



*ugr* | Universidad  
de **Granada**

TRABAJO FIN DE GRADO  
GRADO EN INGENIERÍA INFORMÁTICA

# Sistema de Recuperación de Imágenes

---

Basado en Propiedades de Categorías Visuales

**Autor**  
Fernando Roldán Zafra (alumno)

**Director**  
Jesús Chamorro Martínez



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE  
TELECOMUNICACIÓN

—  
Granada, Septiembre de 2019





Escaneando este código QR se puede acceder al repositorio del trabajo.  
También puede acceder a través del siguiente [enlace](#). [14]

# Sistema de recuperación de Imagenes

---

Basado en Propiedades de Categorías Visuales.

## Autor

Fernando Roldán Zafra (Alumno)

## Tutor

Jesús Chamorro Martínez



# Índice general

<b>Resumen</b>	<b>11</b>
<b>Introducción</b>	<b>19</b>
<b>I Sistemas de Recuperación de Imágenes</b>	<b>23</b>
<b>1. Motivación</b>	<b>25</b>
<b>2. Redes Neuronales convolucionadas</b>	<b>29</b>
2.1. La red Neuronal . . . . .	29
2.2. Arquitecturas . . . . .	31
2.2.1. Redes multicapa . . . . .	32
2.3. Redes Neuronales Convolucionadas o <i>CNN</i> . . . . .	33
<b>3. Sistemas de Recuperación de imágenes CBIR</b>	<b>35</b>
3.1. Descriptores de Imágenes . . . . .	37
3.1.1. Características de primer nivel . . . . .	38
3.1.2. Características de segundo nivel . . . . .	40
3.1.3. Características de tercer nivel . . . . .	42
3.1.4. Sistemas de recuperación basados en conjuntos difusos	43
3.2. Análisis práctico . . . . .	45
<b>4. Descriptor realizado</b>	<b>49</b>
4.1. Comparadores . . . . .	62
4.1.1. Comparadores centrados en etiquetas . . . . .	62
4.1.2. Comparadores centrados en las propiedades visuales .	69
4.1.3. Comparadores para el descriptor basado en cuadrículas	70
<b>5. Bases de datos utilizadas</b>	<b>75</b>
5.1. Primera base de datos . . . . .	75
5.2. Segunda base de datos . . . . .	76
5.3. Tercera base de datos . . . . .	77

<b>II Implementación del Sistema</b>	<b>83</b>
<b>6. Organización y presupuesto</b>	<b>85</b>
6.1. Diagrama de Gantt . . . . .	85
6.2. Presupuesto . . . . .	87
<b>7. Análisis de Requisitos del sistema</b>	<b>89</b>
7.1. Requisitos de datos . . . . .	89
7.2. Requisitos funcionales . . . . .	90
7.3. Requisitos no funcionales . . . . .	91
<b>8. Casos de uso</b>	<b>95</b>
8.1. Diagrama de casos de uso . . . . .	95
8.2. Descripción de los casos de uso . . . . .	97
<b>9. Diseño</b>	<b>115</b>
9.1. Bibliotecas usadas . . . . .	115
9.2. Aporte a la <i>JMR</i> . . . . .	117
<b>10. Implementación</b>	<b>121</b>
<b>11. Manual</b>	<b>123</b>
11.1. Barra de herramientas . . . . .	124
11.1.1. Abrir y cerrar imágenes . . . . .	124
11.1.2. Base de datos . . . . .	124
11.1.3. Opciones del descriptor . . . . .	125
11.1.4. Búsqueda por etiquetas . . . . .	127
11.2. Escritorio . . . . .	127
11.3. Output . . . . .	128
11.4. Demostración . . . . .	128
11.4.1. Primer ejemplo . . . . .	128
11.4.2. Segundo Ejemplo . . . . .	130
<b>12. Conclusiones</b>	<b>135</b>
<b>Bibliografía</b>	<b>137</b>

# Índice de figuras

1.1. Imagen que será utilizada como consulta en <i>Google imágenes</i> . . . . .	27
1.2. Resultado de la búsqueda de un sacapuntas azul en <i>Google imágenes</i> . . . . .	27
2.1. Esquema de una neurona biológica . . . . .	30
2.2. Grafo de una Neurona artificial . . . . .	31
2.3. Representación de una red neuronal multicapa con varias capas ocultas . . . . .	32
2.4. Diagrama de funcionamiento de la convolución . . . . .	34
3.1. Esquema de funcionamiento de un sistema CBIR . . . . .	37
3.2. Histograma de color . . . . .	38
3.3. Búsqueda de una imagen por su histograma de color . . . . .	40
3.4. Ejemplo de uso del sistema IRIS . . . . .	42
3.5. Visualización de un conjunto de colores difusos [15] . . . . .	45
3.6. Búsqueda de una taza roja basándose en el histograma de color . . . . .	46
3.7. Resultado de buscar una imagen basándose en el histograma de color o en la estructura de colores . . . . .	47
3.8. Búsqueda de una imagen basándose en sus etiquetas lingüísticas . . . . .	48
4.1. Consulta por color . . . . .	50
4.2. Consulta por etiqueta . . . . .	51
4.3. Consulta por etiqueta y color utilizando el sistema desarrollado . . . . .	52
4.4. Colores modelo utilizados para etiquetar los colores . . . . .	54
4.5. Prototipo de colores que serán usados para crear el espacio de colores difusos . . . . .	55
4.6. Resultado de etiquetar el color de la imagen de una taza roja . . . . .	56
4.7. Resultado de etiquetar el color de la imagen con un cubo rojo . . . . .	57
4.8. Resultado de etiquetar el color de la imagen con una taza oliva . . . . .	57
4.9. Resultado de etiquetar el color de la imagen con una taza azul . . . . .	58
4.10. Resultado de etiquetar el color de la imagen con un bolígrafo verde . . . . .	58

4.11. Búsqueda en la base de datos por etiquetas lingüísticas . . . . .	60
4.12. Búsqueda en la base de datos por etiquetas lingüísticas usando conjuntos difusos . . . . .	61
4.13. Imágenes con sus respectivas etiquetas calculadas . . . . .	63
4.14. Búsqueda de una imagen usando el comparador SoftEqual . .	63
4.15. Búsqueda de una imagen usando el comparador de igualdad .	64
4.16. Comparación de los comparadores de igualdad suave, igualdad e inclusión . . . . .	66
4.17. Búsqueda de imagen basándose exclusivamente en el peso de las etiquetas. . . . .	67
4.18. Búsqueda de imagen basándose exclusivamente en el peso de las etiquetas. . . . .	68
4.19. Búsqueda de imagen basándose en el peso de las etiquetas y en el histograma de color . . . . .	68
4.20. Búsqueda de imagen basándose en el peso de las etiquetas y en la estructura de color . . . . .	69
4.21. Búsqueda de un objeto usando el comparador basado en el mínimo . . . . .	71
4.22. Búsqueda de dos objetos usando el comparador basado en el mínimo . . . . .	72
4.23. Búsqueda de tres objetos usando el comparador basado en el mínimo . . . . .	72
4.24. Búsqueda de dos objetos usando el comparador basado en el máximo . . . . .	73
4.25. Búsqueda de tres objetos usando el comparador basado en el máximo . . . . .	74
5.1. Ejemplo de búsqueda realizada para la creación de la base de datos . . . . .	76
5.2. Imágenes pertenecientes a la categoría 'púa de guitarra' . . . . .	79
5.3. Imágenes pertenecientes a la categoría 'taza' . . . . .	79
5.4. Búsqueda por términos lingüísticos . . . . .	79
5.5. Imagen consulta y resultado en Google . . . . .	80
5.6. Fotografías con un objeto . . . . .	80
5.7. Fotografías con dos objetos . . . . .	80
5.8. Fotografías con tres objetos . . . . .	81
5.9. Fotografía de ejemplo de la tercera base de datos utilizada . .	81
6.1. Diagrama de Gantt . . . . .	86
8.1. Descripción del Actor . . . . .	96
8.2. Diagrama de casos de uso . . . . .	96
9.1. Diagrama UML de paquetes de la librería <i>JMR</i> . . . . .	116
9.2. Diagrama UML de paquetes de la librería <i>JFI</i> . . . . .	117

9.3. Diagrama de clases del proyecto realizado . . . . .	119
10.1. Captura de pantalla de la documentación del proyecto . . . . .	122
11.1. Pantalla principal de la interfaz . . . . .	123
11.2. Barra de herramientas de la interfaz . . . . .	124
11.3. Botones para abrir y cerrar imágenes . . . . .	124
11.4. Sección de la base de datos . . . . .	124
11.5. Opciones de configuración del descriptor . . . . .	125
11.6. Panel de configuración del grid . . . . .	126
11.7. Barra de herramientas de búsqueda por etiqueta . . . . .	127
11.8. Ventana de selección del clasificador . . . . .	129
11.9. Barra de herramientas tras elegir el descriptor y el Grid . . . . .	129
11.10. Ventana de selección de las opciones del Grid . . . . .	131
11.11. Resultado final del primer ejemplo de consulta . . . . .	132
11.12. Visualización del estado del segundo ejemplo . . . . .	132
11.13. Pantalla de guardado de la base de datos . . . . .	133
11.14. Resultado de realizar la búsqueda sin utilizar los conjuntos difusos . . . . .	133
11.15. Resultado de realizar la búsqueda utilizando los conjuntos difusos . . . . .	134







# **Sistema de Recuperación de Imágenes basado en Propiedades de Categorías Visuales**

Fernando Roldán Zafra

**Palabras clave:** Imagen, Descriptor, Comparador, Búsqueda, Distancia, Consulta, Etiquetado, Color

## **Resumen**

Hoy en día a causa del enorme crecimiento de las bases de datos multimedia parece obvio que se necesita algún tipo de mecanismo por el cual se gestione y se procese dicha información de forma automática. Una primera aproximación puede consistir en obtener información general de una imagen basada en sus propiedades visuales (como pueden ser los colores dominantes o el histograma de color). Otra posible aproximación puede verse en las ya omnipresentes Redes Neuronales, las cuales pueden identificar por ejemplo, que objeto aparece en una imagen.

En el presente proyecto se optará por una nueva alternativa, de la cual no existe ninguna literatura al respecto; esta consiste en el uso combinado de las técnicas mencionadas anteriormente, es decir, realizar un prototipo *CBIR* (Content Based Image Retrieval) el cual utilice tanto el etiquetado automático de la imagen, como las propiedades visuales de esta.

Por tanto en el presente documento se podrá encontrar toda la información referente a la realización de la solución mencionada anteriormente.



# **Image Retrieval System based on Visuals Categories Properties**

Fernando Roldán Zafra (student)

**Keywords:** Image, Descriptor, Comparator, Search, Distance, Query, Labeling, Color

## **Abstract**

Nowadays due to the growth of multimedia databases, it seems obvious that some kind of mechanism for managing and processing this information automatically is necessary. A first approach can consist on getting general information from one image based on its visual properties (such as dominant colors or color histogram). Another possible approach can be seen on the omnipresents Neural Networks, which can identify, for example, what object appears in the image.

In this document a new alternative will be opted for, and does not exist any literature about it. This solution consists on the combined use of the techniques previously mentioned, that is to carry out a CBIR prototype (Content Based Image Retrieval) that uses both automatic image labeling and the visual properties of the image.

Therefore, in this document, all the relevant information about the conduction of the previously mentioned solution could be found.



---

Yo, **Fernando Roldán Zafra**, alumno de la titulación Grado en Ingeniería Informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 50604172-D, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Fernando Roldán Zafra

Granada a 5 de Septiembre de 2019 .



---

D. **Jesús Chamorro Martínez**, Profesor del Área de Visión por Computador del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

**Informa:**

Que el presente trabajo, titulado *Sistema de Recuperación de Imágenes basado en Propiedades de Categorías Visuales*, ha sido realizado bajo su supervisión por **Fernando Roldán Zafra**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 5 de Septiembre de 2019.

**El director:**

**Jesús Chamorro Martínez**



# Agradecimientos

Lo primero y ante todo, darle las gracias a mis padres por todo el apoyo y la confianza que han depositado en mí todos estos años. Sin su paciencia no habría llegado hasta aquí.

También tengo que darle especial mención a toda la gente que me ha acompañado en todo este viaje, tanto a los que siguen en mi vida como a los que no. Gracias por hacer todos estos años mas llevaderos y por ayudarme a no caer rendido en esas noches interminables de estudio.

Muchas gracias a mi tutor Jesús, por la paciencia que ha tenido conmigo, por como trasmite su pasión por lo que hace, por toda la ayuda que me ha aportado en estos meses y por ser un excelente docente.



# Introducción

En los últimos años, la invención de la cámara digital y sobre todo, su integración en los dispositivos móviles han provocado que un usuario medio, pase de ser un mero consumidor de contenido visual a un usuario que es capaz de capturar el mundo a su alrededor en fotografías. Esto a llevado a que hoy en día haya una enorme cantidad de información en formato imagen que por su volumen requiere de alguna técnica o sistema que nos permita consultar y obtener imágenes de forma eficiente en una gran base de datos.

En este contexto aparecen los Sistemas de Recuperación de Imágenes y estos están ampliamente integrados en nuestro mundo. Los sistemas de recuperación de imágenes, suelen utilizar algún tipo de descriptor para describir una imagen. Ya sea su color, la información que contienen en sus metadatos o una etiqueta lingüística que describa lo que se puede ver en la imagen. Pero en este punto aparece un problema y es que una etiqueta lingüística no describe las propiedades visuales de una imagen debido a la gran cantidad de formas y colores que puede presentar un objeto.

Es en este punto donde se desarrolla el presente proyecto. Hay que tener en cuenta que no existe ninguna literatura referente a la combinación de etiquetas lingüísticas extraídas de la imagen con las características visuales de la misma imagen. Para alcanzar este objetivo se utilizaran Redes Neuronales convolucionadas a la par que descriptores de características visuales como puede ser el histograma de color.

## Objetivos

Se definen, por tanto los siguientes objetivos para el presente proyecto:

- Estudiar arquitecturas *CNN* para el etiquetado automático de imágenes.
- Desarrollar descriptores de bajo nivel (color, textura, etc.) para modelar propiedades asociadas a categorías visuales.

- Integrar los descriptores desarrollados en la biblioteca *JMR* (*Java Multimedia Retrieval*©) de software libre.
- Desarrollar un prototipo *CBIR* basado en los descriptores anteriores.

## Organización

La presente memoria, para facilitar su estructuración y comprensión, se encuentra dividida en 2 partes:

Primero se encuentra la parte teórica centrada en el estado actual de estas tecnologías, tanto de las *CNN* y los conjuntos de colores difusos, como de los sistemas de recuperación de imágenes que podemos encontrar hoy en día. Pasando por la motivación que nos lleva a proponer la solución que se ha desarrollado y a su utilización aportando ejemplos que reflejen sus virtudes. Además de esto, también se hablará del descriptor desarrollado.

En la segunda parte se pasará al diseño que se ha utilizado para el prototipo *CBIR*. Se hablará por tanto, de todas las fases inherentes al proceso de desarrollo de un software, pasando por el diseño, análisis de requisitos, casos de uso, etc.

Para finalizar se aportará un manual de usuario referente al sistema realizado y las conclusiones finales y las futuras vías de trabajo, donde se hablará sobre si se ha conseguido realizar el objetivo inicial y como podría mejorarse el proyecto.

## Parte I

# Sistemas de Recuperación de Imágenes



# Capítulo 1

## Motivación

En los últimos años hemos podido observar como el uso y almacenamiento de imágenes se ha extendido y crecido de forma exponencial, todo el mundo puede realizar fotografías o crear imágenes fácilmente. Este hecho a contribuido a que el uso de esta información este presente en cualquier ámbito de la sociedad. Aun a pesar de este hecho y de los enormes esfuerzos que se realizan, no hemos conseguido acceder a esta información de forma eficiente y fiable. Debido a esto, desde hace tiempo la investigación en el campo de la Recuperación de imágenes no ha parado de mejorar y de aproximarse a este problema, aunque sin llegar a solucionarlo totalmente.

Dentro de los Sistemas de Recuperación de Imágenes nos encontramos con los sistemas basados en contenido, estos sistemas consisten en la recuperación de imágenes basándose en el contenido de la imagen y no en sus metadatos como se venia haciendo anteriormente. Es por esto que hoy en día los Sistemas de Recuperación de Imágenes Basados en Contenido *CBIR* están en auge, y es que estos sistemas se basan en las propiedades visuales de las imágenes, permitiéndonos consultas mucho mas complejas y obteniendo resultados mucho mas precisos. Sin embargo, esta solución no esta exenta de errores.

Uno de los problemas de los sistemas CBIR es que la mayoría de estas propiedades visuales se calculan en torno a las características globales de la imagen (como puede ser textura o color) y dada la inmensa variedad de colores y formas que puede tener un objeto estas soluciones se hacen insuficientes. Si además de esto tenemos en cuenta que estos sistemas trabajan analizando las propiedades visuales de la imagen y no del objeto obtenemos que difícilmente obtendremos los resultados deseados.

Todos estos problemas han contribuido al uso de nuevas e innovadoras técnicas las cuales mediante el uso del *Machine learning* pretenden dar una

etiqueta lingüística a cada imagen en función a lo que se encuentra representado en ellas. Esto, sin embargo, tampoco está exento de errores y es que esta solución solo nos permitirá recuperar imágenes basándonos exclusivamente en su categoría y no en sus propiedades visuales. Esto nos limita mucho y es que obtendremos siempre imágenes en las cuales las formas o colores de los objetos variarán mucho.

Para ejemplificar los problemas anteriores, acudiremos al buscador de imágenes *Google Imágenes*. En este sistema hay dos posibles formatos de búsqueda por texto o por imagen. Centrándonos en la búsqueda por imagen, podremos ver una de las carencias que este sistema presenta. Si realizamos una búsqueda de una imagen con dos objetos, siendo cada uno de ellos diferente en categoría y color, podremos ver como este sistema se fija solo en las propiedades globales de la imagen. Como ejemplo utilizaremos una fotografía de un mechero rojo y una goma de borrar verde. La fotografía que se ha usado como consulta se puede ver en la figura 1.1 y el resultado en la figura 1.2.

Como se puede ver, el sistema ha sido capaz de reconocer uno de los objetos (el que ha considerado mas relevante) y nos sugiere como búsqueda relacionada 'Milan 430' siendo esto el tipo de goma de borrar. Además de esto, en las 'imágenes visualmente similares' se puede ver como también ha sido capaz de reconocer los colores que aparecen en la imagen. Sin embargo, no ha sido capaz de asociar cada uno de los objetos con su color ya que hace un análisis global de la imagen. Por lo tanto, veremos imágenes que tengan los colores de la imagen consulta, pero solo se centrará en el objeto que ha sido capaz de reconocer, devolviendo objetos de la categoría de la imagen consulta pero mezclando las propiedades visuales de ambos objetos.

Por lo tanto, la motivación de este proyecto consiste en la realización de un Sistema de recuperación de imágenes capaz de unificar tanto el etiquetado de imágenes como el reconocimiento de las propiedades visuales, sin dejar atrás el análisis local o global de la imagen. Cabe destacar, que no hay ninguna literatura referente a este tema y es que los sistemas tradicionales han venido basándose en una u otra técnica, pero no en la combinación de ambas. Por lo tanto, mediante el sistema aportado por este proyecto se espera que el usuario pueda realizar consultas semánticamente complejas basándose tanto en etiquetas lingüísticas como en propiedades visuales de objetos. En los capítulos siguientes hablaremos por tanto de las redes neuronales, la extracción de propiedades visuales y de las distintas técnicas que se han utilizado para realizar este proyecto.



Figura 1.1: Imagen que será utilizada como consulta en *Google imágenes*

Aproximadamente 2 resultados (0,73 segundos)

 Tamaño de imagen:  
800 × 600

No se ha encontrado esta imagen en otros tamaños.

Possible búsqueda relacionada: [milan 430](#)

[Gomas marca Milan 430 \(73083\) - Materialescolar.es](#)

<https://www.materialescolar.es> > ... > Gomas, gomillas y correctores ▾

Entrega 24h de Gomas marca Milan 430 (73083). ¡Envío GRATIS!

[Milan 430 - Caja de 30 gomas de borrar, migas de pan: Amazon.e...](#)

<https://www.amazon.es> > CAJA-GOMAS-BORRAR-CUADRADA-MILÁN ▾

Compra online Milan 430 - Caja de 30 gomas de borrar, migas de pan. Envío en 1 día GRATIS con Amazon Prime.

Imágenes visualmente similares



Figura 1.2: Resultado de la búsqueda de un sacapuntas azul en *Google imágenes*



## Capítulo 2

# Redes Neuronales convolucionadas

Aun a pesar de que no ha sido tarea de este proyecto desarrollar una *CNN* cabe destacar la relevancia que estas tienen en este proyecto, ya que la *CNN* que ha sido utilizada será la encargada de etiquetar todas las imágenes, sin embargo no podemos hablar de este tipo concreto de red neuronal sin hablar antes de las redes neuronales de forma general. Por eso, en este capítulo se hablará sobre las redes neuronales de una forma más general. Para acabar hablando del tipo concreto de Red Neuronal que nos ocupa.

### 2.1. La red Neuronal

Aun hoy en día el cerebro humano sigue siendo una de las maquinas mas potentes que existen y es que un cerebro humano es capaz de superar al ordenador mas potente en tareas biológicamente relevantes. Es por esto que en muchas ocasiones, nos fijamos en su comportamiento para intentar transmitir las increíbles capacidades que presenta a una máquina de forma de que no sea necesaria la intervención humana para realizar alguna tarea. Y es que desde que se descubrió su enorme potencial allá por finales de los años 80, las redes neuronales han ido implantándose en casi cualquier ámbito de la sociedad.

De esta manera, las redes neuronales artificiales intentan imitar el funcionamiento del cerebro humano para conseguir algunas de sus características y aunque aún no se ha conseguido desarrollar un cerebro artificial que sea capaz de competir con el humano hemos conseguido imitar algunas de sus funciones. Algunos ejemplos de estas funciones son el reconocimiento facial, el reconocimiento de caracteres escritos a mano, reconocimiento de correos no deseados en servicios de correo electrónico y un largo etcétera.

Para conseguir todas estas características se ha partido de como el cerebro humano funciona y es que, nuestro cerebro, no deja de ser un conjunto de piezas mas pequeñas que trabajan individualmente. Estas piezas son conocidas como neuronas, y de la misma forma que lo hace un cerebro humano, una red neuronal también lo hace, estando esta dividida en neuronas artificiales. Debido a esto, si queremos entender el funcionamiento de las neuronas artificiales nos tendremos que fijar en las neuronas biológicas y es que como ya se ha comentado, las artificiales imitan el funcionamiento de las biológicas.

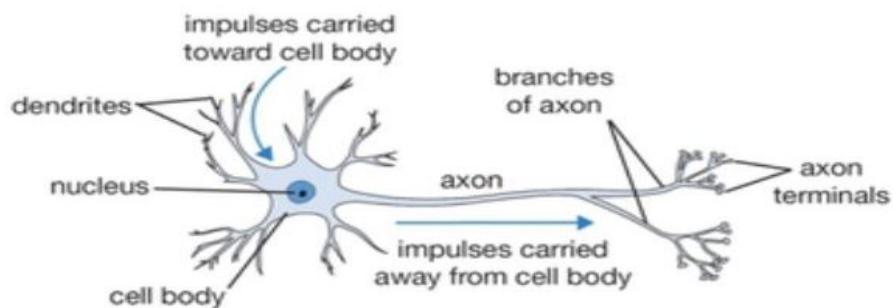


Figura 2.1: Esquema de una neurona biológica

Como se puede ver en la figura 2.1, una neurona es una célula la cual recibe señales electromagnéticas provenientes del exterior o de otras neuronas. La comunicación entre neuronas se produce a través de la sinapsis de las dendritas. Si una neurona recibe estímulos superiores a un cierto umbral, esta se activará, lo que supondrá el envío a través del axón de una señal que será recibida por otras neuronas a través de sus conexiones sinápticas de las dendritas.

Por tanto, una de las claves es la potencia con la que se recibe la señal desde otra neurona y es que la sinapsis de las dendritas es dinámica, puede potenciarse o debilitarse. Esto se realiza a través del aprendizaje.

Si intentamos trasladar esta idea a las neuronas artificiales obtendremos el grafo representado en la figura 2.2. En este grafo, podremos ver como la neurona recibirá datos o señales, representados por  $x_0, x_1, x_2, \text{ etc.}$ , a través de las dendritas y mediante la sinapsis de estas, representada como  $w_0, w_1, w_2, \text{ etc.}$ , se excitara o no la neurona.

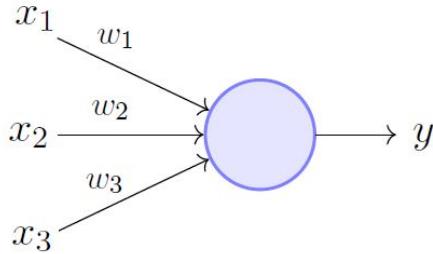


Figura 2.2: Grafo de una Neurona artificial

Esta sinapsis puede ser traducida como el peso que se le asignara a dicha entrada. De esta manera el calculo de la salida queda segmentado en 2 componentes, la función de entrada y la función de activación:

- Función de entrada: Se trata de una componente lineal obtenida al sumar todas las entradas con sus respectivos pesos.
- Función de activación: La función de activación  $f$  es una componente no lineal que transforma la función de entrada en un valor final. Esta será la función que nos marcará si se activa o no la neurona, es decir, si la función de entrada supera cierto umbral la neurona se activará. Para ello se utilizan diversas funciones como pueden ser la función sigmoidal, la función tangente hiperbólica o la Función de Unidad Lineal Rectificada (*RELU* por sus siglas en inglés).

De esta manera se obtiene que la salida ' $y$ ' de la neurona vendrá dada por la siguiente formula:

$$y = f(\sum x_i + w_i)$$

Una vez definidos los componentes de una neurona podemos pasar a ver como se conectan entre ellas.

## 2.2. Arquitecturas

Si hablamos de las arquitecturas de las redes neuronales, antes de entrar en las diferentes formas de organizar una red neuronal debemos definir cuales son sus partes o capas y es que una red neuronal se organiza en capas distinguiendo cada una de ellas en función a la tarea que desempeñan. Podemos distinguir tres capas:

- Capa de entrada: Las capas de entrada, proveen información del mundo exterior a la red, en esta capa no se realiza ningún tipo de calculo, simplemente le pasan la información a la capa oculta.

- Capa oculta: La capa oculta no tiene ningún tipo de conexión con el mundo fuera de la red neuronal. Es en este nivel donde se realizan todos los cálculos recibiendo la información de la capa de entrada y enviándola hacia la capa de salida. Al contrario del resto de capas, una red neuronal puede tener mas de una capa oculta.
- Capa de salida: En esta capa se realizan los cálculos finales y se envía la información al mundo fuera de la red.

Una vez vistas las partes de una red podemos distinguir varios tipos de arquitecturas y es que el uso de unas u otras vendrá dado por el problema que queramos resolver. En este caso y por motivos de espacio hablaremos de las más conocidas, estas son las redes neuronales multicapa.

### 2.2.1. Redes multicapa

Como ya vimos anteriormente las redes neuronales no son más que un conjunto de neuronas conectadas entre si, en el caso de las redes neuronales multicapa, estas se organizan en los tres tipos de capas definidos anteriormente: capas de entrada, capas de salida y capas ocultas.

