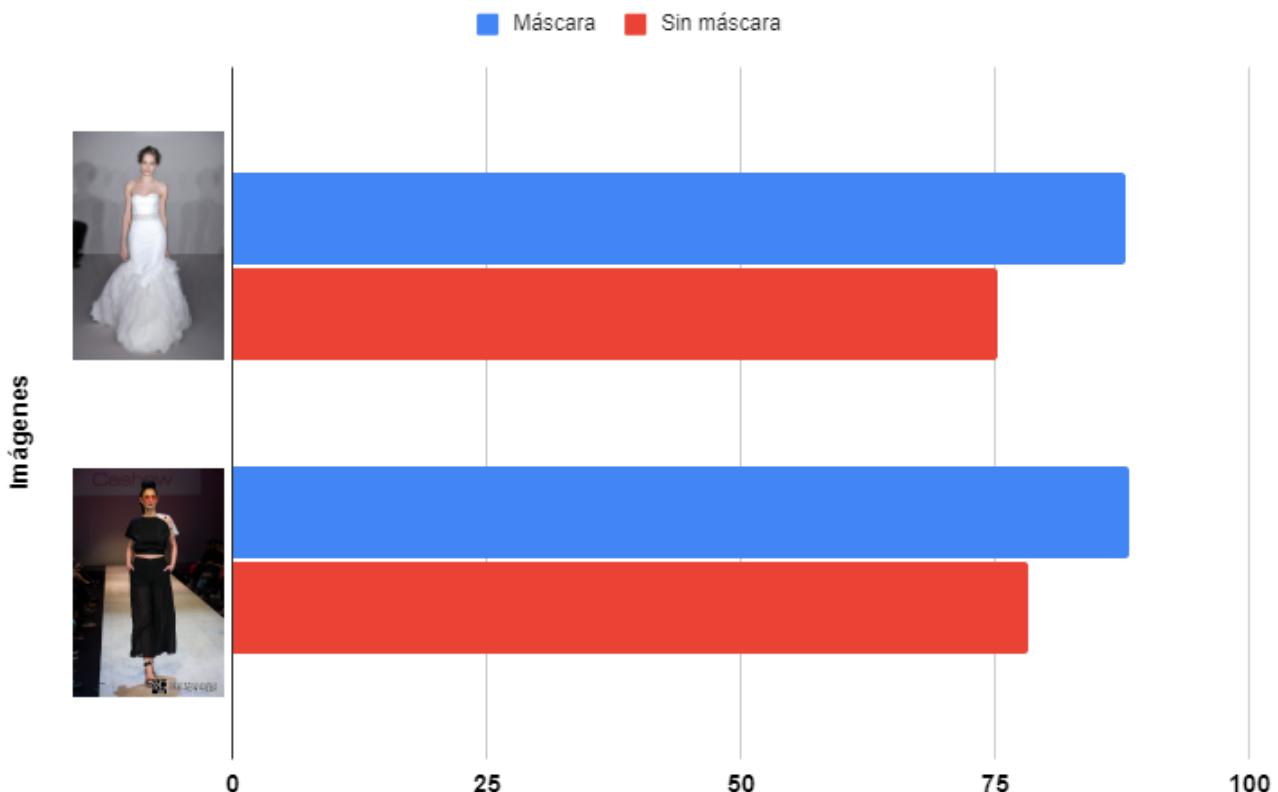


Fig. 4.5. Gráfica de Similitud Coseno

Similitud Coseno Media	
Con Máscara	Sin Máscara
	0,8789
	0,7527
	0,8823
	0,7819

TABLA 4.5. MEDIAS DE SIMILITUD COSENO DE IMÁGENES

4.3. Uso de atributos

En este apartado se analiza el uso de los atributos en las recomendaciones y como estos afectan a la calidad de estas. En el siguiente ejemplo se selecciona el atributo “blazer” (chaqueta) para exemplificar le uso de estos.

Imagen original



Recomendaciones



Fig. 4.6. Recomendaciones con selección del atributo “blazer”

Recomendación	Similitud Coseno
Primera recomendación	0,8779
Segunda recomendación	0,8723
Tercera recomendación	0,8679

TABLA 4.6. TABLA DE SIMILITUD DE RECOMENDACIONES

Se puede observar que los porcentajes de similitud coseno siguen siendo buenos, a la altura de los anteriores. Además podemos comprobar que todas las recomendaciones tienen el atributo “blazer” solicitado.

Ahora vamos a probar a añadir otro atributo a la consulta. Este atributo será “loose (fit)”.