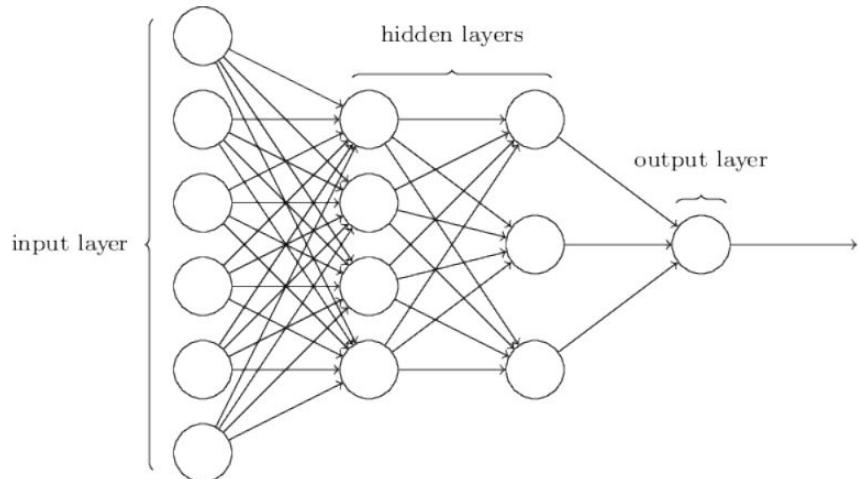


Figura 2.3: Representación de una red neuronal multicapa con varias capas ocultas

Este tipo de distribuciones se puede representar como un grafo acíclico que conecta todas las neuronas de una capa con la siguiente, es decir, se envía la información de una capa a la siguiente sin volver atrás, de lo contrario, podrían producirse bucles infinitos.

Por norma general, el numero de neuronas en la entrada y en la salida suelen ser bastante fácil de calcular ciñéndonos al problema que queramos resolver, por ejemplo, imaginémonos que queremos crear una red neuronal que nos permita reconocer si un numero escrito a mano es un 5 o no. En este caso, la imagen en escala de gris tendrá un tamaño de 100x100 píxeles,  $100 \times 100 = 10.000$  neuronas de entrada por las que introduciremos un píxel en cada una de ellas y una neurona de salida la cual nos dará un valor, si ese valor se encuentra por encima de un determinado umbral, el número será un 5. Sin embargo, el numero de capas a utilizar en la capa oculta es más difícil de estimar.

La principal diferencia entre usar una red con una o una con varias capas ocultas, es que cuantas mas capas tenga la red, problemas mas complejos podrá resolver. Esto se debe, a que cada una de las neuronas ocultas reciben la información de la capa anterior, mejorando cada vez mas los resultados obtenidos. Aunque como se comentó anteriormente el proceso de elegir el numero de capas a utilizar es un problema complejo y en cada caso la solución a este problema puede variar.

En este punto se nos presenta un problema en el caso del uso de redes neuronales para el reconocimiento de imágenes, y es que en una red neuronal multicapa como las descritas anteriormente, cada neurona esta conectada a todas las neuronas de la capa siguiente, a excepción de las que se encuentran en la capa de salida. Esto puede llevar a asociar un píxel de la esquina superior izquierda de una imagen con el píxel de la esquina opuesta, lo que puede no tener mucho sentido. Es en este contexto donde surgen las Redes Neuronales Convolucionadas.

### 2.3. Redes Neuronales Convolucionadas o *CNN*

En esta sección, una vez ya hemos introducido las bases de las redes neuronales, hablaremos de forma muy general de las redes neuronales convolucionadas. Estas redes serán las encargadas de asignar una etiqueta lingüística a cada imagen que se use en este proyecto. Este tipo de redes se caracterizan por ser un tipo de redes neuronales multicapa en las cuales tienen arquitectura especialmente eficiente en el reconocimiento de patrones. Esto, las hace especialmente útiles para el reconocimiento de imágenes. La eficiencia antes mencionada se consigue analizando los píxeles de la imagen no como un único píxel que es independiente al resto de los demás si no leyendo los píxeles por conjuntos y analizando estos. Una vez analizados, se reducen a un único dato y este sera el que se enviara a la capa siguiente. Este proceso es el que se conoce como 'convolución'. De esta forma se consigue

que los patrones de la imagen sean mas fácilmente reconocibles.

Este proceso se realiza basándonos en la afirmación de que dada una imagen, un píxel de una zona de la imagen, se encontrará más relacionado con un píxel que se encuentre muy próximo a el que con uno que se encuentre muy lejano. Un ejemplo de como se analiza una imagen se puede ver en la figura 2.4 en la cual se puede observar como se recorren los píxeles de la imagen de entrada. De esta forma y fijándonos en la imagen, esta se recorre comenzando por la esquina superior izquierda de la imagen y se irá aplicando la convolución en cada región de píxeles.

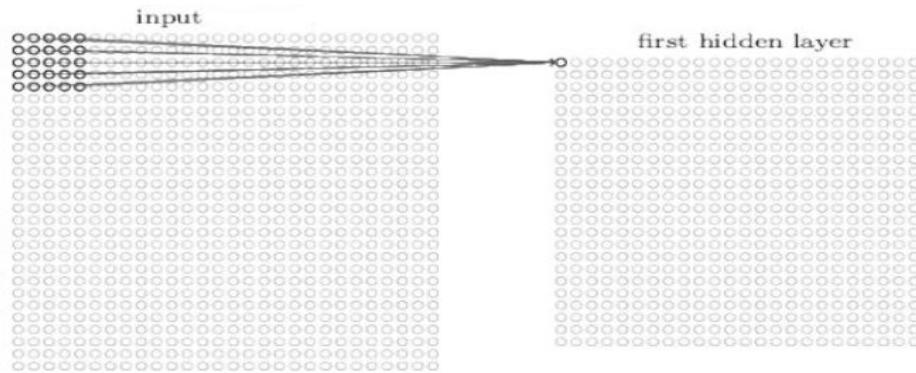


Figura 2.4: Diagrama de funcionamiento de la convolución

Cabe destacar que en este proyecto se ha usado una red neuronal ya entrenada y en pleno funcionamiento, por lo que no ha sido necesario el estudio de estas redes más allá de unas bases para conocer su funcionamiento.

## Capítulo 3

# Sistemas de Recuperación de imágenes CBIR

Como ya se adelanto anteriormente, hoy en día nos enfrentamos a un problema derivado de la facilidad con la que cualquier persona puede realizar una fotografía. Y es que debido a la miniaturización de las cámaras digitales y a la gran expansión de internet todo el mundo realiza y comparte fotografías e imágenes de diversa índole. Esto ha provocado que sea necesario el desarrollo de algún tipo de técnica que nos permita procesar y almacenar imágenes de manera similar a como lo haría un ser humano. Pero aun hoy en día y a pesar de los esfuerzos que se realizan, no se ha conseguido hallar un mecanismo que imite la visión humana y más concretamente como se interpretan las imágenes. Y es que, aunque seamos capaces de capturar digitalmente una imagen, no somos capaces de crear un modelo que imite como interpretamos esa información. Sin embargo, para intentar solucionar este problema se han creado los Sistemas de Recuperación de Imágenes.

Los sistemas de recuperación de imágenes tratan de recuperar imágenes de una base de datos en función a una búsqueda que se haya realizado. Estos resultados han de ser semánticamente relevantes en función a la búsqueda que se haya realizado. Pero ¿como podemos saber que es lo que es realmente semánticamente relevante para un humano? Este problema se denomina brecha semántica y podemos definirlo como la diferencia entre lo que interpreta un humano al ver una imagen y lo que interpreta una computadora.[20]

Tradicionalmente, para resolver el problema de la recuperación de imágenes, los sistemas estaban basados en sistemas de recuperación basados en texto. Esto, se realizaba anotando a mano por un experto las diferentes etiquetas que una imagen podía tener. Sin embargo, esta solución y a pesar de que fue ampliamente utilizada en su momento, dejo de ser utilizada ya que acarreaba varios problemas. Por un lado, tenemos que a medida que el

numero de imágenes crecía, el coste de etiquetar todas esas imágenes a mano también crecía. Por otro lado, las etiquetas añadidas a la imagen son muy personales y pueden variar mucho de una persona a otra. Además de todo esto, en los sistemas en los que las etiquetas se añadían automáticamente a partir del texto que rodea la imagen se producían resultados que no eran relevantes con la consulta. Por todo esto se decidió tomar otro camino.

Para resolver este problema, se crearon los sistemas de recuperación de imágenes basados en contenido o *CBIR* por sus siglas en inglés. Estos sistemas se basan en el análisis de las propiedades visuales de la imagen para, a partir de estas propiedades visuales obtener un conjunto de imágenes semánticamente relevantes como resultado.

En estos sistemas el usuario introduce una consulta y el sistema aportará resultados relevantes en función a la consulta realizada. Esta consulta puede ser una cadena de texto u otra imagen. Dicha consulta será procesada y se calculará una medida de similitud con las medidas calculadas de otras imágenes. Todo este proceso puede ser dividido en varias etapas según [11]:

- Adquisición de imágenes: En esta etapa se adquieren las imágenes que serán las que 'alimentarán' el sistema, usándolas como posibles resultados.
- Preprocesado de imágenes: Las imágenes obtenidas anteriormente son preprocesadas para, por ejemplo, eliminar el ruido
- Extracción de características: Como se puede ver en la imagen 3.1, esta etapa se realizará tanto para la imagen consulta como para las imágenes en la base de datos y es en esta etapa donde analizaremos las imágenes y obtendremos una medida característica de la imagen en función a sus propiedades visuales.
- Búsqueda por similitud: A partir de los resultados obtenidos en la etapa anterior, buscaremos entre las medidas obtenidas de la base de datos las más cercanas a la obtenida con la imagen consulta.
- Obtención de imágenes: Una vez obtenidas las medidas más cercanas se aportan las imágenes asociadas a ellas, siendo estas imágenes las más relevantes con respecto a la imagen consulta. Este resultado aparecerá ordenadas de mayor a menor relevancia.
- Interacción del usuario: El usuario comunica si se tratan de imágenes relevantes. Esto se repite hasta que el usuario quede satisfecho.

Como se puede ver, la fase de extracción de características es fundamental, tanto en la base de datos como en la imagen consulta, ya que de

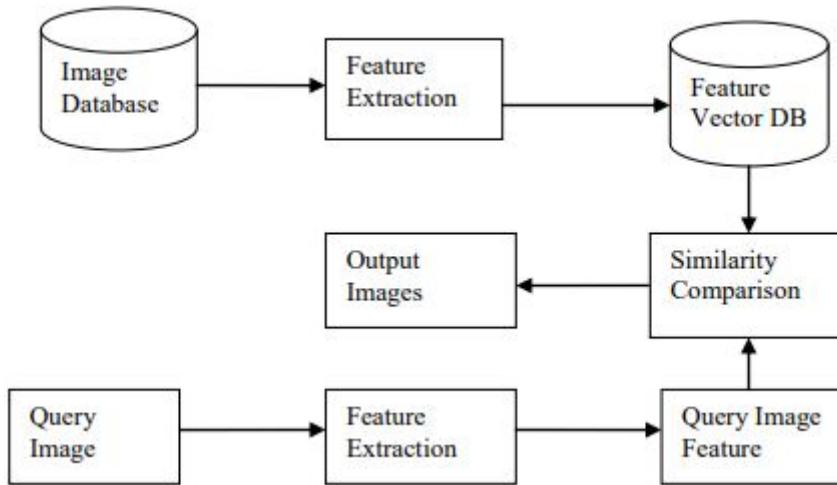


Figura 3.1: Esquema de funcionamiento de un sistema CBIR

ella dependerán los resultados que obtendremos en la búsqueda. Por esto, podemos definir estas características como 'Descriptores' siendo estos una medida calculada a partir de la imagen y que representará las características visuales de dicha imagen. Pero el problema no acaba ahí ya que si queremos realizar un sistema *CBIR*, a parte de hallar algún tipo de técnica para calcular el descriptor, debemos definir como será calculada la similitud entre dos imágenes basándonos en sus descriptores. Estos descriptores, podrán ser calculados en función a varios tipos de características de la imagen.

### 3.1. Descriptores de Imágenes

Podemos llamar 'Descriptor' a una medida calculada a partir de una imagen la cual trata de representar o describir su contenido de forma computacional. Tal y como se mencionó anteriormente, la necesidad de estos descriptores viene dada a partir de la brecha semántica que surge de la diferencia a la hora de interpretar una cadena de píxeles por una computadora y la imagen, a la que pertenecen esos píxeles, que vemos los seres humanos. Para calcular estos descriptores podemos basarnos en multitud de características, las cuales pueden ser separadas tal y como se menciona en [21] en tres niveles atendiendo a las posibles consultas que puede necesitar realizar un usuario al sistema. Estas consultas pueden ser clasificadas en tres niveles de complejidad, siendo cada nivel más complejo que el anterior.

### 3.1.1. Características de primer nivel

A este nivel pertenecen las características con una semántica más "primitiva" como son las referentes a color, textura, forma o localización espacial de los objetos que en la imagen aparecen representados. Estas características, podemos decir que son las que pueden ser extraídas de la imagen directamente. Un ejemplo de consultas de este nivel puede ser "Busca imágenes parecidas a esta." "Busca imágenes en la que la región superior derecha sea mas oscura".

Podríamos decir que estas son las características más extendidas y utilizadas y estas como se comentó anteriormente se dividen en los siguientes tipos:

#### Color

Hoy en día han surgido diversos métodos para la búsqueda de imágenes basadas en color pero la mayoría de ellas parten de la misma idea. Esta idea consiste en que una vez definida una colección de imágenes, se analiza cada una de ellas y se calcula su histograma de color, el cual muestra la proporción de píxeles de cada color en la imagen. Una vez calculado, se almacena el histograma de color en la base de datos.

Más tarde, cuando queramos buscar alguna imagen en el sistema, el usuario tiene dos opciones. Por un lado puede proporcionar una imagen a partir de la cual el histograma sera calculado o por otro lado puede especificar la proporción de color que desea para cada uno de los colores que elija. En este momento se buscará en la base de datos el histograma de color que más se acerque a lo especificado por el usuario. Cabe destacar que los histogramas de color son resistentes a las rotaciones o a la escala de las imágenes. Un ejemplo de histograma puede verse en la figura 3.2

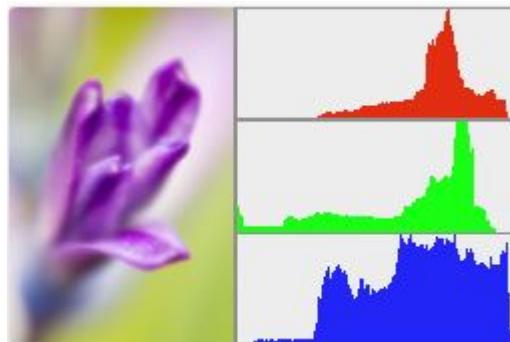


Figura 3.2: Histograma de color

### Textura

Aun a pesar de puede parecer que los descriptores basados en textura pueden ser poco útiles no hay que menospreciar su utilizar. Esta utilidad puede quedar patente en el caso de que queramos distinguir entre regiones de una imagen con colores similares, como por ejemplo entre el cielo y el mar, o entre las hojas y la hierva.

Hay diversas formas de analizar la textura en una imagen, aunque la mas extendida consiste en dada una imagen calcular el brillo relativo de pares de píxeles consecutivos permitiendo calcular medidas de textura tales como contraste, regularidad, direccionalidad, etc...

Las consultas basadas en este tipo de características pueden ser formuladas de una manera muy similar a las consultas por color, o seleccionando una determinada textura de una paleta de texturas o aportando una imagen para posteriormente aportar el resultado que mas se acerque.

### Forma

Esta posiblemente es la mas importante de todas las características de este nivel y es que aunque el resto de características estén bien definidas y nos sean de gran utilidad, los humanos reconocemos la mayor parte de los objetos por su forma.

Dentro de esta característica podemos distinguir dos tipos. Los basados en características globales, como la circularidad, y los basados en características locales, como los puntos característicos.

### Posición

Este tipo de características, son también de gran importancia y pueden muy útiles en diversos campos, como por ejemplo en los sistemas de información geográfica. Para calcular este tipo de características, partiendo de una imagen se buscan sus puntos, lineas o regiones y su localización en la imagen. Una vez han sido localizados permiten realizar búsquedas en función a relaciones direccionables (como 'izquierda' o 'Arriba') o topológicas (como la unión o la disyunción)

Una consulta utilizando descriptores de este nivel se puede observar en la figura 3.3 la cual se ha realizado basandonos en el histograma de color. Esta búsqueda se ha realizado en el sistema desarrollado para este proyecto y usando una base de datos de imágenes descargadas de internet, la cual

contiene varios objetos de diversos colores, como púas de guitarra, tazas o cubos. Como se puede observar las imágenes resultantes tienen un color muy parecido, pero no es capaz de distinguir entre lo que son púas o lo que no.



Figura 3.3: Búsqueda de una imagen por su histograma de color

Llegados a este punto se nos presenta un problema y es que el histograma de color no es fácilmente interpretable por los humanos y es que aunque estén relacionados a conceptos de bajo nivel, como el color, no es posible utilizando características de este nivel distinguir conceptos de mas alto nivel, como la actividad que se esta desarrollando en la imagen o el objeto que aparece en ellas. Nada de esto nos sera posible basandonos exclusivamente en las propiedades visuales de las imágenes, para ello se utilizan otro tipo de técnicas.

### 3.1.2. Características de segundo nivel

Como se comentó en el apartado anterior los descriptores de bajo nivel presentan el problema de que no siguen la semántica que seguiría un humano. Y es que la mayor parte de las veces, cuando buscamos una imagen solemos buscar por términos de más alto nivel semántico. Ejemplos de consultas de este nivel a un sistema CBIR podrían ser 'Busca imágenes de la Torre Eiffel' o 'Busca imágenes de tazas'.

Para etiquetar las imágenes de forma automática y de forma que podamos realizar consultas como las anteriores, necesitaremos por tanto usar algún tipo de mecanismo de conocimiento externo y es que estas etiquetas no pueden ser directamente calculadas a partir de las propiedades visuales. En todo este contexto se basan las características de segundo nivel o características lógicas. Podemos denominarlas como 'lógicas' ya que implican algún grado de inferencia lógica sobre la identidad de los objetos representados en la imagen.

Según [21], podemos dividir estas características en dos tipos: El primero podría estar basado en la obtención de objetos de un determinado tipo, como por ejemplo "Buscar imágenes de una taza". El segundo tipo, podría estar representado por las búsquedas de objetos individuales o personas, como por ejemplo "Busca imágenes de la torre Eiffel".

Como podemos imaginarnos, este tipo de consultas no pueden ser resueltas basándonos en propiedades como el color o la textura por lo que necesitaremos basarnos en algún tipo de mecanismo de mas alto nivel. Este mecanismo se hace aun más patente en las consultas del primer tipo y es que es necesario algún tipo de conocimiento para identificar un objeto como una taza y no como un vaso. Y si hablamos del segundo ejemplo, necesitaremos algún tipo de conocimiento que reconozca que se le ha dado el nombre 'Torre Eiffel' a un determinada estructura en concreto.

Además de todo esto podemos decir que un buen descriptor de contenido visual debe ser resistente a factores accidentales de la imagen, como cambios de brillo o posición en la escena. Aun así, un descriptor de contenido visual puede ser global o local. Un descriptor global trata de reconocer un objeto atendiéndose a la imagen al completo, mientras que un descriptor local usa regiones para describir el contenido de la imagen.

Para obtener los descriptores visuales locales, una imagen es dividida usando una partición. Esto nos proporciona una imagen cortada en secciones de igual tamaño y forma. Otra forma que tenemos de partir una imagen es dividir la imagen en regiones homogéneas basándonos en algún criterio, como por ejemplo seleccionando regiones las cuales tengan texturas similares.

Como nos podemos imaginar estos tipos de sistemas han sido ampliamente investigados y se han realizado grandes esfuerzos en ellos, un ejemplo que muestra esto es el sistema *IRIS* [22]. Este sistema, basándose en características de primer nivel intentaba reconocer la escena mas probable. De esta manera se generaban descriptores textuales por los que podíamos realizar búsquedas mas tarde. Un ejemplo de uso de este sistema se puede ver en la figura siguiente (3.4), la cual representa la imagen consulta y los respectivos resultados.

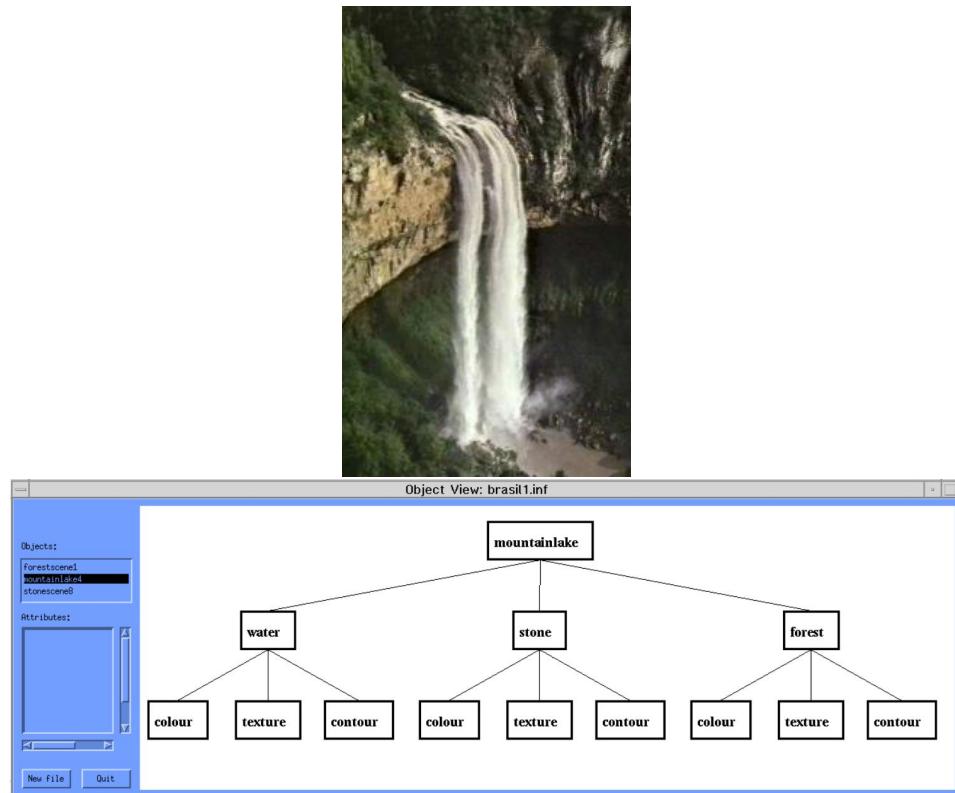


Figura 3.4: Ejemplo de uso del sistema IRIS

Como podemos ver en la imagen, este sistema resolvía parcialmente el problema de reconocimiento de escenas, pero no podía ser utilizado para el reconocimiento de objetos. Para ello, hay diversas técnicas, siendo el uso del *machine learning* una de las técnicas más relevantes. Es por eso que en este proyecto, como se habló anteriormente nos centraremos en el uso de ellas y más concretamente en el uso de redes neuronales convolucionadas.

### 3.1.3. Características de tercer nivel

Este tipo de características se centran en el reconocimiento de atributos abstractos que conllevan una cantidad elevada de razonamiento sobre los objetos y escenas que se representan en la imagen. Dentro de esta categoría podemos incluir eventos con nombres propios, imágenes que hacen referencia a emociones o sentimientos religiosos. Ejemplos de consultas de este tipo son 'Imágenes de un grupo de personas felices' o 'Imágenes de una persona triste'.

Como nos podemos imaginar este tipo de características son muy poco frecuentes aunque siguen en pleno desarrollo y es que aun no se ha logrado

encontrar una solución plena a este problema. Sin embargo, y a pesar de su complejidad, una posible solución al reconocimiento de emociones, según se puede leer en [23] pasa por el reconocimiento facial y la posterior extracción de características faciales basándose en ojos, boca y cejas. Una vez obtenidas estas características se pasa a la clasificación de las emociones utilizando una red neuronal.

### 3.1.4. Sistemas de recuperación basados en conjuntos difusos

En esta sección haremos especial mención al uso de los conjuntos difusos y es que esta característica ha sido de gran relevancia en el presente proyecto ya que además de usar las características mencionadas anteriormente incluiremos la capacidad de recuperar imágenes usando esta novedosa técnica. Tal y como se menciona en [24] el objetivo de estos sistemas no consiste en simplemente la clasificación de los colores, si no en la recuperación de imágenes basándonos en sus colores dominantes expresados a través de expresiones lingüísticas.

Pero ¿Por que utilizar conjuntos de colores difusos? ¿Son Realmente necesarios? Si nos fijamos en los sistemas tradicionales, estos se basan en su mayoría en el espacio de color RGB, el cual representa los tres colores primarios; siendo estos el rojo, verde y azul. Este sistema nos marca entonces las coordenadas en un espacio de color de cualquier color que queramos representar. Por un lado y a modo de ejemplo tendríamos que el color negro esta representado como la ausencia de color, siendo sus coordenadas por tanto  $(0,0,0)$ . Por otro lado, el blanco estaría formado por el máximo de todos los colores, es decir,  $(1,1,1)$ .

Este espacio de color es normalmente usado en histogramas de color anotando la distribución de píxeles en los tres ejes mencionados anteriormente. Pero ante esto surge un problema ¿Como podemos definir si un color pertenece más o menos a un termino lingüístico de color como por ejemplo el rojo? Por esto se utilizan los conjuntos difusos, para asignar un grado de pertenencia a un color.

Este problema como podemos ver en [2] es un problema que puede ser clasificado también en el ámbito de la visión del color, el cual se encarga del estudio de la percepción del color a diferentes niveles (físico, psicológico, informático, etc) y es que el objetivo de estos sistemas es imitar la percepción humana del color, de forma que dada una imagen el sistema debe generar una expresión referente al color igual que lo haría un humano si observase lo representado en la fotografía en el mundo real. Sin embargo, uno de los problemas que surgen es la homogeneidad del color y es que los humanos vemos el color de un objeto de forma homogénea, independientemente de

las sombras o los cambios en la luminosidad que se produzcan en el. Por otra parte, un computador distingue estas diferencias de color debido a la luminosidad o a la perspectiva.

Otro de los problemas en este campo es la ya mencionada brecha semántica y es que, a pesar de que todo el mundo es capaz de reconocer si un color es Rojo o es Naranja, ¿cómo podemos definir la diferencia entre ambos? ¿Cómo representamos o 'enseñamos' a un computador lo que significa que un color sea Rojo? Una aproximación a este problema está representada por los sistemas de denominación de color.

Una aproximación muy natural a este problema puede pasar por realizar una cuantización del espacio de color, realizando así una partición del espacio de color en diferentes colores, de forma que la denominación de un color pueda estar basada en la similitud de ese color a cada una de las particiones que hemos realizado. Pero esta solución tampoco esta exenta de errores. Y es que a todos nos ha pasado, que para una persona, un determinado color es 'verde', por ejemplo, mientras que para nosotros ha sido 'azul' asignándole cada uno un color diferente. Esto se debe a que la denominación que le da cada persona a un color es muy personal y esta basada en varios aspectos, como pueden ser la experiencia de cada persona o el ámbito cultural. Otro ejemplo de esto puede verse en los casos en los que una persona puede ser capaz de distinguir entre varios tonos de rojo, como por ejemplo el Bermellón, Rojo cadmio, Granate, Rojo Carmín, etc. Mientras que para otra persona todos serán simplemente Rojo.