Imagen original



Recomendaciones



Fig. 4.7. Recomendaciones con selección de los atributos “blazer” y “loose (fit)”

Recomendación	Similitud Coseno
Primera recomendación	0,8406
Segunda recomendación	0,8405
Tercera recomendación	0,8357

TABLA 4.7. TABLA DE SIMILITUD DE RECOMENDACIONES

Por último vamos a añadir el atributo “geometric textile pattern”.



Fig. 4.8. Recomendaciones con selección de los atributos “blazer”, “loose (fit)” y “geometric textile pattern”

Recomendación	Similitud Coseno
Primera recomendación	0,6712
Segunda recomendación	0,5589

TABLA 4.8. TABLA DE SIMILITUD DE RECOMENDACIONES

Como podemos comprobar a través de estos resultados, el uso de los atributos es un arma de doble filo. Resulta muy útil, pero se sacrifica porcentaje de similitud. Esto se debe a que cuantos más atributos queramos menos imágenes hay en el *dataset* que contengan estos atributos, de hecho en el último ejemplo solo hay dos de ellas. Si tuvieramos un *dataset* de un tamaño mucho mayor, como así sería en un ejemplo práctico en la vida real, este problema no se daría tan pronto. Aun así la selección de atributos resulta de gran utilidad.

5. CONCLUSIONES

En este capítulo se llevarán a cabo las conclusiones generales del trabajo. Se hará un análisis que consistirá en comprobar si los objetivos iniciales han sido cumplidos.

El objetivo principal de este trabajo era crear una interfaz gráfica que fuera capaz de recomendar imágenes de ropa a partir de, no solo una imagen dada por el usuario, sino también una serie de atributos deseados en esta recomendación. Este objetivo se ha podido cumplir por completo. Además se planteaban los siguientes sub-objetivos:

- Desarrollar un modelo de inteligencia artificial que sea capaz de recomendar imágenes en base a atributos multimodales y otra imagen.
- Crear una API que permita la comunicación entre el modelo previamente mencionado y una interfaz.
- Diseñar una interfaz intuitiva que permita a los usuarios cargar una imagen, seleccionar los atributos de interés y visualizar los resultados proporcionados por el modelo.

Se ha conseguido construir una aplicación web que lleva a cabo todas las tareas descritas anteriormente.

La arquitectura VGG16 ha resultado ser útil para extraer las características de las imágenes. Esta ha sido efectiva y eficaz. El uso de máscaras, aunque haya ocasionado una solución más compleja debido a la tarea de segmentación, ha resultado aportar claras mejoras al modelo de recomendación. Las imágenes recomendadas dejan de estar influidas por aspectos ajenos a las prendas, como pueden ser el fondo de la imagen, marcas de agua o el aspecto físico de los modelos. Gracias a la combinación de la segmentación de las imágenes para la realización de las máscaras y la extracción de características de las imágenes, para el posterior cálculo de su similitud coseno, se ha podido llegar a obtener unos valores de similitud altos para las recomendaciones.

El uso de los atributos también ha permitido al modelo filtrar el conjunto de datos y recomendar únicamente las prendas que contienen estos. Aun así, cuando se usan excesivos atributos juntos o se seleccionan algunos poco comunes, el modelo puede tener problemas recomendando debido al tamaño, no demasiado grande, del conjunto de datos.

La principal limitación de este sistema de recomendación es el número de imágenes en el conjunto de datos. Añadir imágenes a este conjunto es costoso ya que se necesita ayuda de profesionales de la moda que clasifiquen correctamente los atributos de estas antes de añadirlas al conjunto. Además, cada vez que se añadan imágenes al conjunto de datos sería necesario volver a computar los vectores característicos, lo que puede hacer el proceso lento. Otra limitación de este trabajo es la posible falta de comprensión semántica

de las imágenes por parte de la red neuronal. Esto puede provocar que en algunos casos las recomendaciones no sean las óptimas. Por último, este sistema cuenta con las limitaciones de no poder usar más de una única imagen como entrada para la recomendación ni permitir hacer una selección de colores deseados en esta.

En conclusión, este trabajo cumple con los objetivos propuestos, brindando una interfaz que es capaz de servir un sistema de recomendación de imágenes con un alto porcentaje de similitud en las recomendaciones que aporta. Además, se ha podido introducir el uso de atributos, lo cual permite a los usuarios obtener recomendaciones más precisas y ajustadas a sus deseos. Si se contara con un conjunto de datos más grande esta última funcionalidad sería aún más precisa.

6. TRABAJOS FUTUROS

Este trabajo tiene mucho margen de mejora y se pueden plantear una variedad de futuros trabajos a partir de este.

En primer lugar, se puede Ampliar la base de datos sobre la que se trabaja, ya que a mayor cantidad de datos disponibles mayor será la calidad de las recomendaciones. Esto ayudaría a corregir la limitación que se comentó en las conclusiones. Además, se podría llevar a cabo la implementación de una red neuronal más enfocada en la ropa y no tan generalista como la arquitectura VGG16. Entrenar una red que se centre en prendas en específico podrá ayudar a obtener mejores recomendaciones.

Con el fin de resolver la limitación de la falta de comprensión semántica de la red neuronal al extraer las características de la imagen se podría añadir *feedback* de usuarios reales durante el entrenamiento de la red. Esto podría ser de gran utilidad, ya que la calidad de las recomendaciones es muchas veces es subjetiva al gusto del usuario.

Otro posible trabajo futuro es permitir la posibilidad al usuario de utilizar la cámara del dispositivo desde el que se acceda a la aplicación web, en lugar de tener que subir una foto ya hecha, para así hacer más fácil el uso de la interfaz. Además, se podría permitir subir varias imágenes, en lugar de solo una, para obtener resultados conjuntos. Por último, sería de gran utilidad permitir a los usuarios seleccionar los colores que desean obtener en sus recomendaciones.

7. PLANIFICACIÓN Y PRESUPUESTOS

7.1. Planificación

En este apartado se presenta la planificación que se ha llevado a cabo para la realización de este proyecto. La planificación ha consistido en las siguientes etapas:

- **Análisis del conjunto de datos.**

Esta etapa consistió en analizar el conjunto de datos, prepararlo para su uso y ver como se podía hacer uso de este. Se llevaron a cabo estadísticas que ayudaron al posterior diseño de la solución. Esta fase duró siete días.

- **Análisis de las técnicas posibles para la implementación del modelo y de la aplicación web.**

En esta etapa se analizan las diferentes tecnologías posibles para la resolución del problema. Se investiga cuales son las mejores técnicas para el modelo de segmentación, de extracción de características y se exploran las diferentes tecnologías web. Esta fase duró siete días.

- **Diseño de la solución.**

Se diseña la mejor solución a partir del análisis de tecnologías previamente hecho. Esta fase duró once días

- **Desarrollo, pruebas y mejoras.**

Esta es la etapa más larga del proyecto, que consiste en el desarrollo. Una vez se finaliza el desarrollo del proyecto se itera a través de pruebas y se mejora la solución. Esta fase duró cincuenta días.

- **Elaboración de la memoria.**

Esta etapa consiste en documentar el trabajo realizado y duró treinta días.

Dadas estas etapas se puede estimar que el proyecto se ha elaborado en un total de 105 días. Cada uno de estos días se ha trabajado un total de ocho horas, lo que supone que el proyecto ha requerido 840 horas de trabajo. En la siguiente tabla se muestran las fechas detalladas de la planificación del proyecto.

Fase	Duración (días)	Inicio	Fin
Análisis del conjunto de datos	7	15/05/2023	22/05/2023
Análisis de técnicas y algoritmos	7	23/05/2023	30/05/2023
Diseño de la solución	11	31/05/2023	11/06/2023
Desarrollo, pruebas y mejoras	50	12/06/2023	31/07/2023
Elaboración de la memoria	30	01/08/2023	31/08/2023

TABLA 7.1. PLANIFICACIÓN DEL PROYECTO

7.2. Presupuestos

7.2.1. Coste del personal

Este trabajo ha sido realizado por un solo ingeniero. Un ingeniero informático junior tiene un sueldo promedio en España de 2.208 €/mes [26], lo que supone un sueldo de 13,8 €/hora. Se han trabajado ocho horas diarias por lo que el sueldo diario sería de 110,4 €.

Empleado	Días de trabajo	Sueldo diario	Coste total
Ingeniero junior	105	110,4 €	11.592 €

TABLA 7.2. PLANIFICACIÓN DEL PROYECTO

7.2.2. Coste del equipo

El coste del equipo consiste en un ordenador de sobremesa y un portátil, teniendo en cuenta su uso. Para el cálculo del coste estimado teniendo en cuenta su uso se aplica la siguiente fórmula:

$$\text{Coste imputable} = \frac{\text{Tiempo de uso}}{\text{Tiempo de vida}} * \text{Coste total} \quad (7.1)$$

- **Ordenador de sobremesa.** Uso de ochenta horas con un coste total de 450 €
- **Portátil.** Uso de veinticinco horas con un coste total de 849,25 €
- **Teclado.** Uso de ochenta horas con un coste de 95 €
- **Monitor.** Uso de ochenta horas con un coste 240 €

Concepto	Coste total (€)	Horas de uso	Tiempo de vida	Coste imputable (€)
Sobremesa	450	80	1.200	30
Portátil	849,25	25	800	26,54
Teclado	95	No aplica	No aplica	95
Monitor	240	No aplica	No aplica	240
Total	1634,25			390,54

TABLA 7.3. COSTES MATERIALES

7.2.3. Costes directos

Los costes directos incluyen los costes del personal y los costes del equipo.