Para resolver todos estos problemas Jesús Chamorro *et al.* describe en [2] una posible solución basándose en el uso de conjuntos difusos pero antes de esto, hay que definir algunos conceptos básicos para la comprensión del funcionamiento de los conjuntos difusos.

- **Color difuso:** Podemos definir un color difuso como la representación computacional de un color, definido y nombrado por humanos. Debido al carácter de los conjuntos difusos este color tendrá asociado al menos un color el cual tendrá un grado de pertenencia máximo a ese color difuso de forma que no puede haber ningún color difuso el cual este representado por el conjunto vacío.
- **Espacio de color difuso:** Podemos decir que un conjunto de colores difusos está representado por todos los colores difusos que incluyamos en él.

Esta solución antes mencionada se basa a su vez en los diagramas de Voronoi o teselaciones de voronoi [25]. Estas teselaciones no son mas que la partición de un espacio en regiones llamadas células de Voronoi, produciendo

una por cada color difuso que aportemos. Y es que dado un espacio de color, como por ejemplo el RGB podemos crear una región espacial basándonos en sus tres dimensiones. Por tanto esta solución pasaría por crear una partición del espacio del RGB en forma de varios volúmenes en 3-D, representando cada uno un color.

Con el modelo propuesto, el grado de pertenencia a uno u otro color pasaría por determinar a cual de estos volúmenes pertenece un color dado y además de esto calcular la distancia al prototipo a partir del cual se ha creado el color difuso al que pertenece ese volumen. De esta manera podemos obtener una medida difusa de la pertenencia de un color dado a uno u otro color.

Una visualizaciones de estas regiones espaciales incluyendo las células de Voronoi pueden ser vistas en la figura 3.5.

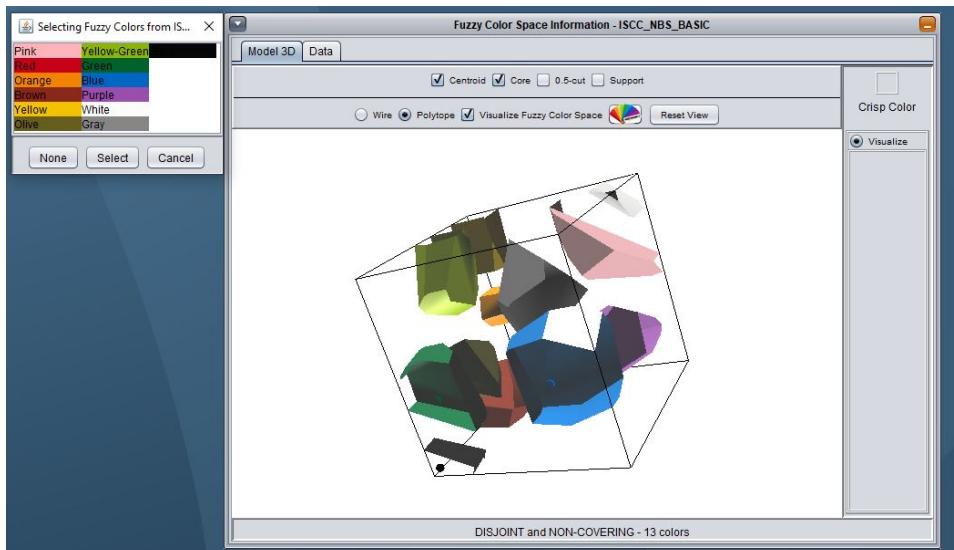


Figura 3.5: Visualización de un conjunto de colores difusos [15]

### 3.2. Análisis práctico

Como hemos visto en la anterior sección, los sistemas de recuperación de imágenes han pasado por diversas fases a lo largo de su evolución. Y es que, partiendo de soluciones muy poco prácticas y susceptibles a fallos, como los sistemas basados en texto los cuales recurren a etiquetas lingüísticas añadidas a mano, hemos llegado a sistemas altamente complejos y que utilizan las más novedosas técnicas para que analizando una imagen consulta obtenamos imágenes que sean relevantes en función a la consulta que hayamos realizado. Pero aun así y a pesar de los enormes esfuerzos que se realizan,

estos sistemas no son infalibles y en muchas ocasiones serán susceptibles a fallos debido a la gran variedad de propiedades diferentes que puede tener un objeto. De estas características ya se habló anteriormente y es que una taza, por ejemplo, puede tener gran variedad de formas o colores haciendo muy difícil que obtengamos un resultado relevante al realizar una consulta basándonos en este tipo de propiedades. Debido a esto, en esta sección analizaremos de forma práctica como se comportan estos sistemas realizando pruebas para poder ver sus puntos fuertes y sus puntos débiles.

Primero partiremos de una de las consultas más simples que podemos realizar, buscar la imagen de una taza roja en una de las bibliotecas de imágenes que he realizado. Esta base de datos esta formada por 800 imágenes, 100 de cada categoría más 16 imágenes de objetos que he descargado de internet que no pertenecen a ninguna de las otras categorías. La consulta se realizada en el sistema que he desarrollado para este proyecto utilizando el histograma de color. Dada la propiedad por la que estamos buscando, esta consulta podría ser traducida a 'Imágenes en las que haya mucha ocurrencia de rojo', por lo tanto es de esperar que los resultados que se obtengan sean de imágenes muy similares con respecto a la tonalidad del color que estamos buscando. Esta consulta se puede ver en la figura 3.6.



Figura 3.6: Búsqueda de una taza roja basándonos en el histograma de color

Como se puede observar en la imagen anterior los resultados se ajustan a lo que esperábamos, las imágenes resultante son imágenes en las cuales el color rojo esta claramente presente, además de esto se tiene muy en cuenta el hecho de que el fondo sea blanco ya que de esta manera el único color que aparece en la imagen es el rojo, por lo tanto podemos decir que nuestra consulta ha sido resuelta satisfactoriamente.

Sin embargo, el problema que surge de los descriptores basados en este tipo de métricas es que en el momento en el que introducimos una imagen con varios colores, el resultado es totalmente inesperado. Pudiendo obtener imágenes que difícilmente podremos relacionar con la imagen consulta. Un ejemplo de esto se puede ver en la consulta realizada en la figura 3.7. En

esta imagen, podemos ver en la hilera superior de resultados, el resultado al buscar utilizando únicamente el histograma de color. Como se puede ver, los resultados no están relacionados de ninguna manera (al menos a simple vista) con la búsqueda que hemos realizado. Sin embargo siempre podemos utilizar técnicas mas avanzadas para obtener mejores resultados.

Al contrario de lo obtenido al realizar la búsqueda usando el histograma de color, podemos ver en la segunda hilera de la figura 3.7, un resultado mucho más acertado al utilizar otro descriptor basado en estructuras de color. Como se puede observar este resultado se acerca mucho más a lo que esperamos obtener al realizar una búsqueda de este tipo. Y es que, aunque no obtengamos imágenes pertenecientes al mismo tipo de objeto, los colores, si que son mucho mas parecidos. Se puede ver que en todas las imágenes aparecen los colores de la imagen consulta y además en tonos relativamente parecidos.

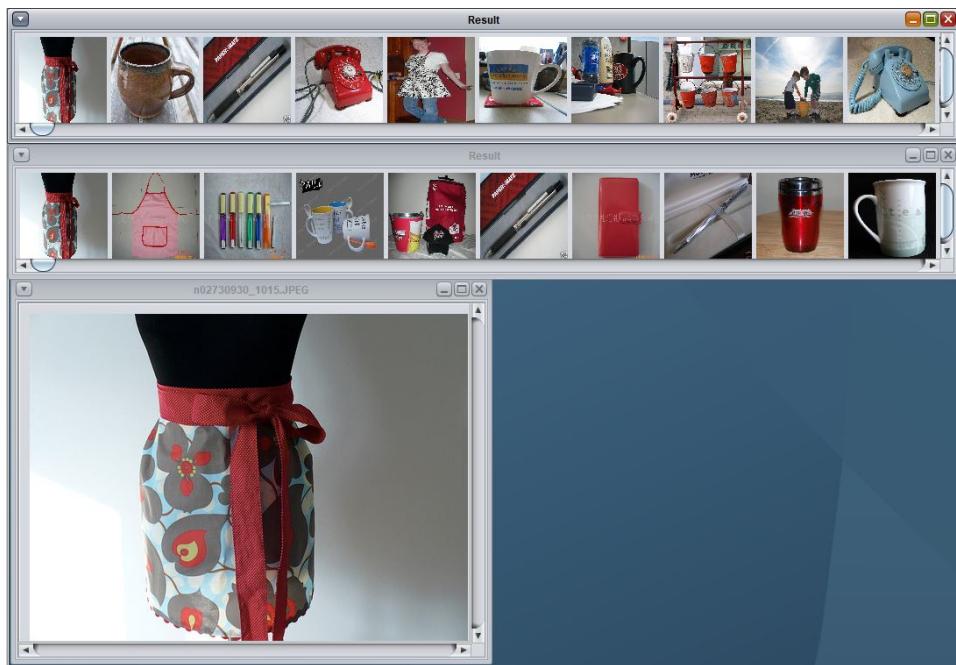


Figura 3.7: Resultado de buscar una imagen basándose en el histograma de color o en la estructura de colores

Sea como sea, podemos ver que ni siquiera en el caso mas simple (una imagen con un único color en ella) podemos obtener objetos del mismo tipo. Es por este motivo por el que aparecieron los, antes mencionados, descriptores basados en características de segundo nivel. Estos se centran en el uso de etiquetas lingüísticas descriptivas de la imagen. Como ejemplo, realiza-

remos una consulta basándonos únicamente en sus etiquetas aportando una imagen. Por lo tanto se espera encontrar resultados de la misma categoría de imágenes. Cabe destacar que para todas las consultas que implican la clasificación de objetos se usará una red neuronal convolucionada de tipo ResNet. Por lo tanto la consulta que realizaremos sera la de una taza roja y se espera encontrar en los resultados imágenes que contengan solo tazas. Esta consulta se puede ver en la figura 3.8



Figura 3.8: Búsqueda de una imagen basándose en sus etiquetas lingüísticas

Como se puede ver en la imagen anterior las imágenes que obtenemos se ajustan a lo esperado y es que aunque las imágenes no sean parecidas a nivel de color, en todas ellas aparecen tazas.

Viendo todos estos ejemplos, queda patente la potencia que tienen los sistemas que se han utilizado, sin embargo cada uno tiene sus fallos. En el primer ejemplo de la figura 3.6 obteníamos imágenes que se parecían visualmente mucho a la consulta, pero que sin embargo, no tenían ninguna relación con el tipo de objeto que en ellas se representaba. Por otro lado, en el tercer ejemplo, representado en la figura 3.8, obteníamos resultados relevantes en cuanto al tipo de objeto pero que no guardaban ninguna relación en cuanto a sus propiedades visuales. Sin embargo, hay una solución a los problemas derivados de las dos técnicas, en el siguiente capítulo se abordará dicha solución.

## Capítulo 4

# Descriptor realizado

En este capítulo, se continuará con el tema tratado al final del capítulo anterior. Este problema consiste en la solución a los inconvenientes que presentan tanto los sistemas *CBIR* basados en características de primer nivel como los basados en características del segundo nivel.

La solución propuesta pasa por el desarrollo de un descriptor. Pero antes de abordar el problema, debemos definir que es un descriptor y que es lo que lo compone: Un descriptor es un tipo de objeto o métrica el cual describe, en este caso, una imagen. Para realizar esta tarea, los descriptores, se basan en algún tipo de característica o propiedad de la imagen que quieran describir. Los tipos de características que presentan los descriptores enfocados a la recuperación de imágenes se describieron en capítulos anteriores (Capítulo 3).

Para continuar definiendo de que tratará este capítulo, debemos decir que en él, se hablará del diseño de la solución y de por qué se han tomado las decisiones que se han tomado, sin entrar en exemplificar estas soluciones. Los ejemplos del funcionamiento se aportarán en el capítulo dedicado a la validación del sistema por estar esta fase incluida en el proceso de desarrollo del sistema, perteneciente a la segunda parte de esta memoria.

Para comenzar recordaremos los problemas por los que ha sido necesaria la creación de este descriptor. Como mencionamos en el capítulo anterior, si realizamos la búsqueda de una imagen en una base de datos en función a una propiedad visual, como puede ser el color medio o el histograma de color obtendremos gran variedad de imágenes las cuales estarán relacionadas o no con nuestra consulta. Esto se debe a que si en una imagen el color rojo es muy relevante obtendremos imágenes en las que este color también lo sea. De esta forma, puede ocurrir que al buscar una imagen con una taza Roja, obtengamos imágenes de cualquier otro objeto que también sea rojo. Por

otro lado, si buscamos imágenes y nos fijamos en el tipo de objeto que estas representan no se tendrá en cuenta el color, con lo que obtendremos objetos de todos los colores. Ejemplos de esto se pueden ver en las figuras 4.1 y 4.2. Cabe destacar, que en la figura 4.2 aparecen imágenes que aparentemente están ordenadas en función al color, esto se debe a que a la hora de añadir las imágenes a la base de datos, estas se han introducido en ese orden. El color no se está tomando como referencia en ningun caso para mostrar ese resultado.

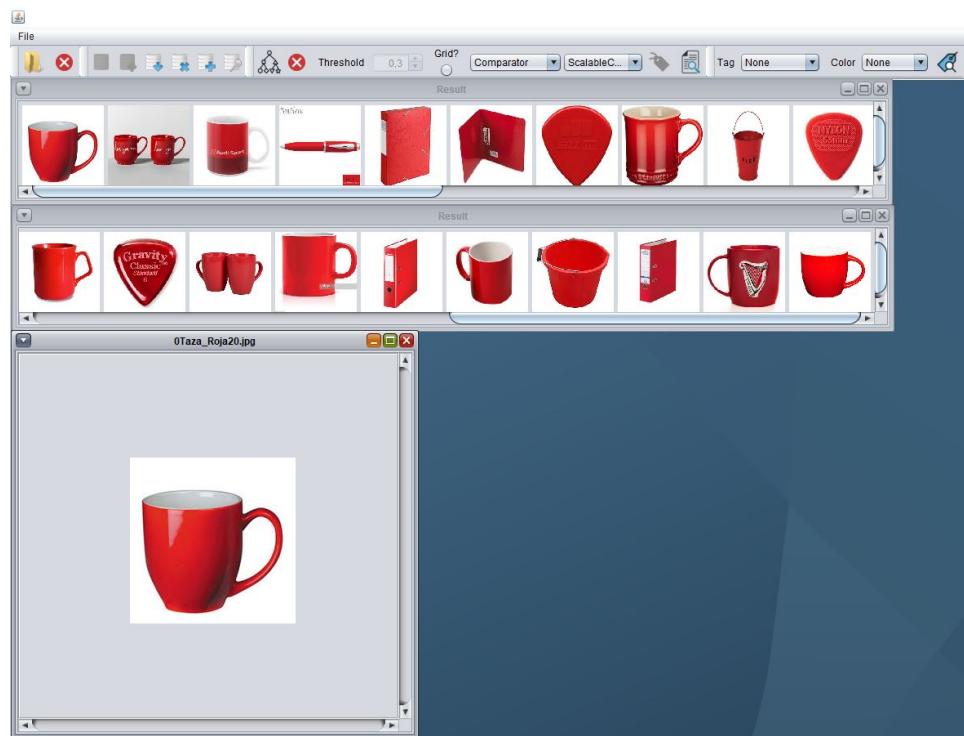


Figura 4.1: Consulta por color

Como se puede ver en las imágenes, dependiendo del tipo de consulta que queramos realizar obtendremos mejor o peor resultado. Si queremos buscar imágenes rojas, podremos hacerlo. Si queremos buscar tazas, podremos hacerlo. Pero si queremos buscar tazas rojas, no podremos hacerlo.

Llegados a este punto surge la posibilidad de unificar ambas características, de este modo podremos realizar consultas en base a etiquetas lingüísticas y a propiedades visuales de las imágenes. Sin embargo, podemos llegar aun más lejos y es que ¿Por que no realizar consultas en base a varias propiedades visuales a la vez? Esto nos puede ser útil en determinados casos, imaginémonos que queremos buscar imágenes de un objeto con una deter-

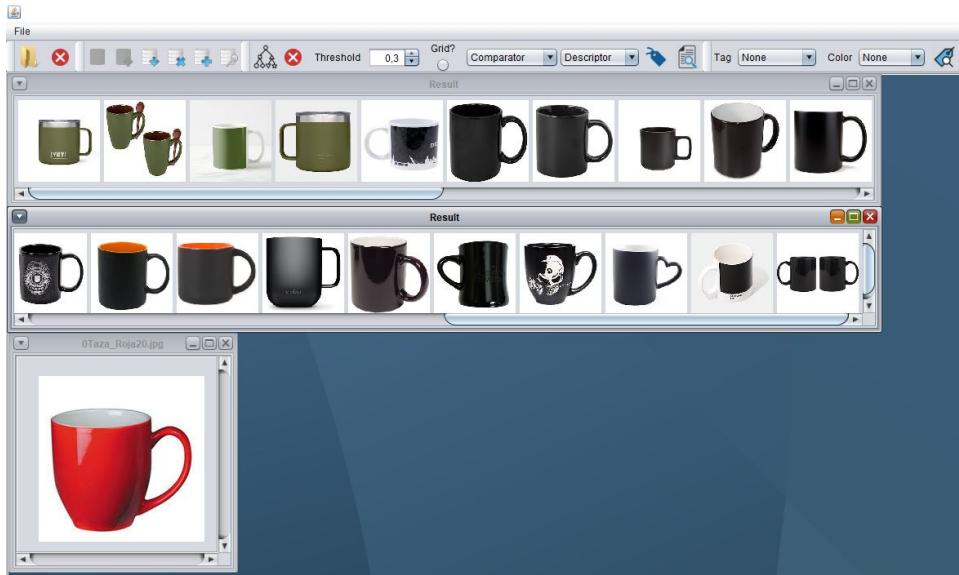


Figura 4.2: Consulta por etiqueta

minada forma y un determinado color. Para realizar esta búsqueda no nos bastaría con el uso de, por una parte, una etiqueta lingüística que describa el objeto que aparece representado en la imagen y, por otra, una medida de sus propiedades visuales, como puede ser el color. Para resolver esto necesitaríamos unificar de alguna forma las propiedades visuales de primer nivel de un objeto. Para resolver esto, las propiedades visuales son una lista de descriptores, teniendo cada una de estas un peso o relevancia aportado por el usuario.

El resultado obtenido a partir de estas ideas, nos permite solventar los problemas observados en las imágenes 4.1 y 4.2. Un ejemplo de ello se puede ver en la figura 4.3. Como se puede ver, realizando la misma consulta que en los dos casos anteriores podemos observar una clara mejoría ya que en este caso, el sistema nos proporciona imágenes relevantes independientemente del aspecto en el que nos fijemos, ya sea fijándonos en el color del objeto o en el tipo de objeto que aparece representado.

Una vez ya hemos obtenido el descriptor anterior, se nos presenta un problema a resolver. Este problema consiste en la necesidad de tener una imagen para realizar una consulta y es que de esa manera el sistema solo nos permite realizar consultas del tipo "Busca imágenes que se parezcan a esta". Para solucionar este problema hay varias soluciones posibles. Una de estas posibles soluciones, podría ser aportar una gama de ejemplos lo suficientemente grande como para satisfacer las necesidades de consulta más

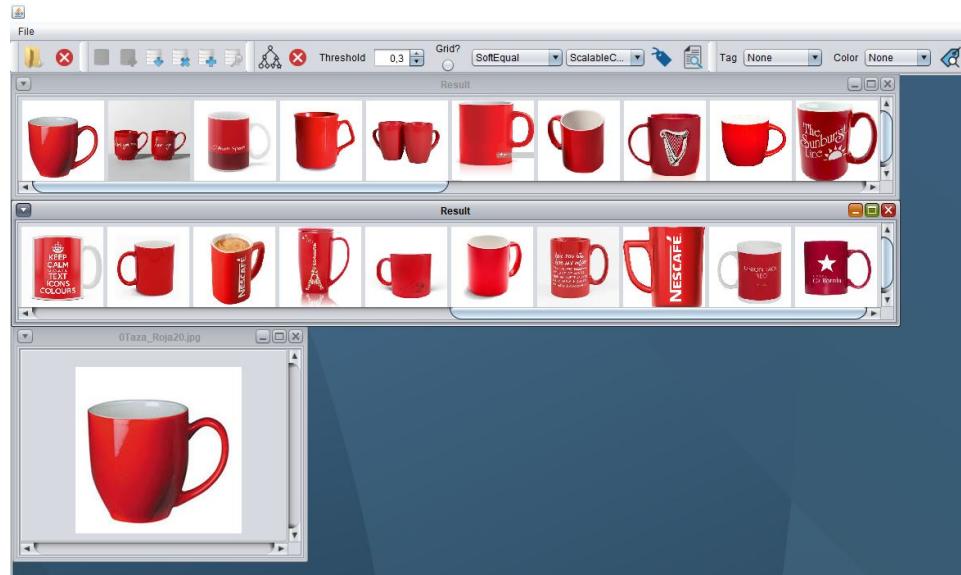


Figura 4.3: Consulta por etiqueta y color utilizando el sistema desarrollado

probables del usuario. De esta manera el usuario podría elegir un ejemplo de consulta y buscar en base a él. Otra posibilidad podría ser aportar la opción de buscar en función a una etiqueta lingüística por un lado y a un selector de propiedades visuales por otro. De esta forma el usuario podría realizar consultas del tipo "Busca imágenes que contengan tazas y que su color medio sea en RGB el (128,128,0)". Sin embargo, y a pesar de que estas soluciones podrían ser validas no resuelven el problema de la brecha semántica. Y es que para nosotros, los humanos, no nos es natural identificar los objetos por su color medio y mucho menos por otras características como el histograma de color. Debido a esto se ha decidido implementar la posibilidad de realizar consultas de una forma mucho mas natural e intuitiva, pasando esta por el etiquetado de colores. De esta forma el usuario podría realizar consultas del tipo "Busca tazas rojas".

Para realizar esto, se ha creado otro descriptor como caso especial del descrito anteriormente. En este nuevo descriptor simplemente tomaremos las propiedades visuales del objeto como una etiqueta lingüística. Esta será calculada a partir de los colores dominantes de la imagen y es que hay que destacar que en este caso, no se etiqueta el color del objeto en sí, si no que etiquetaremos los colores dominantes de la imagen, esperando que uno de estos colores sea el color del objeto que queremos recuperar.

Como se comentó en anteriores capítulos podemos tomar varios caminos para realizar la tarea del etiquetado de colores. En este descriptor se apor-

tarán dos posibles soluciones, siendo la primera de estas la más intuitiva, pero la menos efectiva en términos de ordenación de resultados. La segunda de estas pasa por el uso de conjuntos de colores difusos, siendo esta solución menos natural pero más efectiva en lo que a ordenación de resultados se refiere. Por lo tanto y debido al carácter que debe tener un sistema de recuperación de información en cuanto a la aportación de resultados diremos que esta ultima solución es la correcta o la que mejor resuelve el problema.

En el primer caso, la resolución del problema ha consistido en el cálculo de la similitud de un color dominante a un conjunto de colores de muestra. Para ello se han definido varios colores por cada etiqueta de color, por lo que tendremos, por ejemplo, varios tonos de rojo haciendo referencia al color rojo. De esta manera dado un color se comprobara cual es el color más cercano de entre la lista de colores, si ese color es por ejemplo, un tono de rojo, asumiremos que el color que queremos etiquetar será el Rojo.

Los colores de muestra que serán usados en la primera solución están basados en los colores definidos en el sistema básico del *ISCC-NBS* (*Inter-Society Color Council-National Bureau of Standards*) [19]. Este conjunto de colores esta compuesto por trece colores los cuales representan los colores que son reconocibles e innatos para cualquier persona del mundo. Estos, fueron propuestos por primera vez por Deane B. y Kenneth L. en [12]. Una lista de los colores que se han usado se puede ver en la figura 4.4

Por otra parte, en la segunda solución, utilizaremos un conjunto de colores los cuales representarán por si solos un único color en concreto. A partir de este conjunto de colores, crearemos un espacio de colores difusos usando cada uno de estos colores como prototipo representante de un color. En este espacio de colores difusos solo habrá, por tanto, un color por cada posible etiqueta lingüística.

El uso de esta técnica nos permite que dado un color el sistema le asigne una etiqueta lingüística con un peso asociado que representará el grado de pertenencia a un color o a otro. De esta forma resolvemos el problema que se plantea al intentar definir si un color es más rosa que rojo, por ejemplo. Una lista de los colores que se han usado para crear esta última solución se puede ver en la figura 4.5.

<b>Pink</b>	<b>Orange</b>	<b>Brown</b>
RGB	Color	RGB
254, 181, 186		138,73,37
196, 131, 121		180,116,94
		66, 37, 24
<b>Yellow-Green</b>	<b>Purple</b>	<b>White</b>
RGB	Color	RGB
141,182,20		240,240,240
<b>Red</b>	<b>Yellow</b>	<b>Olive</b>
RGB	Color	RGB
190, 1, 50		102,93,30
150,10,10		120, 120, 60
180, 60, 70		60,60,20
130, 20, 45		
150,50,50		
<b>Green</b>	<b>Blue</b>	<b>Gray</b>
RGB	Color	RGB
0,152,70		135,134,134
39,166,76		214,214,214
138,154,91		170,170,170
23,54,32		100,100,100
103, 146, 103		
147,197,146		
<b>Black</b>		
RGB	Color	
20,20,20		
40,40,40		

Figura 4.4: Colores modelo utilizados para etiquetar los colores

Prototipos		
Name	RGB	Color
Pink	220, 160, 161	
Red	230, 0, 38	
Orange	243, 132, 1	
Brown	180, 116, 94	
Yellow	243, 195, 1	
Olive	102, 93, 30	
Yellow-Green	141, 182, 1	
Green	70, 180, 20	
Blue	67, 90, 230	
Purple	154, 78, 174	
White	252, 252, 249	
Gray	135, 134, 134	
Black	7, 7, 7	

Figura 4.5: Prototipo de colores que serán usados para crear el espacio de colores difusos

Como se ha visto anteriormente el cálculo de los colores dominantes es esencial para el etiquetado de las imágenes ya que estos serán los que etiquetaremos. Sin embargo, hay que destacar que no todos los colores de la imagen serán etiquetados y es que se ha intentado hacer una simplificación de todos ellos, de forma que se etiqueten la menor cantidad de ellos, esto se hace unificando varios colores en el mismo, de modo que las imágenes con gran variedad de colores nos aportará resultados que no serán válidos. Esto se ha hecho ya que en el caso de buscar un objeto y un color, el color que se buscará no será el del objeto por sí solo, si no el color de la imagen. Esto se puede ver más fácilmente en la figura 4.6.

Si nos fijamos en la figura antes mencionada nos ha dado unas etiquetas de '*Grey*', '*Brown*' y '*Red*'. Este resultado tiene mucho sentido si nos fijamos en la imagen: El gris viene por el color de dentro de la taza, el cual como se puede ver no es totalmente blanco, si no que es un poco gris. El Rojo está claro que viene del color de la imagen y el marrón podemos asumir que viene debido a la superficie sobre la que se encuentra la imagen.

Como podemos ver, la salida parece correcta, pero aún así esto presenta un problema y es que si realizásemos la búsqueda '*Taza gris*' o '*Taza marrón*' nos daría como resultado la imagen anterior, cosa que no sería correcta. Además de esto, si nos fijamos en la superficie sobre la que se encuentra la taza,



Figura 4.6: Resultado de etiquetar el color de la imagen de una taza roja

en la parte superior de la imagen esta se ve mucho más rojiza que en la parte inferior, lo que podría dar lugar a dos colores diferentes. Esto provoca que el numero de etiquetas de colores que no son relevantes aumente, haciendo más difícil la identificación del color del objeto. Otro ejemplo se puede ver en la figura 4.7.

En este caso, se puede ver que nos ha dado una única etiqueta, siendo esta '*Brown*', la cual es difícil relacionar con la imagen. En este caso podemos ver el problema mencionado anteriormente y es que, al haber gran variedad de colores no se elige uno en concreto si no que los colores dominantes se obtienen en función a la media de varios colores dominantes.

Por el problema antes mencionado se ha preferido utilizar la base de datos de imágenes sin fondo ya que de esta manera podremos considerar que todos los colores relevantes en la imagen serán colores del objeto, obteniendo así resultados donde las etiquetas de color estarán ligadas al objeto y no a su entorno. Ejemplos de esto se pueden ver en las figuras 4.8, 4.9 y 4.10.

Una vez, comprobado que el etiquetado de color funciona correctamente, podemos cargar la base de datos con imágenes y realizar búsquedas. Estas consultas se realizarán utilizando el comparador implementado '*OnlyLabels-Comparator*'. Este comparador, se ha desarrollado con un único propósito, realizar una consulta por etiquetas lingüísticas estas etiquetas pueden hacer referencia a las de la imagen únicamente o a las de la imagen y a las del

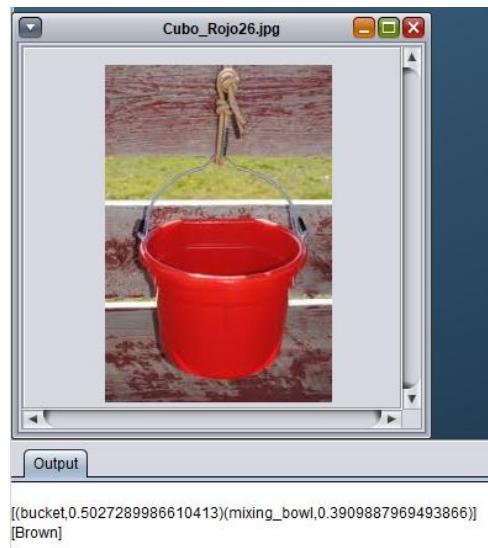


Figura 4.7: Resultado de etiquetar el color de la imagen con un cubo rojo

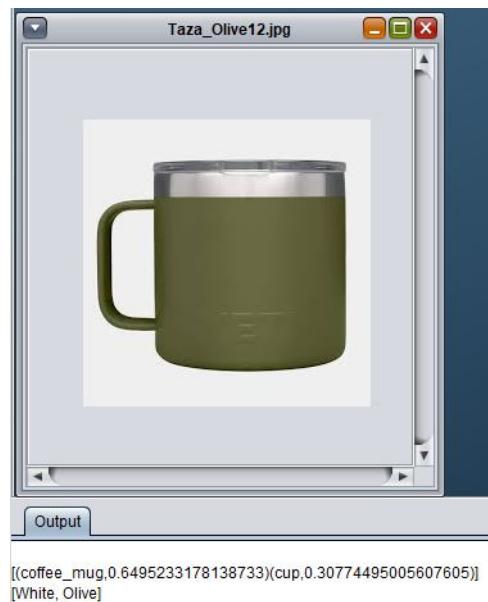


Figura 4.8: Resultado de etiquetar el color de la imagen con una taza oliva



Figura 4.9: Resultado de etiquetar el color de la imagen con una taza azul



Figura 4.10: Resultado de etiquetar el color de la imagen con un bolígrafo verde

color. En la figura 4.11 se pueden ver consultas de tazas de café variando el color.

Como se puede ver en la figura 4.11 los resultados que se obtienen al buscar por objeto y color son bastante buenos. Sin embargo, hay un resultado que no se corresponde a lo que esperado. En las tazas amarillas, la primera de ellas es una taza roja con letras amarillas. Aun así podemos considerar que el resultado es bueno ya que estamos etiquetando los colores dominantes en la imagen, siendo el amarillo uno de estos colores dominantes.

Por otro lado, podemos ver que aunque todos los resultados en la figura 4.11 son buenos no parece que estén ordenados en función a ningún criterio. Dicho de otra forma, los colores no están agrupados en función a su similitud. Sin ir más lejos, en las tazas rojas, el primer y el tercer resultado son parecidos en el tono de rojo mientras que el segundo no se parece a ninguno de ellos al ser este último un rojo más intenso. Esto se soluciona haciendo uso de la opción de los conjuntos difusos, de esta manera los colores si aparecerán agrupados en función a su tonalidad. La anterior consulta activando los conjuntos difusos se puede ver en la figura 4.12.

En el caso de la figura anterior, se puede ver que en el resultado obtenido, los colores aparecen distribuidos de una forma mucho más ordenada. Si es cierto que sigue habiendo algún resultado en el cual se da un salto de tonalidad, pero esto es normal, ya que se ordena en función a la distancia al color que se ha usado como prototipo, por lo tanto, una tonalidad más clara que el prototipo puede tener la misma distancia que una tonalidad más oscura si ambas se alejan lo mismo de él.

Sin embargo y como se comentó en el capítulo anterior, el descriptor no es solo la representación computacional de la imagen (en este caso, una etiqueta y las propiedades visuales de la imagen), si no que también es la forma de definir la similitud entre dos imágenes basándonos en su representación. Para resolver este último problema, se han desarrollado varios comparadores, intentando cubrir la mayor cantidad de posibles consultas que quiera hacer el usuario, de estos descriptores y del por que de cada uno de ellos se hablará a continuación.



Figura 4.11: Búsqueda en la base de datos por etiquetas lingüísticas



Figura 4.12: Búsqueda en la base de datos por etiquetas lingüísticas usando conjuntos difusos

## 4.1. Comparadores

En el capítulo anterior se habló de como a partir de las propiedades visuales y mediante el uso de técnicas de alto nivel, se ha definido un descriptor el cual es capaz de proveernos de unos resultados precisos si realizamos consultas a partir de él. Para hacer esto, además de realizar las operaciones referentes al procesamiento de las propiedades visuales del objeto es necesario definir algún tipo de mecanismo para poder establecer la similitud entre dos descriptores. Precisamente esto son los comparadores, una herramienta la cual nos permite comparar dos descriptores del mismo tipo y que como resultado nos aporta una medida de distancia entre ambos descriptores. Para ello hay que tener en cuenta que dados dos descriptores, ambos van a tener por un lado unas etiquetas lingüísticas referentes a la categoría del objeto y otra propiedad, la cual hará referencia a las múltiples propiedades que se pueden tener en cuenta en este descriptor.

Podemos dividir los comparadores usados en dos grupos: Por un lado están los comparadores centrados en las etiquetas y el comparador creado centrado en las propiedades. De forma paralela también se han creado comparadores para el descriptor basado en cuadriculas, ya que el comparador que este descriptor aportaba no nos era suficiente.

### 4.1.1. Comparadores centrados en etiquetas

Para ver la necesidad que surge a raíz de las etiquetas lingüísticas lo más fácil es ver dos imágenes con sus respectivas etiquetas y ver las posibilidades que surgen. Y es que el problema se presenta en los objetos que pertenecen a varias categorías, esto provoca que la red neuronal le asigne al objeto las etiquetas pertenecientes a ambas categorías. Un ejemplo muy claro de esto es el que se puede ver en la figura 4.13. Estos comparadores ya estaban implementados en la biblioteca utilizada, pero se explicará brevemente su funcionamiento y como están integrados con las propiedades visuales.

Como se puede ver en la figura 4.13, ambos objetos son tazas, por lo que la CNN le asigna a cada una de ellas la etiqueta '*Coffee\_mug*' pero sin embargo a la taza amarilla, también se le asigna la etiqueta '*Cup*' (la cual hace referencia a '*tea cup*') esto provoca que en casos como este podamos comparar ambas etiquetas de diferentes formas. De ahí surgen los comparadores usados.

#### Comparador de igualdad suave

Este es el comparador más permisivo, ya que para asumir que dos descriptores están relacionados basta con que alguna de las etiquetas de los

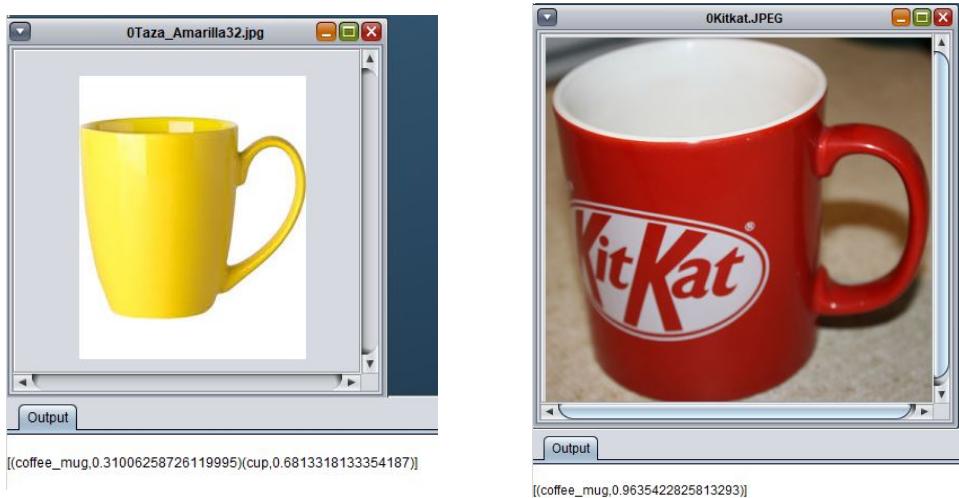


Figura 4.13: Imágenes con sus respectivas etiquetas calculadas

descriptores sean iguales. Si ocurre que se determina que los descriptores están relacionados la distancia entre ambos será la que determinen las propiedades visuales. Por lo tanto, este será el único caso en el que si nos fijamos en las imágenes de la figura 4.13 obtendremos que ambas imágenes están relacionadas busquemos una u otra.

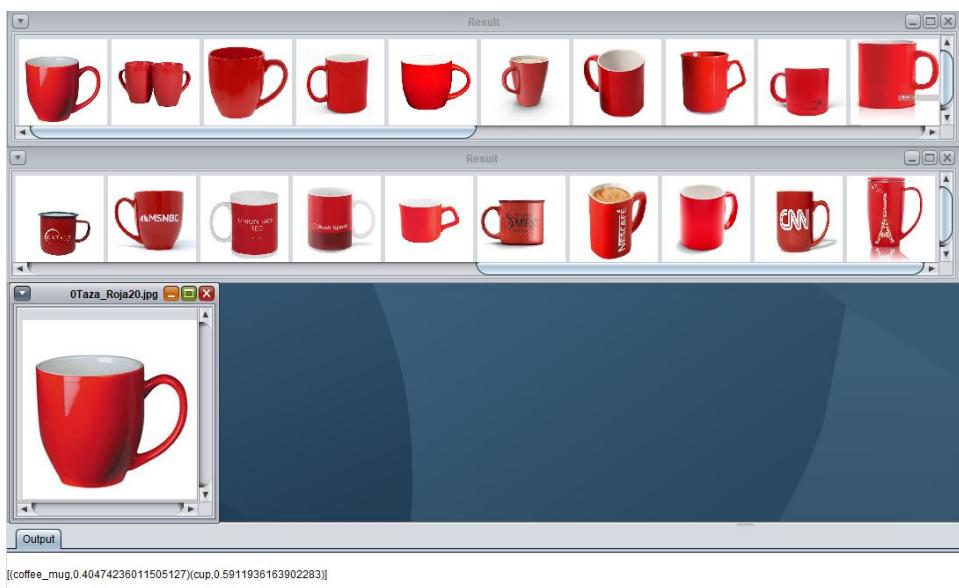


Figura 4.14: Búsqueda de una imagen usando el comparador SoftEqual

En el caso de la figura 4.14 como se puede ver, la imagen que se ha buscado tiene dos etiquetas, las cuales son *cup* y *coffee mug* por lo que en

los resultados aparecerán imágenes que tengan alguna etiqueta en común.

### Comparador de igualdad

En este caso, el comparador comprobará si las etiquetas de los dos descriptores que este comparando son iguales, en el caso de que así sea. Se tomará como distancia la distancia que haya entre sus propiedades visuales. En el caso de que las etiquetas no coincidan totalmente, la distancia será máxima, es decir, ambas imágenes no están relacionadas.

Esto presenta un problema y es que hay casos en los que al ser tan restrictivo este comparador tomamos como imágenes no relacionadas aquellas que si puedan estarlo, un ejemplo de esto son las imágenes de la figura 4.13 las cuales según este comparador no están relacionadas.



Figura 4.15: Búsqueda de una imagen usando el comparador de igualdad

Como se puede ver en la figura 4.15, al realizar la búsqueda utilizando este comparador, obtenemos muchos menos resultados relevantes. En este caso incluso obtenemos una taza azul al final, esto se debe a que no hay más tazas rojas que cumplan con el criterio de este comparador por lo que empieza a mostrar imágenes de otros colores.

### Comparador de inclusión

Al contrario que en el caso anterior, este comparador es más permisivo, permitiendo que si las etiquetas del descriptor consulta están incluidas en las etiquetas del otro descriptor se dará dicho resultado como válido y por lo tanto se tomará como distancia la distancia que haya entre sus propiedades

visuales. En el caso de que las etiquetas de uno no estén incluidas en el otro, ambas imágenes no estarán relacionadas.

En el ejemplo de la figura 4.13 si se busca utilizando la taza roja, nos dirá que la amarilla si está relacionada. Mientras que si buscamos la taza amarilla nos dirá que la taza Roja no está relacionada. En este caso, la consulta devuelve el mismo resultado que si utilizamos el comparador de igualdad de la figura 4.15 ya que buscara imágenes que incluyan las dos etiquetas de la imagen y al no haber imágenes que contengan 3 etiquetas, siendo dos de ellas las buscadas, nos dará el mismo resultado.

Por otro lado si utilizamos una imagen con una sola etiqueta nos dará el mismo resultado que en la figura 4.14, esto se debe a que al tener una sola etiqueta buscaremos que esta se encuentre incluida en la otra imagen, cosa que hace el comparador SoftEqual.

Todo esto se debe a que una imagen de una taza como las de las imágenes anteriores rara vez serán etiquetas con algo que no sea o *Coffee\_mug* o *cup* pero sin embargo, si utilizamos imágenes mas complejas podremos ver una clara diferencia. Para demostrarlo utilizaremos imágenes de coches descapotables. Y faremos una búsqueda con cada uno de los comparadores anteriores. El resultado se puede ver en la figura 4.16, los resultados se encuentran estructurados de la siguiente forma: Arriba nos encontramos el comparador de igualdad, en el centro el comparador de inclusión y abajo el comparador de igualdad suave.

Como se puede ver el comparador de inclusión se encuentra a mitad de camino de los otros dos comparadores ya que este nos incluye unas cuantas imágenes más que el comparador de igualdad pero menos que el comparador de igualdad suave.

### Comparador basado en pesos de las etiquetas lingüísticas

En este caso, en lugar de fijarnos solo en las etiquetas lingüísticas, nos fijamos en el grado de certeza de estas etiquetas de forma que a parte de poder utilizar la igualdad o la inclusión en las etiquetas el caso de que las etiquetas sean iguales o estén incluidas respectivamente se utilizará la distancia entre etiquetas. Esta distancia puede ser calculada mediante 4 operaciones o tipos de distancia:

- Mínimo: Se calcula la distancia entre etiquetas iguales y se utiliza la que sea más pequeña.
- Máximo: Entre las etiquetas iguales se utiliza la diferencia mayor.
- Media: En lugar de utilizar las distancias mayores o menores como en los dos casos anteriores se hace una media de todas estas distancias.

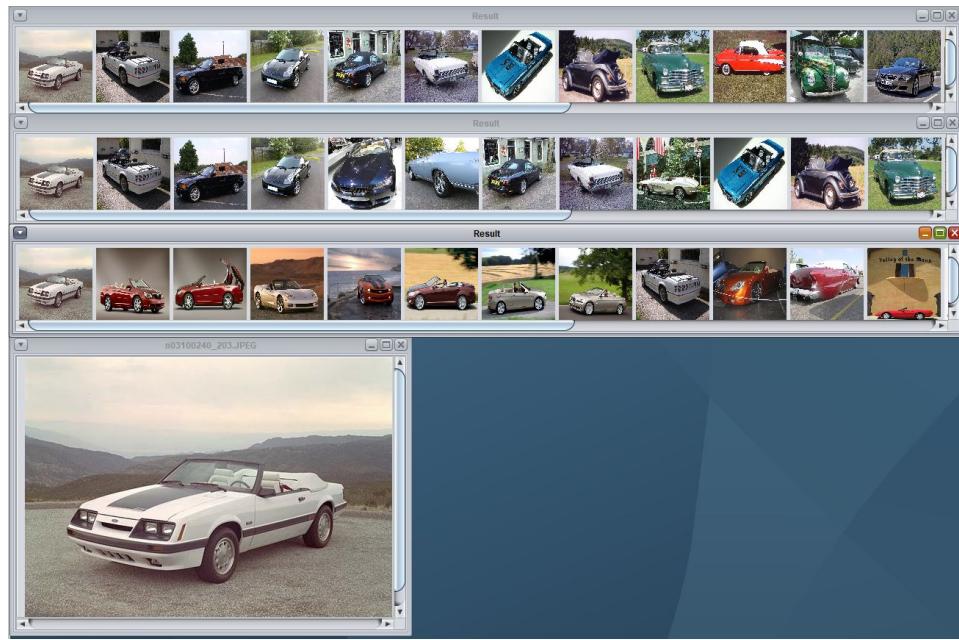


Figura 4.16: Comparación de los comparadores de igualdad suave, igualdad e inclusión

- Euclídea: De la misma forma que en el caso anterior salvo que en lugar de hacer la media entre todas las distancias se utiliza la distancia euclídea

Una vez se ha obtenido alguna de las distancias anteriores; siendo ' $dL$ ' la distancia entre etiquetas, ' $dP$ ' la distancia entre las propiedades y ' $Dist$ ' la distancia entre ambas imágenes se realiza la siguiente operación:

$$Dist = (dL + 1) * dP$$

Esta formula puede ser interpretada de la siguiente forma: Si en la *CNN* se interpreta que las imágenes son muy parecidas casi con total seguridad las imágenes estarán relacionadas por lo que la distancia entre ambas vendrá dada por la similitud en sus propiedades. Sin embargo, si para la *CNN* las imágenes difieren mucho, puede ocurrir que realmente estas no estén muy relacionadas y que aunque las propiedades de la imagen sean muy parecidas, no estén realmente relacionadas. En este último caso se penalizarían estas propiedades aumentándolas al doble de su valor como máximo.

Para mostrar el uso de este comparador, primero realizaremos una consulta sin tener en cuenta las propiedades visuales, utilizando el comparador basado en pesos y calculando la distancia en función al mínimo. Esto se realizará en una base de datos de coches descapotables. El resultado se puede ver en la siguiente imagen.



Figura 4.17: Búsqueda de imagen basándose exclusivamente en el peso de las etiquetas.

Como se puede ver en la figura, en este caso vemos que las imágenes que nos aporta están todas relacionadas, y si nos fijamos en los colores de la imagen podríamos decir que realmente se parecen. Esto último sucede porque de forma habitual los coches suelen estar sobre asfalto, siendo este siempre del mismo color, y tampoco es extraño que aparezcan o arboles o césped en la imagen. Esto se puede ver en la figura 4.18 en ella se puede ver que aunque aparezcan coches rojos es por la misma razón. En este caso da igual en base a que se calcule la distancia ya que la imagen solo tiene una etiqueta.

Sin embargo, podremos ver unos resultados mucho mejores si además de en las etiquetas nos fijamos en las propiedades visuales. En la figura 4.19 podemos ver que ocurre cuando utilizamos el descriptor de histograma de color. Como se puede ver todos los resultados salvo 2 podríamos decir que son relevantes ya que son coches rojos. Las otras dos imágenes que aparecen se deben a que son imágenes muy verdes, y por el paisaje de la imagen consulta, están relacionadas en color.

Podríamos ver otro ejemplo usando la estructura de color. El resultado de la consulta usando este descriptor se puede ver en la figura 4.20. Se puede ver que en este caso funciona peor que en el caso anterior, esto se debe a la forma de calcular este tipo de descriptores. Aun así el resultado sigue siendo mejor que en la opción sin descriptores de color.



Figura 4.18: Búsqueda de imagen basándose exclusivamente en el peso de las etiquetas.



Figura 4.19: Búsqueda de imagen basándose en el peso de las etiquetas y en el histograma de color



Figura 4.20: Búsqueda de imagen basándonos en el peso de las etiquetas y en la estructura de color

#### 4.1.2. Comparadores centrados en las propiedades visuales

Si nos fijamos en la parte de las propiedades visuales, cada uno de los descriptores implementados en la biblioteca usada para realizar este proyecto, aporta su propio comparador. El problema viene debido a la característica del descriptor que nos permite añadir más de un descriptor a una consulta, de forma que podríamos realizar consultas en función a la forma y al color de un objeto. Para esto, ya hay un descriptor implementado en la biblioteca. Este representa una lista de descriptores y su comparador nos aporta la distancia euclídea entre los descriptores de la lista. Esto tiene sus desventajas y es que en el caso de, por ejemplo, realizar consultas por la textura y el histograma de color de una imagen no podemos definir a cual de dichas características queremos darle más importancia.

Por el motivo antes mencionado, se ha desarrollado un nuevo comparador el cual dada una lista de pesos asociados a los descriptores de la lista, pondera las distancias de los descriptores y realiza la distancia euclídea entre todas ellas. La expresión matemática de esta operación es la siguiente:

$$Dist = \sqrt{\sum(w_i * D_i)^2}$$

Siendo  $w_i$  el peso de cada descriptor y  $D_i$  la distancia entre descriptores de color. En el caso de que la lista de pesos este vacía, se realizará la distancia euclídea entre todas las distancias.

Hay que destacar que no se ha usado este comparador en ninguna prueba, ya que al ser todos los descriptores usados descriptores de color, no se puede ver una clara diferencia entre asociarle un peso más alto a uno u a otro. Este comparador sería muy útil en el caso de utilizar descriptores de textura o forma pudiendo realizar consultas más complejas. Queda esto como una posible vía de futuro trabajo.

#### 4.1.3. Comparadores para el descriptor basado en cuadrículas

En esta sección hablaremos del descriptor basado en cuadrículas, este descriptor, dada una imagen la divide en secciones las cuales serán tratadas independientemente, calculando para cada una de las secciones un descriptor especificado. El comparador que nos ofrecía este descriptor, estaba basado en la distancia euclídea entre las secciones. La forma de calcularlo es comparando las secciones que se hallen en posiciones iguales y realizando la distancia euclídea a estos resultados. Esto provoca que para que el resultado sea bueno la distribución de todos los elementos en la imagen ha de ser exactamente la misma.

Para poder realizar consultas más complejas se han realizado dos comparadores para esta clase, en ambos de ellos no se tiene en cuenta la posición de los objetos, ni se tiene en cuenta si están repetidos. Para ver el uso de estos comparadores, se ha creado un grid usando en cada sección de la imagen el descriptor realizado. De forma que en cada sección tendremos una etiqueta y una propiedad, esto nos permitirá realizar búsquedas por alguno de los objetos de la imagen o por todos ellos.

##### **Comparador basado en el mínimo**

El objetivo de este comparador es que dado una imagen con varios objetos se busquen imágenes en las que aparezca al menos uno de los objetos.

Esto se consigue siguiendo el siguiente método: Para cada sección de la imagen consulta, se calcula la distancia a cada una de las secciones de las imágenes de la base de datos, almacenando por cada sección de la imagen consulta su distancia mínima. Una vez calculados todos estos mínimos se busca el menor de todos ellos, siendo este la distancia entre imágenes.

Si nombramos  $A$  a la imagen consulta,  $B$  a la imagen con la cual se le está comparando,  $A_i$  y  $B_j$  a cada una de las regiones de estas e  $i$  y  $j$  los índices de las secciones de cada una de las imágenes, esta operación puede ser descrita de la siguiente forma:

$$Dist = \min(\min(A_i - B_j), \forall i, \forall j)$$

En la figura 4.21 se ve el uso del comparador *minimumGridComparator* aplicado a una imagen con un objeto, en este caso un sacapuntas azul.

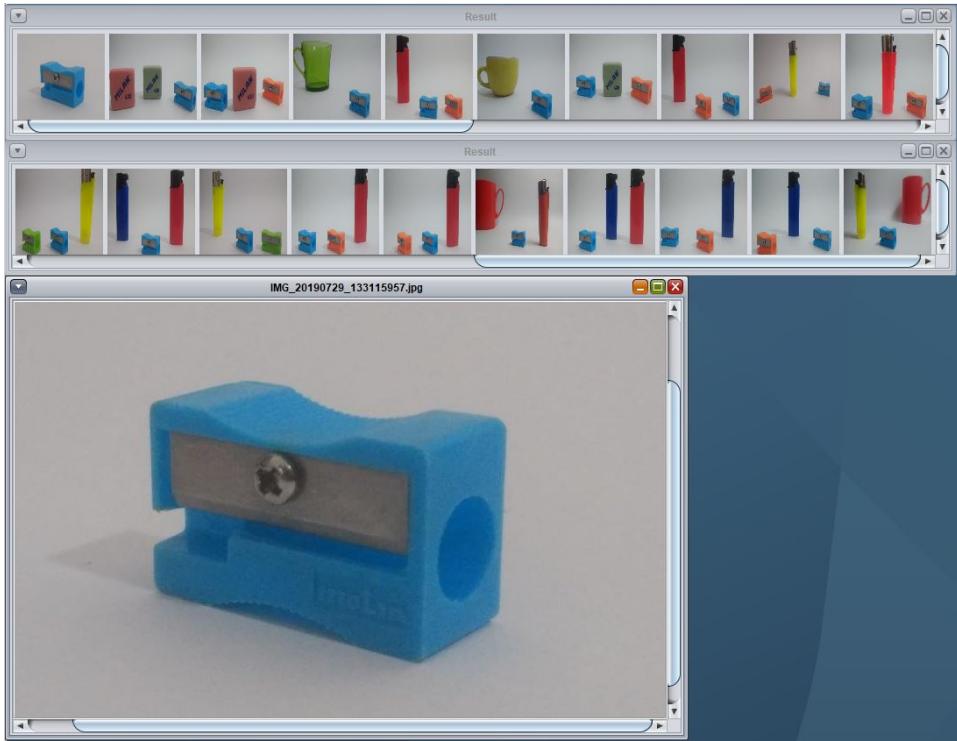


Figura 4.21: Búsqueda de un objeto usando el comparador basado en el mínimo

Como se puede ver, nos muestra imágenes en las que aparezca dicho objeto con dicho color independientemente del numero de objetos que aparezcan en las imágenes resultado, cumpliendo así con lo que se esperaba. Para continuar veremos su funcionamiento con dos y tres objetos respectivamente. Estos ejemplos se pueden ver en la figura 4.22 y ?? respectivamente.