Descripción	Coste (€)
Costes de personal	11.592
Costes de equipo	390,54
Total	11.982,54

TABLA 7.4. COSTES DIRECTOS

7.2.4. Costes indirectos

Los costes indirectos incluyen gastos como agua, luz, gas, alquiler. Se estima que los costes indirectos son un 15 % del total de los costes directos, lo que supone un total de 1797,38 €.

7.2.5. Costes totales

Los costes totales son la suma de todos los costes descritos anteriormente.

Descripción	Coste (€)
Costes directos	11.982,54
Costes indirectos	1.797,38
Total	13.779,92

TABLA 7.5. COSTES DIRECTOS

Por lo tanto se concluye que el coste total del proyecto estimado es de 13.779,92 €.

BIBLIOGRAFÍA

- [1] A. Orús, *Peso de las ventas moda online sobre el total de los ingresos generados por las ventas de ropa en España de 2012 a 2022*, 2023. [En línea]. Disponible en: <https://es.statista.com/estadisticas/1027850/cuota-de-venta-de-prendas-de-ropa-por-internet-en-espana/>.
- [2] C. Kaeser-Chen et al., *iMaterialist (Fashion) 2020 at FGVC7*, 2020. [En línea]. Disponible en: <https://www.kaggle.com/competitions/imaterialist-fashion-2020-fgvc7/data>.
- [3] E. J. del Estado, *Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales*, 2021.
- [4] *Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia*. 1996. [En línea]. Disponible en: <https://www.boe.es/eli/es/rdlg/1996/04/12/1/con>.
- [5] G. de España, *Ley 14/2011, de 1 de junio, de la Ciencia, la Tecnología y la Innovación*, 2011.
- [6] M. Xirau, *El sector de la moda en España, en cifras*, 2020. [En línea]. Disponible en: <https://forbes.es/empresas/78279/el-sector-de-la-moda-en-espana-en-cifras/>.
- [7] A. Muñoz, *Cómo los sistemas de recomendación pueden ayudarte a conseguir más ventas*, 2021. [En línea]. Disponible en: <https://blog.saleslayer.com/es/sistemas-recomendacion-ecommerce>.
- [8] *Detección de caras con el software OpenCV*, 1998. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Face_detection_example_opencv.jpg.
- [9] L. González, *¿Qué son las Redes Neuronales Artificiales?* 2021. [En línea]. Disponible en: <https://aprendeia.com/que-son-las-redes-neuronales-artificiales/>.
- [10] A. Krizhevsky, I. Sutskever y G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, 2012.
- [11] C. Szegedy et al., “Going deeper with convolutions,” en *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1-9.
- [12] K. He, X. Zhang, S. Ren y J. Sun, “Deep residual learning for image recognition,” en *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

- [13] M. D. Zeiler y R. Fergus, “Visualizing and understanding convolutional networks,” en *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I* 13, Springer, 2014, pp. 818-833.
- [14] K. Simonyan y A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014.
- [15] VGG-16 | CNN model. [En línea]. Disponible en: <https://www.geeksforgeeks.org/vgg-16-cnn-model/>.
- [16] J. Deng et al., “ImageNet: A large-scale hierarchical image database,” en *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248-255. doi: [10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- [17] L. Chen, F. Yang y H. Yang, “Image-based product recommendation system with convolutional neural networks,” 2017.
- [18] F. Ullah, B. Zhang y R. U. Khan, “Image-based service recommendation system: A JPEG-coefficient RFs approach,” *IEEE access*, vol. 8, pp. 3308-3318, 2019.
- [19] O. Ronneberger, P. Fischer y T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” en *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III* 18, Springer, 2015, pp. 234-241.
- [20] V. Badrinarayanan, A. Kendall y R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, n.º 12, pp. 2481-2495, 2017.
- [21] K. He, G. Gkioxari, P. Dollár y R. Girshick, “Mask r-cnn,” en *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961-2969.
- [22] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy y A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, n.º 4, pp. 834-848, 2017.
- [23] X. Qin et al., “U2-Net: Going deeper with nested U-structure for salient object detection,” *Pattern recognition*, vol. 106, p. 107404, 2020.
- [24] X. Qin et al., “U-2-Net,” 2022. [En línea]. Disponible en: <https://github.com/xuebinqin/U-2-Net>.
- [25] “VGG: ¿Qué es este modelo? ¡Daniel te lo cuenta todo!,” 2022. [En línea]. Disponible en: <https://datascientest.com/es/vgg-que-es-este-modelo-daniel-te-lo-cuenta-todo>.
- [26] “Salario medio para Ingeniero Informático en España, 2023,” 2023. [En línea]. Disponible en: <https://es.talent.com/salary?job=ingeniero+inform%C3%A1tico>.