Se puede ver como en ambos ejemplos se cumple con lo esperado, en ambos casos nos muestra imágenes en las que aparezca alguno de los objetos por los que se ha buscado y además estos aparecen con el mismo color. Sin embargo se puede observar como no aparecen las imágenes en las que el objeto está solo. Esto se debe a que en estas imágenes el color del objeto, al estar este solo, es muy relevante en la imagen mientras que en las imágenes buscadas, al ser el color menos relevante debido al tamaño aparente, se considera que el color de la imagen consulta está más relacionado con las imágenes de dos o tres objetos.

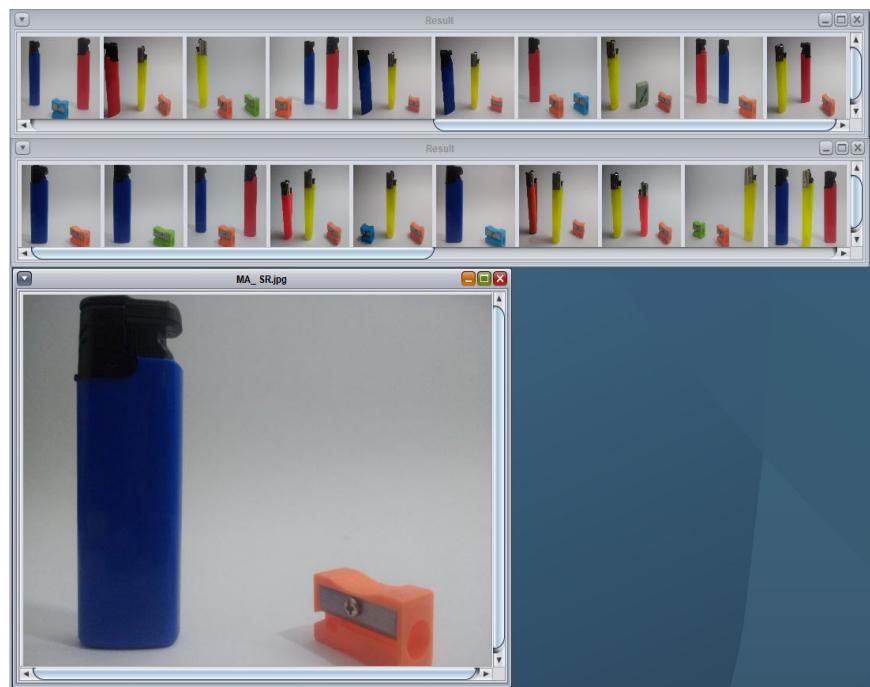


Figura 4.22: Búsqueda de dos objetos usando el comparador basado en el mínimo



Figura 4.23: Búsqueda de tres objetos usando el comparador basado en el mínimo

### Comparador basado en el máximo

Este comparador es muy similar al anterior, pero en lugar de buscar al menos uno de los elementos de la imagen se buscan imágenes en los que aparezcan todos ellos. Para conseguir esto se sigue un proceso similar al anterior: Primero se busca el mínimo de la comparación de todas las secciones de la imagen consultando con la imagen analizada en ese momento y de entre todos esos mínimos se selecciona el máximo. Si seguimos la misma notación que en la formulación anterior obtenemos que esta operación se puede expresar de la siguiente forma:

$$Dist = \max(\min(A_i - B_j), \forall i, \forall j)$$

Llegados a este punto no tiene mucho sentido el uso de este descriptor con una imagen, ya que produciría la misma salida que en el comparador basado en el mínimo, por lo que continuaremos con las consultas por dos y tres objetos. Para ello se buscarán los mismos objetos que en el comparador anterior, un mechero y un sacapuntas. Los resultados se pueden ver en la figura 4.24 y 4.25



Figura 4.24: Búsqueda de dos objetos usando el comparador basado en el máximo

En las figuras antes mencionadas se puede ver como en ambos casos obtenemos la salida correcta, en ellos se puede ver como se busca que estén todos los objetos en la imagen y además intentará que estén todos ellos con

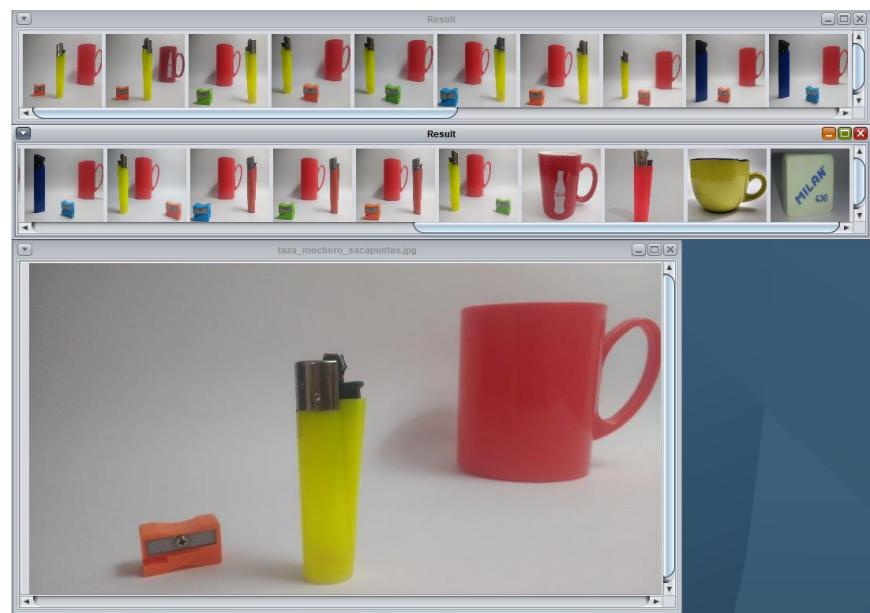


Figura 4.25: Búsqueda de tres objetos usando el comparador basado en el máximo

el mismo color. Por otro lado, si nos fijamos en las consultas con tres objetos obtendremos solo las imágenes que contienen los 3 objetos cambiando su posición. Al haber menos imágenes de este tipo, empezarán a aparecer los mismos objetos pero con diferente color y cuando ya no haya suficientes imágenes que cumplan el requisito de que sean los mismos objetos empezarán a aparecer las imágenes que haya en la base de datos por orden.

Como hemos podido ver, con la utilización de todos los comparadores mencionados, podremos satisfacer la mayoría de las necesidades de consulta que podamos tener.

## Capítulo 5

# Bases de datos utilizadas

Para poder mostrar el uso del descriptor implementado se han creado tres bases de datos bien diferenciadas; la primera, esta formada por imágenes descargadas a través de la herramienta *Google imágenes* [13], la segunda se ha creado realizando fotografías a objetos cotidianos que todo el mundo puede tener por casa y la tercera es un conjunto de imágenes de diversas categorías descargado de internet.

### 5.1. Primera base de datos

Para esta base de datos se han utilizado imágenes recuperadas a través de la herramienta *Google Imágenes* como se mencionó anteriormente. Sin embargo se planteaban varios requisitos:

- Las categorías de los objetos han de ser reconocidos por la red neuronal que sea utilizada.
- Ha de haber un conjunto de imágenes lo suficientemente grande como para que las consultas que se realicen al sistema demuestren claramente su correcto funcionamiento.
- El número de categorías de imágenes ha de ser lo suficientemente grande como para poder realizar diferentes consultas con diferentes categorías de objetos.
- Dentro de cada categoría de imágenes ha de haber la suficiente variedad de colores.
- En las imágenes, el color del objeto ha de ser lo suficientemente relevante. De no ser así, al tratarse el color de forma global se pueden producir errores.

Para cumplir dichos requisitos se han realizado búsquedas en la herramienta antes mencionada del tipo '*mug*' (Taza en inglés) y seleccionado en el apartado de 'herramientas' del buscador un color en concreto como por ejemplo '*Azul*' obteniendo como ejemplo el resultado de la figura 5.1.

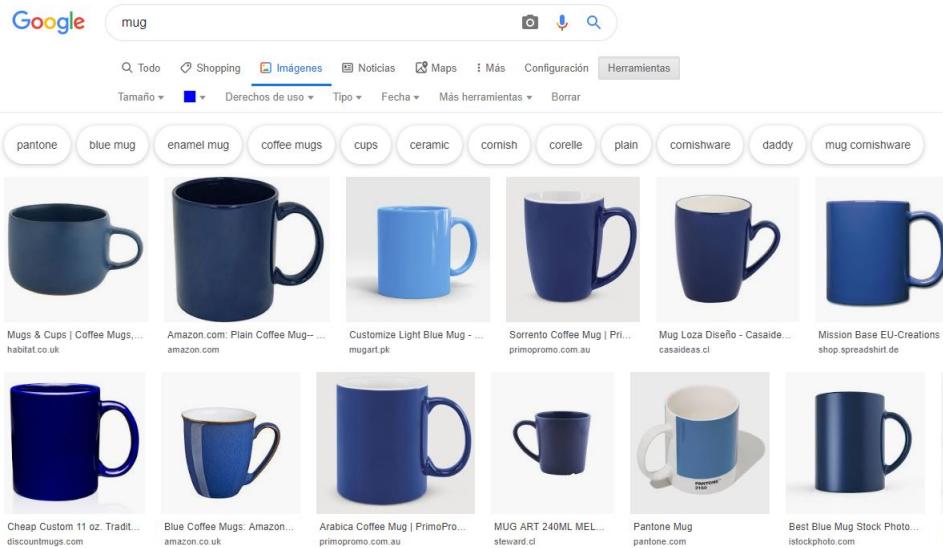


Figura 5.1: Ejemplo de búsqueda realizada para la creación de la base de datos

De esta manera se han obtenido una base de datos con 913 imágenes repartidas en 5 categorías: '*Bucket*', '*Pick*', '*Coffee mug*', '*Ballpoint*' y '*Bin-de*'. Algunos ejemplos de las categorías de imágenes obtenidas se pueden ver en las figuras 5.2 y 5.3.

Para las 5 categorías mencionadas anteriormente se han intentado encontrar al menos cuatro colores; '*Rojo*', '*Azul*', '*Verde*' y '*negro*' ya que estos son los colores más comunes, sin embargo, para las que ha sido posible se ha buscado la mayor variedad posible de colores (es más fácil encontrar una taza amarilla que un bolígrafo o un cubo amarillo).

## 5.2. Segunda base de datos

La segunda base de datos creada para el proyecto pretendía resolver otro objetivo diferente aunque parecido. Obtener varias imágenes en las cuales se puedan observar varias composiciones de objetos, es decir, imágenes con varios objetos en diferentes posiciones. Se añaden por tanto a los objetivos anteriores los siguientes:

- Las imágenes deben contener entre uno y tres objetos
- Ha de haber por cada categoría de objetos, al menos dos colores diferentes.
- Las posiciones o las perspectivas de los objetos han de variar entre imágenes.

La resolución de estos objetivos han sido más difícil y es que, en esta ocasión la herramienta *Google Imágenes* no nos es de utilidad. Esto se puede comprobar haciendo una búsqueda en dicha herramienta de, por ejemplo, 'Archivador negro' y 'Taza Roja'. Como resultado obtendremos imágenes que estarán relacionadas con uno u otro concepto, mezclando colores. Incluso si hacemos una búsqueda por imagen de una imagen compuesta por dos objetos no obtendremos ningún resultado. Esto puede verse en las imágenes 5.4 y 5.5.

Como se puede ver, este método no nos sirve por lo que tendremos que crear nuestras propias imágenes. Para ello se han utilizado varios objetos que pueden ser encontrados por casa. Se han utilizado por tanto:

- Dos tazas Rojas
- Tres Tacitas de diversos colores: Roja, Amarilla y Azul
- Una taza de cristal verde.
- Dos mecheros rojos, un mechero azul y otro amarillo
- Tres sacapuntas de diversos colores: Naranja, Azul y Verde
- Una goma de borrar verde y otra Rosa

Con la combinación de estos objetos se ha creado una base de datos de 447 fotografías que serán utilizadas más adelante. Una muestra de estas fotografías se puede ver a continuación.

Estas imágenes han sido tomadas de forma casera, con la cámara de un teléfono móvil y una cartulina blanca para intentar que el fondo sea blanco.

### 5.3. Tercera base de datos

Esta base de datos, al contrario que las dos anteriores, no son imágenes que haya recogido yo mismo una a una, si no que se trata de imágenes descargadas del repositorio de imágenes proporcionado por '*ImageNet*' [26] para la competición de localización de objetos en imágenes, la cual se desarrolla en la plataforma '*Kaggle*' [27].

Debido al gran tamaño que presenta la base de datos descargada (cerca de 250GB) se ha reducido el numero de categorías que esta presentaba reduciéndola de 1000 a 9 categorías de 1300 imágenes cada una. Un ejemplo de esta base de datos se puede ver en la figura 5.9.

Las tres bases de datos pueden ser encontradas de forma completa a través de la página del repositorio [14]. La primera base de datos se encuentra en la carpeta 'imágenes\_internet', la segunda en la carpeta 'fotografías' y la tercera en la carpeta 'imágenes\_coloridas'.



Figura 5.2: Imágenes pertenecientes a la categoría 'púa de guitarra'



Figura 5.3: Imágenes pertenecientes a la categoría 'taza'

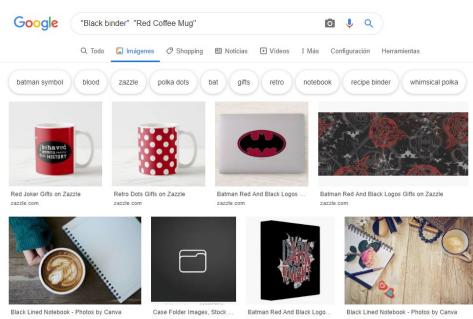


Figura 5.4: Búsqueda por términos lingüísticos

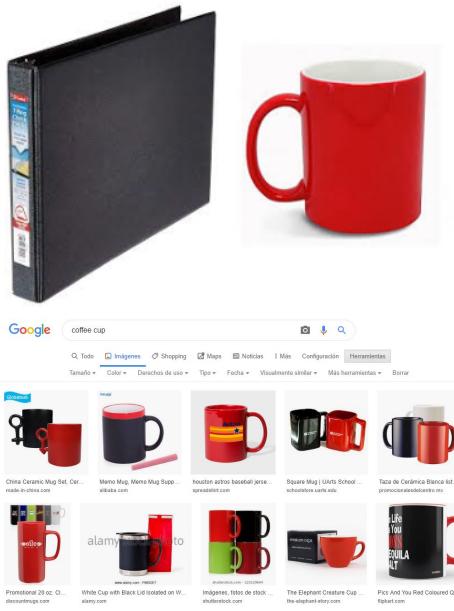


Figura 5.5: Imagen consulta y resultado en Google

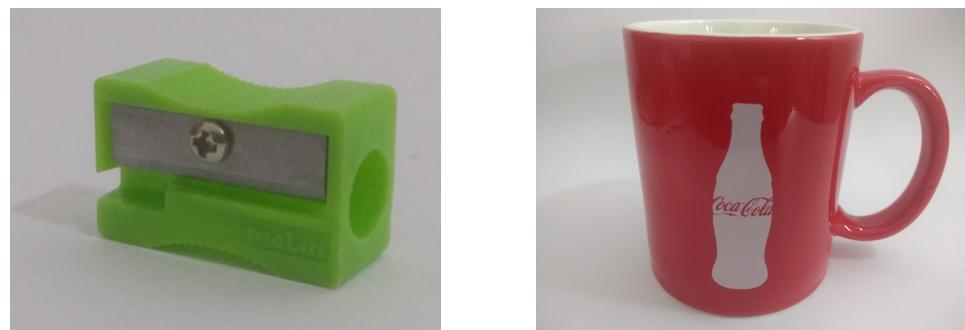


Figura 5.6: Fotografías con un objeto



Figura 5.7: Fotografías con dos objetos



Figura 5.8: Fotografías con tres objetos



Figura 5.9: Fotografía de ejemplo de la tercera base de datos utilizada



## **Parte II**

# **Implementación del Sistema**



# Capítulo 6

## Organización y presupuesto

En este capítulo, se hablará sobre la organización que se ha seguido en el proyecto y a partir de la cual se estructurará esta segunda parte. Además de esto se proveerá un diagrama de Gantt para poder visualizar donde se han concentrado los esfuerzos en este proyecto y cual ha sido la planificación que se ha seguido. Para finalizar se aportara un presupuesto estimado del coste que tendría realizar este proyecto.

Para empezar, hablaremos sobre la metodología de desarrollo que se ha seguido, ya que dependiendo de la metodología que se use la organización del proyecto puede cambiar ampliamente. En este caso se ha seguido la metodología de desarrollo en cascada, esta, consiste en separar el trabajo en fases teniendo que acabar una para empezar la siguiente. Se ha elegido esta metodología ya que se ajusta a las necesidades del proyecto, además de que nos presenta una buena práctica; Antes de implementar el sistema, analizar que es lo que este necesitará y como funcionará. De esta manera el trabajo estaría claro desde el primer momento. Las fases en las que se decidió separar este proyecto son las siguientes:

- Análisis de requisitos
- Diseño del Sistema
- Implementación

Mas adelante se hablará mas específicamente de cada una de estas etapas individualmente.

### 6.1. Diagrama de Gantt

Este diagrama ampliamente utilizado, pretende representar la organización a seguir a la hora de realizar una actividad. En este diagrama se

muestran las diferentes partes en las que se ha desarrollado dicha actividad. La realización de uno de estos diagramas nos permite tener claros los plazos a los que debemos ceñirnos. Dicho diagrama puede verse en la figura 6.1

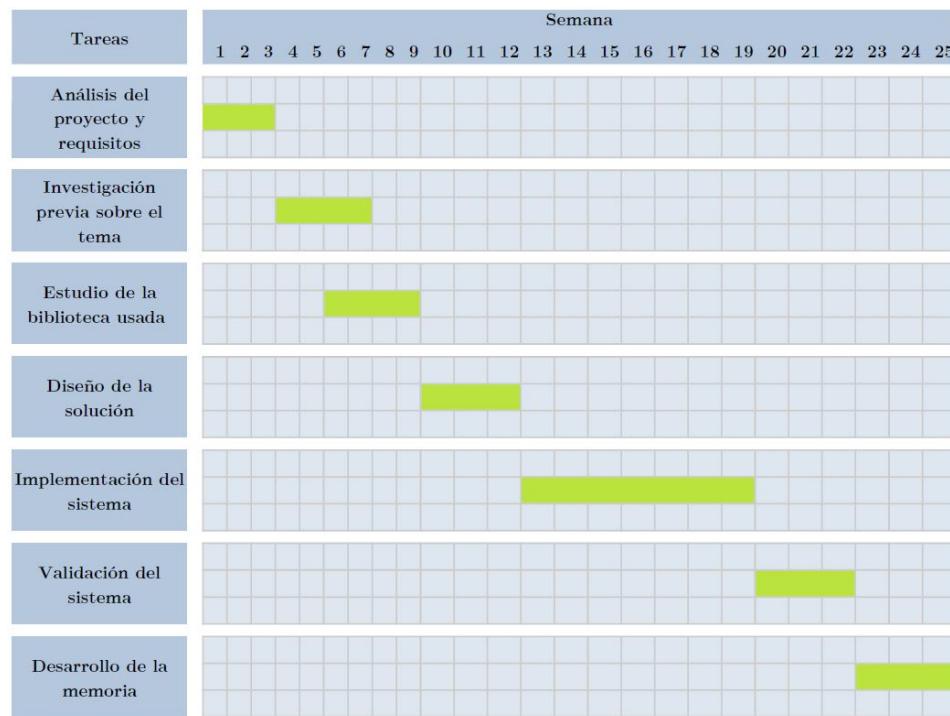


Figura 6.1: Diagrama de Gantt

Como se puede ver en el diagrama de la figura 6.1, se comenzó analizando cuales iban a ser los requisitos necesarios para satisfacer los objetivos propuestos. Una vez claros dichos requisitos se pasó a investigar el estado actual de los sistemas de recuperación de imagen y como funcionan. Una vez claro esto, se paso al estudio de la biblioteca elegida para realizar el proyecto, la *JMR*, este estudio consistió en el análisis de los descriptores y comparadores ya implementados en ella y en la forma en la que podría ser creado otro descriptor que satisficiese nuestras necesidades.

Llegados a este punto se continuamos con la etapa más larga del proceso, la implementación del sistema basándose en la arquitectura de la *JMR*. En este punto también se desarrollaron las bases de datos usadas. Más tarde se pasó a la validación del sistema y al desarrollo de esta memoria.

## 6.2. Presupuesto

En este apartado se detallará el coste aproximado que tendría la realización de este proyecto. Para ello se tendrán en cuenta aspectos como la mano de obra del empleado que realizaría dicho proyecto, el equipo informático utilizado y los diversos gastos que pudiesen surgir.

Por un lado, tenemos la mano de obra del empleado que realiza el sistema. Si estimamos que un informático capacitado cobra al mes unos 1200 € trabajando 240 horas mensuales, es decir, a jornada completa obtenemos 5€/hora. Por otra parte, se han trabajado 4 horas de media al día, incluyendo fines de semana durante 25 semanas obtenemos que se han trabajado 700 horas, lo cual al multiplicarlo por los 5€ /hora obtenemos un coste de mano de obra de 3500€.

Tampoco debemos olvidar el coste del equipo informático usado. Se trata de un ordenador de sobremesa de aproximadamente 1000€. Si añadimos la pantalla nos costaría otros 100€ más. Esto supone 1100€ de gasto. Hay que añadir que la base de datos se ha creado utilizando objetos cotidianos que todo el mundo puede tener por casa y las imágenes se han captado con la cámara de un teléfono móvil de gama baja, como el que puede tener cualquier persona. Por lo tanto no se derivan gastos adicionales en este sentido.

Por tanto se estima que el presupuesto para realizar el sistema es de unos 4600€



## Capítulo 7

# Análisis de Requisitos del sistema

En este nuevo capítulo se analizarán los requisitos que deberá cumplir, o no, nuestro sistema. Y las decisiones que se tomarán para solventarlos. Estos requisitos pueden ser divididos en tres grupos:

- Requisitos de datos: Estos requisitos describen las estructuras a partir de las cuales se construirá el sistema.
- Requisitos Funcionales: Este tipo de requisitos detallan las interacciones entre el usuario y el sistema.
- Requisitos no funcionales: Al contrario de los requisitos funcionales este tipo de requisitos detallan atributos del sistema.

### 7.1. Requisitos de datos

Como se describió anteriormente, los requisitos de datos son los requisitos que hacen referencia a las estructuras necesarias. Estos son esenciales pues serán la base de todo el sistema. Estos son los siguientes:

1. Una estructura que represente una imagen
2. Un descriptor que represente una etiqueta o conjunto de etiquetas lingüísticas
3. Un descriptor que represente una propiedad visual de una imagen
4. Un descriptor de imagen que asocie las etiquetas de una imagen con sus propiedades visuales.
5. Una estructura para almacenar una lista de descriptores

6. Una base de datos que sea capaz de almacenar un descriptor asociado a una imagen junto con la propia imagen o enlace a la imagen.

## **7.2. Requisitos funcionales**

En este apartado se describen los requisitos funcionales, estos son los requisitos referentes a la interacción entre el usuario y el sistema. Es en estos requisitos donde se especificará que puede hacer o no el usuario.

1. Abrir o cerrar una o varias imágenes.
2. Abrir y cerrar un clasificador, que será usado para etiquetar imágenes
3. Modificar el umbral asociado a un clasificador.
4. Crear, cerrar o abrir una base de datos.
5. Realizar búsquedas en la base de datos mediante una imagen consulta.
6. Realizar búsquedas en la base de datos mediante una o varias etiquetas.
7. Añadir imágenes a la base de datos.
8. Realizar búsquedas entre las imágenes abiertas en ese momento.
9. Mostrar varios resultados en una misma ventana.
10. Mostrar las etiquetas pertenecientes a una imagen.
11. Permitir hacer zoom en una imagen, tanto zoom positivo como negativo.
12. Permitir al usuario modificar el descriptor a partir del cual se hará la consulta.
13. Permitir al usuario modificar el comparador que desea usar al realizar una consulta.
14. Permitir al usuario realizar la búsqueda utilizando una cuadricula de tamaño variable.
15. Guardar en disco una base de datos que haya sido creada.

### 7.3. Requisitos no funcionales

Estos requisitos representan las restricciones que marcarán el correcto funcionamiento del sistema, por tanto, son de gran importancia.

1. En caso de hacer una consulta a la base de datos, esta tiene que estar formada por descriptores del mismo tipo.
2. Para comparar dos descriptores formados por varios descriptores estos han de ser los mismos y estar almacenados en el mismo orden.
3. Para realizar consultas por etiquetas es necesario que haya una base de datos con imágenes cargadas.
4. Una base de datos debe de contener descriptores siempre del mismo tipo.
5. Al proporcionar los resultados, estos han de estar ordenados de mayor a menor relevancia.
6. El sistema ha de proporcionar una respuesta en el menor tiempo posible.
7. Para obtener las etiquetas de una imagen, debe de haber un clasificador abierto

Además de estos requisitos no funcionales cabe destacar que el objetivo del sistema no está centrado en la disponibilidad, con lo que se prevé que en ciertos casos el sistema se sature. No obstante, este tipo de fallos no son comunes con lo que la disponibilidad sigue siendo buena.

Una vez definidos los requisitos podemos pasar a analizar que es lo que ya hay implementado y que es lo que necesitaremos implementar.

En lo referente a la representación de una imagen, disponemos de varias alternativas y es que en la mayoría de lenguajes de programación hay ya un objeto que cumple esta necesidad.

Con respecto a las necesidades de descriptores de etiquetas, de propiedades visuales y de la lista de descriptores, la *JMR* ya dispone de todas estas herramientas, por lo que tampoco habrá que implementarlo. Sin embargo, lo que sí tendremos que crear será el Descriptor que unifica las etiquetas y la lista de descriptores.

Con respecto a la base de datos, la *JMR* dispone de una clase para crear una base de datos con el enlace de una imagen y un descriptor por lo que

tampoco deberemos de preocuparnos por este punto.

Con respecto a los requisitos funcionales, todos hacen referencia a una interfaz la cual habrá que crear para que el usuario interactúe con el sistema. Para abrir y cerrar imágenes a parte del típico botón sobre la imagen que permita cerrar esa imagen habrá que crear una imagen para cerrar todas las imágenes en pantalla, por otra parte para abrir y cerrar imágenes habrá que añadir un único botón que permita hacer ambas cosas.

Con respecto al clasificador, será necesario añadir un botón para añadir un clasificador y otro para eliminarlo, de esta manera el usuario podrá usar esta característica o no. Por otra parte, habrá que añadir un desplegable o campo modificable que nos permita cambiar el umbral del clasificador a gusto del usuario.

Será necesario por otra parte añadir varios botones que nos permitan interactuar con la base de datos, estos botones nos permitirán crear, cerrar, añadir registros, abrir y guardar una base de datos. Además deberemos permitir que se realicen búsquedas en la base de datos mediante una imagen consulta. También será necesario añadir campos desplegables para realizar búsquedas en la base de datos por etiquetas lingüísticas. Por tanto, el usuario tendrá 3 posibles botones, uno para realizar una búsqueda con imagen consulta en la base de datos, otro para realizar búsquedas con imagen consulta de entre las imágenes abiertas y otro para realizar búsquedas por etiquetas en la base de datos.

Con las ventanas para mostrar los resultados y para visualizar imágenes se ha optado por utilizar clases ya realizadas, ya que llevaría mucho esfuerzo desarrollar unas propias y no es el objetivo de este trabajo.

Para mostrar las etiquetas pertenecientes a una imagen habrá dos opciones, que el usuario seleccione una imagen de entre las imágenes abiertas y presione un botón o que por otro lado, el usuario elija una imagen de entre las imágenes resultado.

También será necesario añadir algún tipo de funcionalidad que permita hacer *zoom in* y *zoom out* sobre una imagen.

Para modificar los parámetros de la búsqueda será necesario añadir botones y desplegables para seleccionar de entre las múltiples opciones que tenemos.

Con respecto a los requisitos no funcionales, estos requisitos se cumplirán desactivando botones (Para no permitir a un usuario etiquetar una imagen

sin un clasificador seleccionado, por ejemplo) o aportando información suficiente para que el usuario no cometa errores, por ejemplo añadiendo un campo de texto donde puede en base a que descriptor se ha creado la base de datos abierta.



# Capítulo 8

## Casos de uso

Una vez obtenidos los requisitos y teniendo claro que es lo que necesitará el sistema a desarrollar, podemos pasar al siguiente punto del diseño del sistema. Por lo tanto en este punto, se hablará de los casos de uso. Los casos de uso son una parte de la fase de la Ingeniería de Requisitos que nos permiten delimitar el sistema a estudiar, determinar el contexto de uso del sistema y describir el punto de vista de los usuarios que usarán el sistema.

Algunas de las ventajas que aporta este tipo de análisis son las siguientes:

- Puede ser usado como base para el proceso de diseño y su validación.
- Guía el diseño de la interfaz de usuario y facilita la construcción de prototipos.
- Punto de inicio de las ayudas en línea y el manual de usuario.

Estos diagramas se basan en analizar las relaciones que se producen entre los diversos usuarios y el sistema. Especificando las opciones que el usuario podrá realizar y como deberá realizar dichas acciones. En primer lugar, se aporta un diagrama de casos de uso de UML, en el se detallaran cuales son las relaciones entre los usuario y el sistema. Cabe destacar que en este proyecto solo se ha considerado un actor, siendo este el usuario que utilizará el sistema, comunicándose este directamente con el sistema. En la figura 8.1 podemos ver el diagrama asociado a este actor.

### 8.1. Diagrama de casos de uso

A continuación y una vez definido el actor que interactuará con nuestro sistema pasamos a definir los distintos casos de uso que componen el sistema desarrollado. Este diagrama se puede ver en la figura 8.2

Una vez definidos los casos de uso pasamos a describirlos en la siguiente sección.

Actor	Usuario	AC-1
Descripción	Usuario que utilizará nuestro sistema, haciendo uso de todas las características que este aporta	
Características	Puede ser cualquier persona que quiera utilizar el sistema ya que no se necesita ningún tipo de requisito	
Relaciones		
Referencias	CU-1, CU-2, CU-3, CU-4, CU-5, CU-6, CU-7, CU-8, CU-9, CU-10, CU-11, CU-12, CU-13, CU-14, CU-15, CU-16, CU-17, CU-18	

Figura 8.1: Descripción del Actor

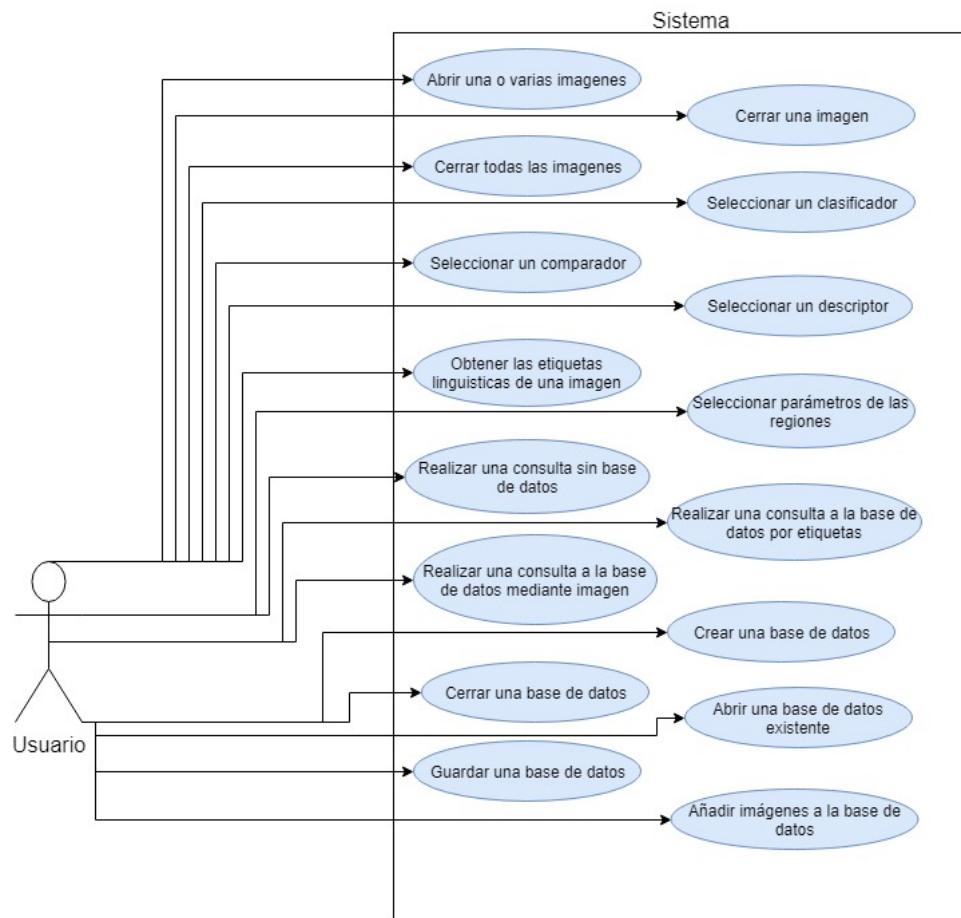


Figura 8.2: Diagrama de casos de uso

## 8.2. Descripción de los casos de uso

Para realizar las descripciones que se aportan mas abajo se ha usado la plantilla aportada por el departamento de Lenguajes y Sistemas Informáticos (*LSI*) de la UGR en la asignatura de 'Fundamentos de Ingeniería del Software'. Estas descripciones pretenden detallar los aspectos relevantes de cada uno de los casos de uso descritos anteriormente. En las siguientes paginas se aportan los cuadros que representan cada uno a los Casos de uso.

<b>Caso de Uso</b>	Abrir una o varias imágenes	CU-1
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RF-1, RD-1	
<b>Precondición</b>	No se presenta ninguna precondición ya que esta será una de las primeras acciones que realice el usuario	
<b>Postcondición</b>	Las imágenes que se hayan seleccionado deberán aparecer en una ventana en pantalla. Siendo cada una de estas pantallas independiente la una de la otra.	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

<b>Propósito</b>
Mediante este caso de uso el usuario proveerá al sistema de imágenes.

<b>Resumen</b>
El usuario, manifiesta su intención de abrir una o más imágenes, el sistema le proporciona una ventana para hacerlo y una vez seleccionadas, estas son mostradas en la ventana principal del sistema.

<b>Curso Normal</b>
1 El usuario, pulsa sobre el botón asociado a la selección de imágenes
2 El sistema abre una ventana de selección de archivos.
3 El usuario seleccionará la imagen o imágenes que quiera utilizar
4 En el caso de que las imágenes tengan un formato correcto, el sistema las abrirá en una ventana creada para tal efecto.

<b>Cursos Alternos</b>
3a En el caso de que el archivo seleccionado no sea correcto el sistema no realizará ninguna acción.

<b>Otros datos</b>			
<b>Frecuencia esperada</b>	Varias veces por ejecución	<b>Rendimiento</b>	
<b>Importancia</b>	Alta	<b>Urgencia</b>	Alta.
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta.

<b>Comentarios</b>

<b>Caso de Uso</b>	Cerrar una imagen	CU-2
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RF-1, RD-1, CU-3	
<b>Precondición</b>	Para poder cerrar una imagen, como es lógico deberá haber una imagen abierta en una ventana dentro del sistema.	
<b>Postcondición</b>	Al cerrar una imagen, la ventana que la contiene debe de cerrarse no apareciendo más. Por otro lado el sistema debe de descargar dicha imagen de la memoria	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**

En este caso el sistema permitira cerrar una imagen cuando esta ya no sea de interes para el caso que se este tratando

**Resumen**

El usuario, una vez ha abierto una o varias imagenes decide cerrar alguna de ellas, para ello pulsa el boton de cierre de ventana de la ventana donde se encuentra contenida la imagen.

**Curso Normal**

- 1 El usuario pulsa sobre el botón de cierre de la ventana que contiene la imagen
- 2 En ese momento, dicha ventana se cerrara y la imagen dejara de aparecer en pantalla.
- 3 El sistema descargara de memoria dicha imagen.

**Cursos Alternos**


**Otros datos**

<b>Frecuencia esperada</b>	Poca	<b>Rendimiento</b>	
<b>Importancia</b>	Alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta

**Comentarios**


<b>Caso de Uso</b>	Cerrar todas las imágenes	CU-3		
<b>Actores</b>	AC-1			
<b>Tipo</b>	Primario, Real			
<b>Referencias</b>	RF-1, RD-1, CU-2			
<b>Precondición</b>	Para poder cerrar varias imágenes debe de haber al menos una imagen abierta en una ventana dentro del sistema.			
<b>Postcondición</b>	Al cerrar todas las imágenes, todas las ventanas que aparecen en la pantalla principal se cerrarán incluyendo resultados e imágenes. Todas estas imágenes serán descargadas de la memoria.			
<b>Autor</b>		<b>Fecha</b>		<b>Versión</b>

**Propósito**

Permite limpiar la ventana principal del programa eliminando todas las ventanas abiertas en el sistema.

**Resumen**

El usuario, una vez ha abierto varias imágenes pulsará sobre el botón de cierre de ventanas, cerrándose todas las ventanas a la vez. Esto incluye ventanas de presentación de resultados.

**Curso Normal**

- |   |                                                                                                               |
|---|---------------------------------------------------------------------------------------------------------------|
| 1 | El usuario pulsa sobre el botón de cierre de todas las ventanas abiertas en la pantalla principal del sistema |
| 2 | En ese momento, todas las ventanas se cerrarán y no se tomarán en cuenta para futuros resultados o consultas  |
| 3 | El sistema descargará de memoria dichas imágenes.                                                             |

**Cursos Alternos**


**Otros datos**

<b>Frecuencia esperada</b>	Alta	<b>Rendimiento</b>	
<b>Importancia</b>	Alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta

**Comentarios**


<b>Caso de Uso</b>	Seleccionar un clasificador	CU-4
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RD-3, RF-2, RF-3, RF-6, RF-10	
<b>Precondición</b>	No se requiere ninguna precondición	
<b>Postcondición</b>	Una vez el usuario haya seleccionado un clasificador, todas las consultas que utilicen etiquetas utilizarán este clasificador.	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**

Seleccionar un clasificador para el etiquetado de una imagen

**Resumen**

El usuario al pulsar la tecla de selección de clasificador, seleccionará de su equipo un clasificador valido que será usado para etiquetar las imágenes.

**Curso Normal**

- 1 El usuario pulsa sobre el botón de selección de clasificador.
- 2 El sistema mostrará una ventana de selección de archivos para seleccionar un clasificador
- 3 El usuario selecciona un clasificador valido
- 4 El sistema, una vez abierto el clasificador, activará todos los botones que no estan disponibles a no ser que se seleccione un clasificador.

**Cursos Alternos**

- |    |                                                                                                         |
|----|---------------------------------------------------------------------------------------------------------|
| 3a | El usuario selecciona un archivo que no tiene la extension correcta o que no es un clasificador valido. |
| 4a | El sistema muestra un mensaje de error informando de este suceso.                                       |
|    |                                                                                                         |

**Otros datos**

<b>Frecuencia esperada</b>	Poco frecuente	<b>Rendimiento</b>	
<b>Importancia</b>	Alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta

**Comentarios**

--

<b>Caso de Uso</b>	Seleccionar un comparador	CU-5
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RF-13	
<b>Precondición</b>	No se presenta ninguna precondición	
<b>Postcondición</b>	El comparador que haya sido seleccionado debe de tenerse en cuenta para realizar las posteriores comparaciones.	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**

El usuario selecciona que tipo de comparador quiere utilizar para el descriptor principal del sistema.

**Resumen**

El usuario, como paso previo a la realización de una consulta selecciona el tipo de comparador que quiere utilizar para el descriptor principal del sistema

**Curso Normal**

- 1 El usuario pulsa sobre la pestaña de selección de comparador
- 2 Se desplegará una lista con los comparadores disponibles para el usuario.
- 3 El usuario selecciona una de las posibles opciones
- 4 Cuando se realice una consulta posterior se utilizará el comparador que se haya seleccionado

**Cursos Alternos**


**Otros datos**

<b>Frecuencia esperada</b>	Alta	<b>Rendimiento</b>	
<b>Importancia</b>	Alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta

**Comentarios**

Si el comparador acepta varios argumentos y opciones de configuración se habilitaran nuevas opciones para la configuración de ese comparador en particular.

<b>Caso de Uso</b>	Seleccionar un descriptor	CU-6
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RD-3, RD-4, RF-5, RF-7, RF-12	
<b>Precondición</b>	No se presenta ninguna precondicion para realizar esta acción	
<b>Postcondición</b>	El descriptor que haya sido seleccionado debe ser el usado para realizar las posteriores consultas	
<b>Autor</b>		Fecha      Versión

**Propósito**

El usuario, para realizar diferentes consultas, selecciona uno u otro descriptor

**Resumen**

Cuando el usuario se disponga a realizar una consulta, deberá seleccionar en base a que quiere realizar esa consulta. Para ello seleccionará que tipo de propiedad quiere utilizar para realizar la búsqueda en el sistema.

**Curso Normal**

- 1 El usuario pulsa sobre la pestaña de selección de descriptor
- 2 Se desplegará una lista con los descriptores disponibles para el usuario.
- 3 El usuario selecciona una de las posibles opciones
- 4 Cuando se realice una consulta posterior se utilizará el descriptor que se haya seleccionado

**Cursos Altemos**

- |    |                                                                                     |
|----|-------------------------------------------------------------------------------------|
| 4a | El sistema informa de que el descriptor seleccionado no admite consultas por imagen |
|    |                                                                                     |
|    |                                                                                     |

**Otros datos**

<b>Frecuencia esperada</b>	Media	<b>Rendimiento</b>	
<b>Importancia</b>	Alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completado	<b>Estabilidad</b>	Alta

**Comentarios**

En el caso de la selección de alguno de los descriptores, la consulta por imagen no estara permitida. En ese caso, si el usuario intenta realizar alguna consulta el sistema le avisara mediante un mensaje de que no puede realizar esa acción.

<b>Caso de Uso</b>	Obtener las etiquetas lingüísticas de una imagen	CU-7
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RD-2, RF-10, RNF-7, RD-1	
<b>Precondición</b>	Para poder obtener las etiquetas de una imagen debe de haber un clasificador abierto y una imagen seleccionada	
<b>Postcondición</b>	Las etiquetas deben ser mostradas al usuario mediante la salida de texto	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**

El usuario, habiendo seleccionado una imagen consulta las etiquetas lingüísticas de esa imagen

**Resumen**

El usuario, una vez ha seleccionado una imagen, consulta sus etiquetas lingüísticas para ello deberá de seleccionar previamente un clasificador.

**Curso Normal**

- 1 El usuario selecciona una imagen.
- 2 El usuario selecciona un clasificador si no lo ha hecho ya
- 3 El usuario presiona el botón para el etiquetado de esa imagen
- 4 El sistema proporciona una lista de etiquetas lingüísticas asociadas a esa imagen

**Cursos Alternos**

- |           |                                                                                                                                       |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------|
| <b>4a</b> | El usuario antes de realizar la consulta ha seleccionado el descriptor que hace referencia a los colores como etiquetas.              |
| <b>4b</b> | Al realizar la consulta, el sistema devuelve, además de las etiquetas de las categorías de los objetos, las etiquetas de los colores. |

**Otros datos**

<b>Frecuencia esperada</b>	Media	<b>Rendimiento</b>	
<b>Importancia</b>	Media	<b>Urgencia</b>	Baja
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta

**Comentarios**

La obtención de las etiquetas de la imagen no estará disponible a no ser que se seleccione un clasificador.

<b>Caso de Uso</b>	Ajustar parametros de las regiones	CU-8
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, real	
<b>Referencias</b>	RF-14	
<b>Precondición</b>	No será necesaria ninguna condición previa	
<b>Postcondición</b>	Las consultas basadas en regiones que se realicen posteriormente se ceñiran a los parametros especificados	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**

El usuario podra modificar los parametros de las regiones que serán usados para realizar consultas posteriormente

**Resumen**

El usuario podra seleccionar diversos tipos de opciones para hacer busquedas basadas en las regiones de una imagen, estas opciones serán: Elegir el numero de secciones que se realizarán, elegir el comparador que será usado para comparar imagenes, elegir el descriptor que se usará para calcular las propiedades de dichas regiones.

**Curso Normal**

- 1 El usuario pulsa el botón proporcionado para abrir la configuración de las regiones.
- 2 Se mostrará en pantalla una nueva ventana en la que el usuario podrá definir varios tipos de propiedades
- 3 Una vez haya terminado de seleccionar las propiedades pulsará el botón 'Ok'
- 4 Cuando se realice una consulta posterior se utilizarán las propiedades que se hayan seleccionado

**Cursos Alternos**


**Otros datos**

<b>Frecuencia esperada</b>	Baja	<b>Rendimiento</b>	
<b>Importancia</b>	Media	<b>Urgencia</b>	Media
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta

**Comentarios**

El selector de descriptor del grid solo se tendrá en cuenta si el usuario ha marcado el descriptor 'Grid Descriptor' en el selector de descriptor.

<b>Caso de Uso</b>	Realizar una consulta sin una base de datos	CU-9
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RD-1, RD-4, RF-8, RF-9	
<b>Precondición</b>	El usuario ha de haber seleccionado una imagen.	
<b>Postcondición</b>	Los resultados de la búsqueda se mostrarán de mayor a menor relevancia en una nueva ventana.	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**

El usuario realiza una búsqueda en el sistema mediante una imagen consulta

**Resumen**

En este caso el usuario realizará una búsqueda utilizando como parámetros los descriptores, comparadores y clasificadores que haya seleccionado previamente. Se buscará la imagen que se encuentre seleccionada de entre todas las imágenes que se encuentren abiertas en ese momento.

**Curso Normal**

- 1 El usuario abre una o más imágenes
- 2 El usuario selecciona la imagen que se quiera utilizar como consulta
- 3 El usuario selecciona los parámetros que se vayan a utilizar para realizar la consulta.
- 4 El usuario presiona el botón de búsqueda.
- 5 El sistema proporciona una ventana con los resultados más relevantes.

**Cursos Alternos**


**Otros datos**

<b>Frecuencia esperada</b>	Alta	<b>Rendimiento</b>	
<b>Importancia</b>	Muy Alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completo	<b>Estabilidad</b>	Alta

**Comentarios**

En el caso de que se seleccione el descriptor basado en colores como etiquetas lingüísticas el usuario no podrá realizar esta opción, se mostrará un mensaje advirtiendo de esto.

Caso de Uso	Realizar una consulta a la base de datos por etiquetas	CU-10
Actores	AC-1	
Tipo	Primario, Real	
Referencias	RD-2, RD-6, RF-7, RF-9, RF-10, RNF-3, RNF-4, RNF-5, RNF-6, RNF-7	
Precondición	Para realizar una consulta de este tipo debe de haber una base de datos y un clasificador abierto.	
Postcondición	Los resultados serán proporcionados en una ventana ordenados en función a su relevancia	
Autor		Fecha   Versión

**Propósito**

El usuario busca imágenes que contengan una o varias etiquetas de las especificadas

**Resumen**

El usuario habiendo seleccionado una base de datos realizará una búsqueda en ella de las imágenes que contengan las mismas etiquetas que las especificadas en la consulta, estas pueden ser etiquetas referentes a la categoría de los objetos o etiquetas referentes a los colores dominantes de la imagen.

**Curso Normal**

- 1 El usuario abre una base de datos en la cual se haya calculado previamente las etiquetas de los objetos
- 2 El usuario selecciona una etiqueta de la lista desplegable añadida para tal efecto.
- 3 En caso de que el usuario lo desee, se selecciona uno de los colores de la lista desplegable añadida para tal efecto.
- 4 El usuario pulsa el botón de búsqueda
- 5 El sistema devuelve una lista de imágenes que contengan las etiquetas por las que se ha consultado.

**Cursos Alternos**

- 1 Si el usuario quiere realizar una consulta basándose en las etiquetas de objetos y en los colores, seleccionará el descriptor 'Labeled Colors'
- 2 Si además de esto quiere utilizar la opción de los conjuntos difusos para una mejor ordenación, marcará la opción 'Fuzzy Colors'

**Otros datos**

Frecuencia esperada	Media	Rendimiento	
Importancia	Alta	Urgencia	Alta
Estado	Completado	Estabilidad	Alta

**Comentarios**

<b>Caso de Uso</b>	Realizar una consulta a la base de datos mediante imagen	CU-11
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RD-1, RD-4, RF-4, RF-5, RF-9, RF-13, RNF-1, RNF-5	
<b>Precondición</b>	Debe de haber una base de datos y una imagen abierta antes de realizar una consulta	
<b>Postcondición</b>	Se mostrarán los resultados en una ventana independiente y ordenados de mayor a menor relevancia	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**  
El usuario se propone buscar una imagen en la base de datos.

**Resumen**  
En este caso, el usuario realiza una consulta a la base de datos mediante una imagen. Para ello deberá de haber seleccionado un comparador y haber seleccionado el mismo descriptor usado para realizar la base de datos.

- Curso Normal**
- |   |                                                                                           |
|---|-------------------------------------------------------------------------------------------|
| 1 | El usuario abre una base de datos                                                         |
| 2 | El usuario selecciona las mismas opciones que se usaron para crear la base de datos.      |
| 3 | El usuario abre la imagen que quiera buscar.                                              |
| 4 | El usuario pulsa el botón de búsqueda en la base de datos                                 |
| 5 | El sistema devuelve las imágenes más relevantes de forma ordenada en una ventana a parte. |

**Cursos Alternos**


**Otros datos**

<b>Frecuencia esperada</b>	Alta	<b>Rendimiento</b>	
<b>Importancia</b>	Muy alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completado	<b>Estabilidad</b>	Alta

**Comentarios**

--

Caso de Uso	Crear una base de datos	CU-12
Actores	AC-1	
Tipo	Secundario, Real	
Referencias	RD-6, RF-4	
Precondición	No debe de haber ninguna base de datos abierta en el sistema.	
Postcondición	Se creará una base de datos vacía con la estructura que haya sido seleccionada.	
Autor		Fecha      Versión

**Propósito**

El usuario crea una base de datos siguiendo la estructura que haya especificado

**Resumen**

El usuario crea una base de datos siguiendo la estructura que se haya especificado. Posteriormente el usuario podrá realizar diversas operaciones con ella.

**Curso Normal**

- 1 El usuario selecciona el descriptor con el que quiera que se cree la base de datos
- 2 El usuario pulsará el botón destinado a crear la base de datos
- 3 El sistema creará una base de datos en función a lo seleccionado.
- 4 El sistema mostrara el estado de la base de datos en la parte inferior derecha de la ventana del programa

**Cursos Altermos**


**Otros datos**

Frecuencia esperada	Baja	Rendimiento	
Importancia	Alta	Urgencia	Alta
Estado	Completado	Estabilidad	Alta

**Comentarios**


<b>Caso de Uso</b>	Cerrar una base de datos	CU-13		
<b>Actores</b>	AC-1			
<b>Tipo</b>	Secundario, Real			
<b>Referencias</b>	RD-6, RF-4			
<b>Precondición</b>	Debe de haber una base de datos abierta			
<b>Postcondición</b>	No habrá ninguna base de datos abierta en el sistema			
<b>Autor</b>		<b>Fecha</b>		<b>Versión</b>

**Propósito**

El usuario se propone cerrar una base de datos abierta.

**Resumen**

El usuario cerrara una base de datos que se encuentre abierta en el sistema, permitiéndole crear o abrir otra base de datos.

**Curso Normal**

- |   |                                                                     |
|---|---------------------------------------------------------------------|
| 1 | El usuario pulsa el botón de cierre de la base de datos.            |
| 2 | El sistema cierra la base de datos que esté abierta en ese momento. |

**Cursos Alternos**


**Otros datos**

<b>Frecuencia esperada</b>	Media	<b>Rendimiento</b>	
<b>Importancia</b>	Alta	<b>Urgencia</b>	Media
<b>Estado</b>	Completado	<b>Estabilidad</b>	Alta

**Comentarios**


<b>Caso de Uso</b>	Abrir una base de datos existente	CU-14
<b>Actores</b>	AC-1	
<b>Tipo</b>	Primario, Real	
<b>Referencias</b>	RD-6, RF-4	
<b>Precondición</b>	No debe haber ninguna base de datos abierta en ese momento	
<b>Postcondición</b>	Habrá en el sistema una base de datos abierta	
<b>Autor</b>		<b>Fecha</b>
		<b>Versión</b>

**Propósito**

El usuario abre una base de datos.

**Resumen**

El usuario mediante este caso de uso abrirá una base de datos la cual sido creada anteriormente. La base de datos podra contener o no imagenes.

**Curso Normal**

- 1 El usuario pulsa el boton de apertura de base de datos.
- 2 El sistema muestra una ventana de selección de archivo.
- 3 El usuario selecciona la base de datos que quiera abrir.
- 4 El sistema carga la base de datos que haya sido seleccionada

**Cursos Altermos**

- |    |                                                                                                                       |
|----|-----------------------------------------------------------------------------------------------------------------------|
| 3a | En el caso de que el archivo seleccionado no sea valido, se mostrará por la salida de texto un mensaje notificandolo. |
|    |                                                                                                                       |
|    |                                                                                                                       |

**Otros datos**

<b>Frecuencia esperada</b>	Media	<b>Rendimiento</b>	
<b>Importancia</b>	Muy alta	<b>Urgencia</b>	Alta
<b>Estado</b>	Completado	<b>Estabilidad</b>	Alta

**Comentarios**


Caso de Uso	Guardar una base de datos	CU-15
Actores	AC-1	
Tipo	Secundario, Real	
Referencias	RD-6, RF-15	
Precondición	Debe de haber una base de datos abierta	
Postcondición	La base de datos sera almacenada en disco	
Autor		Fecha
		Versión

**Propósito**

El usuario guarda una base de datos que haya sido creada

**Resumen**

El usuario, despues de crear una base de datos y haber añadido imágenes, puede que quiera guardar esta base de datos en el disco de forma que pueda ser abierta en el futuro

**Curso Normal**

- 1 El usuario, teniendo una base de datos abierta, pulsa el boton de guardar base de datos
- 2 El sistema, abre una ventana de selección de archivo en la que el usuario podra elegir la localización y el nombre del archivo en el que se almacenará la base de datos
- 3 El usuario elige el nombre y la ubicación donde guardar la base de datos
- 4 El sistema guarda en la ubicación especificada un archivo que representa la base de datos

**Cursos Alternos**


**Otros datos**

Frecuencia esperada	Media	Rendimiento	
Importancia	Alta	Urgencia	Alta
Estado	Completado	Estabilidad	Alta

**Comentarios**


Caso de Uso	Añadir imágenes a la base de datos	CU-16
Actores	AC-1	
Tipo	Primario, Real	
Referencias	RD-6, RF-7	
Precondición	Debe de haber una base de datos abierta en el sistema y al menos una imagen	
Postcondición	Las imágenes abiertas quedan añadidas a la base de datos	
Autor		Fecha
		Versión

Propósito
El usuario añade imágenes a la base de datos.