- [27] *¿Qué son las redes neuronales convolucionales?* [En línea]. Disponible en: <https://www.ibm.com/es-es/topics/convolutional-neural-networks>.

ANEXO A: ENGLISH SUMMARY

Introduction

Work motivation

The fashion industry has always been one of the most powerful in our society. This industry has been adapting to the rise of information technologies, and in the year 2022, online clothing sales accounted for 21.1 % of total sales in our country.

This is where recommendation systems come into play. A fashion recommendation system helps people find garments that align with their preferences. Customers can enhance their shopping experience and be content with their choices by receiving personalized and accurate recommendations. As a result, there is an increase in sales for fashion-focused websites. While an image-based recommendation system is useful, incorporating the option to specify garment attributes further enhances it.

Enabling users to receive recommendations saves them a significant amount of time that would otherwise be spent on potentially tedious searches. In conclusion, the creation of a multimodal attribute-based fashion recommendation system proves highly beneficial for both online store owners, as it improves their sales, and for users, who find it easier to locate the garments they are seeking, thus saving a considerable amount of time.

Objectives

The goal of this work is to create a web interface capable of recommending clothing images with a certain similarity based on another image provided by the user. Additionally, users will be able to specify the attributes they desire in the output images (for example: long sleeves, loose fit, etc.).

With this being the primary objective, we can distinguish the following sub-objectives:

- Develop an artificial intelligence model capable of recommending images based on multimodal attributes and another image.
- Create an API that facilitates communication between the aforementioned model and an interface.
- Design an intuitive interface that allows users to upload an image, select desired attributes, and view the results provided by the model.

These objectives aim to address the presented problem, resulting in a graphical web

interface that empowers users to obtain clothing recommendations based on garments they like and the attributes they consider.

Legal framework

For the execution of this project, a set of images from the competition iMaterialist (Fashion) 2020 at FGVC7^[2] is utilized. These images contain real people's faces, so it is necessary to consider the Organic Law 3/2018 of December 5, regarding the Protection of Personal Data and the guarantee of digital rights [3]. Nonetheless, the dataset has been appropriately cited. For the same reason, attention must be paid to the Intellectual Property Law [4]. Furthermore, if we regard this as a research work, it would be necessary to take into account the Law 14/2011 of June 1, regarding Science, Technology, and Innovation [5].

The tools used for the development of this project are open-source and freely available, such as Python, Visual Studio Code, Tensorflow, Keras, and React.

Socio-economic framework

As mentioned previously in the motivation section, the fashion industry is one of the largest on a global scale. It generates significant interest and constitutes 2.8 % of the Spanish GDP and 4 % of the labor market [6]. This is why a fashion recommendation system holds great interest and social impact. Some of the benefits are as follows, according to Alma Muñoz's article "How recommendation systems can help you achieve more sales" [7]:

- "Between 15 % and 45 % increase in conversions."
- "25 % average increase in the average purchase value."
- "Extension of the customer's lifecycle."
- "Generation of buyer loyalty."

Furthermore, Alma mentions that "Amazon states that around 35 % of its revenue is attributed to product recommendations."

Specifically, this product aims to be a fashion recommendation system. Therefore, it could be beneficial for any company engaged in online clothing sales due to the aforementioned benefits. Moreover, implementing this system on a clothing sales website provides a significant advantage over competitors.

A recommendation system also holds great societal relevance. Users of such systems can not only save substantial amounts of time but also discover garments more aligned

with their taste and personality. This results in greater customer satisfaction, as they need to invest less time in finding better products.

In conclusion, this project offers numerous economic advantages for both sellers and buyers, saving time and achieving greater satisfaction with purchases. In the subsequent section, the planning and budget will be detailed.

Structure of the document

The document is presented with a cover page, followed by an abstract along with keywords. Next, there is a general table of contents, a list of figures, and an acronym list. From here on, the document is divided into seven marked chapters:

- **Introduction:** In this section, both the motivation for undertaking the work and the objectives it aims to achieve are presented. It also includes the regulatory framework and the socioeconomic context.
- **State of the Art:** An analysis of the current state related to the project's objective is elaborated upon. Similar works and the state of the employed technologies are examined. Additionally, the argument for how this project distinguishes itself from previously conducted works is provided.
- **System Design:** The solution to the proposed problem is explained, divided into its respective pre-established sub-objectives.
- **Results:** The obtained results are presented visually, with examples. Furthermore, their quality is analyzed using similarity percentages.
- **Conclusions:** A reflection that evaluates whether the proposed objectives have been met.
- **Future Work:** Possible continuations and improvements that can be undertaken based on this work are analyzed.
- **Planning and Budgets:** A summary of how the project has been planned and an explanation of the budget calculations are provided.