Resumen
El usuario, teniendo una base de datos abierta y varias imágenes, decide añadir estas imágenes a la base de datos basándose en los descriptores que tenga seleccionados

Curso Normal
1 El usuario abre o crea una base de datos
2 El usuario abre una o varias imágenes
3 El usuario selecciona las características que quiera almacenar en la base de datos
4 El usuario pulsa un botón específico para añadir estas imágenes
5 El sistema calcula las propiedades de todas las imágenes que se encuentren abiertas y las almacena en la base de datos

Cursos Altermos

Otros datos			
Frecuencia esperada	Alta	Rendimiento	
Importancia	Muy alta	Urgencia	Alta
Estado	Completado	Estabilidad	Alta

Comentarios
Todas las imágenes que se añadan a una misma base de datos han de estar calculadas con el mismo tipo de descriptor, en caso contrario cuando se realice una consulta surgira un error.



# Capítulo 9

## Diseño

En este capítulo hablaremos de todos los aspectos relacionados con el diseño del proyecto, presentando diagramas de paquetes y de clases. Además de esto hablaremos de la estructura de la biblioteca sobre la que se ha realizado, la *JMR*.

### 9.1. Bibliotecas usadas

Como se mencionó anteriormente, la biblioteca más usada en este proyecto y sobre la que se ha formado una propuesta para ella, ha sido la biblioteca *JMR*. Se ha elegido utilizar esta biblioteca como base, por ser una biblioteca enfocada a la recuperación de objetos multimedia, por ser de código abierto y por estar diseñada enteramente en Java [16].

Esta librería está formada por diferentes paquetes, cada uno referente a un propósito o ámbito concreto. De entre todos estos paquetes se han usado los referentes a las propiedades visuales de los objetos. Un diagrama de paquetes UML se puede ver en la figura 9.1

De entre todos los paquetes que se pueden observar en la figura 9.1 se han utilizado solo algunos de ellos, estos son los siguientes:

- **JMR.db:** Este paquete ha sido el utilizado para la realización de la base de datos y es que este paquete implementa una clase (*ListDB.java*) la cual representa una base de datos. Dentro de la clase antes mencionada se encuentran todos los métodos necesarios para crear, añadir elementos y hacer consultas a la base de datos.
- **JMR.initial:** Dentro de este paquete, se encuentra la clase que nos permite que dada una imagen nos devuelva una lista con sus colores dominantes. Estos colores dominantes serán los que etiquetaremos.
- **JMR.descriptor:** Este ha sido el paquete más utilizado, pues ofrece la mayoría de las clases referentes a las propiedades visuales de un objeto.

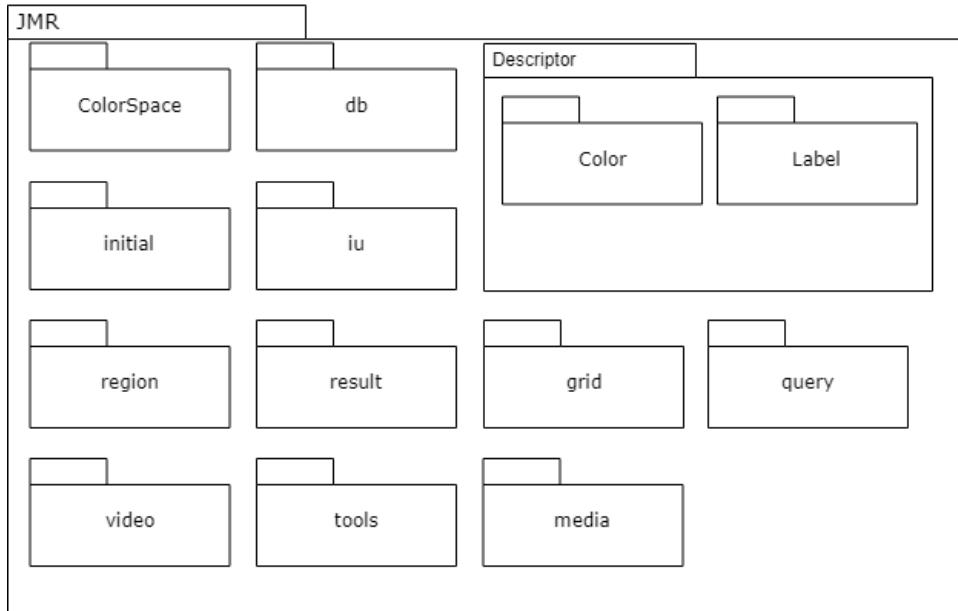


Figura 9.1: Diagrama UML de paquetes de la librería *JMR*

Además de esto podemos encontrar dos paquetes en su interior, estos son los siguientes:

- JMR.descriptor.color: Este paquete incluye las clases que representan las propiedades visuales de los objetos basados en su color, como por ejemplo el histograma de color.
- JMR.descriptor.label: Este paquete implementa la clase que representa una o varias etiquetas lingüísticas.

Dentro de todos estos paquetes cabe destacar el papel que desempeña el paquete *jmr.descriptor* y es que en él, están las bases sobre las cuales se construirá el descriptor desarrollado, estas son las clases *MediaDescriptor*, *MediaDescriptorAdapter* y *Comparator*. La primera de ellas representa la idea de descriptor y es que todas los descriptores desarrollados en la librería han de heredar de esta clase. Esta clase está formada por un medio sobre el que se calculará el descriptor, un método para calcularlo y un método para compararlo.

La clase *MediaDescriptorAdapter* presenta una interfaz heredada de la clase anterior, en ella se requiere un constructor en el cual se aporte un medio sobre el que calcular el descriptor y un comparador. El comparador, por otra parte, será la clase en la que se definirá como se comparan dos objetos del tipo *MediaDescriptor*.

Además de la *JMR* también se ha usado la librería *Java Fuzzy Imaging* o

*JFI* la cual aporta herramientas para el uso de conjuntos de colores difusos. Además de esto, esta biblioteca al igual que la *JMR* esta desarrollada en Java y es de código abierto. El diagrama de paquetes de la biblioteca se puede ver en la figura 9.2. Estas dos librerías usadas, además de todo esto tienen la ventaja de que implementan la interfaz *Serializable*[17], lo que nos permitirá guardar en disco bases de datos de este descriptor.

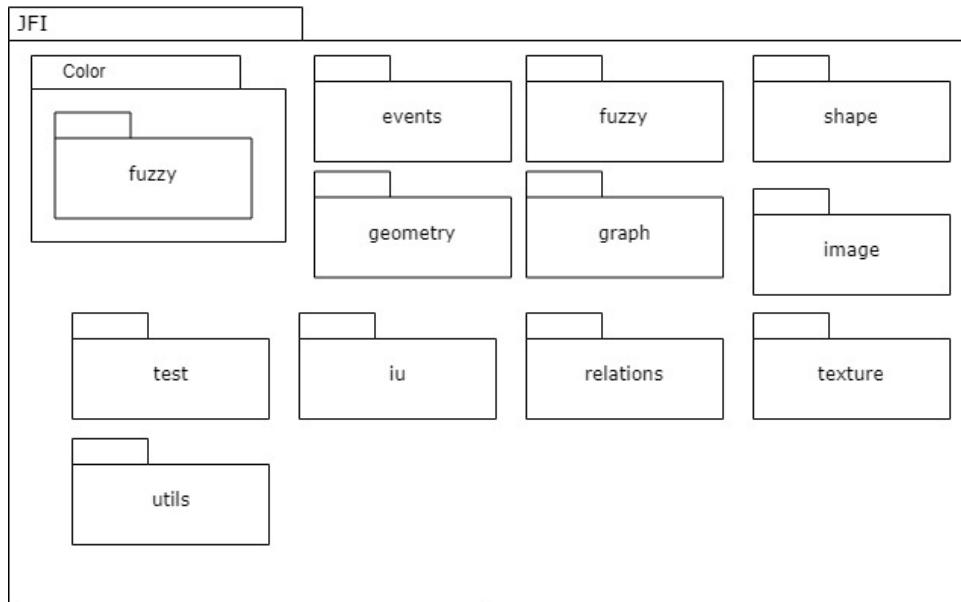


Figura 9.2: Diagrama UML de paquetes de la librería *JFI*

En este caso, el paquete utilizado ha sido el paquete *Color* el cual implementa tanto un mapa de colores como una clase que representa un conjunto difuso de colores construido con el mapa de colores previo. Utilizando ambas herramientas podemos obtener una etiqueta lingüística que represente un color dado.

## 9.2. Aporte a la *JMR*

Vistos los paquetes que se han utilizado se pasa a describir la propuesta que se ha desarrollado para la *JMR*, esta propuesta pretende ser una solución al problema de la recuperación de imágenes aprovechando todo el potencial que nos brinda la librería antes mencionada. Se ha desarrollado un único paquete, el cual contiene varios comparadores y dos descriptores.

### Descriptores

Para esta solución se ha desarrollado dos descriptores, siendo uno de ellos un caso especial del otro. El descriptor principal desarrollado esta represen-

tado por la clase *LabelProperties*, la cual esta creada a partir de la interfaz *MediaDescriptorAdapter*. Esta clase como se comentó anteriormente, esta formada por una etiqueta, representada por el descriptor *LabelDescriptor* y por una lista de etiquetas, representada por el descriptor *DescriptorList*. Este descriptor nos permite que dado una imagen del tipo *BufferedImage* nos devuelva una etiqueta y una lista de propiedades visuales.

El segundo descriptor desarrollado, es el descriptor *LabeledColorDescriptor* hereda del descriptor *LabelProperties* y es que este consiste en un caso especial en el cual las propiedades visuales del objeto son etiquetas lingüísticas representando los colores dominantes de la imagen. Para realizar este proceso de etiquetado se ha usado la biblioteca *JFI*, descrita anteriormente.

### Comparadores

Además de estos dos descriptores, se han creado diversos comparadores, seis para la clase *LabelProperties*, los cuales combinan los descriptores de las clases *LabelDescriptor* y *DescriptorList*. Para esta última clase, se ha creado otro comparador, que será usado por todos los comparadores del descriptor desarrollado. Este comparador tiene en cuenta la posibilidad de que queramos tener más en cuenta un descriptor que otro y asignándoles pesos diferentes.

Por otra parte, se han creado dos comparadores para la clase *GriddedDescriptor*. El primero de ellos es el comparador *MinimumGridComparator*, este comparador trata de representar el caso en el que dada una imagen queremos buscar al menos uno de los elementos que aparezcan en ella. Para ello, el tamaño de las regiones en las cuales se dividirá la imagen deberá ser el correcto. De lo contrario no se asignarán bien las etiquetas y las propiedades visuales.

El segundo comparador de la clase *GriddedDescriptor* es el comparador *MaximumGridComparator*, el cual representa el caso en el que queremos realizar una consulta buscando todos los objetos que aparezcan en ella. Al igual que en el caso anterior, el buen resultado de este comparador estará directamente relacionado con la buena segmentación de la imagen añadiendo el tamaño del grid correspondiente a cada caso.

En la figura 9.3 se presenta el diagrama de clases del proyecto realizado, esto incluye tanto descriptores como comparadores. Como se puede ver en el diagrama, se ha desarrollado el paquete *jmr.expansion* el cual incluye la clase *LabelProperties* y la clase *LabeledDescriptor*. Además de esto incluye diversos comparadores para la clase *LabelProperties* junto con el paquete

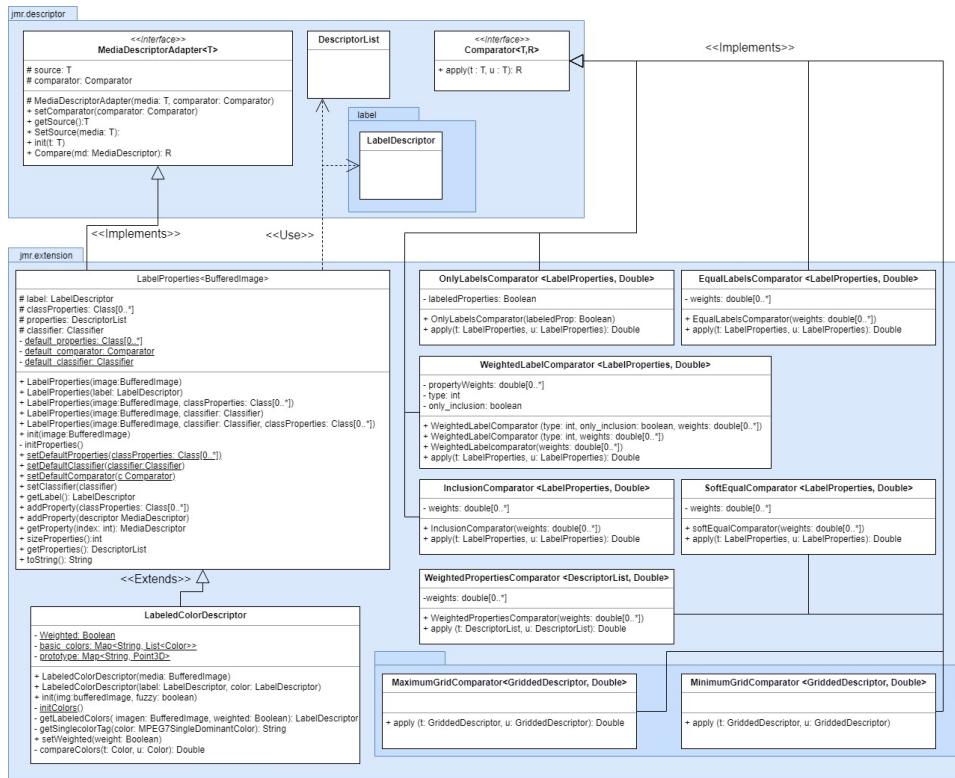


Figura 9.3: Diagrama de clases del proyecto realizado

*jmr.expansion.grid*. Este último paquete incluye los comparadores que se han desarrollado para la clase *GriddedDescriptor*



# Capítulo 10

## Implementación

Una de las características que se han buscado en este proyecto ha sido la versatilidad y es que al estar desarrollado usando la biblioteca *JMR* se ha seguido el estilo y el espíritu de esta biblioteca, permitiendo así que el proyecto realizado pueda ser usado por cualquier persona cuyo objetivo sea la recuperación de imágenes. Esto se ha conseguido siguiendo el hilo marcado por la biblioteca, desarrollando todo en Java y de forma que cualquier persona pueda realizar un descriptor propio y pueda usarlo en combinación con este proyecto.

El entorno de programación que se ha elegido para este proyecto ha sido *Netbeans IDE 8.0.1* [18] ya que este entorno provee varias herramientas para el desarrollo de una GUI a la par que herramientas que ayudan al programador, como la posibilidad de utilizar el gestor de proyectos GitHub de forma fácil e intuitiva. El repositorio de Github donde se encuentra alojado este proyecto puede ser encontrado en el enlace de la bibliografía [14].

Además de lo mencionado anteriormente, se ha utilizado *Javadoc* para generar una documentación lo mas clara posible. Esta documentación puede ser encontrada en la entrega del proyecto, junto a esta memoria. En la figura 10.1 se puede ver una captura de dicha documentación.

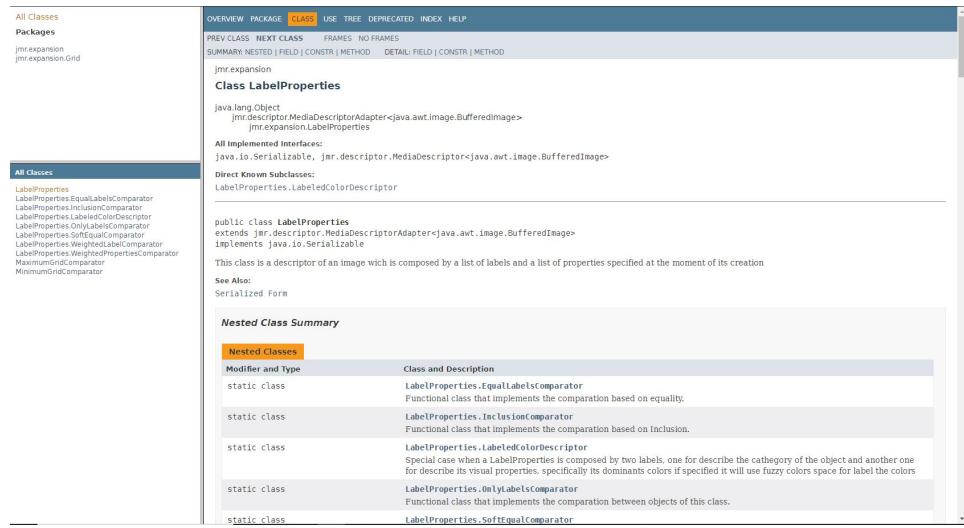


Figura 10.1: Captura de pantalla de la documentación del proyecto

Todo el código y la documentación de este, han sido escritos en inglés siguiendo el hilo de la biblioteca *JMR*. De esta manera el código desarrollado podrá ser reutilizado por cualquier persona, independientemente de su idioma.

# Capítulo 11

## Manual

En este capítulo se hablará sobre la interfaz que se aporta para comprobar el correcto funcionamiento del sistema creado. Esta interfaz ha sido creada adaptando algunas clases y métodos tanto de la *JMR* como de la *JFI* y creando las funcionalidades que han ido siendo necesarias. Por tanto, este capítulo puede ser entendido como unas instrucciones para poder manejar la interfaz con soltura y sin ninguna duda.

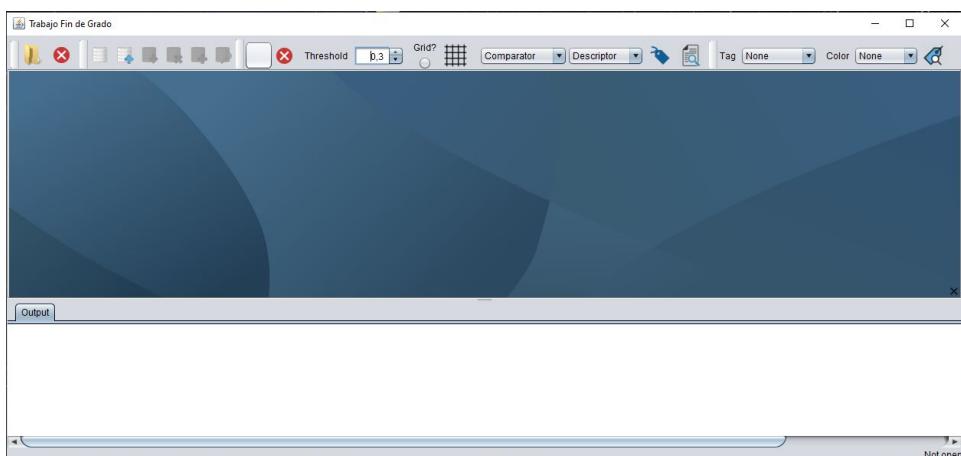


Figura 11.1: Pantalla principal de la interfaz

Para comenzar hablaremos de las distintas zonas que se pueden ver en la pantalla principal (11.1). En esta pantalla se pueden distinguir tres zonas a simple vista:

- En la parte superior podemos ver la barra de herramientas, la cual aporta la mayoría de las funcionalidades del sistema.
- En la parte media se encuentra el escritorio, sera en este escritorio donde aparecerán todas las ventanas que contienen las imágenes

- En la parte inferior nos encontramos la zona de 'Output'. Es en este lugar donde podremos ver las etiquetas lingüísticas de las distintas imágenes.

## 11.1. Barra de herramientas



Figura 11.2: Barra de herramientas de la interfaz

En la barra de herramientas de la figura 11.2 podemos ver que hay separaciones entre los distintos botones. Estas separaciones se basan en la funcionalidad de las herramientas que se encuentran a cada lado de ellas. Podemos distinguir por tanto 4 zonas dentro de la barra de herramientas: Primero una sección que hace referencia a los archivos (11.3), segundo la sección de la base de datos (11.4), en tercer lugar el descriptor creado con sus opciones (11.5) y por ultimo, la búsqueda por etiqueta (11.7).

### 11.1.1. Abrir y cerrar imágenes



Figura 11.3: Botones para abrir y cerrar imágenes

Por lo tanto empezaremos por la primera de estas secciones (11.3), siendo esta una de las más simples. Si pulsamos el botón con el ícono de la carpeta, se abrirá un selector de archivos permitiéndonos buscar imágenes en nuestro equipo pudiendo seleccionar varias imágenes de una sola vez. Si pulsamos el botón de la derecha, cerraremos todas las ventanas de imágenes que se encuentren abiertas en el escritorio (incluyendo las ventanas de los resultados).

### 11.1.2. Base de datos



Figura 11.4: Sección de la base de datos

A la derecha de la sección de los archivos, nos encontraremos las opciones de la base de datos. Estas opciones se encuentran en el siguiente orden: Crear nueva base de datos, Abrir una base de datos existente (Se abrirá un selector de archivos), guardar base de datos (se abrirá un selector de archivos donde podremos elegir donde guardar la base de datos), cerrar base de datos, añadir imágenes a la base de datos (Se añadirán todas y cada una de las imágenes que se encuentren abiertas) y buscar en la base de datos la imagen seleccionada.

En esta sección hay que aclarar que en el momento de la creación de la base de datos, han de estar seleccionadas las propiedades que queramos utilizar y en el caso de querer utilizar un grid esta opción ha de estar también seleccionada. Esto ocurre por que una vez creada la base de datos a la hora de añadir nuevas imágenes, se utilizarán las opciones que se encuentren seleccionadas y no se podrán añadir registros a la base de datos que sean de un tipo diferente al de la base de datos. Además de esto, en la consulta de imágenes ocurrirá lo mismo, la consulta ha de realizarse con las mismas opciones que con las que se añadieron las imágenes.

### 11.1.3. Opciones del descriptor



Figura 11.5: Opciones de configuración del descriptor

En este caso, la sección es mas compleja ya que aglutina varias opciones. Primero tenemos dos botones para seleccionar el clasificador y el botón para cerrarlo. Este clasificador será el usado para etiquetar las imágenes y en caso de no seleccionar uno o de estar cerrado no se tendrán en cuenta las etiquetas y se realizaran las búsquedas solo por las propiedades visuales. Además de esto podremos modificar el umbral del clasificador con la barra de la derecha, etiquetada como '*Threshold*' esta opción hará que todas las etiquetas por debajo de ese umbral sean descartadas.

A la derecha de estas tres opciones tenemos el botón de selección del grid, en el caso de que este botón este activado se creara un grid con el descriptor que se ha creado, permitiendo realizar consultas por varios objetos (Consultas del tipo 'mechero amarillo y taza roja').

A la derecha del botón de selección del grid tenemos las opciones del grid en forma de un único botón, si lo pulsamos aparecerá una ventana con las opciones de configuración 11.6. Estas son el tamaño del grid, el compa-

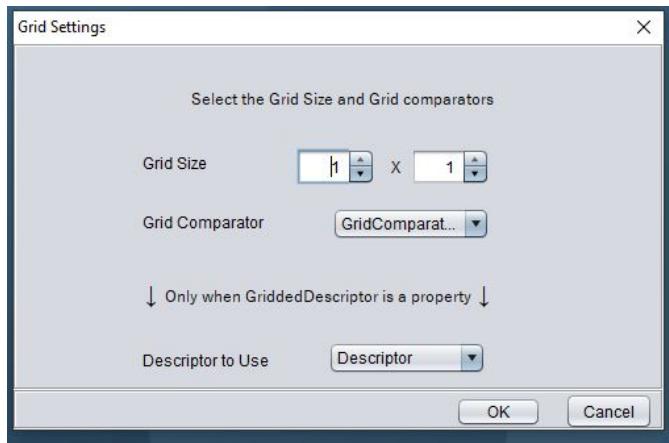


Figura 11.6: Panel de configuración del grid

rador (Si no se selecciona nada se usará el *MinimumGridComparator*), y en el caso de que el grid sea tomado como una propiedad visual, el descriptor que utilizará este.

Si continuamos por la barra de herramientas, nos encontramos la selección del comparador del descriptor realizado. Dentro de esta opción cabe destacar que si seleccionamos el comparador *Weighted* aparecerán mas opciones permitiéndonos ajustar los parámetros de este comparador.

Continuando por la sección nos encontramos el selector de descriptor de propiedades visuales, aquí poco hay que decir, salvo que en el caso de que seleccionemos *GriddedDescriptor* se tendrá en cuenta la opción que se encuentre marcada en la ventana de configuración del grid.

Para finalizar tenemos el botón de etiquetado y búsqueda. En el primer caso, el sistema etiquetara la imagen que tengamos seleccionada en ese momento y nos mostrara la salida en la parte inferior del programa, en la sección de output. Cabe destacar que en caso de haber seleccionado el descriptor *LabeledColor* las etiquetas de color también aparecerán en el output. En el segundo caso realizaremos una búsqueda, utilizando las opciones que se hayan marcado, de la imagen seleccionada entre todas las imágenes que tengamos abiertas en el sistema. La salida aparecerá en una nueva ventana con las imágenes en miniatura y ordenadas de mayor a menor relevancia.

#### 11.1.4. Búsqueda por etiquetas



Figura 11.7: Barra de herramientas de búsqueda por etiqueta

En esta sección aparece todo lo relevante para el caso de la búsqueda por etiquetas de color, primero vemos un selector de etiqueta en el cual aparecerán todas las categorías que se encuentren en la base de datos. Por otro lado nos encontramos un selector para seleccionar si queremos usar o no los conjuntos difusos. Esta selección sera relevante en el caso de que queramos etiquetar una imagen junto con sus colores o de que queramos añadir imágenes a la base de datos. Por ultimo encontramos el selector de etiqueta de color y el botón de búsqueda en la base de datos según las etiquetas especificadas. Cabe destacar que podremos realizar una búsqueda solo por etiqueta, sin seleccionar ningún color. Para ello basta con seleccionar en el desplegable '*None*'

Para finalizar con la barra de herramientas, cabe destacar que en el caso de que no se cumplan algunas condiciones no se podrán pulsar algunos botones o no se mostrarán. Por ejemplo, no se podrá guardar o buscar en una base de datos si no hay ninguna base de datos abierta.

## 11.2. Escritorio

En esta sección se hablará de la parte central del programa y es que aunque no presente ningún botón tiene algunas funcionalidades. Estas funcionalidades son las siguientes:

- Una vez obtenido un resultado, si hacemos doble click en alguna de las miniaturas que aparecen en el resultado obtendremos las etiquetas de dicha imagen. En el caso de que el descriptor *LabeledColor* este seleccionado, al hacer doble click también obtendremos las etiquetas de color, pudiendo seleccionar también si queremos usar conjuntos difusos o no.
- Al abrir una imagen esta no aparece escalada, si no que aparece en su tamaño original, si esta es muy grande puede que queramos reducir su tamaño, o al contrario aumentarlo si la imagen es muy pequeña. Esto podremos hacerlo con los botones '-' y '+'. Cabe destacar que en algunas ocasiones no se detecta bien la pulsación de estas teclas en algunas imágenes, cosa que se podrá arreglar pulsando en otra imagen y haciendo 'zoom in' o 'zoom out'. Si vemos que el zoom se ha aplicado

el problema se habrá resuelto y podremos hacer lo mismo en la imagen anterior.

### **11.3. Output**

En esta sección no hay mucho de lo que hablar y es que esta parte del programa solo nos muestra las etiquetas cuando lo solicitemos. Además de esto también podremos hacer click derecho para limpiar la salida.

Cabe destacar que la separación entre el escritorio y el output se puede mover verticalmente para ajustar el tamaño de estos dos componentes.

### **11.4. Demostración**

Para que quede claro como usar este sistema, realizaremos dos consultas de ejemplo aportando capturas de pantalla del procedimiento que debemos seguir para ello. El primer ejemplo consistirá en realizar una consulta partiendo de una imagen con dos objetos y buscando imágenes que nos muestren alguno de los dos objetos. El segundo ejemplo consistirá en la creación de una base de datos de imágenes, con sus respectivos colores dominantes etiquetados para luego realizar una consulta por etiquetas lingüísticas.

Partiremos en ambos casos de la ventana principal del programa que podemos ver en la figura 11.1.

#### **11.4.1. Primer ejemplo**

Partiendo de la ventana principal, lo primero que deberemos hacer es seleccionar el clasificador que utilizaremos. Para ello pulsamos el botón de selección de clasificador con lo que obtendremos una ventana de selección de archivo como la que se puede ver en 11.8. En esta ventana nos desplazaremos hasta la ubicación donde se encuentre el clasificador que vayamos a usar y seleccionaremos el archivo con la extensión 'xml'.

Una vez hemos cargado el clasificador seleccionaremos las opciones que queramos utilizar y abriremos la base de datos. eligiendo primero las opciones que vayamos a usar. Estas opciones serán marcar la opción del Grid y seleccionar los descriptores que queramos usar, en este caso, serán los de la estructura de color. La barra de herramientas quedaría tal y como se puede ver en la figura 11.9. Una vez en este punto, podremos crear la base de datos pulsando el primer botón de la sección de la barra de herramientas dedicada a la base de datos. Se observará que la base de datos ha sido creada porque nos aparece el mensaje 'New DB (not saved)[#0][GriddedDescriptor]' en la parte inferior izquierda de la ventana del programa, justo debajo de la salida

de texto.

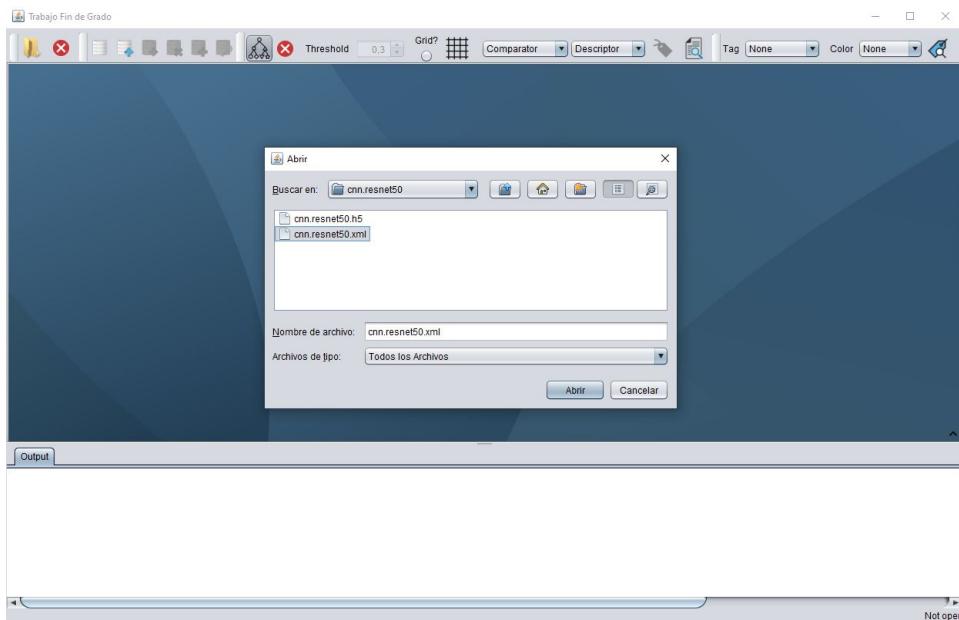


Figura 11.8: Ventana de selección del clasificador

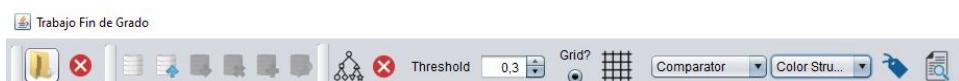


Figura 11.9: Barra de herramientas tras elegir el descriptor y el Grid

Llegados a este punto, debemos cargar la base de datos. Para ello debemos pulsar el primer botón de la barra de herramientas, obteniendo una ventana de selección de archivos similar a la obtenida en la selección del clasificador. En esta ventana seleccionaremos las imágenes con las que queremos cargar la base de datos. En este caso, no podemos cargar en la base de datos todas las imágenes a la vez y es que antes de añadirlas tenemos que seleccionar en cuantas regiones queremos partir las imágenes. De esta forma todas las imágenes con un objeto se añadirán con un tamaño de Grid de (1x1). En el caso de tener dos objetos, uno a cada lado usaremos un tamaño de grid de (2x1) y así sucesivamente. Para ello abriremos primero las imágenes de un objeto y pulsaremos el botón de configuración del Grid obteniendo la pantalla de la figura 11.10. En esta pantalla seleccionaremos el tamaño del Grid, en este primer caso será de (1x1). Cuando hayamos seleccionado el tamaño pulsaremos 'Ok' y el botón de 'Añadir registros a la base de datos' (el quinto en la sección de la base de datos del panel de herramientas). Realizaremos esto, con las imágenes de uno, dos y tres objetos.

Cuando tengamos la base de datos cargada, solo faltará abrir la imagen que queramos buscar, seleccionar el tamaño del grid como se ha hecho antes y seleccionar los comparadores del grid y del descriptor principal. Los comparadores del Grid pueden ser elegidos en la pantalla que se ve en la imagen de la figura 11.10, estos se encuentran justo bajo el selector del tamaño del Grid. Por otra parte los comparadores del descriptor principal se pueden elegir en la lista desplegable que se encuentra a la izquierda del selector de descriptor de la barra de herramientas. Para este ejemplo utilizaremos el descriptor 'SoftEqual'

Llegados a este punto podremos realizar la consulta, obteniendo la imagen de la figura 11.11.

#### 11.4.2. Segundo Ejemplo

En esta sección como se comentó anteriormente realizaremos un ejemplo de uso del sistema el cual consistirá en realizar dos bases de datos las cuales estén formadas por las etiquetas que describen el tipo de objeto y por las etiquetas de los colores dominantes de estas imágenes. La primera de estas bases de datos no le asignara peso a las etiquetas de color, mientras que la segunda si que lo hará. Una vez obtenidas dichas base de datos la guardaremos y realizaremos dos búsquedas de imágenes por etiquetas, primero sin utilizar los conjuntos difusos y luego utilizando estos. Partimos por lo tanto y como se comentó anteriormente de un sistema recién iniciado, sin ninguna imagen ni característica seleccionada.

Primero, al igual que en el caso anterior, abriremos el clasificador que queramos usar utilizando el mismo método que describimos en el ejemplo anterior. (Figura 11.8). Una vez hayamos seleccionado el descriptor utilizaremos el selector de descriptor y elegiremos la opción 'Labeled Colors'. En este momento crearemos la base de datos y las imágenes que queramos añadir a esta. Quedando la pantalla como se puede ver en la figura 11.12.

En este punto posiblemente nos hayamos percatado de que ha aparecido un botón seleccionable a la izquierda del ultimo ícono de la barra de herramientas, en su sección de búsqueda por etiqueta. Este botón contiene el texto 'Fuzzy Colors' y se puede ver en la figura 11.12. Esta opción está disponible únicamente cuando seleccionamos el descriptor 'Labeled Colors'. De momento lo dejaremos sin seleccionar y pulsaremos el botón de añadir imágenes a la base de datos. Una vez el proceso haya terminado, pulsaremos el botón de guardar base de datos obteniendo un selector de archivos que nos permite guardar la base de datos donde queramos y elegir un nombre para esta. Esto se puede ver en la figura 11.13. Como se puede ver guardaremos

la base de datos con el nombre 'Base\_de\_datos\_ejemplo.db'.

Una vez guardada, la cerraremos y abriremos una nueva y en este caso marcaremos la opción 'Fuzzy Colors' antes de añadir las imágenes. Una vez añadidas las imágenes, guardaremos la base de datos con un nombre diferente para poder diferenciarlas. En mi caso utilizaré el nombre 'Base\_de\_datos\_ejemplo\_2.db'. Ahora que tenemos las dos bases de datos podemos cerrar todas las imágenes de la pantalla y abrir la primera base de datos.

Para realizar la consulta, en la parte derecha de la barra de herramientas seleccionaremos la primera etiqueta, referente a la categoría de los objetos y la segunda, referente al color dominante de la imagen y pulsaremos el botón de búsqueda (sin estar el botón de 'Fuzzy Colors' seleccionado). El resultado se puede ver en la figura 11.14. Para realizar la ultima consulta, cerraremos la base de datos, abriremos la segunda que creamos y, esta vez si, pulsaremos el botón de 'Fuzzy Colors' antes de buscar. El resultado de esta ultima consulta se puede ver en la figura 11.15. Destacar que el nombre de las bases de datos utilizadas se puede ver en la esquina inferior derecha de las imágenes anteriormente mencionadas.

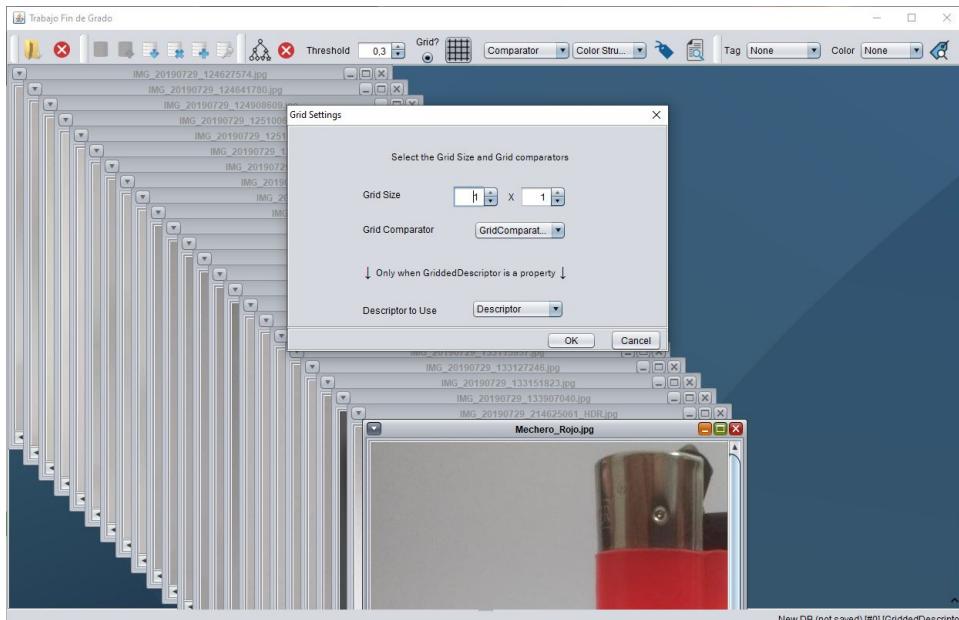


Figura 11.10: Ventana de selección de las opciones del Grid

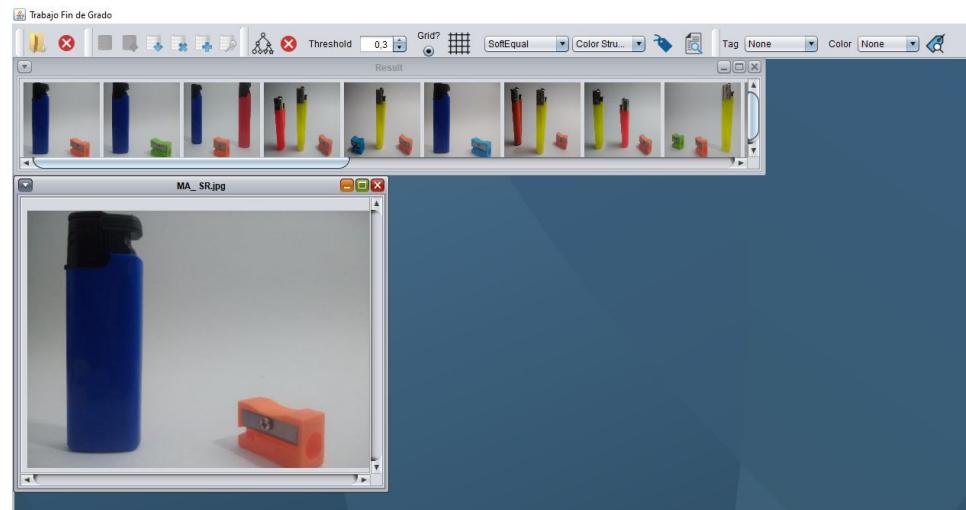


Figura 11.11: Resultado final del primer ejemplo de consulta

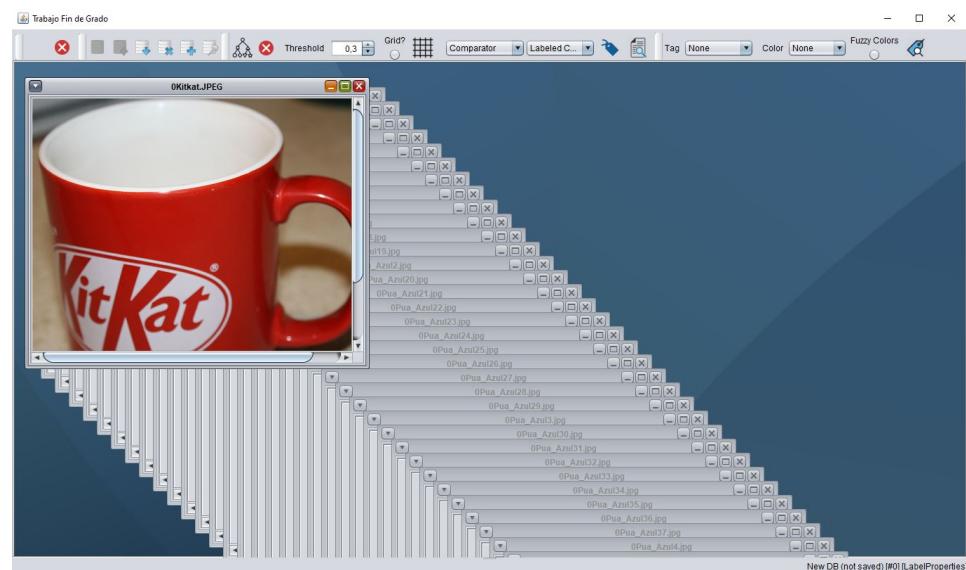


Figura 11.12: Visualización del estado del segundo ejemplo

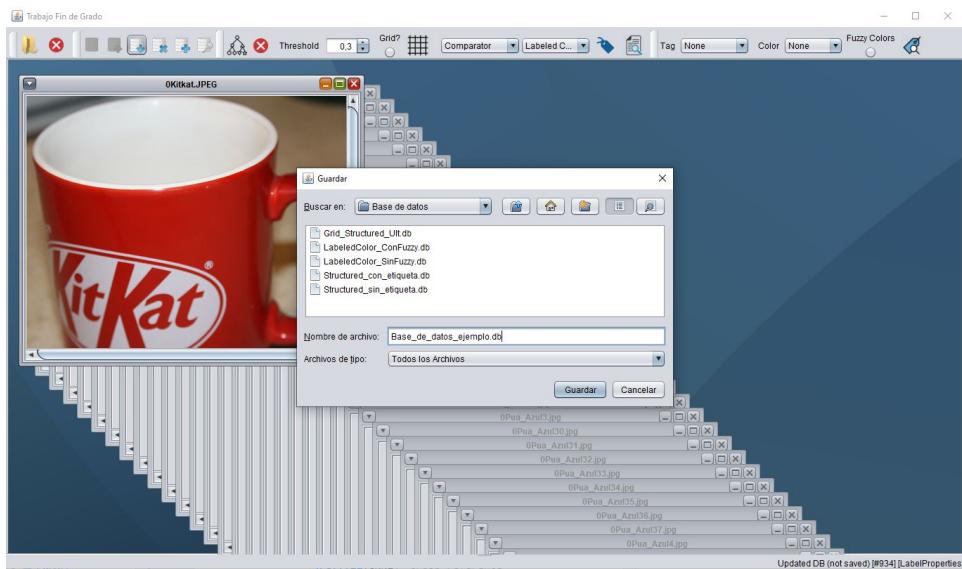


Figura 11.13: Pantalla de guardado de la base de datos

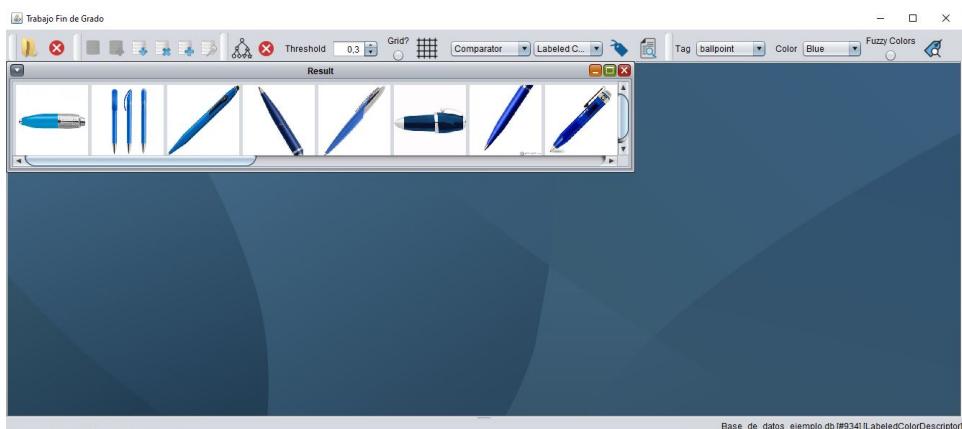


Figura 11.14: Resultado de realizar la búsqueda sin utilizar los conjuntos difusos

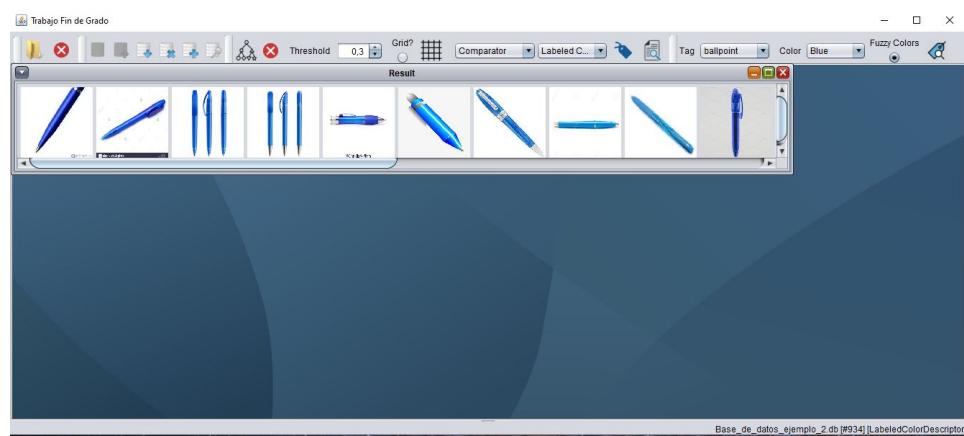


Figura 11.15: Resultado de realizar la búsqueda utilizando los conjuntos difusos

# Capítulo 12

## Conclusiones

En este ultimo capítulo se hablará sobre las conclusiones generales del proyecto teniendo en cuenta las dos partes principales de este, el uso de las *CNNs* y de los métodos disponibles que hay para reconocer las propiedades visuales de una imagen. Además de esto hablaremos sobre las posibles vías de trabajo futuro de este proyecto.

Como se ha podido ver, el sistema que se ha desarrollado, bebe de dos campos diferentes, siendo uno las técnicas de análisis de las propiedades visuales y otro el uso de redes neuronales para el etiquetado de imágenes. El estudio de este ultimo campo, aun a pesar de que no se ha desarrollado la red neuronal en este proyecto ya que no era el objetivo de este, me ha permito conocer mucho mejor estas estructuras y conocer las diferentes arquitecturas que podemos encontrar dentro de las redes neuronales. Esto era algo fundamental pues es un campo que se encuentra en auge y que esta presente en casi cualquier tecnología que veamos.

Por otro lado, el estudio de las diferentes métricas que podemos utilizar para reconocer las propiedades visuales de un objeto nos ha permitido tener una visión mucho más clara sobre las diferentes problemáticas que surgen al intentar analizar una imagen de forma global y no de forma local. Estos problemas como hemos visto son el fondo que estas presentan y es que este sera tomado en cuenta para el análisis del color del objeto.

Por otro lado mediante el etiquetado de los colores dominantes de la imagen se ha podido hacer una pequeña aproximación al problema del etiquetado de colores o *Color Naming* y es que este problema es algo que aun hoy en día sigue siendo materia de estudio después de varios años ya que no es una tarea baladí. Esto último ha quedado patente en este trabajo y es que aun habiendo aportado una solución que funciona bien, aunque queda trabajo por delante.

Además de todo esto, al haber desarrollado el trabajo siguiendo la filosofía utilizada en la *JMR* he podido aprender de la importancia de desarrollar un código lo más modularizado posible, de forma que en el supuesto de que alguien quisiese utilizar el descriptor creado aquí pueda desarrollar sus propias herramientas y métodos partiendo de todo lo que ya se aporta.

Si hablamos de las posibles vías de trabajo futuro podríamos intentar mejorar este proyecto añadiendo la función de que las imágenes se dividan automáticamente en subsecciones variando la posición de esta hasta que la red neuronal encuentre una etiqueta con un valor por encima de un umbral dado. Esto nos permitiría que dada una imagen no nos importe el fondo que haya en esta ya que estaríamos analizando solo una región en la cual el color del objeto sera muy relevante, no teniendo en cuenta el color de fondo de la imagen original.

Por otra parte como se comentó durante la validación del sistema, otra posible vía de trabajo futuro podría consistir en desarrollar descriptores de textura o forma, permitiéndonos el uso del comparador basado en el peso de las propiedades. Esto nos permitiría realizar consultas mucho más complejas variando si queremos centrarnos más en el color o en la textura.

# Bibliografía

- [1] Ritendra Datta, Dhiraj joshi, Jia Li, James Z. Wang (The Pennsylvania State University, Pennsylvania) *Image Retrieval: Ideas, Influences, and Trends of the New Age* <http://infolab.stanford.edu/~wangz/project/imsearch/review/JOUR/datta.pdf>
- [2] J. Chamorro-Martínez, J.M. Soto-Hidalgo, P.M. Martínez-Jiménez and D. Sánchez *Fuzzy Color Spaces: A Conceptual Approach to Color Vision* IEEE Transactions on Fuzzy Systems, vol.25, no 5, pp 1264 - 1280, Oct. 2017. DOI: <http://dx.doi.org/10.1109/TFUZZ.2016.2612259>
- [3] A. Chandra Lagandula <https://towardsdatascience.com/perceptron-the-artificial-neuron-4d8c70d5cc8d>
- [4] *The perceptron: A probabilistic model for information storage and organization in the brain* <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.335.3398&rep=rep1&type=pdf>
- [5] T.M. Mitchell *Machine Learning* <http://profsite.um.ac.ir/~monsefi/\machine-learning/pdf/Machine-Learning- Tom-Mitchell.pdf>
- [6] S. Russell, P. Norvig *Artificial Intelligence (A modern approach)* <https://www.cin.ufpe.br/~tf12/\artificial-intelligence-modern-approach.9780131038059.25368.pdf>
- [7] *Convolutional Neural Networks for Visual Recognition.* <http://cs231n.github.io/neural-networks-1/>
- [8] M. Nielsen *Neural Networks and Deep Learning* <http://static.latexstudio.net/article/2018/0912/neuralnetworksanddeeplearning.pdf>
- [9] *Understanding Convolution, the core of Convolutional Neural Networks* <https://beckernick.github.io/convolutions/>
- [10] Robert Benavente, Maria Vanrell, Ramon Baldrich *A Data Set For Fuzzy Color Naming* [http://www.cat.cvc.uab.es/Public/Publications/2006/BVB2006/CRA2006\\_preprint\\_web.pdf](http://www.cat.cvc.uab.es/Public/Publications/2006/BVB2006/CRA2006_preprint_web.pdf)

- [11] Nikita Upadhyaya, Manish Dixit *A Review: Relating Low Level Features to High Level Semantics in CBIR* International Journal of Signal Processing, Image Processing and Pattern Recognition Vol.9, No.3 (2016), pp.433-444 DOI:<http://dx.doi.org/10.14257/ij sip.2016.9.3.37>
- [12] Deane B. Judd, Kenneth L. Kelly *Method of Designating Colors* Journal of Research of the National Bureau of Standards, Volume 23, September 1939 Research Paper RP1239 [https://nvlpubs.nist.gov/nistpubs/jres/23/jresv23n3p355\\_A1b.pdf](https://nvlpubs.nist.gov/nistpubs/jres/23/jresv23n3p355_A1b.pdf)
- [13] Google Imagenes <https://www.google.es/imghp>
- [14] Repositorio del proyecto <https://github.com/FernandoRoldan93/TFG>
- [15] *Java Fuzzy Color System* <http://www.jfcsssoftware.com/>
- [16] Java <https://www.oracle.com/java/>
- [17] Documentación de la interfaz Serializable <https://docs.oracle.com/javase/7/docs/api/java/io/Serializable.html>
- [18] Página de Netbeans <https://netbeans.org/>
- [19] Página del ISCC, (*Inter-Society Color Council*) <https://www.iscc.org>
- [20] Tyagi, Vipin. *Content-Based Image Retrieval Ideas, Influences, and Current Trends* <https://link.springer.com/book/10.1007%2F978-981-10-6759-4>
- [21] Eakins John, Graham Margaret University of Northumbria at Newcastle *Content-based Image Retrieval*[http://www.leeds.ac.uk/educol/documents/00001240.htm#\\_Toc442192699](http://www.leeds.ac.uk/educol/documents/00001240.htm#_Toc442192699)
- [22] P.Alshuth, Th.Hermes, Ch.Klauck, J.Krey and M.Röper. Universidad de Bremen, Alemania *IRIS - Image retrieval for images and videos* <https://www.semanticscholar.org/paper/IRIS-Image-retrieval-for-images-and-videos-Klauck-Krey/7588ed311d25da2c55bc8fa41279c1190653eeb5>
- [23] Widanagamaachchi, Wathsala. Universidad de Utah *Emotion Recognition with Image Processing and Neural Networks* [https://www.researchgate.net/publication/261250781\\_Emotion\\_Recognition\\_with\\_Image\\_Processing\\_and\\_Neural\\_Networks](https://www.researchgate.net/publication/261250781_Emotion_Recognition_with_Image_Processing_and_Neural_Networks)
- [24] Aït Younes, Amine. Truck, Isis. Akdag, Herman. *Image Retrieval using Fuzzy Representation of Colors* Soft Computing, vol:

11(3), pp: 287-298, February 2007, DOI: 10.1007/s00500-006-0070-x  
[https://www.researchgate.net/publication/225152193\\_Image\\_Retrieval\\_using\\_Fuzzy\\_Representation\\_of\\_Colors](https://www.researchgate.net/publication/225152193_Image_Retrieval_using_Fuzzy_Representation_of_Colors)

[25] A. Okabe, B. Boots, and K. Sugihara, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*

[26] Pagina de ImageNet <http://image-net.org/>

[27] Página de la competición de localización de objetos en imágenes <https://www.kaggle.com/c/imagenet-object-localization-challenge>