State of the art

In this chapter, the current state of the project will be analyzed, taking a journey through the necessary technologies for its realization. Furthermore, an analysis of similar projects will be conducted, along with an argument regarding how this one can distinguish itself from them.

Computer vision

Computer vision is the discipline of machine learning that deals with images. Through it, an attempt is made to computationally understand images. Among other things, computer vision is used for object recognition, image restoration, facial recognition, and a myriad of other tasks. Given that this project involves working with images, the importance of this discipline is paramount.

Computer vision has made significant progress in recent years, primarily due to improvements in hardware. We are now capable of performing a greater number of calculations on a larger amount of data, thanks to a significant increase in computing power. These advancements enable deeper investigation and exploration in this field.

With the advent of neural networks, it has become possible to perform complex tasks such as image classification, object detection, and segmentation. Convolutional neural networks have become the standard in this scientific field. They are capable of recognizing visual patterns that would be challenging, if not impossible, for a human to perceive.

Convolutional networks

Neural networks are composed of different layers, which in turn are made up of different neurons. All networks have an input layer, one or more intermediate layers, and an output layer. The nodes in the layers are interconnected, and they can activate or not, depending on the outcome of their associated function, for the node to which they are connected.

Convolutional neural networks are the ones used for computer vision tasks. These networks are much more efficient when dealing with images, which is why they are employed in this field of machine learning. In the initial layers, elements of the image are identified at a larger scale, such as edges or colors. As the network goes deeper, larger parts of the image are recognized. Upon reaching the end, the network can identify the type of image. Convolutional networks consist of three layers:

- **Convolutional Layer:** In this layer, the process known as convolution is applied. A filter (kernel) is used to detect the desired features in the image.
- **Pooling Layer:** In this layer, dimensionality reduction is applied. This layer is crucial because, although information is lost, efficiency is gained.
- **Fully Connected Layer:** In this layer, unlike the others, all nodes are connected to nodes in the previous layer.

There are several types of convolutional neural networks, including AlexNet, GoogleNet, ResNet, ZFNet, and VGGNet. For the completion of this project, the latter has been chosen.

Characteristics extraction

For feature extraction, the VGG16 architecture has been chosen. This architecture was proposed by Karen Simonyan and Andrew Zisserman in the article titled “VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION” [14].

This model achieved a 92.7% accuracy on the “ImageNet” dataset, which contains over fourteen million images. It is one of the most popular models for working with feature extraction in images. It is highly useful for classifying images or, in the case of this project, calculating similarity between them.

The VGG16 architecture has been selected for several reasons. Firstly, it is extensively documented and easily accessible through *Python* libraries. Despite being a simpler architecture than other options, it has been proven to have excellent performance in tasks related to computer vision, making it a solid choice for an image recommendation system. Furthermore, this architecture demonstrated its effectiveness in the article titled “Image-based Product Recommendation System with Convolutional Neural Networks” [18].

Images segmentation

When it comes to image segmentation, U2-Net stands out as the chosen architecture. Presented by researchers from a Canadian university in the year 2020, this architecture has rapidly gained prominence in the field of computer vision due to its high effectiveness.

U2-Net achieves highly precise segmentation by utilizing convolutional and deconvolutional layers, enabling it to capture features at a deep level. It has a remarkable ability to preserve fine details, making it well-suited for fashion image segmentation.

U2-Net is a leading architecture in the realm of image segmentation. Furthermore, its usage is straightforward in Python, and for these reasons, the decision has been made to employ it.

Similar work

There are many previous works similar to this one. Among them, the following stand out:

- “Image-based Product Recommendation System with Convolutional Neural Networks” [17]. This article describes the techniques used to create a recommendation system for online stores based on image similarity. The SVG, AlexNet, and VGG algorithms are tested. After evaluating all three, the latter proves to be the most accurate. Once image vectors are obtained, cosine distance is calculated between them to measure similarity.

- “Image-Based Service Recommendation System: A JPEG-Coefficient RFs Approach” [18]. This article consists of two phases. In the first phase, they classify images according to the type of product, while in the second, a recommendation system is created to find products most similar to a given one. For the second phase, which applies in our case, JPEG coefficients are used to extract image features. Subsequently, Euclidean distance is calculated between these coefficients.

This work differs in several aspects from the aforementioned ones. Not only does it seek recommendations based on an initial image, but it also allows users to select attributes they want the recommended images to have. In addition to this primary difference, this project goes beyond a mere model; it also includes a graphical interface that enables users to utilize the model.

System’s design

Dataset

As mentioned in the “Regulatory Framework” section, this work utilizes the dataset “iMaterialist (Fashion) 2020 at FGVC7” [2]. This dataset consists of several files, which are explained below.

Firstly, there are two sets of images: “train” and “test”. The “train” set comprises 43,793 images, while the “test” set contains 3,200 images. Both sets include these images labeled with an identifier. Additionally, there is a file named “train.csv” that contains the following fields:

- **imageId:** Unique identifier for the images. For each image, there are multiple entries in the file, corresponding to the number of segmentations the image has.
- **encodedPixels:** These pixels represent the image segmentation.
- **height:** Height of the image.
- **width:** Width of the image.
- **classId:** Identifier of the class to which they belong.
- **attributeIds:** List of attribute identifiers contained in this image segment.

Segmentation is highly useful as it is used to create image masks. The “attributeIds” field is also noteworthy, as it allows us to filter images according to user preferences. These attributes are described in a file named “label_descriptions.json”. Each attribute has a description and a supercategory associated with its unique identifier.

Model

Segmentation

Firstly, the images need to be preprocessed. The objective of the task is to provide recommendations for similar clothing items, so anything other than clothing in the image can negatively impact the recommendation. For example, recommendations could include garments with a similar background in the image, images with the same watermark, or images where the models wearing them have physical similarities, among other things. These aspects should not influence the recommendation, which is why the first step is to create masks for the images. This is achieved using the “encodedPixels” attribute. These pixels, which represent the segmentation of the clothing, allow for the extraction of an image containing only the clothes.

However, this is not sufficient, as the user provides an image that is not in the database, meaning its segmentation is not available. Therefore, it is necessary to utilize a segmentation model to generate a mask for the user-provided image. There are numerous architectures that could address this issue. Some of them are described below:

- **U-Net:** Named due to its U-shaped architecture, it combines convolutional and deconvolutional layers to achieve image segmentation.
- **SegNet:** Uses pooling layers to achieve image segmentation.
- **Mask R-CNN:** This architecture combines Faster R-CNN, used for object detection, with mask generation.
- **DeepLab:** Utilizes dilated convolution to achieve image segmentation.
- **U2-Net:** An extension of U-Net that includes mask generation.

Ultimately, the decision is made to use the U2-Net architecture, which encompasses all the advantages of U-Net and is additionally tailored for mask creation. This architecture demonstrates high accuracy even when dealing with high-resolution images, making it a strong candidate for fashion image segmentation.

U2-Net originated in 2020 at the University of Alberta, Canada, with the publication of the article “U2-Net: Going Deeper with Nested U-Structure for Salient Object Detection” [23]. This article received recognition in 2020 as the best in pattern recognition.

Reccomendation algorithm

The next step involves extracting features from all the images. To achieve this, a convolutional neural network is employed. As mentioned earlier, the decision has been made

to use the VGG16 model. This model has an easily comprehensible structure and is deeper than its predecessors, such as AlexNet, enabling the extraction of more complex features.

The model utilizes the VGG16 architecture to extract the feature vector from all the masks. These vectors are stored in a matrix. When the user provides an image, its mask is generated using the previously mentioned procedure. Next, its feature vector is also obtained, and the cosine similarity is computed with the vectors from all other images. Finally, the dataset is filtered based on garments containing attributes selected by the user, and the top N images with the highest similarity are returned, where N is the desired number of images the user wants to obtain.

API

In order to create a user-friendly graphical interface for the common user, it is necessary to have an API that serves the model. An API is a way to connect different applications to make them work together. The API enables the graphical interface to consume the previously described artificial intelligence model.

Python has been chosen for building this API, as it facilitates development since the model is also created using the same language. The chosen library for constructing this API is Flask, a framework designed for building CRUD applications (Create, Read, Update, Delete). A CRUD application can have multiple endpoints, which are routes to which a request can be sent. Each of these routes can perform actions like creating, reading, updating, or deleting information in the API.

For our API, only two endpoints have been necessary:

- “/recommendations”: Uses the POST method. This endpoint expects to receive an image, a number of recommendations, and an optional list of attribute identifiers. Once it receives these arguments, the API utilizes the model and returns a list of image identifiers that, while having the required attributes, exhibit the highest similarity.
- “/image/<id>”: Through this endpoint, the API allows the retrieval of images based on their identifier, where “id” is the identifier of the desired image. It only requires a GET request with a valid identifier in the route.

Interface

Several options were considered for the graphical interface. The first of these was StreamLit, a platform for developing interfaces for artificial intelligence projects written in Python. It enables the deployment of such applications in a very straightforward manner, requiring only a few lines of code. Additionally, Gradio was considered, which

operates in a similar way to StreamLit. Both options are good, but in the end, the decision was made to use the React development framework, a JavaScript language. While the first two tools were interesting for achieving rapid development, they had limitations in terms of interface customization.

The inability to create the interface exactly as desired led me to choose React, which allows for building a complete web application from start to finish, including all its features. Furthermore, TypeScript is used instead of JavaScript for greater robustness due to its static typing. This helps prevent hard-to-debug errors during development, even though marking all the types may sometimes feel tedious. Using React proves highly advantageous as it allows for organizing code into reusable components where HTML and JavaScript come together.

The project has the following structure:

- “**attributesSearch.tsx**”: This component is used to search for attributes that the user wants to select for their recommendations. It includes the complete list of attributes (selectable elements) and a search feature, given that there are 341 different attributes.
- “**attributesSelector.tsx**”: This component contains the attribute search. It also has a button that, if desired by the user, is used to obtain clothing recommendations with various attributes.
- “**header.tsx**”: This represents the page header.
- “**imageUploader**”: This component is a box that allows the user to upload the image they want to use for obtaining recommendations. It enables exploring local files and uploading an image file.
- “**stepper**”: This component is used to navigate through the screens of the information-giving process, as the application consists of a single page.
- “**services.ts**”: This file handles API requests.
- “**services.interfaces.ts**”: This file contains the necessary interfaces for the services.
- “**constants.js**”: In this file, variables that remain constant throughout the project are stored.
- “**globals.css**”: This file contains the CSS configuration that affects the entire project.
- “**label_descriptions.json**”: This file is used to associate attribute identifiers with their descriptions.
- “**page.tsx**”: This is the main page where all the components come together to compose the application.

Upon loading the page, a window prompts us to upload an image. Once the image is uploaded, you can press the “Next” button, which takes you to selecting the number of recommendations through a number selection bar. Once again, you can press the “Next” button; this time, it leads you to the final window, the attribute selection. This window contains the complete list of selectable attributes. Selected attributes are marked at the top, where they can be deselected. Additionally, there is a search bar given the large number of attributes. Once everything is set, you can press the “Process” button. This button sends a request to the API which, utilizing the model, returns the identifiers of the most similar images. With these identifiers, a request is made for each of them to obtain their respective images. These images are displayed in a gallery that can be viewed in full size. If you wish to repeat the process, you can simply press the “Reset” button.

Conclusions

In this chapter, we will present the overall conclusions of the work carried out. An analysis will be conducted to verify the fulfillment of the initial objectives. The main objective of this work was to create a graphical interface capable of recommending clothing images based not only on a user-provided image but also on a set of desired attributes for the recommendation. This objective has been fully achieved. Additionally, the following sub-objectives were outlined:

- Develop an artificial intelligence model capable of recommending images based on multimodal attributes and another image.
- Label the images to be used with their respective attributes.
- Create an API that allows communication between the aforementioned model and an interface.
- Design an intuitive interface that enables users to upload an image, select attributes of interest, and visualize the results provided by the model.

A web application has been successfully built to perform the tasks described above, leading to the conclusion that the work has been successful.

The VGG16 architecture has proven to be useful for extracting image features. It has been effective and efficient. The use of masks, although leading to a more complex solution due to the segmentation task, has clearly improved the recommendation model. Recommended images are no longer influenced by elements unrelated to the garments, such as image backgrounds, watermarks, or the physical appearance of models. Through the combination of image segmentation for mask creation and feature extraction for subsequent cosine similarity calculation, very good similarity percentages have been achieved for recommendations.

The use of attributes has also been successful. The model is capable of filtering the dataset and recommending only garments that possess these attributes. However, when an excessive number of attributes are used together or some uncommon attributes are selected, the model may encounter recommendation challenges due to the not overly large size of the dataset.

In conclusion, this work has been successful, as it has perfectly fulfilled the proposed objectives, providing a clean interface capable of serving an image recommendation system with a high percentage of similarity in the provided recommendations. Furthermore, the use of attributes has been integrated, allowing users to obtain more precise and tailored recommendations. If a larger dataset were available, this latter functionality would be even more accurate.

Future work

The field of computer vision could be considered relatively new. Therefore, there is significant room for improvement, and a variety of future work can be proposed from this point. Some of these possibilities include:

- Expanding the working database, as having a greater quantity of available data will lead to higher-quality recommendations.
- Implementing a neural network more focused on clothing and less generalized than the VGG16 architecture. Training a network specifically tailored to clothing could enhance recommendation quality.
- Incorporating feedback from real users during network training could be highly beneficial, as recommendation quality is often subjective and depends on user preferences.
- Exploring the possibility of utilizing the device's camera when accessing the web application to facilitate the recommendation process.
- Allowing users to provide multiple images to obtain collective results.
- Enabling users to select colors for their recommendations.

These potential directions could contribute to advancing the field of computer vision for fashion and enhancing the capabilities of the recommendation system.